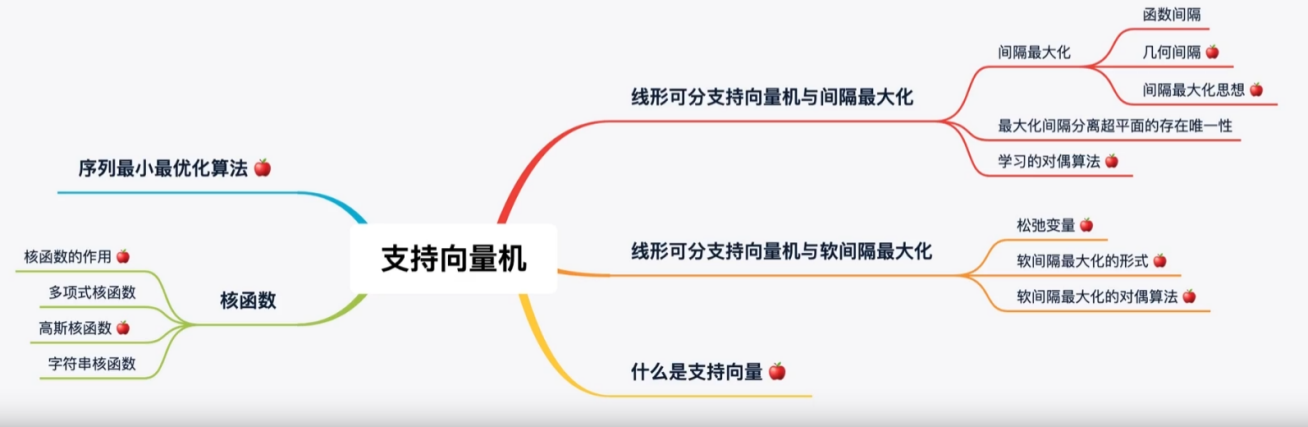
# 统计学习方法——支持向量机

create: 2021/11/23 广州　　update：2021/11/27 广州



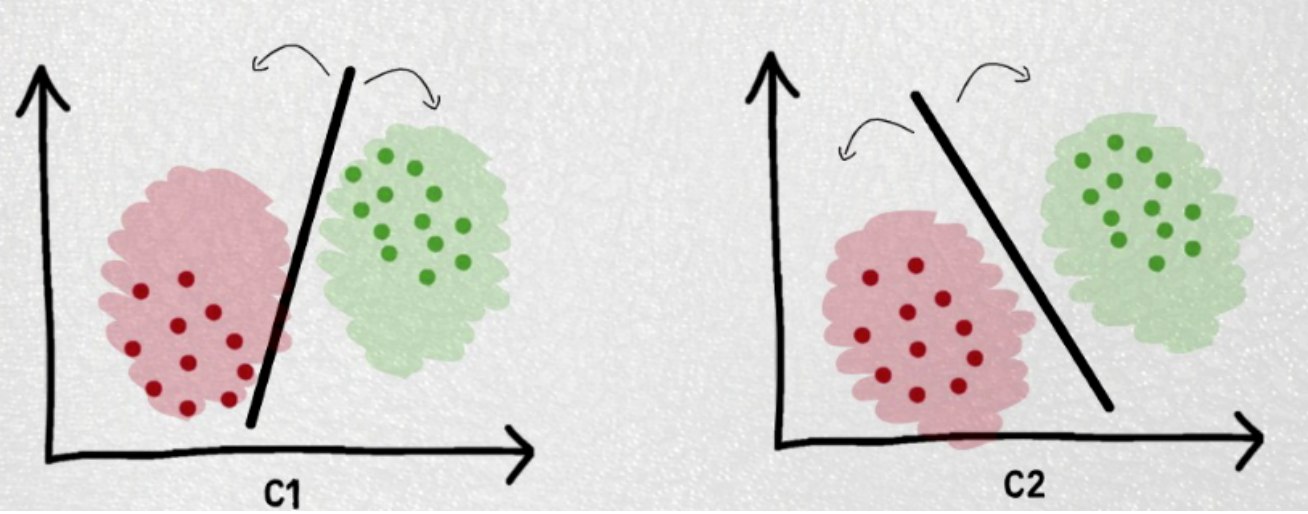
注：李航的统计学习方法中有关硬间隔和软间隔支持向量的部分没写（7.2.3）。

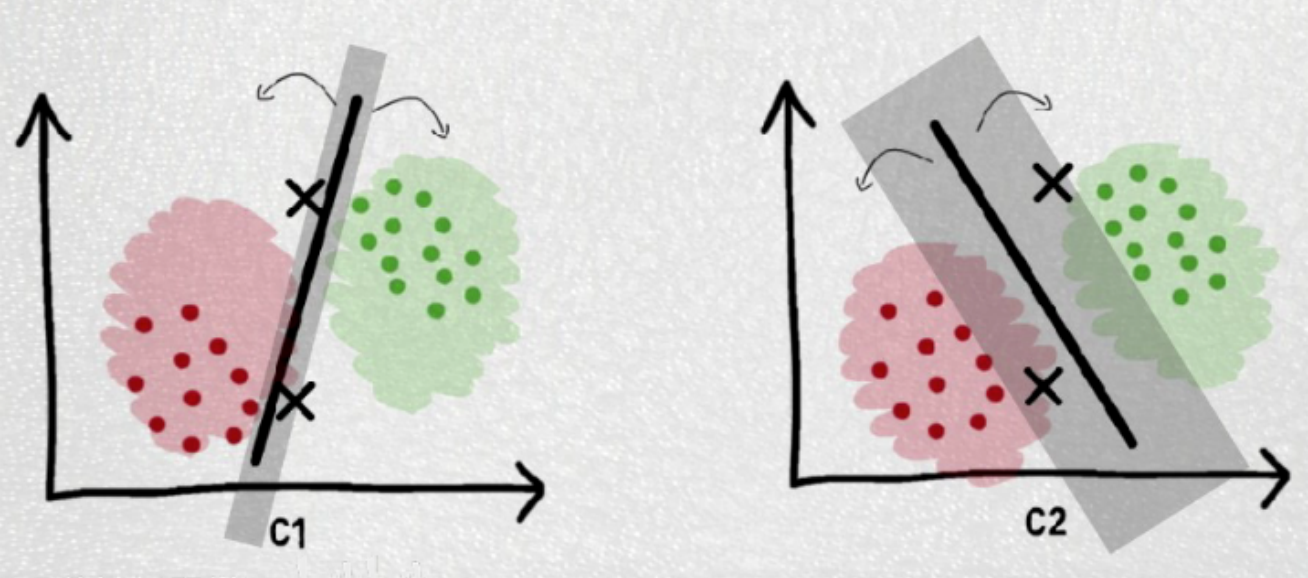



## SVM简介

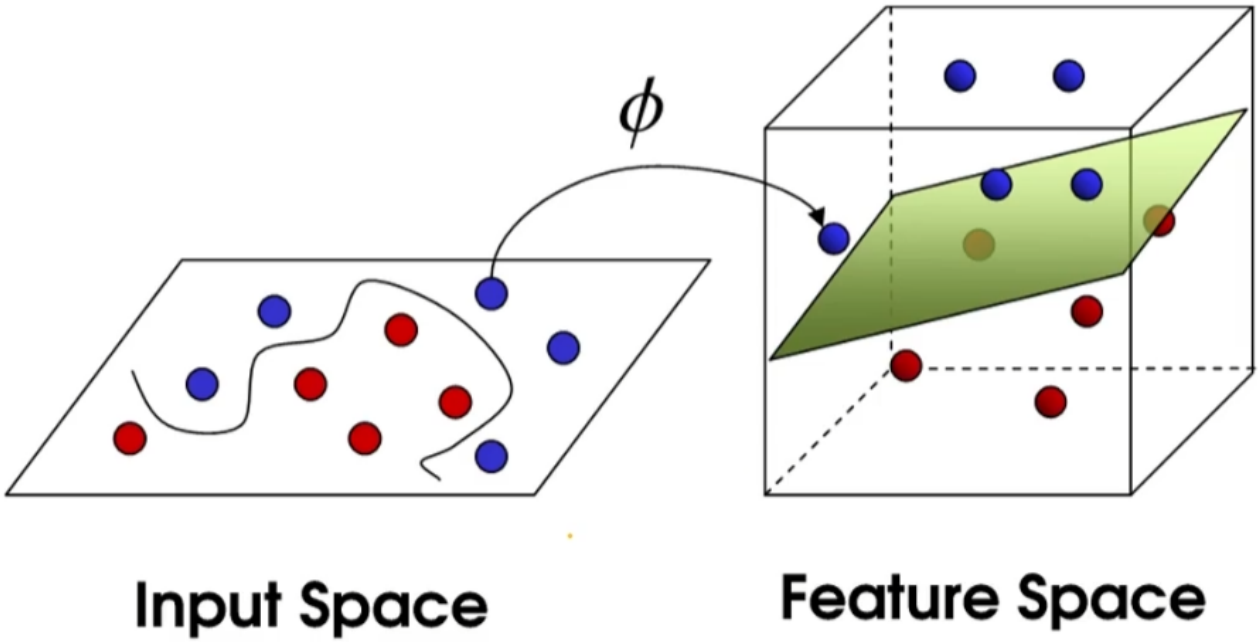
支持向量机（support vector machines, SVM）是一种二分类模型，它的基本模型是定义在特征空间上的**间隔最大的线性分类器**，间隔最大使它有别于感知机；SVM还包括**核技巧**，这使它成为实质上的非线性分类器。SVM的学习策略就是间隔最大化。

**间隔最大化**：寻找一个超平面，这个超平面使得与它距离最近的样本点的距离必须大于其他所有超平面划分与最近的样本点的距离。




SVM可以划分非线性平面，将低维空间转换为高维空间，再划分高维空间。**在低维空间上的非线性划分的数据在某一高维度上一定是线性可分的。**



# 函数间隔

函数间隔通常表示为:$|w \cdot x_1 + b|$

