第 6 章　支持向量机

6.1 间隔与支持向量

给定训练样本集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_m,y_m)\}$, $y_i\in\{-1,+1\}$, 分类学习最基本的想法就是基于训练集 D 在样本空间中找到一个划分超平面, 将不同类别的样本分开. 但能将训练样本分开的划分超平面可能有很多, 如图 6.1 所示, 我们应该努力去找到哪一个呢?

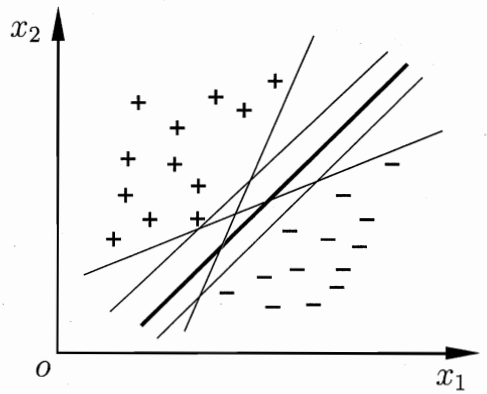

图 6.1　存在多个划分超平面将两类训练样本分开

直观上看, 应该去找位于两类训练样本“正中间”的划分超平面, 即图 6.1 中红色的那个, 因为该划分超平面对训练样本局部扰动的“容忍”性最好. 例如, 由于训练集的局限性或噪声的因素, 训练集外的样本可能比图 6.1 中的训练样本更接近两个类的分隔界, 这将使许多划分超平面出现错误, 而红色的超平面受影响最小. 换言之, 这个划分超平面所产生的分类结果是最鲁棒的, 对未见示例的泛化能力最强.

在样本空间中, 划分超平面可通过如下线性方程来描述:

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0\,, \tag{6.1}$$

其中 $\boldsymbol{w}=(w_1;w_2;\ldots;w_d)$ 为法向量, 决定了超平面的方向; b 为位移项, 决定了超平面与原点之间的距离. 显然, 划分超平面可被法向量 $\boldsymbol{w}$ 和位移 b 确定,

参见习题 6.1.

下面我们将其记为 $(\boldsymbol{w}, b)$. 样本空间中任意点 $\boldsymbol{x}$ 到超平面 $(\boldsymbol{w}, b)$ 的距离可写为

$$r = \frac{|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b|}{||\boldsymbol{w}||} . \tag{6.2}$$

若超平面 $(\boldsymbol{w}', b')$ 能将训练样本正确分类, 则总存在缩放变换 $\varsigma\boldsymbol{w} \mapsto \boldsymbol{w}'$ 和 $\varsigma b \mapsto b'$ 使式(6.3)成立.

假设超平面 $(\boldsymbol{w}, b)$ 能将训练样本正确分类, 即对于 $(\boldsymbol{x}_i, y_i) \in D$, 若 $y_i = +1$, 则有 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b > 0$; 若 $y_i = -1$, 则有 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b < 0$. 令

$$\begin{cases} \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b \geqslant +1, & y_i = +1 ; \\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b \leqslant -1, & y_i = -1 . \end{cases} \tag{6.3}$$

每个样本点对应一个特征向量.

如图 6.2 所示, 距离超平面最近的这几个训练样本点使式(6.3)的等号成立, 它们被称为“支持向量”(support vector), 两个异类支持向量到超平面的距离之和为

$$\gamma = \frac{2}{||\boldsymbol{w}||} , \tag{6.4}$$

它被称为“间隔”(margin).

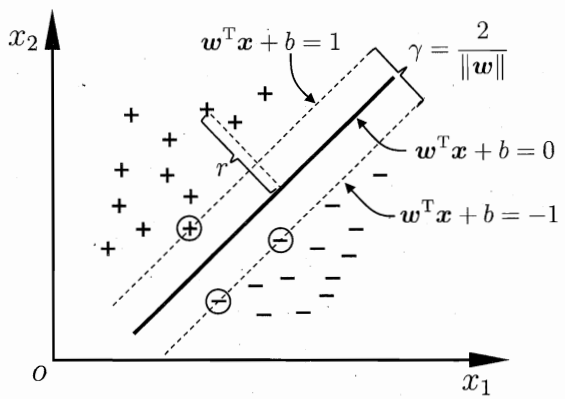

图 6.2 支持向量与间隔

欲找到具有“最大间隔”(maximum margin)的划分超平面, 也就是要找到能满足式(6.3)中约束的参数 $\boldsymbol{w}$ 和 b, 使得 γ 最大, 即

$$\begin{aligned} &\max_{\boldsymbol{w}, b} \quad \frac{2}{||\boldsymbol{w}||} \\ &\text{s.t.} \quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{6.5}$$

间隔貌似仅与 $\boldsymbol{w}$ 有关, 但事实上 b 通过约束隐式地影响着 $\boldsymbol{w}$ 的取值, 进而对间隔产生影响.

显然, 为了最大化间隔, 仅需最大化 $\|\boldsymbol{w}\|^{-1}$, 这等价于最小化 $\|\boldsymbol{w}\|^2$. 于是, 式(6.5)可重写为

$$\begin{aligned} \min_{\boldsymbol{w},b} \ & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \ & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{6.6}$$

这就是支持向量机(Support Vector Machine, 简称 SVM)的基本型.

6.2 对偶问题

我们希望求解式(6.6)来得到大间隔划分超平面所对应的模型

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \,, \tag{6.7}$$

其中 $\boldsymbol{w}$ 和 b 是模型参数. 注意到式(6.6)本身是一个凸二次规划(convex quadratic programming) 问题, 能直接用现成的优化计算包求解, 但我们可以有更高效的办法.

参见附录 B.1.

