人工智能的数学基础

华东师范大学 数学科学学院 黎芳(教授) 2019年10月21日

Chapter 7 支持向量机(support vector machines, SVM)

Table of Contents

人工智能的数学基础 1
Chapter 7 支持向量机(support vector machines, SVM) 1
7.1 线性可分支持向量机与硬间隔最大化 1
7.1.1 线性可分支持向量机 1
7.1.2 函数间隔和几何间隔 2
7.1.3 间隔最大化 2
7.1.4 学习的对偶算法 5
7.2 线性支持向量机和软间隔最大化 7
7.2.1 线性支持向量机 7
7.2.2 学习的对偶算法 8
7.2.3 支持向量 9
7.2.4 合页损失函数 10
7.3 非线性支持向量机与核函数 11
7.3.1 核技巧 11
1.非线性分类问题 11
2.核函数的定义 12
3. 核技巧在支持向量机中的应用 12
7.3.2 正定核 12
7.3.3 常用核函数 15
7.3.4 非线性支持向量分类机 15
7.4 序列最小最优化算法(sequential minimal optimization, SMO) 16
7.4.1 两个变量二次规划的求解方法 16
7.4.2 变量的选择方法 18
7.4.3 SMO算法 19
Train SVM Classifier 20

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

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

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

以及相应的分类决策函数

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

称为线性可分支持向量机.

7.1.2 函数间隔和几何间隔

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: $\widehat{\gamma}_i = y_i(w\cdot x_i + b)$

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，

$$\widehat{\gamma} = \min_{i\to 1,\cdots,N} \widehat{\gamma}_i$$

$$w \to 2w, b \to 2b \Rightarrow \widehat{\gamma}_i \to 2\widehat{\gamma}_i$$

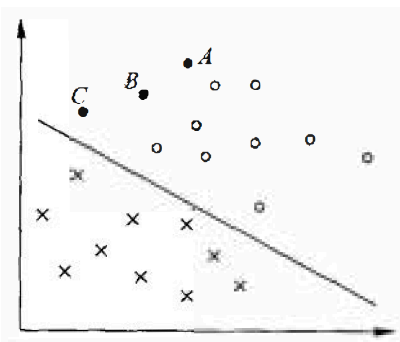


图 7.1　二类分类问题

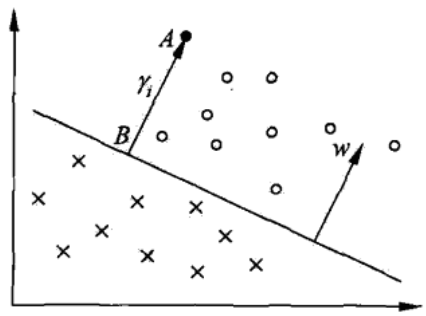


图 7.2　几何间隔

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

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

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，

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

函数间隔与几何间隔的关系: $\gamma_i = \dfrac{\widehat{\gamma}_i}{\|w\|},\ \gamma = \dfrac{\widehat{\gamma}}{\|w\|}$

7.1.3 间隔最大化

1. <u>最大间隔分离超平面</u>

$$\begin{aligned} &\max_{w,b} \quad \gamma \\ &\text{s.t.} \quad y_i\left(\frac{w}{\|w\|}\cdot x_i + \frac{b}{\|w\|}\right) \geq \gamma, \quad i = 1,2,\cdots,N \end{aligned}$$

$$\Longleftrightarrow \begin{array}{ll} \max\limits_{w,b} & \dfrac{\hat{\gamma}}{\|w\|} \\ \text{s.t.} & y_i(w \cdot x_i + b) \geq \hat{\gamma}, \quad i = 1, 2, \cdots, N \end{array}$$

函数间隔$\hat{\gamma}$的取值并不影响最优化问题的解, 不妨取$\hat{\gamma} = 1$.

$$\Longleftrightarrow \begin{array}{ll} \min\limits_{w,b} & \dfrac{1}{2}\|w\|^2 \\ \text{s.t.} & y_i(w \cdot x_i + b) - 1 \geq 0, \quad i = 1, 2, \cdots, N \end{array}$$

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

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$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，$\mathcal{X} = \mathbf{R}^n, \quad y_i \in \mathcal{Y} = \{-1, +1\}, \quad i = 1, 2, \cdots, N$

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

$$\begin{array}{ll} \min\limits_{x,b} & \dfrac{1}{2}\|w\|^2 \\ \text{s.t.} & y_i(w \cdot x_i + b) - 1 \geq 0, \quad i = 1, 2, \cdots, N \end{array} \qquad (1)$$

求得最优解w^*, b^*.

(2) 由此得到分离超平面和分类决策函数

$$w^* \cdot x + b^* = 0$$
$$f(x) = \text{sign}(w^* \cdot x + b^*)$$

2. 最大间隔分离超平面的存在唯一性

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， 则可将训练数据操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.

存在性：数据集可分，存在可行解；函数有下界，优化问题存在解.

唯一性：反证法

假设存在两个最优解 (w_1^*, b_1^*)和(w_2^*, b_2^*)，先证$w_1^* = w_2^*$. 显然$\|w_1^*\| = \|w_2^*\| = c$，其中c是一个常数.

