

支持向量机

样本较少的情况下，使用支持向量机基本上能得到好的结果

1. 间隔与支持向量

划分超平面：

$$\mathbf{w}^T \mathbf{x} + b = 0$$

\mathbf{w} 为法向量决定了超平面的方向， b 位移项决定了超平面与原点的距离

将样本空间中任意一点 \mathbf{x} 到超平面的距离写为：

$$r = \frac{|\mathbf{w}^T \mathbf{x} + b|}{\|\mathbf{w}\|}$$

为了最大化间隔，仅需最大化

$$\|\mathbf{w}\|^{-1}$$

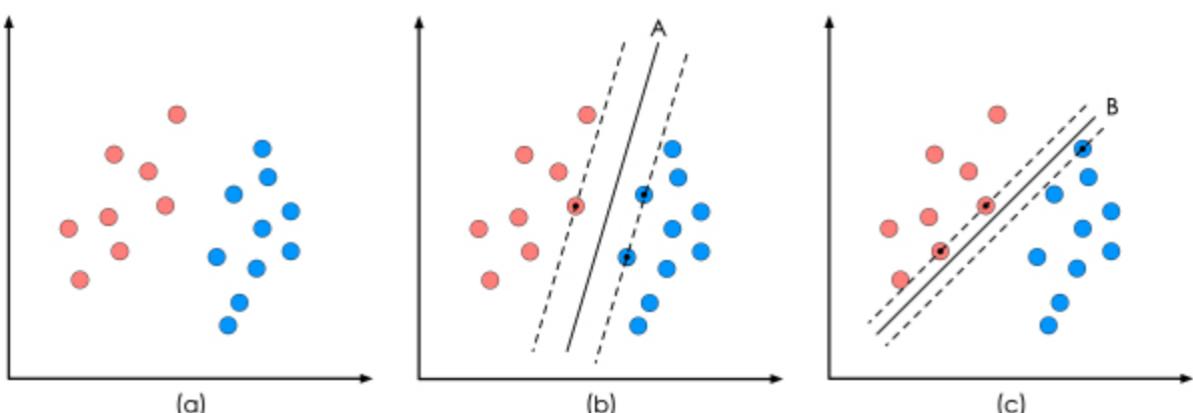
这就是支持向量机的基本型

SVM处理线性问题

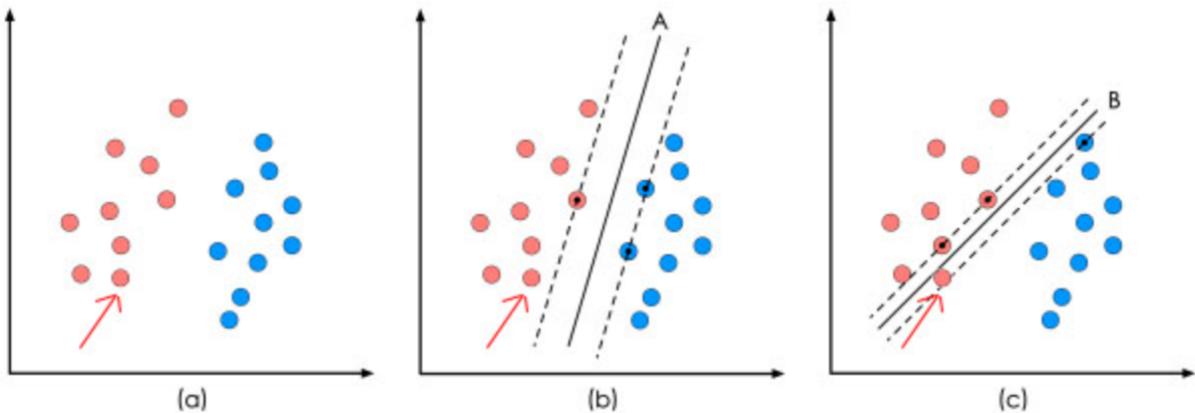
最大分类间隔器

线性模型，就是找到最好的一个平面，最大限度地容忍误差，那大概怎么画

- 得定义一个性能指标----即 d 【间隔距离 Margin】但是得处于最中间的，即到两边的边界距离相等



- 上图中的(a)是已有的数据，红色和蓝色分别代表两个不同的类别。数据显然是线性可分的，但是将两类数据点分开的直线显然不止一条。上图的(b)和(c)分别给出了B、C两种不同的分类方案，其中黑色实线为分界线，术语称为“决策面”。每个决策面对应了一个线性分类器。虽然从分类结果上看，分类器A和分类器B的效果是相同的。但是他们的性能是有差距的，看下图：



- 在“决策面”不变的情况下，又添加了一个红点。分类器B依然能很好的分类结果，而分类器C则出现了分类错误。

- 【SVM算法独有的“分类概念”】

在保证决策面方向不变且不会出现错分样本的情况下移动决策面，会在原来的决策面两侧找到两个极限位置（越过该位置就会产生错分现象），如虚线所示。

- 虚线的位置由决策面的方向和距离原决策面最近的几个样本的位置决定。而这两条平行虚线正中间的分界线就是在保持当前决策面方向不变的前提下得的**最优决策面**。两条虚线之间的**垂直距离**就是这个最优决策面对应的**分类间隔**。具有“**最大间隔**”的决策面就是SVM要寻找的**最优解**，这个真正的最优解对应的两侧虚线所穿过的样本点，就是SVM中的支持样本点，称为“支持向量”。