对式(6.6)使用拉格朗日乘子法可得到其“对偶问题” (dual problem). 具体来说, 对式(6.6) 的每条约束添加拉格朗日乘子 $\alpha_i \geqslant 0$, 则该问题的拉格朗日函数可写为

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)\right) \,, \tag{6.8}$$

其中 $\boldsymbol{\alpha} = (\alpha_1; \alpha_2; \ldots; \alpha_m)$. 令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 对 $\boldsymbol{w}$ 和 b 的偏导为零可得

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i \,, \tag{6.9}$$

$$0 = \sum_{i=1}^{m} \alpha_i y_i \,. \tag{6.10}$$

将式(6.9)代入(6.8), 即可将 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 中的 $\boldsymbol{w}$ 和 b 消去, 再考虑式(6.10)的约束, 就得到式(6.6)的对偶问题

$$\max_{\boldsymbol{\alpha}} \ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \tag{6.11}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0 \,,$$
$$\alpha_i \geqslant 0 \,, \quad i = 1, 2, \ldots, m \,.$$

解出 $\boldsymbol{\alpha}$ 后, 求出 $\boldsymbol{w}$ 与 b 即可得到模型

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b \\ &= \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x} + b \,. \end{aligned} \tag{6.12}$$

从对偶问题(6.11)解出的 α_i 是式(6.8)中的拉格朗日乘子, 它恰对应着训练样本 $(\boldsymbol{x}_i, y_i)$. 注意到式(6.6)中有不等式约束, 因此上述过程需满足 KKT (Karush-Kuhn-Tucker) 条件, 即要求

参见附录 B.1.

$$\begin{cases} \alpha_i \geqslant 0 \,; \\ y_i f(\boldsymbol{x}_i) - 1 \geqslant 0 \,; \\ \alpha_i \left(y_i f(\boldsymbol{x}_i) - 1 \right) = 0 \,. \end{cases} \tag{6.13}$$

于是, 对任意训练样本 $(\boldsymbol{x}_i, y_i)$, 总有 $\alpha_i = 0$ 或 $y_i f(\boldsymbol{x}_i) = 1$. 若 $\alpha_i = 0$, 则该样本将不会在式(6.12) 的求和中出现, 也就不会对 $f(\boldsymbol{x})$ 有任何影响; 若 $\alpha_i > 0$, 则必有 $y_i f(\boldsymbol{x}_i) = 1$, 所对应的样本点位于最大间隔边界上, 是一个支持向量. 这显示出支持向量机的一个重要性质: 训练完成后, 大部分的训练样本都不需保留, 最终模型仅与支持向量有关.

如 [Vapnik, 1999] 所述, 支持向量机这个名字强调了此类学习器的关键是如何从支持向量构建出解; 同时也暗示着其复杂度主要与支持向量的数目有关.

那么, 如何求解式(6.11) 呢? 不难发现, 这是一个二次规划问题, 可使用通用的二次规划算法来求解; 然而, 该问题的规模正比于训练样本数, 这会在实际任务中造成很大的开销. 为了避开这个障碍, 人们通过利用问题本身的特性, 提出了很多高效算法, SMO (Sequential Minimal Optimization) 是其中一个著名的代表 [Platt, 1998].

二次规划参见附录 B.2.

SMO 的基本思路是先固定 α_i 之外的所有参数, 然后求 α_i 上的极值. 由于存在约束 $\sum_{i=1}^{m} \alpha_i y_i = 0$, 若固定 α_i 之外的其他变量, 则 α_i 可由其他变量导出. 于是, SMO 每次选择两个变量 α_i 和 α_j, 并固定其他参数. 这样, 在参数初始化后, SMO 不断执行如下两个步骤直至收敛:

- 选取一对需更新的变量 α_i 和 α_j;

- 固定 α_i 和 α_j 以外的参数, 求解式(6.11)获得更新后的 α_i 和 α_j.

注意到只需选取的 α_i 和 α_j 中有一个不满足 KKT 条件(6.13), 目标函数就会在迭代后减小 [Osuna et al., 1997]. 直观来看, KKT 条件违背的程度越大, 则变量更新后可能导致的目标函数值减幅越大. 于是, SMO 先选取违背 KKT 条件程度最大的变量. 第二个变量应选择一个使目标函数值减小最快的变量, 但由于比较各变量所对应的目标函数值减幅的复杂度过高, 因此 SMO 采用了一个启发式: 使选取的两变量所对应样本之间的间隔最大. 一种直观的解释是, 这样的两个变量有很大的差别, 与对两个相似的变量进行更新相比, 对它们进行更新会带给目标函数值更大的变化.

SMO 算法之所以高效, 恰由于在固定其他参数后, 仅优化两个参数的过程能做到非常高效. 具体来说, 仅考虑 α_i 和 α_j 时, 式(6.11)中的约束可重写为

$$\alpha_i y_i + \alpha_j y_j = c\ ,\quad \alpha_i \geqslant 0\ ,\quad \alpha_j \geqslant 0\ , \tag{6.14}$$

其中

$$c = -\sum_{k\neq i,j} \alpha_k y_k \tag{6.15}$$

是使 $\sum_{i=1}^{m} \alpha_i y_i = 0$ 成立的常数. 用

$$\alpha_i y_i + \alpha_j y_j = c \tag{6.16}$$

消去式(6.11)中的变量 α_j, 则得到一个关于 α_i 的单变量二次规划问题, 仅有的约束是 $\alpha_i \geqslant 0$. 不难发现, 这样的二次规划问题具有闭式解, 于是不必调用数值优化算法即可高效地计算出更新后的 α_i 和 α_j.

如何确定偏移项 b 呢? 注意到对任意支持向量 $(\boldsymbol{x}_s, y_s)$ 都有 $y_s f(\boldsymbol{x}_s) = 1$, 即

$$y_s \left(\sum_{i\in S} \alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_s + b \right) = 1\ , \tag{6.17}$$

其中 $S = \{i \mid \alpha_i > 0,\ i = 1, 2, \ldots, m\}$ 为所有支持向量的下标集. 理论上, 可选取任意支持向量并通过求解式(6.17)获得 b, 但现实任务中常采用一种更鲁棒的做法: 使用所有支持向量求解的平均值

$$b = \frac{1}{|S|} \sum_{s\in S} \left(y_s - \sum_{i\in S} \alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_s \right)\ . \tag{6.18}$$

6.3 核函数

在本章前面的讨论中, 我们假设训练样本是线性可分的, 即存在一个划分超平面能将训练样本正确分类. 然而在现实任务中, 原始样本空间内也许并不存在一个能正确划分两类样本的超平面. 例如图 6.3 中的“异或”问题就不是线性可分的.