令$w = \dfrac{w_1^* + w_2^*}{2}, \quad b = \dfrac{b_1^* + b_2^*}{2}$，易知，$(w, b)$也是（1）的可行解，从而有

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

$$\Rightarrow \|w\| = \frac{1}{2}\|w_1^*\| + \frac{1}{2}\|w_2^*\|$$
$$\Rightarrow w_1^* = \lambda w_2^*, |\lambda| = 1$$

若$\lambda = -1$，$(w = 0, b)$不是（1）的可行解，矛盾. 所以$\lambda = 1$，$w_1^* = w_2^* = w^*$.

再证$b_1^* = b_2^*$.

设$x_1{'}$和$x_2{'}$是集合$\{x_i|y_i = +1\}$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$\{x_i|y_i = -1\}$中分别对应(w^*, b_1^*)和(w^*, b_2^*)使得问题的不等式等号成立的点，即

$$w^* \cdot x_1{'} + b_1^* = 1, w^* \cdot x_2{'} + b_2^* = 1$$
$$w^* \cdot x_1{''} + b_1^* = -1, w^* \cdot x_2{''} + b_2^* = -1$$
$$\Rightarrow w^* \cdot \frac{(x_1{'} + x_1{''})}{2} + b_1^* = 0, w^* \cdot \frac{(x_2{'} + x_2{''})}{2} + b_2^* = 0$$
$$\Rightarrow b_1^* - b_2^* = -\frac{1}{2}[w^* \cdot (x_1{'} - x_2{'}) + w^* \cdot (x_1{''} - x_2{''})]$$

又因为

$$w^* \cdot x_2{'} + b_1^* \geq 1 = w^* \cdot x_1{'} + b_1^* \Rightarrow w^* \cdot (x_2{'} - x_1{'}) \geq 0$$
$$w^* \cdot x_1{'} + b_2^* \geq 1 = w^* \cdot x_2{'} + b_2^* \Rightarrow w^* \cdot (x_1{'} - x_2{'}) \geq 0$$

$w^* \cdot \left(x_1{'} - x_2{'}\right) = 0$，同理，$w^* \cdot \left(x_1{''} - x_2{''}\right) = 0$

因此，$b_1^* = b_2^*$.

3. 支持向量和间隔边界

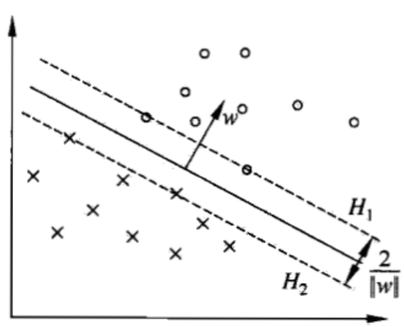


图 7.3　支持向量

对$y_i = 1$的正实例点，支持向量在超平面$H_1 : w \cdot x + b = 1$

对$y_i = -1$的负实例点，支持向量在超平面$H_2 : w \cdot x + b = -1$

例**7.1** 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.

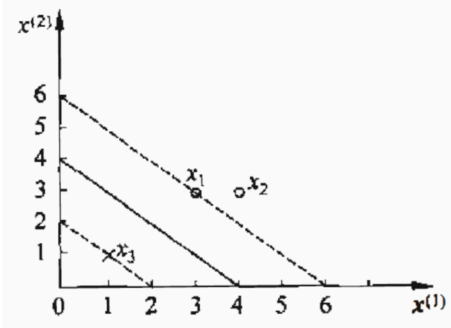


图 7.4　间隔最大分离超平面示例

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

$$w_1 = w_2 = \frac{1}{2}, \quad b=-2,$$

分类超平面：$\frac{1}{2}x^{(1)}+\frac{1}{2}x^{(2)}-2=0$

支持向量：x_1, x_3.

7.1.4 学习的对偶算法

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i(w \cdot x_i + b) - 1 \geq 0, \quad i=1,2,\cdots,N \end{aligned} \tag{P}$$

拉格朗日函数

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

（1）求 $\min\limits_{w,b} L(w,b,\alpha)$

$$\left.\begin{aligned} \nabla_w L(w,b,\alpha) &= w-\sum_{i=1}^N \alpha_i y_i x_i = 0 \\ \nabla_b L(w,b,\alpha) &= -\sum_{i=1}^N \alpha_i y_i = 0 \end{aligned}\right. \Longrightarrow w=\sum_{i=1}^N \alpha_i y_i x_i, \quad \sum_{i=1}^N \alpha_i y_i = 0$$

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

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

（2）对α求极大

$$\begin{array}{ll} \max\limits_{\alpha} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i\cdot x_j) + \sum_{i=1}^{N}\alpha_i & \min\limits_{\alpha} \ \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i\cdot x_j) - \sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \ \sum_{i=1}^{N}\alpha_i y_i = 0 \quad \Longleftrightarrow & \text{s.t.} \ \sum_{i=1}^{N}\alpha_i y_i = 0 \qquad \text{(D)} \\ \alpha_i \geq 0, \quad i = 1,2,\cdots,N & \alpha_i \geq 0, \quad i = 1,2,\cdots,N \end{array}$$

定理**7.2** 设$\alpha^* = \left(\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*\right)^T$是对偶最优化问题(D)的解，则存在下标$j$，使得$\alpha_j^* \geq 0$，并可按下式求得原始最优化问题(P)的解$w^*, b^*$：

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

证明：根据定理C.3(见附录), KKT条件成立

$$\begin{aligned} &\nabla_w L(w^*, b^*, \alpha^*) = w^* - \sum_{i=1}^{N}\alpha_i^* y_i x_i = 0 \\ &\nabla_\delta L(w^*, b^*, \alpha^*) = -\sum_{i=1}^{N}\alpha_i^* y_i = 0 \\ &\alpha_i^*\left(y_i\left(w^*\cdot x_i + b^*\right) - 1\right) = 0, \quad i = 1,2,\cdots,N \\ &y_i\left(w^* x_i + b^*\right) - 1 \geq 0, \quad i = 1,2,\cdots,N \\ &\alpha_i^* \geq 0, \quad i = 1,2,\cdots,N \end{aligned}$$

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

其中至少一个$\alpha_j^* > 0$，否则$w^* = 0$，矛盾. 对此j，$y_j\left(w^*\cdot x_j + b^*\right) - 1 = 0$.