其中$w$和$x_1$都是n维的向量，$x_1$为第一组样本的数据向量，这个式子表示的就是$x_1$这个数据点到超平面$w \cdot x + b = 0$的距离。但是函数间隔会带来一个问题，计算机在优化的时候，会同时缩小$w$和$b$的值来减小$|w \cdot x_1 + b|$

假设 $|w \cdot x_1 + b| = 1000$，如果去优化这个式子，计算机很有可能同时缩小$w$和$b$变成 $|\frac{1}{1000} w \cdot x_1 + \frac{1}{1000} b| = 1$，但是这样肯定是不行的，**因为w的值本质上是没有变化的只是等比例缩小了。**所以就需要几何间隔。

# 几何间隔

几何间隔通常表示为:$\frac{1}{||w||} |w \cdot x_1 + b|$

就是在函数间隔中加入了$w$的$L_2$范数

$L_2$范数定义为$||w||_2 = \sqrt{\sum_{i=1}^{N} w_i^2}$

因此当出现$w$和$b$同时缩小1000倍的情况时，

$\frac{1}{||w||} |\frac{1}{1000} w \cdot x_1 + \frac{1}{1000} b|$， 其中$||w||$为$\frac{1}{1000} \sqrt{\sum_{i=1}^{N} w_i^2}$，取个倒数就能将公式里面的$\frac{1}{1000}$化掉。