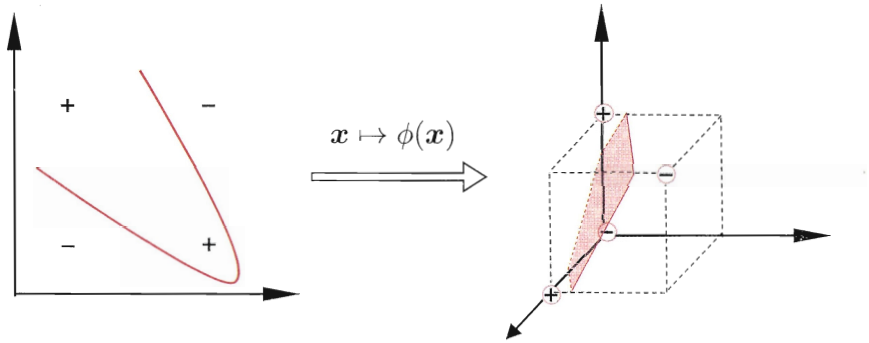

图 6.3 异或问题与非线性映射

对这样的问题, 可将样本从原始空间映射到一个更高维的特征空间, 使得样本在这个特征空间内线性可分. 例如在图 6.3 中, 若将原始的二维空间映射到一个合适的三维空间, 就能找到一个合适的划分超平面. 幸运的是, 如果原始空间是有限维, 即属性数有限, 那么一定存在一个高维特征空间使样本可分.

参见第 12 章.

令 $\phi(\boldsymbol{x})$ 表示将 $\boldsymbol{x}$ 映射后的特征向量, 于是, 在特征空间中划分超平面所对应的模型可表示为

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b\ , \tag{6.19}$$

其中 $\boldsymbol{w}$ 和 b 是模型参数. 类似式(6.6), 有

$$\begin{aligned} &\min_{\boldsymbol{w},b}\ \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t.}\ \ y_i(\boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}_i) + b) \geqslant 1, \quad i = 1, 2, \ldots, m. \end{aligned} \tag{6.20}$$

其对偶问题是

$$\max_{\boldsymbol{\alpha}}\ \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j) \tag{6.21}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \ , \\ & \alpha_i \geqslant 0 \ , \quad i = 1, 2, \ldots, m \ . \end{aligned}$$

求解式(6.21)涉及到计算 $\phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$, 这是样本 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 映射到特征空间之后的内积. 由于特征空间维数可能很高, 甚至可能是无穷维, 因此直接计算 $\phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$ 通常是困难的. 为了避开这个障碍, 可以设想这样一个函数:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j) \ , \tag{6.22}$$

这称为“核技巧” (kernel trick).

即 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 在特征空间的内积等于它们在原始样本空间中通过函数 $\kappa(\cdot, \cdot)$ 计算的结果. 有了这样的函数, 我们就不必直接去计算高维甚至无穷维特征空间中的内积, 于是式(6.21)可重写为

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \ , \\ & \alpha_i \geqslant 0 \ , \quad i = 1, 2, \ldots, m \ . \end{aligned} \tag{6.23}$$

求解后即可得到

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b \\ &= \sum_{i=1}^{m} \alpha_i y_i \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}) + b \\ &= \sum_{i=1}^{m} \alpha_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b \ . \end{aligned} \tag{6.24}$$

这里的函数 $\kappa(\cdot, \cdot)$ 就是“核函数” (kernel function). 式(6.24) 显示出模型最优解可通过训练样本的核函数展开, 这一展式亦称“支持向量展式” (support vector expansion).

显然, 若已知合适映射 $\phi(\cdot)$ 的具体形式, 则可写出核函数 $\kappa(\cdot, \cdot)$. 但在现实任务中我们通常不知道 $\phi(\cdot)$ 是什么形式, 那么, 合适的核函数是否一定存在呢? 什么样的函数能做核函数呢? 我们有下面的定理:

证明可参阅 [Schölkopf and Smola, 2002].

定理 6.1 (核函数) 令 $\mathcal{X}$ 为输入空间, $\kappa(\cdot,\cdot)$ 是定义在 $\mathcal{X}\times\mathcal{X}$ 上的对称函数, 则 κ 是核函数当且仅当对于任意数据 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$, "核矩阵" (kernel matrix) $\mathbf{K}$ 总是半正定的:

$$\mathbf{K}=\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_m)\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_i,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_i,\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x}_i,\boldsymbol{x}_m)\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_m,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_m,\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x}_m,\boldsymbol{x}_m)\end{bmatrix}.$$

定理 6.1 表明, 只要一个对称函数所对应的核矩阵半正定, 它就能作为核函数使用. 事实上, 对于一个半正定核矩阵, 总能找到一个与之对应的映射 ϕ. 换言之, 任何一个核函数都隐式地定义了一个称为 "再生核希尔伯特空间" (Reproducing Kernel Hilbert Space, 简称 RKHS)的特征空间.

通过前面的讨论可知, 我们希望样本在特征空间内线性可分, 因此特征空间的好坏对支持向量机的性能至关重要. 需注意的是, 在不知道特征映射的形式时, 我们并不知道什么样的核函数是合适的, 而核函数也仅是隐式地定义了这个特征空间. 于是, "核函数选择" 成为支持向量机的最大变数. 若核函数选择不合适, 则意味着将样本映射到了一个不合适的特征空间, 很可能导致性能不佳.

这方面有一些基本的经验, 例如对文本数据通常采用线性核, 情况不明时可先尝试高斯核.

表 6.1 列出了几种常用的核函数.

表 6.1 常用核函数

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j)^d$	$d\geqslant 1$ 为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma^2}\right)$	$\sigma>0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|}{\sigma}\right)$	$\sigma>0$
Sigmoid 核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\tanh(\beta\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j+\theta)$	tanh 为双曲正切函数, $\beta>0$, $\theta<0$

$d=1$ 时退化为线性核.

高斯核亦称 RBF 核.

此外, 还可通过函数组合得到, 例如:

- 若 κ_1 和 κ_2 为核函数, 则对于任意正数 γ_1、γ_2, 其线性组合

$$\gamma_1\kappa_1+\gamma_2\kappa_2 \tag{6.25}$$

也是核函数;

- 若 κ_1 和 κ_2 为核函数, 则核函数的直积

$$\kappa_1 \otimes \kappa_2(\boldsymbol{x}, \boldsymbol{z}) = \kappa_1(\boldsymbol{x}, \boldsymbol{z})\kappa_2(\boldsymbol{x}, \boldsymbol{z}) \tag{6.26}$$

也是核函数;

- 若 κ_1 为核函数, 则对于任意函数 $g(\boldsymbol{x})$,

$$\kappa(\boldsymbol{x}, \boldsymbol{z}) = g(\boldsymbol{x})\kappa_1(\boldsymbol{x}, \boldsymbol{z})g(\boldsymbol{z}) \tag{6.27}$$

也是核函数.