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

由此定理可知，分离超平面可以写成：$\sum_{i=1}^{N}\alpha_i^* y_i (x\cdot x_i) + b^* = 0$

分类决策函数可以写成：$f(x) = \text{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i (x\cdot x_i) + b^*\right)$

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

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(D)，得α^*.

（2）根据定理7.2求w^*, b^*.

（3）得分离超平面$w^* \cdot x + b^* = 0$， 分类决策函数 $f(x) = \operatorname{sign}(w^* \cdot x + b^*)$

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

由对偶互补条件可得 $y_i\left(w^* \cdot x_i + b^*\right) - 1 = 0$,或$w^* \cdot x_i + b^* = \pm 1$

例**7.2** 试用算法7.2求例7.1的线性可分支持向量机.

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

将$\alpha_3 = \alpha_1 + \alpha_2$ 代入目标函数并记为

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

$$\alpha_1^* = \alpha_3^* = \frac{1}{4}, \alpha_2^* = 0.$$

$$w_1^* = w_2^* = \frac{1}{2},\ b^* = -2.$$

7.2 线性支持向量机和软间隔最大化

7.2.1 线性支持向量机

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

$$
\begin{aligned}
\min_{w,b,\xi} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\
\text{s.t.} \quad & y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \cdots, N \\
& \xi_i \geq 0, \quad i = 1, 2, \cdots, N
\end{aligned}
\qquad \text{(P1)}
$$

得到的分离超平面为 $w^* \cdot x + b^* = 0$，以及相应的分类决策函数 $f(x) = \operatorname{sign}(w^* \cdot x + b^*)$

称为线性支持向最机.

7.2.2 学习的对偶算法

问题(P1)的拉格朗日函数为

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

$$\alpha_i \ge 0, \mu_i \ge 0.$$

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

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

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

得 $w = \sum_{i=1}^{N}\alpha_i y_i x_i, \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \quad C-\alpha_i-\mu_i=0$，带入拉格朗日函数得

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

对偶问题为

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

$$\begin{aligned} \text{s.t. } & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & C-\alpha_i-\mu_i = 0 \\ & \alpha_i \ge 0 \\ & \mu_i \ge 0, \quad i=1,2,\cdots,N \end{aligned} \qquad \text{(D1)}$$

定理**7.3** 设 $\alpha^* = \left(\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*\right)^{\mathrm{T}}$ 是对偶问题（D1）的一个解，若存在 α^* 的一个分量 α_j^* 满足 $0 < \alpha_j^* < C$，则原问题(P1)的解可按下式求得：

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

证明：KKT条件

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

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

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

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

$$\mu_i^* \xi_i^* = 0$$

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

$$\xi_i^* \geq 0$$

$$\alpha_i^* \geq 0$$

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

算法7.3（线性支持向量机学习算法）

输入：线性可分训练集T

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(D1)，得α^*.

（2）根据定理7.3求w^*, b^*.

（3）得分离超平面$w^* \cdot x + b^* = 0$， 分类决策函数 $f(x) = \operatorname{sign}(w^* \cdot x + b^*)$

7.2.3 支持向量

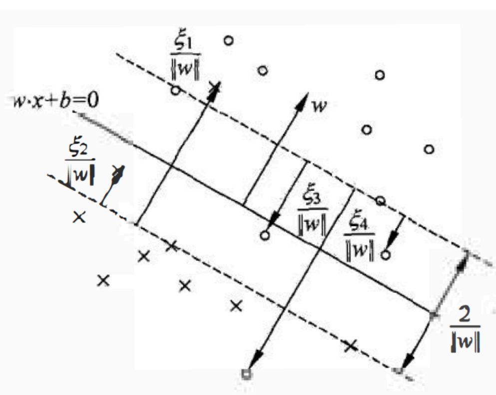


图 7.5 软间隔的支持向量

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）

若$\alpha_i^* < C$，则$\mu_i^* > 0, \xi_i^* = 0$，$y_i\left(w^* \cdot x_i + b^*\right) = 1$，$x_i$恰好在间隔边界上.