结果变为$\frac{1}{\sqrt{\sum_{i=1}^{N} w_i^2}} |w \cdot x_1 + b|$。此时w向量的值就能被优化了。

# 线性可分支持向量机与硬间隔最大化

定义一个特征空间上的训练数据集:

$$T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

其中$x_i \in \chi = R^n$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$i = 1, 2, \ldots, N$，其中$x_i$为第i个向量特征也就是实例，$y_i$是$x_i$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。

在SVM中对于给定的训练数据集$T$和超平面$(w, b)$，我们可以定义超平面$(w, b)$关于样本点$(x_i, y_i)$的**函数间隔**:

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

定义超平面$(w, b)$关于训练数据集$T$所有样本点的函数间隔的最小值为:

$$\tilde{\gamma} = \min_{i=1,\ldots,N} \tilde{\gamma}_i$$

定义几何间隔$\gamma_i$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$T$所有样本点的几何间隔的最小值为:

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

因此很容易可以得出函数间隔和几何间隔的关系:

$$\gamma_i = \frac{\tilde{\gamma}_i}{||w||}$$

$$\gamma = \frac{\tilde{\gamma}}{||w||}$$

如果$||w|| = 1$那么函数间隔和几何间隔是相等的。

# 间隔最大化

对于线性可分的数据而言，支持向量机希望找到能够正确划分数据集并且几何间隔最大的分离超平面，线性可分的超平面有无穷多个，但是几何间隔最大的分离超平面是唯一的。

最大间隔分离超平面可以转换为下面的约束最优化问题：

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

$$s.t. \quad y_i \left( \frac{w}{||w||} \cdot x_i + \frac{b}{||w||} \right) \geq \gamma, \quad i = 1, 2, \dots, N$$

最大化超平面$(w, b)$关于训练数据集的几何间隔$\gamma$

根据几何间隔的函数间隔的关系，可以将上式写为：

$$\max_{w,b} \quad \frac{\tilde{\gamma}}{||w||}$$

$$s.t. \quad y_i \left( w \cdot x_i + b \right) \geq \tilde{\gamma}, \quad i = 1, 2, \dots, N$$

在上面对于函数间隔和几何间隔的介绍中可以看到，将$w$和$b$同时按比例放大或者缩小，函数间隔$\tilde{\gamma}$也会按照同样的比例变化，因为函数间隔对于优化问题的不等式约束没有影响，所以我们可以取$\tilde{\gamma} = 1$，带入上面的式子就可以得到：

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

$$s.t. \quad y_i \left( w \cdot x_i + b \right) - 1 \geq 0, \quad i = 1, 2, \dots, N$$

最大化$\frac{1}{||w||}$和最小化$\frac{1}{2} ||w||^2$是等价的，加入$1\backslash 2$和$2$次方是方便下面计算中求导数。

到此就形成了一个完整的算法流程如下：

**线性可分支持向量机学习算法（最大间隔法 maximum margin method）：**

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 $x_i\in\mathcal{X}=R^n, y_i\in\mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$

输出：最大间隔分离超平面和分类决策函数

1. 构建并求解约束最优化问题

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

求得最优解 $w^*,b^*$ 。

2. 得到分离超平面

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

以及分类决策函数

$$f(x)=sign\left(w^*\cdot x+b^*\right)$$

# 学习的对偶算法

为了求解约束问题，我们构建拉格朗日乘数如下：

$$\begin{aligned}L(w,b,\alpha)&=\frac{1}{2}||w||^2+\sum_{i=1}^{n}\alpha_i(1-y_if(x_i))\\&=\frac{1}{2}||w||^2+\sum_{i=1}^{n}\alpha_i(1-y_i(w\cdot x_i+b))\end{aligned}$$

$\alpha=(\alpha_1,\alpha_2,\ldots,\alpha_N)^T$为拉格朗日乘子向量。

该部分有3点需要补充1.关于$\alpha_i\geq 0$的说明。2.拉格朗日对偶性的说明。3.KKT条件的说明。

根据拉格朗日对偶性，可以转换成极大极小问题（这里不细说了，Reference中给出的链接有详细的解释）

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

我们需要求L对$w,b$的极小值，再求对$\alpha$的极大值。

- 求$\min_{w,b}L(w,b,\alpha)$

  对w和b求偏导：

$$\begin{aligned}\frac{\partial L(w,b,\alpha)}{\partial w}&=w-\sum_{i=1}^{N}\alpha_iy_ix_i=0\\\frac{\partial L(w,b,\alpha)}{\partial b}&=-\sum_{i=1}^{N}\alpha_iy_i=0\end{aligned}$$

  得：

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

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

将结果带入拉格朗日函数，得到结果：

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

$$=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_{_i} y_i \alpha_j y_j (x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_{_i}$$

因此得到了:

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

- 求$\min_{w,b}L(w,b,\alpha)$对$\alpha$的极大，是对偶问题

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

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

$$\alpha_i\geq 0, i=1,2,\dots,n$$

等价于

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

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

$$\alpha_i\geq 0, i=1,2,\dots,n$$

因为原始问题满足Slater条件，所以存在$w^*,\alpha^*,\beta^*$ ，使$w^*$是原始问题的解，$\alpha^*,\beta^*$是对偶问题的解。所以求解原始问题可以转换为求解对偶问题。

算法流程如下:

**线性可分支持向量机（硬间隔支持向量机）学习算法：**

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 $x_i\in\mathcal{X}=R^n, y_i\in\mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$

输出：最大间隔分离超平面和分类决策函数

1. 构建并求解约束最优化问题

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

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

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

求得最优解 $\alpha^*=\left(\alpha_1^*,\alpha_1^*,\cdots,\alpha_N^*\right)$ 。

2. 计算（根据KKT得到的等式）

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

并选择 $\alpha^*$ 的一个正分量 $\alpha_j^*>0$ ，计算

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

3. 得到分离超平面

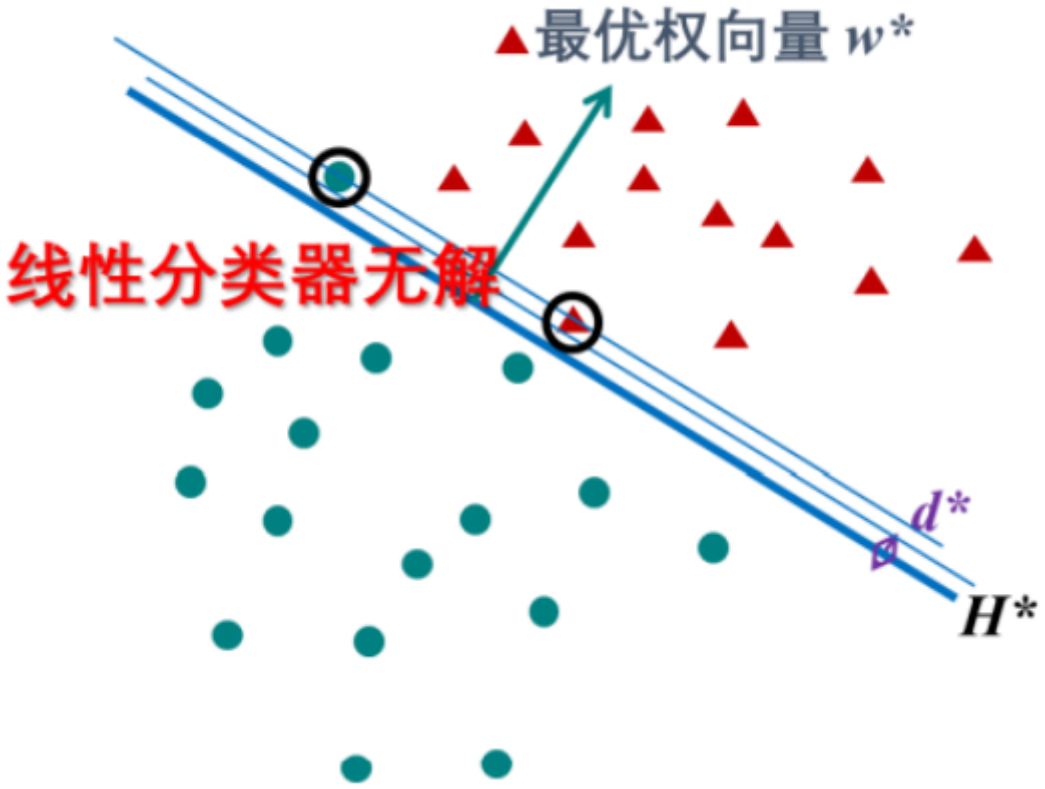
$$w^*\cdot x+b^*=0$$

以及分类决策函数

$$f(x)=sign\left(w^*\cdot x+b^*\right)$$

# 线性支持向量机与软间隔最大化

有时候本来数据的确是可分的，也就是说可以用线性分类SVM的学习方法来求解，但是却因为混入了异常点，导致不能线性可分，比如下图，在▲中存在圆形的的异常点，因此无法按照上面讲的线性可分的方法进行分类，因此SVM引入了软间隔最大化，这**属于线性不可分。**




## 软间隔最大化

线性不可分意味着某些样本点不能满足函数间隔大于等于1的约束条件。为此，对每一个样本点引进一个松弛变量 $\xi_i \geq 0$，使得函数间隔加上松弛变量大于等于1，因此约束条件就变为了：

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, i = 1, 2, \ldots, n$$

因为加入了松弛变量，我们的目标函数也发生了变化：

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

这里C>0为惩罚参数，值越大对误分类的惩罚越大，越容易造成过拟合。

线性不可分的线性支持向量机问题变成了如下的凸二次规划问题（原始问题）：

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

## 软间隔最大化的对偶算法

该问题的拉格朗日函数为:

$$L(w, b, \xi, \alpha, \mu) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(w \cdot x_i + b) - 1 + \xi_i) - \sum_{i=1}^{N}\mu_i\xi_i$$
$$\alpha_i \geq 0, \mu_i \geq 0, i = 1, 2, \ldots, n$$

原问题就变为了：

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

对偶问题为：

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

为了求解对偶问题，需要先求$L(w,b,\xi,\alpha,\mu)$对于$w,b,\xi$的极小，再求对$\alpha,\mu$的极大。

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

得:

$$w=\sum_{i=1}^{N}\alpha_i y_i x_i$$
$$\sum_{i=1}^{N}\alpha_i y_i=0$$
$$C-\alpha_i-\mu_i=0$$

代回到$L(w,b,\xi,\alpha,\mu)$的表达式得：

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

**求$L(w,b,\xi,\alpha,\mu)$对于$\alpha,\mu$的极大，即：**

$$\max\quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i y_i \alpha_j y_j (x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i$$
$$s.t.\quad \sum_{i=1}^{N}\alpha_i y_i=0$$
$$C-\alpha_i-\mu_i=0$$
$$\alpha_i\geq 0$$
$$\mu_i\geq 0, i=1,2,\dots,N$$