6.4 软间隔与正则化

在前面的讨论中, 我们一直假定训练样本在样本空间或特征空间中是线性可分的, 即存在一个超平面能将不同类的样本完全划分开. 然而, 在现实任务中往往很难确定合适的核函数使得训练样本在特征空间中线性可分; 退一步说, 即便恰好找到了某个核函数使训练集在特征空间中线性可分, 也很难断定这个貌似线性可分的结果不是由于过拟合所造成的.

缓解该问题的一个办法是允许支持向量机在一些样本上出错. 为此, 要引入“软间隔”(soft margin)的概念, 如图 6.4 所示.

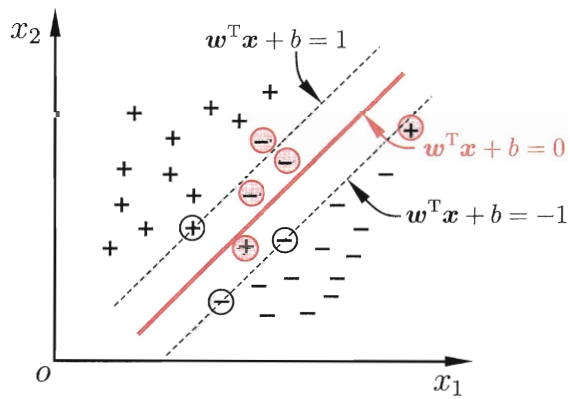

图 6.4 软间隔示意图. 红色圈出了一些不满足约束的样本.

具体来说, 前面介绍的支持向量机形式是要求所有样本均满足约束(6.3), 即所有样本都必须划分正确, 这称为“硬间隔”(hard margin), 而软间隔则是

允许某些样本不满足约束

$$y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1\ . \tag{6.28}$$

当然, 在最大化间隔的同时, 不满足约束的样本应尽可能少. 于是, 优化目标可写为

$$\min_{\boldsymbol{w},b}\ \frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^{m}\ell_{0/1}\left(y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b\right)-1\right)\ , \tag{6.29}$$

其中 $C>0$ 是一个常数, $\ell_{0/1}$ 是“0/1损失函数”

$$\ell_{0/1}(z)=\begin{cases}1, & \text{if } z<0;\\ 0, & \text{otherwise.}\end{cases} \tag{6.30}$$

显然, 当 C 为无穷大时, 式(6.29)迫使所有样本均满足约束(6.28), 于是式(6.29)等价于(6.6); 当 C 取有限值时, 式(6.29)允许一些样本不满足约束.

然而, $\ell_{0/1}$ 非凸、非连续, 数学性质不太好, 使得式(6.29)不易直接求解. 于是, 人们通常用其他一些函数来代替 $\ell_{0/1}$, 称为“替代损失”(surrogate loss). 替代损失函数一般具有较好的数学性质, 如它们通常是凸的连续函数且是 $\ell_{0/1}$ 的上界. 图 6.5 给出了三种常用的替代损失函数:

对率损失是对率函数的变形, 对率函数参见 3.3 节.

对率损失函数通常表示为 $\ell_{log}(\cdot)$, 因此式(6.33)把式(3.15)中的 $\ln(\cdot)$ 改写为 $\log(\cdot)$.

$$\text{hinge 损失: } \ell_{hinge}(z)=\max(0,1-z)\ ; \tag{6.31}$$

$$\text{指数损失(exponential loss): } \ell_{exp}(z)=\exp(-z)\ ; \tag{6.32}$$

$$\text{对率损失(logistic loss): } \ell_{log}(z)=\log(1+\exp(-z))\ . \tag{6.33}$$

若采用 hinge 损失, 则式(6.29)变成

$$\min_{\boldsymbol{w},b}\ \frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^{m}\max\left(0,1-y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b\right)\right)\ . \tag{6.34}$$

引入“松弛变量”(slack variables) $\xi_i\geqslant 0$, 可将式(6.34)重写为

$$\min_{\boldsymbol{w},b,\xi_i}\ \frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^{m}\xi_i \tag{6.35}$$

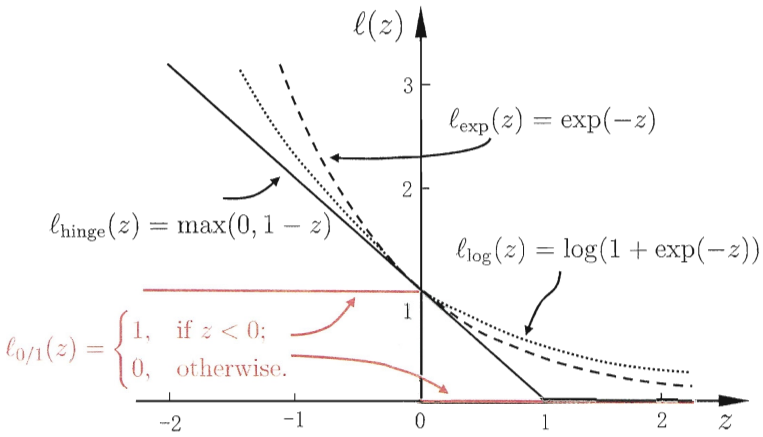

图 6.5 三种常见的替代损失函数: hinge损失、指数损失、对率损失

$$\begin{aligned} \text{s.t.} \quad & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i \\ & \xi_i \geqslant 0 \ , \ i = 1, 2, \ldots, m. \end{aligned}$$

这就是常用的“软间隔支持向量机”.

显然, 式(6.35)中每个样本都有一个对应的松弛变量, 用以表征该样本不满足约束(6.28)的程度. 但是, 与式(6.6)相似, 这仍是一个二次规划问题. 于是, 类似式(6.8), 通过拉格朗日乘子法可得到式(6.35)的拉格朗日函数

$$\begin{aligned} L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\mu}) = & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ & + \sum_{i=1}^{m}\alpha_i\left(1 - \xi_i - y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b\right)\right) - \sum_{i=1}^{m}\mu_i\xi_i \ , \end{aligned} \tag{6.36}$$

其中 $\alpha_i \geqslant 0$, $\mu_i \geqslant 0$ 是拉格朗日乘子.