数学建模

最优化问题通常有两个基本因素：

- 1) 目标函数：“分类间隔”
- 2) 优化对象：决策面
- 数学描述：

- 二维空间下一条直线的方式：

$$y = ax + b$$

- 做个小改变，原来的x轴变为 x_1 ，y轴变为 x_2

$$x_2 = ax_1 + b$$

- 移项得：

$$ax_1 - x_2 + b = 0$$

- 将公式向量化：

$$\begin{bmatrix} a & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + b = 0$$

- 进一步向量化：

$$\mathbf{w}^T \mathbf{x} + \gamma = 0$$

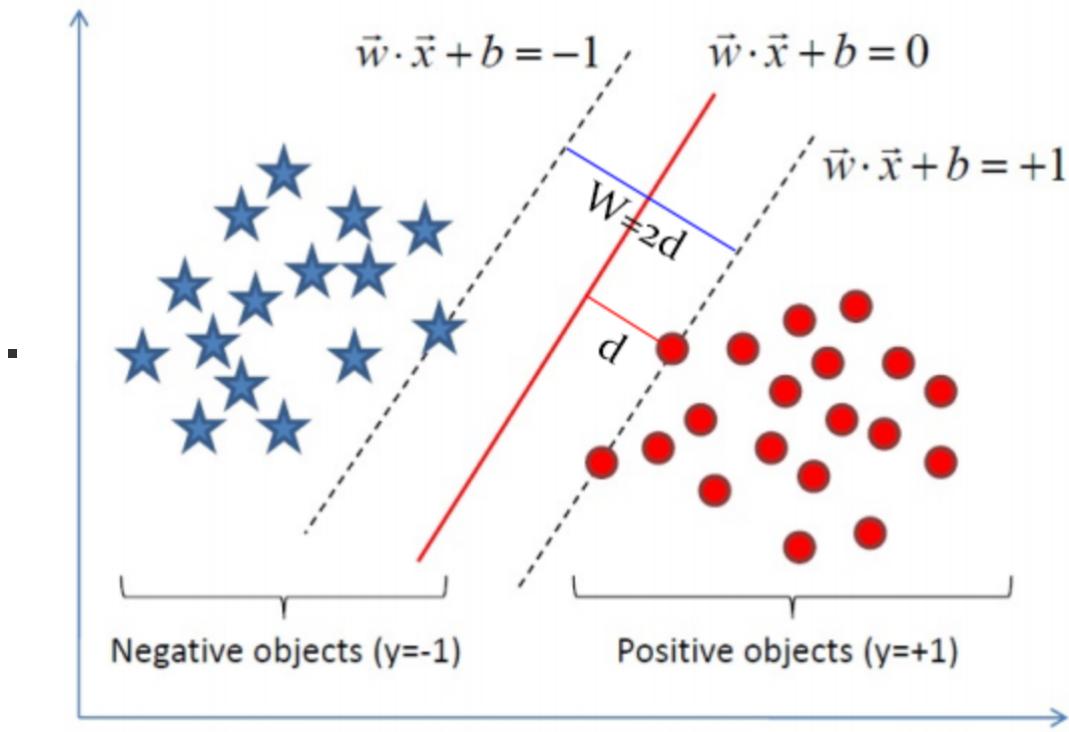
- 这里 $w_1=a$, $w_2=-1$ 。最初的那个直线方程 a 和 b 的几何意义， a 表示直线的斜率， b 表示截距， a 决定了直线与 x 轴正方向的夹角， b 决定了直线与 y 轴交点位置。

- 向量化后的直线的 w 和 r 的几何意义：

w 为直线的法向量， r 代表截距

- 将其推广到 n 维空间，就变成了超平面方程

- 分类间隔方程**



间隔的大小实际上就是支持向量对应的样本点到决策面的距离的二倍。

d得求法是：点到直线得距离

$$d = \frac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$$

直线方程扩展到多维，求得的超平面方程：

$$d = \frac{|\mathbf{w}^T \mathbf{x} + \gamma|}{\|\mathbf{w}\|}$$

这个d就是"分类间隔"。其中 $\|\mathbf{w}\|$ 表示w的二范数，求所有元素的平方和，然后再开方。比如对于二维平面：

$$\mathbf{w} = [w_1, w_2]^T$$

那么：

$$\|\mathbf{w}\| = \sqrt{w_1^2 + w_2^2}$$

目的是为了找出一个分类效果好的超平面作为分类器。分类器的好坏的评定依据是分类间隔W=2d的大小，即分类间隔w越大，则认为这个超平面的分类效果越好。此时，求解超平面的问题就变成了求解分类间隔W最大化为题。W的最大化也就是d最大化的。

■ 3) 约束条件

上图二维平面上有两种点，分别对它们进行标记：

- 红颜色的圆点标记为1，我们人为规定其为正样本；
- 蓝颜色的五角星标记为-1，我们人为规定其为负样本。

对每个样本点 x_i 加上一个类别标签 y_i ：

$$y_i = \begin{cases} +1 & \text{红色点} \\ -1 & \text{蓝色点} \end{cases}$$

- 超平面方程能够完全正确地对上图的样本点进行分类，假设决策面正好处于间隔区域的中轴线上，并且相应的支持向量对应的样本点到决策面的距离为d，得到如下公式：

$$\begin{cases} \frac{\omega^T x_i + \gamma}{\|\omega\|} \geq d & \forall y_i = 1 \\ \frac{\omega^T x_i + \gamma}{\|\omega\|} \leq -d & \forall y_i = -1 \end{cases}$$