利用$C-\alpha_i-\mu_i=0$消去约束条件中的$\mu$并将极大问题转化为极小问题：

$$\max\quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i y_i \alpha_j y_j (x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i$$
$$s.t.\quad \sum_{i=1}^{N}\alpha_i y_i=0$$
$$0\leq\alpha_i\leq C, i=1,2,\dots,N$$

整个算法流程如下：

**线性支持向量机（软间隔支持向量机）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 $x_i \in \mathcal{X}=R^n, y_i \in \mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$

输出：最大间隔分离超平面和分类决策函数

1. 选择惩罚参数 $C \geq 0$ ，构建并求解约束最优化问题

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

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

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

求得最优解 $\alpha^* = (\alpha_1^*, \alpha_1^*, \cdots, \alpha_N^*)$ 。

2. 计算

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

并选择 $\alpha^*$ 的一个分量 $0 < \alpha_j^* < C$ ，计算

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

3. 得到分离超平面

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

以及分类决策函数

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

## 合页损失函数

上面的介绍中，学习算法都是凸二次规划，SVM还有另外一种解释，就是最小化以下的目标函数：

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

第一项函数就被称为合页损失函数，第二项是正则化项。其中下标"+"表示取正值的函数：

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

上面的函数其实就是max函数

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

那么损失函数很好理解了，当样本点被正确分类，并且函数间隔$y_i(w \cdot x_i + b)$大于1时，损失是0。否则损失为$1-y_i(w \cdot x_i + b)$，哪怕被正确分类但是函数间隔小于1也会有损失。

原始最优化问题为：

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

等价于最优化问题:

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

令$1-y_i(w \cdot x_i + b) = \xi_i$，则$\xi_i \geq 0$。当$1-y_i(w \cdot x_i + b) > 0$时，也就是说函数间隔比1小，那么损失就是$1-y_i(w \cdot x_i + b)$也就是$\xi_i$。如果函数间隔比1大，$\xi_i = 0$同时损失也是0，所以损失永远和$\xi_i$保持一致。

即：

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

所以最优化问题就可以写成:

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

取$\lambda = \frac{1}{2C}$，则：

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

该结果与原始最优化问题的公式一样，所以原始的公式得到了证明。

下图是合页损失函数的图

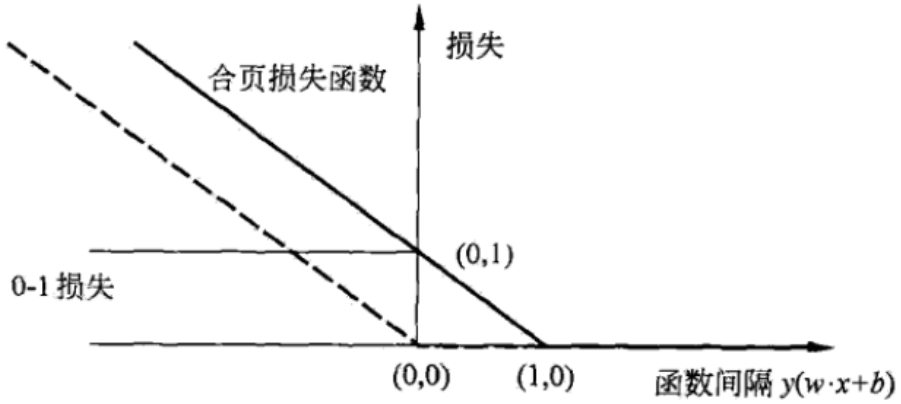



图 7.6　合页损失函数

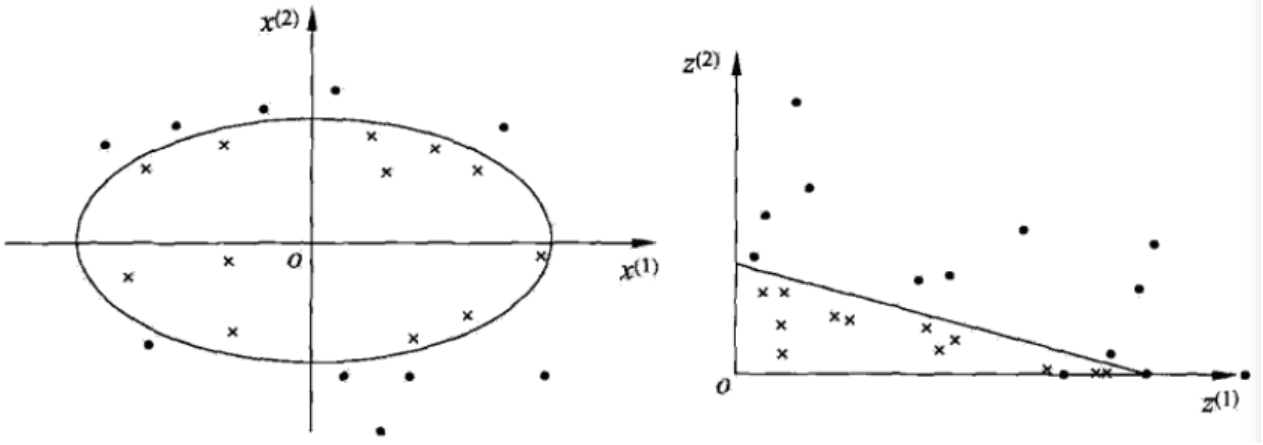
图 7.6 中虚线显示的是感知机的损失函数$[y_i(w\cdot x_i+b)]_+$. 这时，当样本点$(x_i,y_i)$被正确分类时，损失是 0，否则损失是$-y_i(w\cdot x_i+b)$. 相比之下，合页损失函数不仅要分类正确，而且确信度足够高时损失才是 0. 也就是说，合页损失函数对学习有更高的要求.