令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\mu})$ 对 $\boldsymbol{w}$, b, ξ_i 的偏导为零可得

$$\boldsymbol{w} = \sum_{i=1}^{m}\alpha_i y_i \boldsymbol{x}_i \ , \tag{6.37}$$

$$0 = \sum_{i=1}^{m}\alpha_i y_i \ , \tag{6.38}$$

$$C = \alpha_i + \mu_i \ . \tag{6.39}$$

将式(6.37)–(6.39)代入式(6.36)即可得到式(6.35)的对偶问题

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \,, \\ & 0 \leqslant \alpha_i \leqslant C \,, \quad i = 1, 2, \ldots, m \,. \end{aligned} \tag{6.40}$$

将式(6.40)与硬间隔下的对偶问题(6.11)对比可看出, 两者唯一的差别就在于对偶变量的约束不同: 前者是 $0 \leqslant \alpha_i \leqslant C$, 后者是 $0 \leqslant \alpha_i$. 于是, 可采用 6.2 节中同样的算法求解式(6.40); 在引入核函数后能得到与式(6.24)同样的支持向量展式.

类似式(6.13), 对软间隔支持向量机, KKT 条件要求

$$\begin{cases} \alpha_i \geqslant 0 \,, \quad \mu_i \geqslant 0 \,, \\ y_i f(\boldsymbol{x}_i) - 1 + \xi_i \geqslant 0 \,, \\ \alpha_i \left(y_i f\left(\boldsymbol{x}_i\right) - 1 + \xi_i \right) = 0 \,, \\ \xi_i \geqslant 0 \,, \ \mu_i \xi_i = 0 \,. \end{cases} \tag{6.41}$$

于是, 对任意训练样本 $(\boldsymbol{x}_i, y_i)$, 总有 $\alpha_i = 0$ 或 $y_i f(\boldsymbol{x}_i) = 1 - \xi_i$. 若 $\alpha_i = 0$, 则该样本不会对 $f(\boldsymbol{x})$ 有任何影响; 若 $\alpha_i > 0$, 则必有 $y_i f(\boldsymbol{x}_i) = 1 - \xi_i$, 即该样本是支持向量: 由式(6.39) 可知, 若 $\alpha_i < C$, 则 $\mu_i > 0$, 进而有 $\xi_i = 0$, 即该样本恰在最大间隔边界上; 若 $\alpha_i = C$, 则有 $\mu_i = 0$, 此时若 $\xi_i \leqslant 1$ 则该样本落在最大间隔内部, 若 $\xi_i > 1$ 则该样本被错误分类. 由此可看出, 软间隔支持向量机的最终模型仅与支持向量有关, 即通过采用 hinge 损失函数仍保持了稀疏性.

那么, 能否对式(6.29)使用其他的替代损失函数呢?

可以发现, 如果使用对率损失函数 ℓ_{log} 来替代式(6.29)中的 0/1 损失函数, 则几乎就得到了对率回归模型(3.27). 实际上, 支持向量机与对率回归的优化目标相近, 通常情形下它们的性能也相当. 对率回归的优势主要在于其输出具有自然的概率意义, 即在给出预测标记的同时也给出了概率, 而支持向量机的输出不具有概率意义, 欲得到概率输出需进行特殊处理 [Platt, 2000]; 此外, 对率回归能直接用于多分类任务, 支持向量机为此则需进行推广[Hsu and Lin, 2002]. 另一方面, 从图 6.5 可看出, hinge 损失有一块“平坦”的零区域, 这使

得支持向量机的解具有稀疏性, 而对率损失是光滑的单调递减函数, 不能导出类似支持向量的概念, 因此对率回归的解依赖于更多的训练样本, 其预测开销更大.

我们还可以把式(6.29)中的 0/1 损失函数换成别的替代损失函数以得到其他学习模型, 这些模型的性质与所用的替代函数直接相关, 但它们具有一个共性: 优化目标中的第一项用来描述划分超平面的“间隔”大小, 另一项 $\sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i)$ 用来表述训练集上的误差, 可写为更一般的形式

$$\min_{f} \ \Omega(f) + C \sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i), \tag{6.42}$$

其中 $\Omega(f)$ 称为“结构风险”(structural risk), 用于描述模型 f 的某些性质; 第二项 $\sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i)$ 称为“经验风险”(empirical risk), 用于描述模型与训练数据的契合程度; C 用于对二者进行折中. 从经验风险最小化的角度来看, $\Omega(f)$ 表述了我们希望获得具有何种性质的模型(例如希望获得复杂度较小的模型), 这为引入领域知识和用户意图提供了途径; 另一方面, 该信息有助于削减假设空间, 从而降低了最小化训练误差的过拟合风险. 从这个角度来说, 式(6.42)称为“正则化”(regularization)问题, $\Omega(f)$ 称为正则化项, C 则称为正则化常数. L_p 范数 (norm) 是常用的正则化项, 其中 L_2 范数 $\|\boldsymbol{w}\|_2$ 倾向于 $\boldsymbol{w}$ 的分量取值尽量均衡, 即非零分量个数尽量稠密, 而 L_0 范数 $\|\boldsymbol{w}\|_0$ 和 L_1 范数 $\|\boldsymbol{w}\|_1$ 则倾向于 $\boldsymbol{w}$ 的分量尽量稀疏, 即非零分量个数尽量少.

正则化可理解为一种“罚函数法”, 即对不希望得到的结果施以惩罚, 从而使得优化过程趋向于希望目标. 从贝叶斯估计的角度来看, 正则化项可认为是提供了模型的先验概率.

参见 11.4 节.

6.5 支持向量回归

现在我们来考虑回归问题. 给定训练样本 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, $y_i \in \mathbb{R}$, 希望学得一个形如式(6.7)的回归模型, 使得 $f(\boldsymbol{x})$ 与 y 尽可能接近, $\boldsymbol{w}$ 和 b 是待确定的模型参数.

对样本 $(\boldsymbol{x}, y)$, 传统回归模型通常直接基于模型输出 $f(\boldsymbol{x})$ 与真实输出 y 之间的差别来计算损失, 当且仅当 $f(\boldsymbol{x})$ 与 y 完全相同时, 损失才为零. 与此不同, 支持向量回归(Support Vector Regression, 简称 SVR)假设我们能容忍 $f(\boldsymbol{x})$ 与 y 之间最多有 ϵ 的偏差, 即仅当 $f(\boldsymbol{x})$ 与 y 之间的差别绝对值大于 ϵ 时才计算损失. 如图 6.6 所示, 这相当于以 $f(\boldsymbol{x})$ 为中心, 构建了一个宽度为 2ϵ 的间隔带, 若训练样本落入此间隔带, 则认为是被预测正确的.