- 上述公式的解释就是，对于所有分类标签为1和-1样本点，它们到直线的距离都大于等于d(支持向量上的样本点到超平面的距离)。公式两边都除以d，就可以得到：

$$\begin{cases} \omega_d^T x_i + \gamma_d \geq 1 & \forall y_i = 1 \\ \omega_d^T x_i + \gamma_d \leq -1 & \forall y_i = -1 \end{cases}$$

- 其中：

$$\omega_d = \frac{\omega}{\|\omega\|d}, \quad \gamma_d = \frac{\gamma}{\|\omega\|d}$$

- 因为 $\|\omega\|$ 和d都是标量。所以上述公式的两个矢量，依然描述一条直线的法向量和截距。
- 因此，对于存在分类间隔的两类样本点，我们一定可以找到一些超平面，使其对于所有的样本点均满足下面的条件：

$$\begin{cases} \omega^T x_i + \gamma \geq 1 & \forall y_i = 1 \\ \omega^T x_i + \gamma \leq -1 & \forall y_i = -1 \end{cases}$$

- 至于为什么会标记y为1或者-1，是因为方便将约束条件变为一个方程：

$$y_i(\omega^T x_i + \gamma) \geq 1 \quad \forall x_i$$

4) 线性SVM优化问题基本描述

- 目标函数

$$d = \frac{|\omega^T x + \gamma|}{\|\omega\|}$$

- 化简：

$$\min \frac{1}{2} \|\omega\|^2$$

- 目标函数和约束条件：

$$\begin{aligned} & \min \frac{1}{2} \|\omega\|^2 \\ \text{s. t. } & y_i(\omega^T x_i + b) \geq 1, i = 1, 2, \dots, n \end{aligned}$$

- n是样本点的总个数，缩写s.t.表示"Subject to"，是"服从某某条件"的意思。上述公式描述的是一个典型的不等式约束条件下的二次型函数优化问题，同时也是支持向量机的基本数学模型。

5) 优化问题--求解准备（凸优化问题）（二次规划问题）

- (1) 通常我们需求解的最优化问题有如下几类：

- 无约束优化问题
- 有等式约束的优化问题
- 有不等式约束的优化问题

- (2) 分别对应的方法：

- 对于第(a)类的优化问题，尝试使用的方法就是费马大定理(Fermat)，即使用求取函数f(x)的导数，然后令其为零，可以求得候选最优值，再在这些候选值中验证；如果是凸函数，可以保证是最优解。这也就是我们高中

经常使用的求函数的极值的方法。

2. 对于第(b)类的优化问题，常常使用的方法就是拉格朗日乘子法（Lagrange Multiplier），即把等式约束 $h_i(x)$ 用一个系数与 $f(x)$ 写为一个式子，称为拉格朗日函数，而系数称为拉格朗日乘子。通过拉格朗日函数对各个变量求导，令其为零，可以求得候选值集合，然后验证求得最优值。
3. 对于第(c)类的优化问题，常常使用的方法就是KKT条件。同样地，我们把所有的等式、不等式约束与 $f(x)$ 写为一个式子，也叫拉格朗日函数，系数也称拉格朗日乘子，通过一些条件，可以求出最优值的**必要条件**，这个条件称为KKT条件。

6) 拉格朗日函数与对偶问题

- 我们要求解的是最小化问题，所以一个直观的想法是如果我能够构造一个函数，使得该函数在可行解区域内与原目标函数完全一致，而在可行解区域外的数值非常大，甚至是无穷大，那么这个**没有约束条件的新目标函数的优化问题就与原来有约束条件的原始目标函数的优化问题是等价的问题**。这就是使用拉格朗日方程的目的，它将约束条件放到目标函数中，从而将有约束优化问题转换为无约束优化问题。
- 需要进行下面二个步骤：
 - 将有约束的原始目标函数转换为无约束的新构造的拉格朗日目标函数
 - 使用拉格朗日对偶性，将不易求解的优化问题转化为易求解的优化
- 进行第一步：**将有约束的原始目标函数转换为无约束的新构造的拉格朗日目标函数**

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^n \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1)$$

- 其中 α_i 是拉格朗日乘子， α_i 大于等于0，是我们构造新目标函数时引入的系数变量(我们自己设置)。现在我们令：

$$\theta(w) = \max_{\alpha_i \geq 0} \mathcal{L}(w, b, \alpha)$$

- 当样本点不满足约束条件时，即**在可行解区域外**：

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) < 1$$

- 当样本点满足约束条件时，即**在可行解区域内**：

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

- 此时，显然 $\theta(w)$ 为原目标函数本身。我们将上述两种情况结合一下，就得到了新的目标函数：

$$\theta(\mathbf{w}) = \begin{cases} \frac{1}{2} \|\mathbf{w}\|^2 & x \in \text{可区域} \\ +\infty & x \in \text{非可行区域} \end{cases}$$

- 目的：就是为了建立一个在可行解区域内与原目标函数相同，在可行解区域外函数值趋近于无穷大的新函数
- 现在，问题变成了求新目标函数的最小值：

$$\min_{w,b} \theta(w) = \min_{w,b} \max_{\alpha_i \geq 0} \mathcal{L}(w, b, \alpha) = p^*$$