若$\alpha_i^* = C$，则 $y_i(w^* \cdot x_i + b^*) = 1 - \xi_i^*$. 若$0 < \xi_i^* < 1$，则分类正确，$x_i$在间隔边界和分离超平面之间；若$\xi_i^* = 1$，$x_i$在超平面上；若$\xi_i^* > 1$，$x_i$位于超平面的误分一侧。

7.2.4 合页损失函数

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：

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

第一项称为合页损失函数（hinge loss function）

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

定理7.4 线性支持向量机原始最优化问题：

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|^2 + C \textstyle\sum_{i=1}^{N} \xi_i \\ \text{s.t.} \quad & y_i(w \cdot x_i + b) \ge 1 - \xi_i, \quad i = 1, 2, \cdots, N \\ & \xi_i \ge 0, \quad i = 1, 2, \cdots, N \end{aligned} \qquad \text{(A)}$$

等价于最优化问题

$$\min_{w,b} \textstyle\sum_{i=1}^{N} [1 - y_i(w \cdot x_i + b)]_+ + \lambda \|w\|^2 \qquad \text{(B)}$$

证明：(A⇒B) $y_i(w \cdot x_i + b) \ge 1 - \xi_i \Rightarrow \xi_i \ge 1 - y_i(w \cdot x_i + b)$，又因为$\xi_i \ge 0$，所以$\min_{x,b,\xi} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{N} \xi_i$在$\xi_i = \max\{1 - y_i(w \cdot x_i + b), 0\} = [1 - y_i(w \cdot x_i + b)]_+$达到极值.

(B⇒A) 令$\xi_i = [1 - y_i(w \cdot x_i + b)]_+$，则

$\xi_i = 1 - y_i(w \cdot x_i + b)$ if $y_i(w \cdot x_i + b) \le 1$，$\xi_i = 0$ if $y_i(w \cdot x_i + b) > 1$. 同时满足$y_i(w \cdot x_i + b) \ge 1 - \xi_i$.

$\lambda = \frac{1}{2C}$.

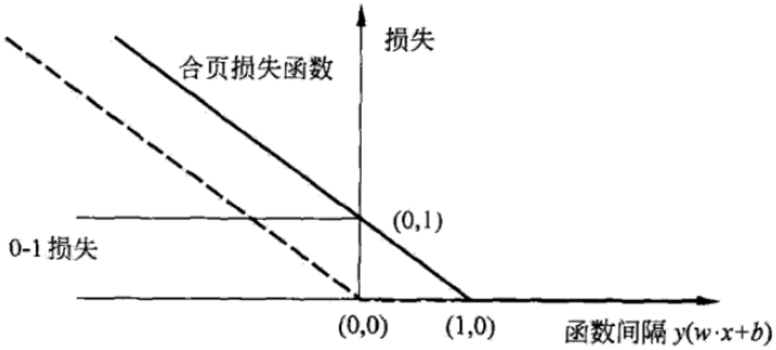


图 7.6　合页损失函数

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.

7.3 非线性支持向量机与核函数

7.3.1 核技巧

1.非线性分类问题

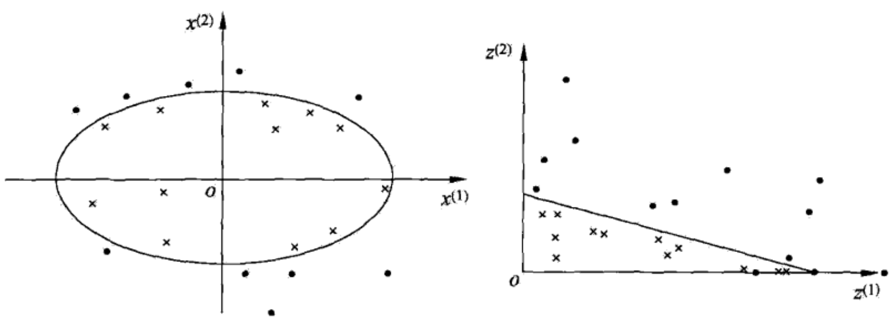


图 7.7　非线性分类问题与核技巧示例

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$\mathcal{Z} \subset \mathbf{R}^2, z = (z^{(1)}, z^{(2)})^{\mathrm{T}} \in \mathcal{Z}$

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

原空间中椭圆：$w_1(x^{(1)})^2 + w_2(x^{(2)})^2 + b = 0$

新空间中直线：$w_1 z^{(1)} + w_2 z^{(2)} + b = 0$

向量空间：空间中的点具有加法和数乘的操作

内积空间：向量空间上定义一个内积操作

赋范空间：根据内积可以定义一个范数

度量空间：范数可以用于定义一个度量

Hilbert Space：如果一个空间在其定义的度量下是完备的，那么这个空间叫做 Hilbert Space。

完备性：一个空间上的任意柯西序列必收敛于空间中的某一点——相当于闭集的定义

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

$$\phi(x):\mathcal{X}\rightarrow\mathcal{H}$$

使得对所有$x\in\mathcal{X}$，函数$K(x,z)$满足条件

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

则称$K(x,z)$为核函数，$\phi(x)$为映射函数.

例**7.3** 假设输入空间是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$\phi(x):R^2\rightarrow\mathcal{H}$.

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

- (1)$\mathcal{H}=\mathbf{R}^3$, $\phi(x)=\left((x^{(1)})^2,\sqrt{2}\,x^{(1)}x^{(2)},(x^{(2)})^2\right)^{\mathrm{T}}$
- (2) $\mathcal{H}=\mathbf{R}^3$, $\phi(x)=\dfrac{1}{\sqrt{2}}\left((x^{(1)})^2-(x^{(2)})^2,2x^{(1)}x^{(2)},(x^{(1)})^2+(x^{(2)})^2\right)^{\mathrm{T}}$
- (3)$\mathcal{H}=\mathbf{R}^4$, $\phi(x)=\left((x^{(1)})^2,x^{(1)}x^{(2)},x^{(1)}x^{(2)},(x^{(2)})^2\right)^{\mathrm{T}}$