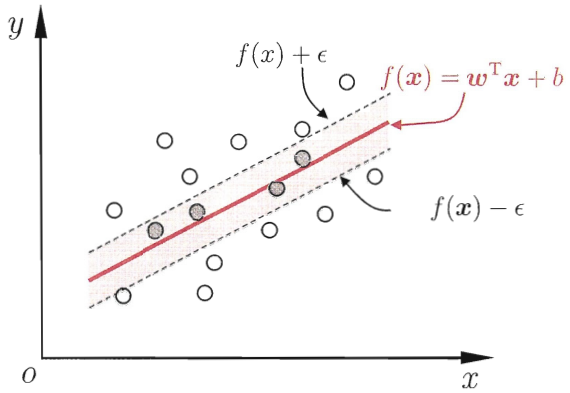

图 6.6 支持向量回归示意图. 红色显示出 ϵ-间隔带, 落入其中的样本不计算损失.

于是, SVR 问题可形式化为

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\ell_\epsilon(f(\boldsymbol{x}_i)-y_i)\ , \tag{6.43}$$

其中 C 为正则化常数, ℓ_ϵ 是图 6.7 所示的 ϵ-不敏感损失 (ϵ-insensitive loss) 函数

$$\ell_\epsilon(z)=\begin{cases}0, & \text{if } |z|\leqslant\epsilon\ ;\\ |z|-\epsilon, & \text{otherwise.}\end{cases} \tag{6.44}$$

间隔带两侧的松弛程度可有所不同.

引入松弛变量 ξ_i 和 $\hat{\xi}_i$, 可将式(6.43)重写为

$$\min_{\boldsymbol{w},b,\xi_i,\hat{\xi}_i} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}(\xi_i+\hat{\xi}_i) \tag{6.45}$$

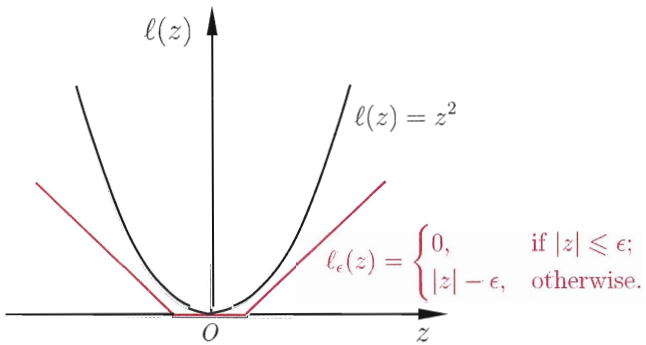

图 6.7 ϵ-不敏感损失函数

$$
\begin{aligned}
\text{s.t.}\ \ & f(\boldsymbol{x}_i) - y_i \leqslant \epsilon + \xi_i \ , \\
& y_i - f(\boldsymbol{x}_i) \leqslant \epsilon + \hat{\xi}_i \ , \\
& \xi_i \geqslant 0,\ \hat{\xi}_i \geqslant 0 \ ,\ i = 1, 2, \ldots, m.
\end{aligned}
$$

类似式(6.36), 通过引入拉格朗日乘子 $\mu_i \geqslant 0$, $\hat{\mu}_i \geqslant 0$, $\alpha_i \geqslant 0$, $\hat{\alpha}_i \geqslant 0$, 由拉格朗日乘子法可得到式(6.45)的拉格朗日函数

$$
\begin{aligned}
& L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}, \boldsymbol{\xi}, \hat{\boldsymbol{\xi}}, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}}) \\
& = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \hat{\xi}_i) - \sum_{i=1}^{m}\mu_i\xi_i - \sum_{i=1}^{m}\hat{\mu}_i\hat{\xi}_i \\
& + \sum_{i=1}^{m}\alpha_i\big(f\left(\boldsymbol{x}_i\right) - y_i - \epsilon - \xi_i\big) + \sum_{i=1}^{m}\hat{\alpha}_i\big(y_i - f\left(\boldsymbol{x}_i\right) - \epsilon - \hat{\xi}_i\big) \ . \qquad (6.46)
\end{aligned}
$$

将式(6.7)代入, 再令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}, \boldsymbol{\xi}, \hat{\boldsymbol{\xi}}, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}})$ 对 $\boldsymbol{w}$, b, ξ_i 和 $\hat{\xi}_i$ 的偏导为零可得

$$
\boldsymbol{w} = \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i)\boldsymbol{x}_i \ , \qquad (6.47)
$$

$$
0 = \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i) \ , \qquad (6.48)
$$

$$
C = \alpha_i + \mu_i \ , \qquad (6.49)
$$

$$
C = \hat{\alpha}_i + \hat{\mu}_i \ . \qquad (6.50)
$$

将式(6.47)–(6.50)代入式(6.46), 即可得到 SVR 的对偶问题

$$
\begin{aligned}
\max_{\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}} \ & \sum_{i=1}^{m} y_i(\hat{\alpha}_i - \alpha_i) - \epsilon(\hat{\alpha}_i + \alpha_i) \qquad (6.51) \\
& - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \\
\text{s.t.}\ \ & \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i) = 0 \ , \\
& 0 \leqslant \alpha_i, \hat{\alpha}_i \leqslant C \ .
\end{aligned}
$$

上述过程中需满足 KKT 条件, 即要求

$$\begin{cases} \alpha_i(f(\boldsymbol{x}_i)-y_i-\epsilon-\xi_i)=0\ , \\ \hat{\alpha}_i(y_i-f(\boldsymbol{x}_i)-\epsilon-\hat{\xi}_i)=0\ , \\ \alpha_i\hat{\alpha}_i=0\ ,\ \xi_i\hat{\xi}_i=0\ , \\ (C-\alpha_i)\xi_i=0\ ,\ (C-\hat{\alpha}_i)\hat{\xi}_i=0\ . \end{cases} \tag{6.52}$$

可以看出, 当且仅当 $f(\boldsymbol{x}_i)-y_i-\epsilon-\xi_i=0$ 时 α_i 能取非零值, 当且仅当 $y_i-f(\boldsymbol{x}_i)-\epsilon-\hat{\xi}_i=0$ 时 $\hat{\alpha}_i$ 能取非零值. 换言之, 仅当样本 $(\boldsymbol{x}_i,y_i)$ 不落入 ϵ-间隔带中, 相应的 α_i 和 $\hat{\alpha}_i$ 才能取非零值. 此外, 约束 $f(\boldsymbol{x}_i)-y_i-\epsilon-\xi_i=0$ 和 $y_i-f(\boldsymbol{x}_i)-\epsilon-\hat{\xi}_i=0$ 不能同时成立, 因此 α_i 和 $\hat{\alpha}_i$ 中至少有一个为零.