- 进行第二步：**将不易求解的优化问题转化为易求解的优化**

- 看一下我们的新目标函数，先求最大值，再求最小值。这样的话，我们首先就要面对带有需要求解的参数 w 和 b 的方程，而 α_i 又是不等式约束，这个求解过程不好做。所以，我们需要使用拉格朗日函数对偶性，将最小和最大的位置交换一下，这样就变成了：

$$\max_{\alpha_i \geq 0} \min_{w,b} \mathcal{L}(w, b, \alpha) = d^*$$

- 交换以后的新问题是原始问题的对偶问题，这个新问题的最优值用 d 来表示。而且 $d \leq p^*$ 。我们关心的是 $d=p$ 的时候，这才是我们要的解。需要什么条件才能让 $d=p$ 呢？
- 首先必须满足这个优化问题是凸优化问题。

- 其次，需要满足KKT条件。

凸优化问题的定义是：**求取最小值的目标函数为凸函数的一类优化问题**。目标函数是凸函数我们已经知道，这个优化问题又是求最小值。所以我们的最优化问题就是凸优化问题。

接下来，就是探讨是否满足KKT条件。

■ 7) KKT条件

一个最优化模型能够表示成下列标准形式：

$$\begin{aligned} & \min f(\mathbf{x}) \\ s.t. \quad & h_j(\mathbf{x}) = 0, j = 1, 2, \dots, p \\ & g_k(\mathbf{x}) \leq 0, k = 1, 2, \dots, q \\ & \mathbf{x} \in X \subset \mathbb{R}^n \end{aligned}$$

- KKT条件的全称是Karush-Kuhn-Tucker条件，KKT条件是说最优值条件必须满足以下条件：

- 条件一：经过拉格朗日函数处理之后的新目标函数 $L(w, b, \alpha)$ 对 w 求导为零；
- 条件二： $h(x) = 0$ ；
- 条件三： $\alpha^*g(x) = 0$ ；

■ 8) 对偶问题求解

- 第一步：

根据上述推导已知：

$$\begin{aligned} & \max_{\alpha_i \geq 0} \min_{w, b} \mathcal{L}(w, b, \alpha) = d^* \\ \mathcal{L}(w, b, \alpha) = & \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i (y_i (w^T \mathbf{x}_i + b) - 1) \end{aligned}$$

首先固定 α ，要让 $L(w, b, \alpha)$ 关于 w 和 b 最小化，我们分别对 w 和 b 偏导数，令其等于0，即：

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w} = 0 & \Rightarrow w = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \Rightarrow \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

将上述结果带回 $L(w, b, \alpha)$ 得到：

$$\begin{aligned}
\mathcal{L}(\mathbf{w}, b, \alpha) &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^n \alpha_i [y_i (\mathbf{w}^\top \mathbf{x}_i + b) - 1] \\
&= \frac{1}{2} \mathbf{w}^\top \mathbf{w} - \mathbf{w}^\top \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i - b \sum_{i=1}^n \alpha_i y_i + \sum_{i=1}^n \alpha_i \\
&= \frac{1}{2} \mathbf{w}^\top \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i - \mathbf{w}^\top \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i - b \cdot 0 + \sum_{i=1}^n \alpha_i \\
&= \sum_{i=1}^n \alpha_i - \frac{1}{2} \left(\sum_{i=1}^n \alpha_i y_i \mathbf{x}_i \right)^\top \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i \\
&= \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j
\end{aligned}$$

从上面的最后一个式子，可以看出，此时的 $\mathcal{L}(w, b, \alpha)$ 函数只含有一个变量，即 α_i 。

■ 第二步：

现在内侧的最小值求解完成，我们求解外侧的最大值，从上面的式子得到：

$$\begin{aligned}
&\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \\
&s.t. \quad \alpha_i \geq 0, i = 1, 2, \dots, n \\
&\quad \sum_{i=1}^n \alpha_i y_i = 0
\end{aligned}$$

现在我们的优化问题变成了如上的形式。对于这个问题，我们有更高效的优化算法，即序列最小优化(SMO)算法。我们通过这个优化算法能得到 α ，再根据 α ，我们就可以求解出 w 和 b ，进而求得我们最初的目的：找到超平面，即“决策平面”。

SMO算法

上述数学推导找到了可以用SMO算法求解的目标函数

(1) Platt的SMO算法

SM表示序列最小化，Platt的SMO算法是将大优化问题分解为多个小优化问题来求解的。这些小优化问题往往很容易求解，并且对它们进行顺序求解的结果与将它们作为整体来求解的结果完全一致的。在结果完全相同的同时，SMO算法的求解时间短很多。

SMO算法的目标是求出一系列alpha和b，一旦求出了这些alpha，就很容易计算出权重向量w并得到分隔超平面。

SMO算法的工作原理是：每次循环中选择两个alpha进行优化处理。一旦找到了一对合适的alpha，那么就增大其中一个同时减小另一个。这里所谓的“合适”就是指两个alpha必须符合以下两个条件，条件之一就是两个alpha必须要在间隔边界之外，而且第二个条件则是这两个alpha还没有进行过区间化处理或者不在边界上。