# 非线性支持向量机与核函数

## 核技巧

如下图所示：一些数据在一个空间是线性不可分的，我们将它投影到另一个空间就可以分割了。

线性分类方法求解非线性问题分为两步：首先是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模型，核技巧就是这样的方法。




设$\chi$是输入空间（欧氏空间$\mathbb{R}^n$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$\chi$到$\mathcal{H}$的映射：

$$\phi(x):\chi\to\mathcal{H}$$

使得对所有的$x,z\in\chi$，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，$\phi(x)\cdot\phi(z)$是这两个向量的内积。

核技巧的想法是：在学习和预测中只定义核函数$K(x,z)$而不显式定义映射函数$\phi(x)$。对于给定的核函数，映射函数并不唯一。

**我们如果直接计算$K(w,z)$比较容易，但是求$\phi(x)\cdot\phi(z)$的结果来计算K就很复杂了。**

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都只涉及输入实例和实例之间的内积，使用核函数替代内积，此时对偶问题变为：

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

分类的决策函数变成了：

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

这等价于经过映射函数$\phi(x)$将原来的输入空间变换到一个新的特征空间，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为非线性支持向量机。学习是隐式地在特征空间中进行的，不需要显示地定义特征空间和映射函数。**这种技巧称为核技巧。**

在实际应用中，往往依赖于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。

通常所说的核函数就是**正定核**，正定核存在以下充要条件：

设$K:\chi*\chi\to\mathbb{R}$是对称函数，则$K(x,z)$为正定核函数的充要条件是对任意$x_i\in\chi,i=1,2,\ldots,m$，$K(x,z)$对应的Gram矩阵：

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

是**半正定矩阵**，则称$K(x,z)$是正定核

# 常用核函数

下面介绍几个常用核函数：

- 线性核

$$K(x,z)=x^Tz$$

- 多项式核

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

- 高斯核

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

如果对应的支持向量机是高斯径向基函数分类器，那么分类决策函数就变成了：

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

因此，整个非线性支持向量机的算法流程如下：

**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 $x_i\in\mathcal{X}=R^n, y_i\in\mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N$

输出：分类决策函数

1. 选择适当的核函数 $K(x,z)$ 和惩罚参数 $C\geq 0$ ，构建并求解约束最优化问题

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

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

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

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_1^*,\cdots,\alpha_N^*)$ 。

2. 计算

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

并选择 $\alpha^*$ 的一个分量 $0<\alpha_j^*<C$ ，计算

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

3. 得到分离超平面

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

以及分类决策函数

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

# 序列最小最优化算法

从上面的介绍中，我们知道支持向量机将问题转换为凸二次规划问题，凸二次规划问题具有全局最优解并且很多优化算法都可以求解，然而当训练样本的容量很大时，这些算法效率极低，所以就需要SMO算法。

SMO算法要求解如下的凸二次规划问题：

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

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

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

在该问题中变量就变成了拉格朗日乘子$\alpha_i$。

SMO算法是一种启发式算法，其解决上述问题的基本思路是：如果所有变量的解都满足此最优化问题的KKT条件，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，因为这会使得原始问题的目标函数变得更小。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。

由于$\sum_{i=1}^{N}\alpha_i y_i = 0$，假如$\alpha_3, \alpha_4, \ldots, \alpha_m$固定，那么$\alpha_1, \alpha_2$的关系也就确定了，这样SMO算法将一个复杂的优化算法转化为一个比较简单的两变量优化问题。

整个SMO算法包括两个部分：求解两个变量的二次规划的解析方法、选择变量的启发式方法。

## 两个变量的二次规划的求解方法

假设所选两个变量为$\alpha_1, \alpha_2$，其他变量$\alpha_i, i = 3, \ldots, n$是固定的，那么SMO的最优化问题的子问题就可以写为：

$$\min_{\alpha_1, \alpha_2} \quad W\left(\alpha_1, \alpha_2\right) = \frac{1}{2}K_{11}\alpha_1^2 + \frac{1}{2}K_{22}\alpha_2^2 + y_1 y_2 K_{12}\alpha_1\alpha_2$$

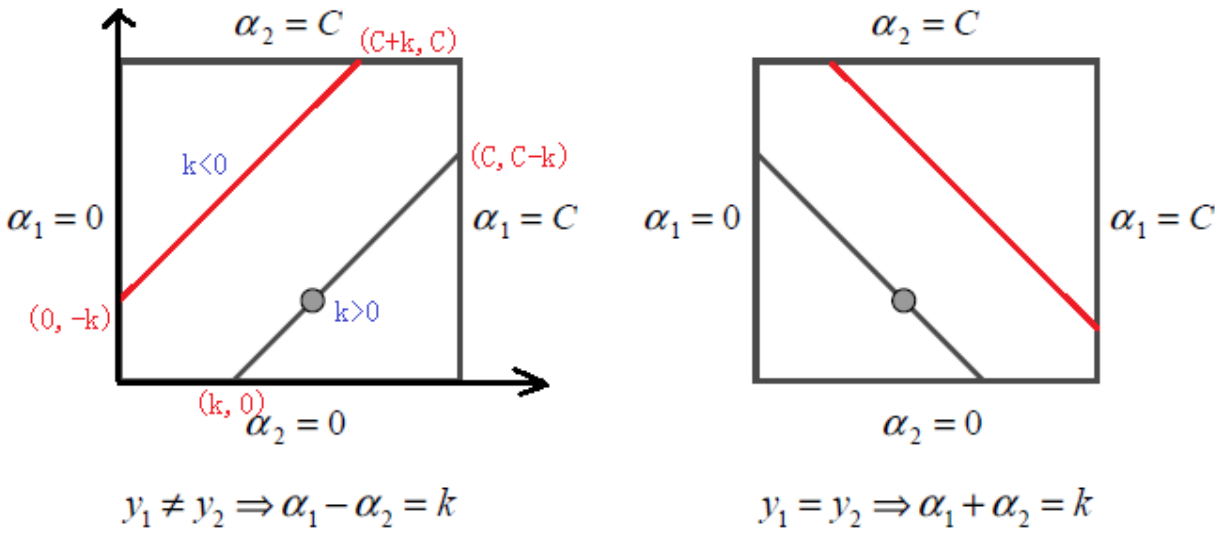
$$-\left(\alpha_1 + \alpha_2\right) + y_1\alpha_1\sum_{i=3}^{N}y_i\alpha_i K_{i1} + y_2\alpha_2\sum_{i=3}^{N}y_i\alpha_i K_{i2}$$