将式(6.47)代入(6.7), 则 SVR 的解形如

$$f(\boldsymbol{x})=\sum_{i=1}^{m}(\hat{\alpha}_i-\alpha_i)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}+b\ . \tag{6.53}$$

落在 ϵ-间隔带中的样本都满足 $\alpha_i=0$ 且 $\hat{\alpha}_i=0$.

能使式(6.53)中的 $(\hat{\alpha}_i-\alpha_i)\neq 0$ 的样本即为 SVR 的支持向量, 它们必落在 ϵ-间隔带之外. 显然, SVR 的支持向量仅是训练样本的一部分, 即其解仍具有稀疏性.

由 KKT 条件(6.52)可看出, 对每个样本 $(\boldsymbol{x}_i,y_i)$ 都有 $(C-\alpha_i)\xi_i=0$ 且 $\alpha_i(f(\boldsymbol{x}_i)-y_i-\epsilon-\xi_i)=0$. 于是, 在得到 α_i 后, 若 $0<\alpha_i<C$, 则必有 $\xi_i=0$, 进而有

$$b=y_i+\epsilon-\sum_{i=1}^{m}(\hat{\alpha}_i-\alpha_i)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}\ . \tag{6.54}$$

因此, 在求解式(6.51)得到 α_i 后, 理论上来说, 可任意选取满足 $0<\alpha_i<C$ 的样本通过式(6.54)求得 b. 实践中常采用一种更鲁棒的办法: 选取多个(或所有)满足条件 $0<\alpha_i<C$ 的样本求解 b 后取平均值.

若考虑特征映射形式(6.19), 则相应的, 式(6.47)将形如

$$\boldsymbol{w}=\sum_{i=1}^{m}(\hat{\alpha}_i-\alpha_i)\phi(\boldsymbol{x}_i)\ . \tag{6.55}$$

将式(6.55)代入(6.19), 则 SVR 可表示为

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} (\hat{\alpha}_i - \alpha_i)\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b \ , \tag{6.56}$$

其中 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$ 为核函数.

6.6 核方法

回顾式(6.24)和(6.56)可发现, 给定训练样本 $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, 若不考虑偏移项 b, 则无论 SVM 还是 SVR, 学得的模型总能表示成核函数 $\kappa(\boldsymbol{x}, \boldsymbol{x}_i)$ 的线性组合. 不仅如此, 事实上我们有下面这个称为“表示定理”(representer theorem)的更一般的结论:

证明参阅 [Schölkopf and Smola, 2002], 其中用到了关于实对称矩阵正定性充要条件的 Mercer 定理.

定理 6.2 (表示定理) 令 $\mathbb{H}$ 为核函数 κ 对应的再生核希尔伯特空间, $\|h\|_{\mathbb{H}}$ 表示 $\mathbb{H}$ 空间中关于 h 的范数, 对于任意单调递增函数 $\Omega : [0, \infty] \mapsto \mathbb{R}$ 和任意非负损失函数 $\ell : \mathbb{R}^m \mapsto [0, \infty]$, 优化问题

$$\min_{h \in \mathbb{H}} \ F(h) \ = \ \Omega(\|h\|_{\mathbb{H}}) + \ell\big(h(\boldsymbol{x}_1), h(\boldsymbol{x}_2), \ldots, h(\boldsymbol{x}_m)\big) \tag{6.57}$$

的解总可写为

$$h^*(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) \ . \tag{6.58}$$

表示定理对损失函数没有限制, 对正则化项 Ω 仅要求单调递增, 甚至不要求 Ω 是凸函数, 意味着对于一般的损失函数和正则化项, 优化问题(6.57)的最优解 $h^*(\boldsymbol{x})$ 都可表示为核函数 $\kappa(\boldsymbol{x}, \boldsymbol{x}_i)$ 的线性组合; 这显示出核函数的巨大威力.

人们发展出一系列基于核函数的学习方法, 统称为“核方法”(kernel methods). 最常见的, 是通过“核化”(即引入核函数)来将线性学习器拓展为非线性学习器. 下面我们以线性判别分析为例来演示如何通过核化来对其进行非线性拓展, 从而得到“核线性判别分析”(Kernelized Linear Discriminant Analysis, 简称 KLDA).

线性判别分析见 3.4 节.

我们先假设可通过某种映射 $\phi : \mathcal{X} \mapsto \mathbb{F}$ 将样本映射到一个特征空间 $\mathbb{F}$, 然后在 $\mathbb{F}$ 中执行线性判别分析, 以求得

$$h(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) \ . \tag{6.59}$$

类似于式(3.35), KLDA 的学习目标是

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = \frac{\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_b^{\phi}\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w^{\phi}\boldsymbol{w}} , \tag{6.60}$$

其中 $\mathbf{S}_b^{\phi}$ 和 $\mathbf{S}_w^{\phi}$ 分别为训练样本在特征空间 $\mathbb{F}$ 中的类间散度矩阵和类内散度矩阵. 令 X_i 表示第 $i \in \{0,1\}$ 类样本的集合, 其样本数为 m_i; 总样本数 $m = m_0 + m_1$. 第 i 类样本在特征空间 $\mathbb{F}$ 中的均值为

$$\boldsymbol{\mu}_i^{\phi} = \frac{1}{m_i}\sum_{\boldsymbol{x}\in X_i}\phi(\boldsymbol{x}) , \tag{6.61}$$

两个散度矩阵分别为

$$\mathbf{S}_b^{\phi} = (\boldsymbol{\mu}_1^{\phi} - \boldsymbol{\mu}_0^{\phi})(\boldsymbol{\mu}_1^{\phi} - \boldsymbol{\mu}_0^{\phi})^{\mathrm{T}} ; \tag{6.62}$$

$$\mathbf{S}_w^{\phi} = \sum_{i=0}^{1}\sum_{\boldsymbol{x}\in X_i}\big(\phi(\boldsymbol{x}) - \boldsymbol{\mu}_i^{\phi}\big)\big(\phi(\boldsymbol{x}) - \boldsymbol{\mu}_i^{\phi}\big)^{\mathrm{T}} . \tag{6.63}$$