(2) SMO算法的解法

先来定义特征到结果的输出函数为：

$$u = \boldsymbol{\omega}^T \mathbf{x} + b$$

接着，我们回忆一下原始优化问题，如下：

$$\begin{aligned} & \min \frac{1}{2} \|\boldsymbol{\omega}\|^2 \\ \text{s.t. } & y_i (\boldsymbol{\omega}^T \mathbf{x}_i + b) \geq 1, i = 1, 2, \dots, n \end{aligned}$$

求导得：

$$\boldsymbol{\omega} = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i$$

将上述公式带入输出函数中：

$$u = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

与此同时，拉格朗日对偶后得到最终的目标化函数：

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

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

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

将目标函数变形，在前面增加一个符号，将最大值问题转换成最小值问题：

$$1 \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \sum_{i=1}^n \alpha_i$$

$$s.t. \quad \alpha_i \geq 0, \quad i = 1, 2, \dots, n$$

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

实际上，对于上述目标函数，是存在一个假设的，即数据100%线性可分。但是，目前为止，我们知道几乎所有数据都不那么"干净"。这时我们就可以通过引入所谓的松弛变量 ξ (slack variable)和惩罚参数C，来允许有些数据点可以处于超平面的错误的一侧。此时我们的约束条件有所改变：

$$s.t. \quad C \geq \alpha_i \geq 0, \quad i = 1, 2, \dots, n$$

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

同时，考虑到松弛变量和惩罚参数C，目标函数变为：

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

原始问题的拉格朗日函数变为：

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

对偶问题拉格朗日函数的极大极小问题，得到以下等价优化问题：

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

则，原始问题的解对偶问题的解相同需要满足KKT对偶互补条件，即：

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

$$\mu_i \xi_i = 0 \quad (2)$$

对样本点 x_i ，记SVM的输出结果为：

$$u_i = w \cdot x_i + b$$

Platt在序列最小优化 (SMO) 方法1中提到，对正定二次优化问题 (a positive definite QP problem) 的优化点的充分必要条件为KKT条件 (Karush-Kuhn-Tucker conditions)。

对于所有的*i*，若满足以下条件，QP问题可解。KKT条件如下：

$$\alpha_i = 0 \Leftrightarrow y_i u_i \geq 1 \quad (3)$$

$$0 < \alpha_i < C \Leftrightarrow y_i u_i = 1 \quad (4)$$

$$\alpha_i = C \Leftrightarrow y_i u_i \leq 1 \quad (5)$$

其中 $y_i u_i$ 就是每个样本点的**函数间隔**。

现在，让我们梳理下SMO算法的步骤：

- 步骤1：计算误差：

$$E_i = f(\mathbf{x}_i) - y_i = \sum_{j=1}^n \alpha_j y_j \mathbf{x}_i^T \mathbf{x}_j + b - y_i$$

- 步骤2：计算上下界L和H：

$$\begin{cases} L = \max(0, \alpha_j^{old} - \alpha_i^{old}), H = \min(C, C + \alpha_j^{old} - \alpha_i^{old}) & \text{if } y_i \neq y_j \\ L = \max(0, \alpha_j^{old} + \alpha_i^{old} - C), H = \min(C, \alpha_j^{old} + \alpha_i^{old}) & \text{if } y_i = y_j \end{cases}$$

- 步骤3：计算 η ：

$$\eta = \mathbf{x}_i^T \mathbf{x}_i + \mathbf{x}_j^T \mathbf{x}_j - 2\mathbf{x}_i^T \mathbf{x}_j$$

- 步骤4：更新 α_j ：

$$\alpha_j^{new} = \alpha_j^{old} + \frac{y_j(E_i - E_j)}{\eta}$$

- 步骤5：根据取值范围修剪 α_j ：

$$\alpha^{new,clipped} = \begin{cases} H & \text{if } \alpha_2^{new} \geq H \\ \alpha_2^{new} & \text{if } L \leq \alpha_2^{new} \leq H \\ L & \text{if } \alpha_2^{new} \leq L \end{cases}$$

- 步骤6：更新 α_i ：

$$\alpha_i^{new} = \alpha_i^{old} + y_i y_j (\alpha_j^{old} - \alpha_j^{new,clipped})$$

- 步骤7：更新 b_1 和 b_2 ：

$$b_1^{new} = b^{old} - E_i - y_i (\alpha_i^{new} - \alpha_i^{old}) \mathbf{x}_i^T \mathbf{x}_i - y_j (\alpha_j^{new} - \alpha_j^{old}) \mathbf{x}_j^T \mathbf{x}_i$$

$$b_2^{new} = b^{old} - E_j - y_i (\alpha_i^{new} - \alpha_i^{old}) \mathbf{x}_i^T \mathbf{x}_j - y_j (\alpha_j^{new} - \alpha_j^{old}) \mathbf{x}_j^T \mathbf{x}_j$$

- 步骤8：根据 b_1 和 b_2 更新 b ：

$$b = \begin{cases} b_1 & 0 < \alpha_1^{new} < C \\ b_2 & 0 < \alpha_2^{new} < C \\ \frac{b_1 + b_2}{2} & otherwise \end{cases}$$

SVM处理非线性

概念引入

- 正则化两种情况
 - 让有些没有解的情况变成有解
 - 或者是得出的解非常凹凸不平，或者函数的模样不是我们需要的
- 改造：
 - C -- 事先设定的参数（会给出范围）不断尝试，来找到最好的

SVM 处理非线性

① 最小化: $\frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i$ 松弛变量 (Slack Variable)