3. 核技巧在支持向量机中的应用

$$x_i\cdot x_j\rightarrow\phi(x_i)\cdot\phi(x_j)$$

对偶问题的目标函数：$W(\alpha)=\dfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_jy_iy_jK(x_i,x_j)-\sum_{i=1}^{N}\alpha_i$

分类决策函数：$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}a_i^*y_i\phi(x_i)\cdot\phi(x)+b^*\right)=\operatorname{sign}\left(\sum_{i=1}^{N}a_i^*y_iK(x_i,x)+b^*\right).$

7.3.2 正定核

已知映射函数$\phi(x)$，可以通过$\phi(x)\cdot\phi(z)$构造$K(x,z)$。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？

正定核的充要条件

假设$K(x,z)$是对称函数，$K(x,z)$关于$x_1,x_2,\cdots,x_m$的Gram矩阵是半正定的, 可以根据$K(x,z)$构造一个希尔伯特空间:

首先定义映射$\phi(x)$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**

$\phi : x \to K(\cdot, x)$

定义线性组合：$f(\cdot) = \sum_{i=1}^{m} \alpha_i K(\cdot, x)$

考虑以线性组合为元素的集合S. 由于集合S对加法以及数乘运算是封闭的，所有构成一个向量空间。

2 在**S**上定义内积，使其成为内积空间

对任意$f, g \in S$

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

定义运算:

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

证明运算*是空间S的内积，为此要证：

(1) $(cf) * g = c(f * g), \quad c \in \mathbf{R}$
(2) $(f+g) * h = f * h + g * h, \quad h \in S$
(3) $f * g = g * f$
(4) $f * f \geq 0, f * f = 0 \Leftrightarrow f = 0$

(1)-(3)以及$f * f \geq 0$由f, g的定义和$K(x,z)$的对称性易得。

证明$f * f = 0 \Rightarrow f = 0$

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

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

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

所以$f * f = 0 \Rightarrow |f(x)| = 0, \forall x$

可以记内积*为·

3 将内积空间**S**完备化为希尔伯特空间

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

这一希尔伯特空间称为再生核希尔伯特空间 (reproducing kernel Hilbert space), 这是由于核K具有再生性， 即满足

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

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

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

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

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$\phi$，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$c_1, c_2, \cdots, c_m \in \mathbf{R}$，

$$\begin{aligned}\sum_{i,j=1}^{m} c_i c_j K(x_i, x_j) &= \sum_{i,j=1}^{m} c_i c_j (\phi(x_i) \cdot \phi(x_j)) \\ &= \Big(\sum_i c_i \phi(x_i)\Big) \cdot \Big(\sum_i c_j \phi(x_j)\Big) = \Big|\sum_i c_i \phi(x_i)\Big|^2 \geq 0\end{aligned}$$

充分性 根据前面的结果，可以构造$\phi : x \to K(\cdot, x)$，满足

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

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

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

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

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

是半正定矩阵，则称$K(x,z)$是正定核.

7.3.3 常用核函数

1.多项式核函数**(polynomial kernel function)**

$$K(x,z)=(x\cdot z+1)^p$$

分类决策函数成为

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

2 高斯核函数**(Gaussian kernel function)**

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

高斯径向基函数(radial basis function)分类器

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

3 字符串核函数**(string kernel function)**

举个例子来说明。

	f-o	f-g	o-g	f-b	o-b
ϕ(fog)	λ^2	λ^3	λ^2	0	0
ϕ(fob)	λ^2	0	0	λ^3	λ^2

表示的是在字母表Σ={f,o,g,b}中长度为2的子串（部分子串两个字符串均没有，故不列出）组成的特征空间，由上面给出的映射函数每一维取值公式可知，

$$\begin{aligned}\phi(\text{fog}) &= (\lambda^2,\lambda^3,\lambda^2,0,0)\\ \phi(\text{fob}) &= (\lambda^2,0,0,\lambda^3,\lambda^2)\end{aligned}$$

$$k(\text{ fog},\ fog)=2\lambda^4+\lambda^6$$
$$k(\text{ fob},\ fob)=2\lambda^4+\lambda^6$$
$$k(\text{ fog },\ fob)=\frac{k(\text{ fog, fog })}{\sqrt{k(\text{ fog }),\ \text{fog })k(fob,\ fob)}}=\frac{\lambda^4}{2\lambda^4+\lambda^6}=\frac{1}{2+\lambda^2}$$

7.3.4 非线性支持向量分类机

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

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

称为非线性支持向量，$K(x,z)$是正定核函数.

算法7.4（非线性支持向量机学习算法）

输入：训练数据集

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

$$\begin{aligned}\min_{a}\quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i\\ \text{s.t.}\quad & \sum_{i=1}^{N}\alpha_i y_i=0\\ & 0\le\alpha_i\le C,\quad i=1,2,\cdots,N\end{aligned}$$

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

(2) 选择$\alpha_j^*\in(0,C)$，$b^*=y_j-\sum_{i=1}^{N}\alpha_i^* y_i K(x_i,x_j)$

(3) 构造决策函数 $f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* yK(x\cdot x_i)+b^*\right)$

7.4 序列最小最优化算法(sequential minimal optimization, SMO)