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

$$0 \leq \alpha_i \leq C, \quad i = 1, 2$$

其中，$K_{ij} = K(x_i, x_j)$，$\varsigma$是常数，上式省略了不含$\alpha_1, \alpha_2$的常数项。

为了求解上面的优化问题，我们首先分析约束条件。



根据上面的约束条件$\alpha_1+\alpha_2=\varsigma$，$\alpha_i\leq C, i=1,2$因为$y_1,y_2$只能取1或者-1，因此我们可以构造上图，$\alpha_1,\alpha_2$在0-C范围的盒子里面，而且斜率只能是1或者-1，左边的图是$y_1\neq y_2$的情形，并且我们规定$\alpha_1-\alpha_2=k$，红色的那条线代表了$\alpha_1<\alpha_2$，因此k小于0，并且我们求出在盒子边缘的坐标。另外一条灰色的线代表了$\alpha_1>\alpha_2$，因此k是大于0的。右边的图表示的是$y_1=y_2$的情况。

由于$\alpha_1,\alpha_2$的关系被限制在盒子里的一条线段上，所以两变量的优化问题实际上仅仅是一个变量的优化问题。不妨我们假设最终是$\alpha_2$的优化问题。由于我们采用的是启发式的迭代法，假设我们上一轮迭代得到的解是$\alpha_1^{old},\alpha_2^{old}$，假设沿着约束方向$\alpha_2$未经剪辑的解是$\alpha_2^{new,unc}$本轮迭代完成后的解为$\alpha_1^{new},\alpha_2^{new}$。

由于$\alpha_2^{new}$必须满足上图中的约束条件，假设L和H是$\alpha_2^{new}$所在的线段的边界，所以就得到了:

$$L\leq\alpha_2^{new}\leq H$$

原本$\alpha_2^{new}$的限制只是0——C这个区间，我们可以根据上面的图缩小$\alpha_2^{new}$的取值区间，只观察纵轴，影响$\alpha_2^{new}$取值的坐标只有$(0,-k)$和$(C,C-k)$，前者可以和0对比，取大的那个作为L，后者可以和C对比，取小的作为H。已知$k=(\alpha_1-\alpha_2)$带入就可以得到结果。

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

同理右边的图可以得到:

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

也就是说我们通过求导得到$\alpha_2^{new,unc}$，最终的$\alpha_2^{new}$应该为:

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

我们将目标函数对$\alpha_2$求偏导就可以得到$\alpha_2^{new,unc}$。

我们再整理下目标函数，

为了简化叙述，记:

$$g(x)=w^*\bullet\phi(x)+b=\sum_{i=1}^{N}\alpha_i^*y_iK\left(x,x_i\right)+b^*$$

令:

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

$$v_i=\sum_{j=3}^{N}\alpha_jy_jK\left(x_i,x_j\right)=g\left(x_i\right)-\sum_{j=1}^{2}\alpha_jy_jK\left(x_i,x_j\right)-b,\quad i=1,2$$

当$i=1,2$时，$E_i$为函数$g(x)$对输入$x_i$的预测值与真实值$y_i$的差，目标函数可以写为:

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

因为$\alpha_1y_1=\varsigma-\alpha_2y_2, y_i^2=1$，所以$\alpha_1$可以表示为:

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

带入，得:

$$\begin{aligned}W\left(\alpha_2\right)=&\tfrac{1}{2}K_{11}[(\varsigma-y_2\alpha_2)y_1]^2+\tfrac{1}{2}K_{22}\alpha_2^2+y_1y_2K_{12}\left(\varsigma-y_2\alpha_2\right)y_1\alpha_2\\&-[(\varsigma-y_2\alpha_2)y_1+\alpha_2]+y_1v_1\left(\varsigma-y_2\alpha_2\right)y_1+y_2v_2\alpha_2\\=&\tfrac{1}{2}K_{11}(\varsigma-y_2\alpha_2)^2+\tfrac{1}{2}K_{22}\alpha_2^2+y_2K_{12}\left(\varsigma-y_2\alpha_2\right)\alpha_2\\&-(\varsigma-y_2\alpha_2)y_1-\alpha_2+v_1\left(\varsigma-y_2\alpha_2\right)+y_2v_2\alpha_2\end{aligned}$$

现在我们可以通过求偏导来得到$\alpha_2^{new,nuc}$

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

整理后得:

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

将$\varsigma = \alpha_1^{\text{old}} y_1 + \alpha_2^{\text{old}} y_2$带入得:

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

整理可得:

$$\alpha_2^{\text{new,unc}} = \alpha_2^{\text{old}} + \frac{y_2\left(E_1 - E_2\right)}{K_{11} + K_{22} - 2K_{12})}$$

得到了$\alpha_2^{new}$后，根据线性关系，可以得到$\alpha_1^{new}$

# 变量选择方法

## 第一个变量的选择:

SMO算法称选择第一个变量为外层循环，这个变量需要选择在训练集中违反KKT条件最严重的样本点。检查样本点是否满足$KKT$条件:

$$\alpha_i^* = 0 \Rightarrow y_i g\left(x_i\right) \geq 1(\text{正常分类})$$
$$0 < \alpha_i^* < C \Rightarrow y_i g\left(x_i\right) = 1(\text{支持向量})$$
$$\alpha_i^* = C \Rightarrow y_i g\left(x_i\right) \leq 1(\text{两边界之间})$$

一般来说我们首先选择违反$0 < \alpha_i^* < C \Rightarrow y_i g\left(x_i\right) = 1$这个条件的点。如果这些支持向量都满足KKT条件，再选择违反$\alpha_i^* = 0 \Rightarrow y_i g\left(x_i\right) \geq 1$和$\alpha_i^* = C \Rightarrow y_i g\left(x_i\right) \leq 1$的点。

## 第二个变量的选择

SMO算法称选择第二个变量为内层循环，假设我们在外层循环已经找到了$\alpha_1$, 第二个变量$\alpha_2$的选择标准是变化足够大，而$\alpha_2$依赖于$|E_1 - E_2|$，所以我们的目标就变成了$|E_1 - E_2|$有足够大的变化。由于$\alpha_1$定了的时候,$E_1$也确定了，所以要想$|E_1 - E_2|$最大，只需要在$E_1$为正时，选择最小的$E_i$作为$E_2$， 在$E_1$为负时，选择最大的$E_i$作为$E_2$，可以将所有的$E_i$保存下来加快迭代。

如果内存循环找到的点不能让目标函数有足够的下降， 可以采用遍历支持向量点来做$\alpha_2$,直到目标函数有足够的下降， 如果所有的支持向量做$\alpha_2$都不能让目标函数有足够的下降，可以跳出循环，重新选择$\alpha_1$。

当完成两个变量的优化后，我们需要重新计算阈值b。当$0 < \alpha_1^{new} < C$，由KKT条件:

$$y_1 - \sum_{i=1}^{N} \alpha_i y_i K_{i1} - b_1 = 0$$

所以更新之后的$b_1^{new}$就等于:

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

$E_1$为:

$$E_1 = g\left(x_1\right) - y_1 = \sum_{i=3}^{N} \alpha_i y_i K_{i1} + \alpha_1^{old} y_1 K_{11} + \alpha_2^{old} y_2 K_{21} + b^{old} - y_1$$

可以看到上面的两个式子都有$y_1 - \sum_{i=3}^{N} \alpha_i y_i K_{i1}$，因此我们可以将$b_1^{new}$用$E_1$表示

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

同理：

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

如果$\alpha_1^{new}, \alpha_2^{new}$满足$0 < \alpha_i^{new} < C, i = 1, 2$，那么：

$$b^{new} = b_1^{new} = b_2^{new}$$

否则$b^{new}$为:

$$b^{\text{new}} = \frac{b_1^{\text{new}} + b_2^{\text{new}}}{2}$$

得到$b^{new}$后，我们更新$E_i$为：

$$E_i = \sum_{S} y_j \alpha_j K\left(x_i, x_j\right) + b^{new} - y_i$$

S是所有支持向量$x_j$的集合。

## SMO算法流程

输入:输入训练集$T = \{(x_1, y_1), \ldots, (x_N, y_N)\}$，$x_i$是n维的特征向量。y是1或者-1，精度$\epsilon$，输出为近似解$\hat{\alpha}$。

1. 取初值$\alpha^0 = 0, k = 0$
2. 通过前面的变量选择$\alpha_1^k$和$\alpha_2^k$，并求$\alpha_2^{new,unc}$

$$\alpha_2^{new,\text{ unc}} = \alpha_2^k + \frac{y_2\left(E_1 - E_2\right)}{K_{11} + K_{22} - 2K_{12})}$$

3. 然后求出$\alpha_2^{k+1}$

$$\alpha_2^{k+1} = \begin{cases} H & \alpha_2^{\text{new,unc}} > H \\ \alpha_2^{\text{new,unc}} & L \le \alpha_2^{\text{new , unc}} \le H \\ L & \alpha_2^{\text{new , unc}} < L \end{cases}$$

4. 利用线性关系求出$\alpha_1^{k+1}$
5. 求出$b^{k+1}$和$E_i$
6. 在精度$\epsilon$范围内检查是否满足如下的终止条件：

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$
$$0 \le \alpha_i \le C, i = 1, 2 \ldots N$$
$$\alpha_i^{k+1} = 0 \Rightarrow y_i g\left(x_i\right) \ge 1$$
$$0 < \alpha_i^{k+1} < C \Rightarrow y_i g\left(x_i\right) = 1$$
$$\alpha_i^{k+1} = C \Rightarrow y_i g\left(x_i\right) \le 1$$

7. 如果满足则结束，返回$\alpha^{k+1}$,否则转到步骤2

# Reference：

https://www.bilibili.com/video/BV1Y7411P7nd?from=search&seid=6845846039550236819&spm_id_from=333.337.0.0

https://zhuanlan.zhihu.com/p/38296513

https://zhuanlan.zhihu.com/p/38163970

https://www.bilibili.com/video/BV1i4411G7Xv?p=7