限制条件: ① $y_i [w^T x_i + b] \geq 1 - \xi_i \quad (i=1 \sim N)$

② $\xi_i \geq 0$

低维到高维

(0,1) $\xrightarrow{(1,1)} X$ (异或问题)

$\xrightarrow{(0,0) \quad (1,0)}$

$x_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \in C_1 \quad x_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \in C_1$

$x_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \in C_2 \quad x_4 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in C_2$

$\varphi(x): x = \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\varphi} \varphi(x) = \begin{bmatrix} a^2 \\ b^2 \\ a \\ b \\ ab \end{bmatrix}$

$\varphi(x_1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \varphi(x_2) = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \in C_1 \quad w = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix}$

$\varphi(x_3) = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \varphi(x_4) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in C_2$

$$\begin{aligned}
 W^T \varphi(x_1) + b &= 1 \in C_1 \\
 W^T \varphi(x_2) + b &= 3 \in C_1 \\
 W^T \varphi(x_3) + b &= -1 \in C_2 \\
 W^T \varphi(x_4) + b &= -1 \in C_2
 \end{aligned}$$

所以刚好可以把不同类分开

核函数

如何找到 $\phi(x)$

$\phi(x)$ 是无限维

但是SVM使用有限维的手段来处理无限维：我们可以不知道无限维映射的显示表达，我们只要知道一个核函数

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2)$$

$K(\cdot)$ 是一个数，函数 $\phi(x)$ 是一个无限维度的函数，但是行列相乘得到的是一个数。那么这个优化问题是可解的。所以我们只需要知道 K ，并不需要知道函数的显示表达是什么。但是 K 也要满足一些条件：

$K(x_1, x_2)$ 能写成 $\phi(x_1)^T \phi(x_2)$ 的充要条件：

$$K(x_1, x_2) = K(x_2, x_1) \quad (\text{交换性})$$

$$\forall C_i, X_i (i = 1, N) \text{ 有} : \quad \sum_{i=1}^N \sum_{j=1}^N C_i C_j K(x_i, x_j) \geq 0 \quad (\text{半正定性})$$

常见核函数

核函数

$$\textcircled{1} \quad K(x_1, x_2) = e^{-\frac{\|x_1 - x_2\|^2}{2\sigma^2}} \quad (\text{高斯核})$$

$$= \varphi(x_1)^T \varphi(x_2)$$

$$\textcircled{2} \quad K(x_1, x_2) = (x_1^T x_2 + 1)^d \quad \text{多项式阶数}$$

2. 原问题和对偶问题

公式推导如图：

Hard-Margin SVM

Data = $\{(x_i, y_i)\}_{i=1}^N, x_i \in \mathbb{R}^p, y_i \in \{+1, -1\}$

$$\begin{cases} \min_{w, b} \frac{1}{2} w^T w \\ \text{s.t. } y_i(w^T x_i + b) \geq 1 \end{cases} \quad \rightarrow \text{primal problem.}$$

$\sum_{i=1}^N$

$$L(w, b, \lambda) = \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i (1 - y_i(w^T x_i + b))$$

$\geq 0 \quad \leq 0$

无约束

$$\begin{cases} \min_{w, b} \max_{\lambda} L(w, b, \lambda) \\ \text{s.t. } \lambda_i \geq 0 \end{cases}$$

对偶

$$\begin{cases} \max_{\lambda} \min_{w, b} L(w, b, \lambda) \\ \text{s.t. } \lambda_i \geq 0 \end{cases}$$

$$\begin{cases} \max_{\lambda} -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i \\ \text{s.t. } \lambda_i \geq 0, \sum_{i=1}^N \lambda_i y_i = 0 \end{cases}$$

$$\min_{w, b} L(w, b, \lambda)$$

$$\frac{\partial L}{\partial b} = \frac{\partial}{\partial b} \left[\sum_{i=1}^N \lambda_i - \sum_{i=1}^N \lambda_i y_i (w^T x_i + b) \right]$$

$$= \frac{\partial}{\partial b} \left[- \sum_{i=1}^N \lambda_i y_i b \right]$$

$$= - \sum_{i=1}^N \lambda_i y_i \triangleq 0 \quad \text{将其代入 } L(w, b, \lambda).$$

$$\begin{aligned} L(w, b, \lambda) &= \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i - \sum_{i=1}^N \lambda_i y_i (w^T x_i + b) \\ &= \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i - \sum_{i=1}^N \lambda_i y_i w^T x_i + \sum_{i=1}^N \lambda_i y_i b \\ &= \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i - \sum_{i=1}^N \lambda_i y_i w^T x_i \end{aligned}$$

$$\frac{\partial L}{\partial w} = \frac{1}{2} \cdot 2 \cdot w - \sum_{i=1}^N \lambda_i y_i x_i \triangleq 0 \Rightarrow w = \sum_{i=1}^N \lambda_i y_i x_i$$

$$\begin{aligned} L(w, b, \lambda) &= \frac{1}{2} \left(\sum_{i=1}^N \lambda_i y_i x_i \right)^T \left(\sum_{j=1}^N \lambda_j y_j x_j \right) - \sum_{i=1}^N \lambda_i y_i \left(\sum_{j=1}^N \lambda_j y_j x_j \right)^T x_i + \sum_{i=1}^N \lambda_i \\ &\quad \left(\sum_{i=1}^N \lambda_i y_i x_i \right) \left(\sum_{j=1}^N \lambda_j y_j x_j \right) = \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j \\ &= -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i \end{aligned}$$