问题：

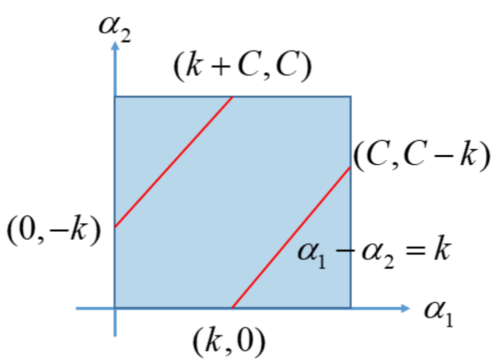
$$\begin{aligned}\min_{\alpha}\quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i\\ \text{s.t.}\quad & \sum_{i=1}^{N}\alpha_i y_i=0\\ & 0\le\alpha_i\le C,\quad i=1,2,\cdots,N\end{aligned}$$

7.4.1 两个变量二次规划的求解方法

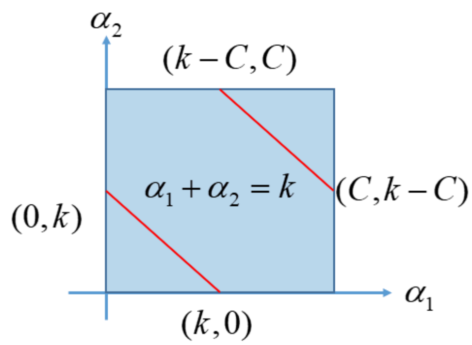
假设选择的两个变量α_1,α_2，

$$
\begin{aligned}
\min_{a,c_2} W(\alpha_1,\alpha_2) = \frac{1}{2}K_{11}\alpha_1^2 + \frac{1}{2}K_{22}\alpha_2^2 + y_1y_2K_{12}\alpha_1\alpha_2 \\
- (\alpha_1+\alpha_2) + y_1\alpha_1\sum_{i=3}^{N} y_i\alpha_iK_{i1} + y_2\alpha_2\sum_{i=3}^{N} y_i\alpha_iK_{i2}
\end{aligned} \tag{D2}
$$

$$
\begin{aligned}
\text{s.t.}\quad & \alpha_1y_1+\alpha_2y_2 = -\sum_{i=3}^{N} y_i\alpha_i = \zeta \\
& 0 \le \alpha_i \le C,\quad i=1,2
\end{aligned}
$$



$y_1 \ne y_2 \Rightarrow \alpha_1-\alpha_2=k$



$y_1 = y_2 \Rightarrow \alpha_1+\alpha_2=k$

假设初始可行解是$\alpha_1^{\text{old}},\alpha_2^{\text{old}}$，若$y_1\ne y_2, k=\alpha_1^{old}-\alpha_2^{old}$. 若$y_1=y_2, k=\alpha_1^{\text{old}}+\alpha_2^{\text{old}}$.

$y_1\ne y_2, L=\max\left(0,\alpha_2^{\text{old}}-\alpha_1^{\text{old}}\right),\quad H=\min\left(C,C+\alpha_2^{\text{old}}-\alpha_1^{\text{old}}\right)$, $L\le\alpha_2^{\text{new}}\le H$

$y_1=y_2, L=\max\left(0,\alpha_2^{\text{old}}+\alpha_1^{\text{old}}-C\right),\quad H=\min\left(C,\alpha_2^{\text{old}}+\alpha_1^{\text{old}}\right)$, $L\le\alpha_2^{\text{new}}\le H$

令
$$g(x)=\sum_{i=1}^{N}\alpha_iy_iK(x_i,x)+b$$

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

定理**7.6** 最优化问题(D2)沿约束方向未经剪辑时的解是

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

其中 $\eta=K_{11}+K_{22}-2K_{12}=\|\Phi(x_1)-\Phi(x_2)\|^2$, $\Phi(x)$是输入空间到特征空间的映射.

$$
a_2^{\text{new}}=\begin{cases}
H, & \alpha_2^{\text{new,unc}}>H \\
\alpha_2^{\text{new,unc}}, & L\le\alpha_2^{\text{new,unc}}\le H \\
L, & a_1^{\text{new,unc}}<L
\end{cases}
$$

$$\alpha_1^{\text{new}} = \alpha_1^{\text{old}} + y_1 y_2 \left(\alpha_2^{\text{old}} - \alpha_2^{\text{acw}}\right)$$

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

$W(\alpha_1, \alpha_2) = \frac{1}{2}K_{11}\alpha_1^2 + \frac{1}{2}K_{22}\alpha_2^2 + y_1 y_2 K_{12}\alpha_1\alpha_2 - (\alpha_1 + \alpha_2) + y_1 v_1 \alpha_1 + y_2 v_2 \alpha_2$

将$\alpha_1 = (\zeta - y_2\alpha_2)y_1$代入

$$\begin{aligned} W(\alpha_2) &= \frac{1}{2}K_{11}(\zeta - \alpha_2 y_2)^2 + \frac{1}{2}K_{22}\alpha_2^2 + y_2 K_{12}(\zeta - \alpha_2 y_2)\alpha_2 \\ &-(\zeta - \alpha_2 y_2)y_1 - \alpha_2 + v_1(\zeta - \alpha_2 y_2) + y_2 v_2 \alpha_2 \end{aligned}$$

$$\frac{\partial W}{\partial \alpha_2} = K_{11}\alpha_2 + K_{22}\alpha_2 - 2K_{12}\alpha_2 - K_{11}\zeta y_2 + K_{12}\zeta y_2 + y_1 y_2 - 1 - v_1 y_2 + y_2 v_2 = 0$$

$$\begin{aligned} (K_{11} + K_{22} - 2K_{12})\alpha_2 &= y_2(y_2 - y_1 + \zeta K_{11} - \zeta K_{12} + v_1 - v_2) \\ &= y_2\left[y_2 - y_1 + \zeta K_{11} - \zeta K_{12} + \left(g(x_1) - \sum_{j=1}^{2} y_j \alpha_j^{\text{old}} K_{1j} - b\right)\right. \\ &\left.-\left(g(x_2) - \sum_{j=1}^{2} y_j \alpha_j^{\text{old}} K_{2j} - b\right)\right] \end{aligned}$$