通常我们难以知道映射 ϕ 的具体形式, 因此使用核函数 $\kappa(\boldsymbol{x},\boldsymbol{x}_i) = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x})$ 来隐式地表达这个映射和特征空间 $\mathbb{F}$. 把 $J(\boldsymbol{w})$ 作为式(6.57)中的损失函数 ℓ, 再令 $\Omega \equiv 0$, 由表示定理, 函数 $h(\boldsymbol{x})$ 可写为

$$h(\boldsymbol{x}) = \sum_{i=1}^{m}\alpha_i\kappa(\boldsymbol{x},\boldsymbol{x}_i) , \tag{6.64}$$

于是由式(6.59)可得

$$\boldsymbol{w} = \sum_{i=1}^{m}\alpha_i\phi(\boldsymbol{x}_i) . \tag{6.65}$$

令 $\mathbf{K} \in \mathbb{R}^{m\times m}$ 为核函数 κ 所对应的核矩阵, $(\mathbf{K})_{ij} = \kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$. 令 $\mathbf{1}_i \in \{1,0\}^{m\times 1}$ 为第 i 类样本的指示向量, 即 $\mathbf{1}_i$ 的第 j 个分量为 1 当且仅当 $\boldsymbol{x}_j \in X_i$, 否则 $\mathbf{1}_i$ 的第 j 个分量为 0. 再令

$$\hat{\boldsymbol{\mu}}_0 = \frac{1}{m_0}\mathbf{K}\mathbf{1}_0 , \tag{6.66}$$

$$\hat{\boldsymbol{\mu}}_1 = \frac{1}{m_1}\mathbf{K}\mathbf{1}_1 , \tag{6.67}$$

$$\mathbf{M} = \big(\hat{\boldsymbol{\mu}}_0 - \hat{\boldsymbol{\mu}}_1\big)\big(\hat{\boldsymbol{\mu}}_0 - \hat{\boldsymbol{\mu}}_1\big)^{\mathrm{T}} , \tag{6.68}$$

$$\mathbf{N} = \mathbf{K}\mathbf{K}^{\mathrm{T}} - \sum_{i=0}^{1} m_i \hat{\boldsymbol{\mu}}_i \hat{\boldsymbol{\mu}}_i^{\mathrm{T}} . \tag{6.69}$$

于是, 式(6.60)等价为

$$\max_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{M}\boldsymbol{\alpha}}{\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{N}\boldsymbol{\alpha}} . \tag{6.70}$$

求解方法参见 3.4 节.

显然, 使用线性判别分析求解方法即可得到 $\boldsymbol{\alpha}$, 进而可由式(6.64)得到投影函数 $h(\boldsymbol{x})$.

6.7 阅读材料

线性核 SVM 迄今仍是文本分类的首选技术. 一个重要原因可能是: 若将每个单词作为文本数据的一个属性, 则该属性空间维数很高, 冗余度很大, 其描述能力足以将不同文档"打散". 关于打散, 参见 12.4 节.

支持向量机于 1995 年正式发表 [Cortes and Vapnik, 1995], 由于在文本分类任务中显示出卓越性能 [Joachims, 1998], 很快成为机器学习的主流技术, 并直接掀起了"统计学习"(statistical learning)在 2000 年前后的高潮. 但实际上, 支持向量的概念早在二十世纪六十年代就已出现, 统计学习理论在七十年代就已成型. 对核函数的研究更早, Mercer 定理 [Cristianini and Shawe-Taylor, 2000] 可追溯到 1909 年, RKHS 则在四十年代就已被研究, 但在统计学习兴起后, 核技巧才真正成为机器学习的通用基本技术. 关于支持向量机和核方法有很多专门书籍和介绍性文章 [Cristianini and Shawe-Taylor, 2000; Burges, 1998; 邓乃扬 与 田英杰, 2009; Schölkopf et al., 1999; Schölkopf and Smola, 2002], 统计学习理论则可参阅 [Vapnik, 1995, 1998, 1999].

m 是样本个数.

支持向量机的求解通常是借助于凸优化技术 [Boyd and Vandenberghe, 2004]. 如何提高效率, 使 SVM 能适用于大规模数据一直是研究重点. 对线性核 SVM 已有很多成果, 例如基于割平面法(cutting plane algorithm)的 SVM$^{\mathrm{perf}}$ 具有线性复杂度 [Joachims, 2006], 基于随机梯度下降的 Pegasos 速度甚至更快 [Shalev-Shwartz et al., 2011], 而坐标下降法则在稀疏数据上有很高的效率 [Hsieh et al., 2008]. 非线性核 SVM 的时间复杂度在理论上不可能低于 $O(m^2)$, 因此研究重点是设计快速近似算法, 如基于采样的 CVM [Tsang et al., 2006]、基于低秩逼近的 Nyström 方法 [Williams and Seeger, 2001]、基于随机傅里叶特征的方法 [Rahimi and Recht, 2007]等. 最近有研究显示, 当核矩阵特征值有很大差别时, Nyström 方法往往优于随机傅里叶特征方法 [Yang et al., 2012].

支持向量机是针对二分类任务设计的, 对多分类任务要进行专门的推广 [Hsu and Lin, 2002], 对带结构输出的任务也已有相应的算法 [Tsochantaridis

et al., 2005]. 支持向量回归的研究始于 [Drucker et al., 1997], [Smola and Schölkopf, 2004] 给出了一个较为全面的介绍.

核函数直接决定了支持向量机与核方法的最终性能, 但遗憾的是, 核函数的选择是一个未决问题. 多核学习(multiple kernel learning) 使用多个核函数并通过学习获得其最优凸组合作为最终的核函数 [Lanckriet et al., 2004; Bach et al., 2004], 这实际上是在借助集成学习机制.

集成学习参见第 8 章.

替代损失函数在机器学习中被广泛使用. 但是, 通过求解替代损失函数得到的是否仍是原问题的解? 这在理论上称为替代损失的“一致性”(consistency)问题. [Vapnik and Chervonenkis, 1991] 给出了基于替代损失进行经验风险最小化的一致性充要条件, [Zhang, 2004] 证明了几种常见凸替代损失函数的一致性.

一致性亦称“相合性”.

SVM 已有很多软件包, 比较著名的有 LIBSVM [Chang and Lin, 2011] 和 LIBLINEAR [Fan et al., 2008] 等.