(x_i, y_i)

① 若 $1 - y_i(w^T x_i + b) > 0$

$$\max_{\lambda} L(\lambda, w, b) = \frac{1}{2} w^T w + \infty = \infty$$

② 若 $1 - y_i(w^T x_i + b) \leq 0$

$$\max_{\lambda} L(\lambda, w, b) = \frac{1}{2} w^T w + 0 = \frac{1}{2} w^T w$$

$$\min_{w, b} \max_{\lambda} L = \min_{w, b} \frac{1}{2} w^T w$$

$$\textcircled{12} \rightarrow \min_{w, b} \max_{\lambda} L = \min_{w, b} \left(\infty, \frac{1}{2} w^T w \right)$$

$$= \min_{w, b} \frac{1}{2} w^T w$$

$$\min \max L \geq \max \min L \quad \text{弱对偶关系.}$$

$$\min \max L = \max \min L \quad \text{强对偶关系}$$

求 $\min_{w, b} L$

$$\begin{cases} \frac{\partial L}{\partial w} \triangleq 0 \Rightarrow w = \sum_{i=1}^N \lambda_i y_i x_i \\ \frac{\partial L}{\partial b} \triangleq 0 \Rightarrow \sum_{i=1}^N \lambda_i y_i = 0 \end{cases}$$

优化理论

几本书的推荐

① <Convex optimization>

Stephen Boyd

凸优化

② <Nonlinear Programming>

■ 原问题

原问题 (Prime Problem) → (非常普遍)

最小化: $f(w)$

限制条件: $g_i(w) \leq 0 \quad (i=1 \sim K)$

$h_i(w) = 0 \quad (i=1 \sim N)$

■ 对偶问题

对偶问题 (Dual Problem)

① 定义: $L(w, \alpha, \beta)$

$$= f(w) + \sum_{i=1}^K \alpha_i g_i(w) + \sum_{i=1}^M \beta_i h_i(w)$$

$$= f(w) + \alpha^T g(w) + \beta^T h(w)$$

② 对偶问题定义

最大化: $\Theta(\alpha, \beta) = \underset{\text{所有 } w}{\inf} \left\{ L(w, \alpha, \beta) \right\}$ 求最小值

限制条件: $\alpha_i \geq 0 \quad (i=1 \sim K)$

定理：如果 w^* 是原问题的解，
而 α^*, β^* 是对偶问题的解，则有：

$$f(w^*) \geq \theta(\alpha^*, \beta^*)$$

$$\begin{aligned} \theta(\alpha^*, \beta^*) &= \inf_{\{w \text{ 可行}\}} L(w, \alpha^*, \beta^*) \\ &\leq L(w^*, \alpha^*, \beta^*) \\ &= f(w^*) + \sum_{i=1}^k \alpha_i^* g_i(w^*) + \sum_{i=1}^m \beta_i^* h_i(w^*) \\ &\quad \geq 0 \leq 0 \\ &\quad \leq 0 \quad = 0 \\ &\leq f(w^*) \end{aligned}$$

一个定义

定义: $G = f(w^*) - \Theta(\alpha^*, \beta^*) \geq 0$
 G 叫做原问题与对偶问题的间距 (Duality Gap)

(对于某些特定优化问题, 可以证明
 $G = 0$)

- 基于这个定义下, 上述推导成立的条件:

对 $\forall i=1 \sim K$, (KKT 条件)

或者 $\alpha_i^* = 0$

或者 $g_i^*(w^*) = 0$

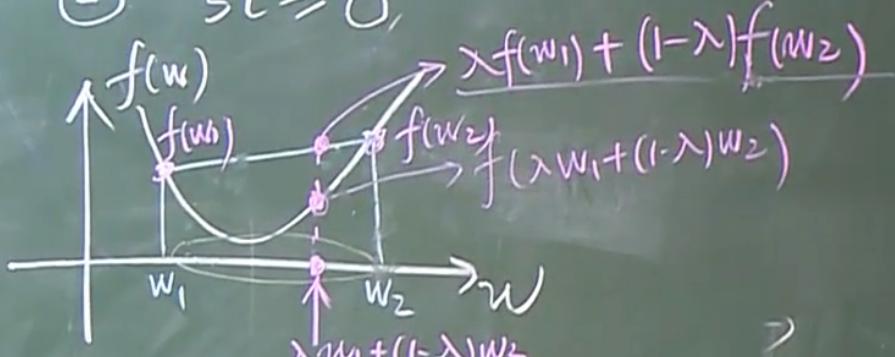
支持向量机的原问题化为对偶问题

- 凸函数的定义

$$\text{最小化: } \frac{\frac{1}{2}\|w\|^2 + C \sum_{i=1}^N \xi_i}{\text{(凸函数)}}$$

$$\text{限制条件: } ① y_i [w^T \varphi(x_i) + b] \geq 1 - \xi_i$$

$$② \xi_i \geq 0$$



$$\forall w_1, w_2; \forall \lambda \in [0, 1]$$

$$f(\lambda w_1 + (1-\lambda)w_2) \leq \lambda f(w_1) + (1-\lambda)f(w_2)$$

① 原问题：

最小化 $f(w)$

限制条件： $g_i(w) \leq 0 \quad (i=1 \dots K)$

$h_i(w) = 0 \quad (i=1 \dots M)$

② 对偶问题：

$$L(w, \alpha, \beta) = f(w) + \sum_{i=1}^K \alpha_i g_i(w) + \sum_{j=1}^M \beta_j h_j(w)$$

$$\text{最大化 } \theta(\alpha, \beta) = \inf_{\{w \in \mathbb{R}^N\}} L(w, \alpha, \beta)$$