将$\zeta = \alpha_1^{\text{old}} y_1 + \alpha_2^{\text{old}} y_2$代入，得到p-、

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

将$\eta = K_{11} + K_{22} - 2K_{12}$代入，得

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

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一个变量。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.

$$\alpha_i = 0 \Leftrightarrow y_i g(x_i) \geq 1$$

$$\begin{aligned} 0 < \alpha_i < C &\Leftrightarrow y_i g(x_i) = 1 \\ \alpha_i = C &\Leftrightarrow y_i g(x_i) \leq 1 \end{aligned}$$

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个变量生．第2个变量选择的标准是希望能使α_2有足够大的变化． 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．若以上方法不能使目标有足够的下降，那么采用启发式规则继续选择，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

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

$$b^{\text{new}} = y_1 - \sum_{i=3}^{N} \alpha_i y_i K_n - \alpha_1^{\text{new}} y_i K_{11} - \alpha_2^{\text{new}} y_2 K_{21}$$

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

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

$$b_1^{\text{new}} = -E_1 - y_1 K_{11} \left(\alpha_1^{\text{new}} - \alpha_1^{\text{old}} \right) - y_2 K_{21} \left(\alpha_2^{\text{new}} - \alpha_2^{\text{old}} \right) + b^{\text{old}}$$

同理，若$0 < \alpha_2^{\text{new}} < C$，有

$$b_2^{\text{new}} = -E_2 - y_1 K_{12} \left(\alpha_1^{\text{ncw}} - \alpha_1^{\text{old}} \right) - y_2 K_{22} \left(\alpha_2^{\text{new}} - \alpha_2^{\text{old}} \right) + b^{\text{old}}$$

若$\alpha_1^{\text{new}}, \alpha_2^{\text{new}}$同时满足条件$0 < \alpha_i^{\text{new}} < C, i = 1, 2$，那么$b_1^{\text{new}} = b_2^{\text{new}}$.

若$\alpha_1^{\text{new}}, \alpha_2^{\text{new}}$同时是0或者C，那么$b_1^{\text{new}}, b_2^{\text{new}}$ 以及它们之间的数都符合KKT条件的阈值，这时选择它们的中点作为b^{new}.

更新 $E_i^{\text{new}} = \sum_S y_j \alpha_j K(x_i, x_j) + b^{\text{new}} - y_i$，S是所有支持向量的集合.

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 $\hat{\alpha}=\alpha^{(k+1)}$. ■

Train SVM Classifier

Load Fisher's iris data set. Remove the sepal lengths and widths and all observed setosa irises.

```
load fisheriris
inds = ~strcmp(species,'setosa');
X = meas(inds,3:4);
y = species(inds);
```

Train an SVM classifier using the processed data set.

```
SVMModel = fitcsvm(X,y,"BoxConstraint",10,"KernelFunction","linear")
```

```
SVMModel = 
  ClassificationSVM
             ResponseName: 'Y'
    CategoricalPredictors: []
               ClassNames: {'versicolor'  'virginica'}
           ScoreTransform: 'none'
          NumObservations: 100
                    Alpha: [15×1 double]
                     Bias: -21.2030
         KernelParameters: [1×1 struct]
           BoxConstraints: [100×1 double]
```

```
      ConvergenceInfo: [1×1 struct]
      IsSupportVector: [100×1 logical]
               Solver: 'SMO'

  Properties, Methods
```

`SVMModel` is a trained `ClassificationSVM` classifier. Display the properties of `SVMModel`. For example, to determine the class order, use dot notation.