限制条件： $\alpha_i \geq 0 \quad (i=1 \dots K)$

③ KKT条件： $\forall i = 1 \dots K, \alpha_i^* = 0$ 或 $g_i^*(w^*) = 0$

↓

$$\text{最小化: } \frac{1}{2} \|w\|^2 - C \sum_{i=1}^N \xi_i \quad (\text{凸函数})$$

限制条件：① $1 + \xi_i - y_i w^T \phi(x_i) - b \leq 0$

② $\xi_i \geq 0 \quad (i=1 \dots N)$

对偶问题：

$$\text{最大化 } \theta(\alpha, \beta) = \inf \left(\frac{1}{2} \|w\|^2 - C \sum_{i=1}^N \xi_i + \sum_{i=1}^N \xi_i + \sum_{i=1}^N \alpha_i [1 + \xi_i - y_i w^T \phi(x_i) - b] \right)$$

限制条件： $\alpha_i \geq 0 \quad (i=1 \dots N)$

$\beta_i \geq 0$

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^N \alpha_i y_i \phi(x_i)$$

=>

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow \beta_i + \alpha_i = C$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^N \alpha_i y_i = 0$$

总之就是把phi函数表达成K函数，最终化简得：

也就是SMO算法的推导

如何化为对偶问题

对 (ω, b, δ_i) 求导并令导数为0

$$(1) \frac{\partial \theta}{\partial \omega} = \omega - \sum_{i=1}^N \alpha_i \varphi(X_i) y_i = 0 \implies \omega = \sum_{i=1}^N \alpha_i y_i \varphi(X_i)$$

$$(2) \frac{\partial \theta}{\partial \delta_i} = -C + \alpha_i + \beta_i = 0 \implies \alpha_i + \beta_i = C$$

$$(3) \frac{\partial \theta}{\partial b} = -\sum_{i=1}^N \alpha_i y_i = 0 \implies \sum_{i=1}^N \alpha_i y_i = 0$$

将这三个式子代入

将支持向量机的原问题化为对偶问题：

$$\text{最大化: } \theta(\alpha, \beta) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \varphi(X_i)^T \varphi(X_j)$$

$$\text{限制条件: (1) } 0 \leq \alpha_i \leq C, (i = 1 \sim N)$$

$$(2) \sum_{i=1}^N \alpha_i y_i = 0, (i = 1 \sim N)$$

如何求b

$$\text{由于 } \omega = \sum_{j=1}^N \alpha_j y_j \varphi(X_j)$$

$$\begin{aligned} \text{则 } \omega^T \varphi(X_i) &= \sum_{j=1}^N \alpha_j y_j \varphi(X_j)^T \varphi(X_i) \\ &= \sum_{j=1}^N \alpha_j y_j K(X_j, X_i) \end{aligned}$$

$$\varphi(X_i)^T \varphi(X_j) = K(X_i, X_j)$$

总结

SVM算法：

① 训练流程：

输入 $\{(x_i, y_i)\}_{i=1 \sim N}$

(解优化问题)

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

限制条件：
① $0 \leq \alpha_i \leq C$
② $\sum_{i=1}^N \alpha_i y_i = 0$ (SMO算法)

算 b ，找一个 $0 < \alpha_i < C$ ，
 $b = \frac{1 - y_i \sum_{j=1}^N \alpha_j y_j K(x_i, x_j)}{y_i}$

测试流程

输入测试样本 X

$\begin{cases} \text{若 } \sum_{i=1}^N \alpha_i y_i K(x_i, X) + b \geq 0, \text{ 则 } y = +1 \\ \text{若 } \sum_{i=1}^N \alpha_i y_i K(x_i, X) + b < 0, \text{ 则 } y = -1 \end{cases}$

只出现了K函数。

一般用混淆矩阵来判断系统的性能

		预测	
		正样本	负样本
实际	正样本	TP (2249)	FN (39)
	负样本	FP (51)	TN (20717)

一般改变阈值就会得到：

支持向量机的判别公式：

系统二

$$\begin{cases} \text{若 } \sum_{i=1}^N \alpha_i y_i K(X_i, X) + b > \text{正数}, \text{ 则 } y = +1 \text{ (判为正样本)} \\ \text{若 } \sum_{i=1}^N \alpha_i y_i K(X_i, X) + b < \text{正数}, \text{ 则 } y = -1 \text{ (判为负样本)} \end{cases}$$

TP ↗ FP ↗

多类问题

1. 1类 对 K-1类

假设总共有K类，我们需要构造K个支持向量机模型。

(1) 类别1 VS 类别2, 3, 4 … K

(2) 类别2 VS 类别1, 3, 4 … K

(3) 类别3 VS 类别1, 2, 4 … K

……

(K) 类别K VS 类别1, 2, 3, … K-1

2. 1类 VS 另一类

假设有三类，那么我们就构建三个支持向量机分类器。

(1) 类别1 VS 类别2

(2) 类别1 VS 类别3

(3) 类别2 VS 类别3

投票