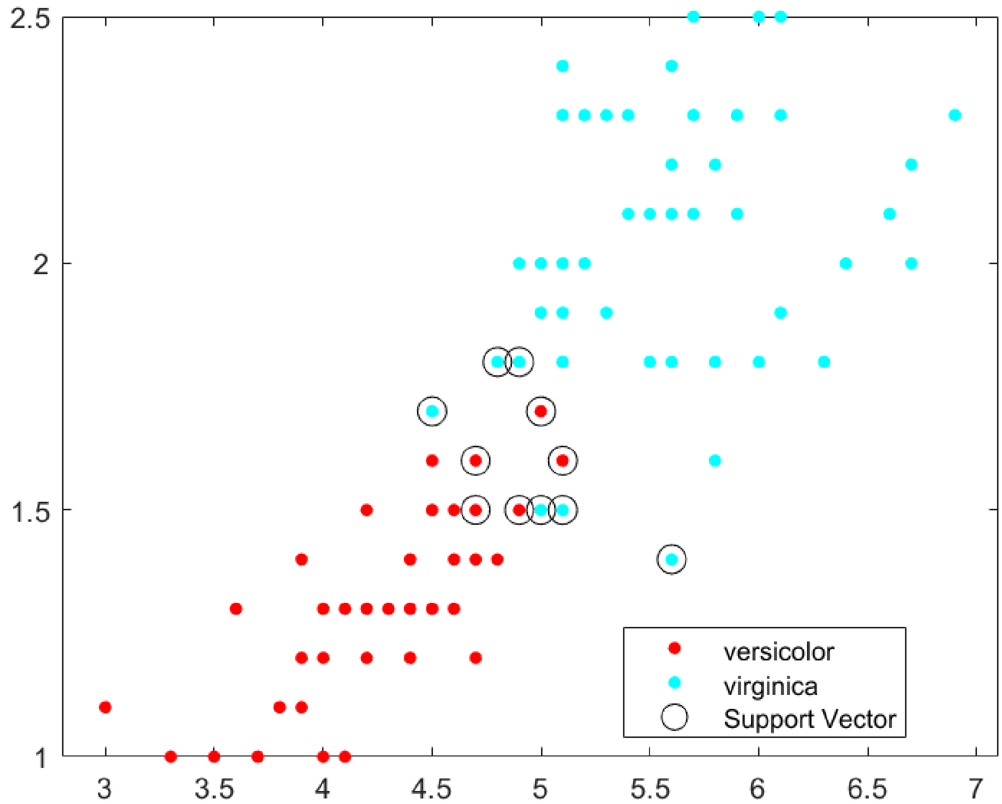
```
classOrder = SVMModel.ClassNames
```

```
classOrder = 2×1 cell array
    {'versicolor'}
    {'virginica' }
```

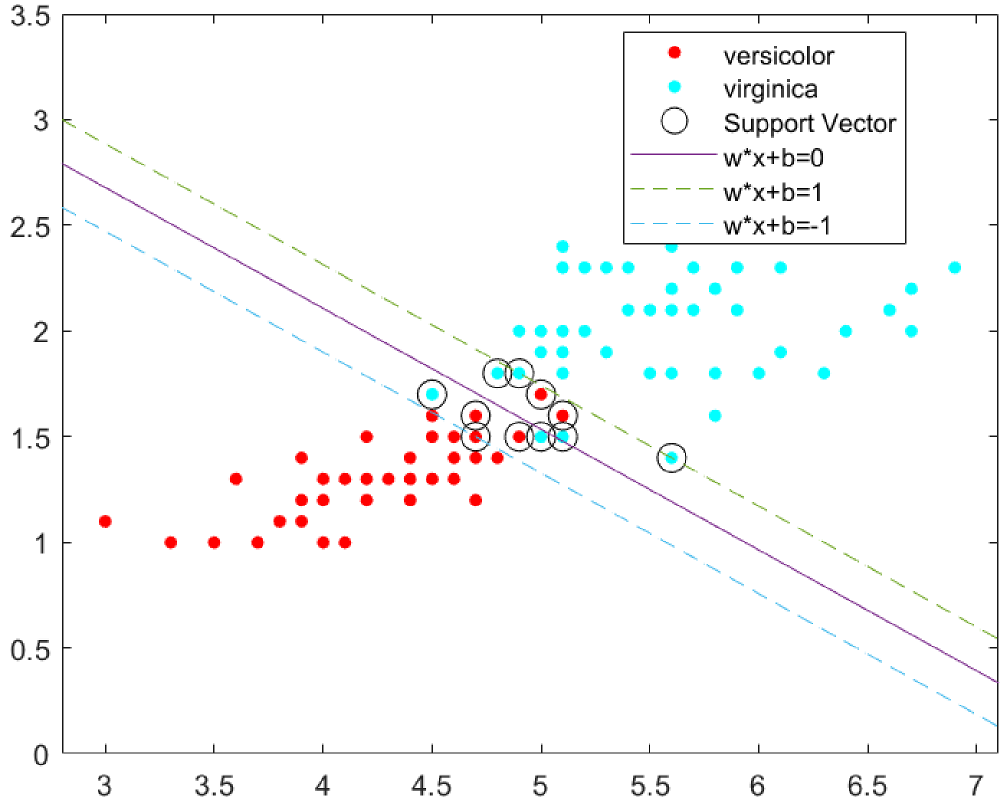
The first class (`'versicolor'`) is the negative class, and the second (`'virginica'`) is the positive class. You can change the class order during training by using the `'ClassNames'` name-value pair argument.

Plot a scatter diagram of the data and circle the support vectors.

```
sv = SVMModel.SupportVectors;
figure
gscatter(X(:,1),X(:,2),y)
hold on
plot(sv(:,1),sv(:,2),'ko','MarkerSize',10)
legend('versicolor','virginica','Support Vector')
```




```
b = SVMModel.Bias;
w = sum(SVMModel.Alpha.*SVMModel.SupportVectorLabels.*SVMModel.SupportVectors);
xx = 2:0.01:7.5;
yy = (-b-w(1)*xx)/w(2); % w*x+b=0
yy1 = (1-b-w(1)*xx)/w(2); % w*x+b=1
yy2 = (-1-b-w(1)*xx)/w(2); % w*x+b=-1
hold on; plot(xx,yy);plot(xx,yy1,'--');plot(xx,yy2,'--');
legend('versicolor','virginica','Support Vector','w*x+b=0','w*x+b=1','w*x+b=-1')
```



The support vectors are observations that occur on or beyond their estimated class boundaries.

You can adjust the boundaries (and, therefore, the number of support vectors) by setting a box constraint during training using the `'BoxConstraint'` name-value pair argument.

作业

习题7.2,7.3,7.4