

SVM

整理：卿来云

SVM

1. SVM基本型

1.1 最大间隔 (Maximize Margin)

1.2 SVM的对偶问题

2 带松弛因子的SVM (C-SVM)

3. 合页损失函数 (Hinge Loss)

4 核方法

4.1 核技巧

4.2 核函数构造

5 SVM优化求解: SMO

5.1 SMO算法原理: 获得没有修剪的原始解

5.2 对原始解进行修剪

5.3 α 选择

5.4 更新截距项 b

5.5 SMO小结

6 支持向量回归 (SVR)

6.1 ϵ 不敏感损失函数

6.2 SVR

7 Scikit learn 中的SVM实现

8 SVM 算法小结

支持向量机 (support vector machine, SVM) 最初用于两类分类任务，后来通过核方法扩展到非线性模型，通过改变损失函数扩展到回归任务。同Logistic回归直接从后验概率出发不同，SVM从几何的角度出发，其基本模型定义为特征空间上的间隔最大的线性分类器，其学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。SVM亦可放到机器学习的一般框架，通过目标函数和正则项定义目标函数，也存在模型选择/超参数调优问题。

1. SVM基本型

1.1 最大间隔 (Maximize Margin)

理解SVM，我们先看看线性分类器和间隔的概念。

考虑一个两类的分类问题，数据点用 \mathbf{x} 表示，为一个 D 维向量；类别用 y 表示，可以取值为1或者-1（有些地方会选0和1，其实分类问题选什么都无所谓，只要是两个不同的数字即可，不过这里选择+1和-1是为了方便SVM的推导），分别代表两个不同的类。一个线性分类器的学习目标就是要在 D 维的数据空间中找到一个分类超平面，其方程可以表示为：

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

在Logistic回归中，分类判别函数为： $a(\mathbf{x}) = \ln \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})} = \mathbf{w}^T \mathbf{x}$ （截距项通常单独处理，在此省略），其中 $p(y=1|\mathbf{x}) = \sigma(a(\mathbf{x})) = \frac{1}{1+e^{-a(\mathbf{x})}} = \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}}}$ ，
 $p(y=0|\mathbf{x}) = 1 - \sigma(a(\mathbf{x})) = \frac{e^{-a(\mathbf{x})}}{1+e^{-a(\mathbf{x})}} = \frac{e^{-\mathbf{w}^T \mathbf{x}}}{1+e^{-\mathbf{w}^T \mathbf{x}}}$

当 $a > 0$, $p(y=1|\mathbf{x}) > p(y=0|\mathbf{x})$, \mathbf{x} 的类别 $y = 1$ ；

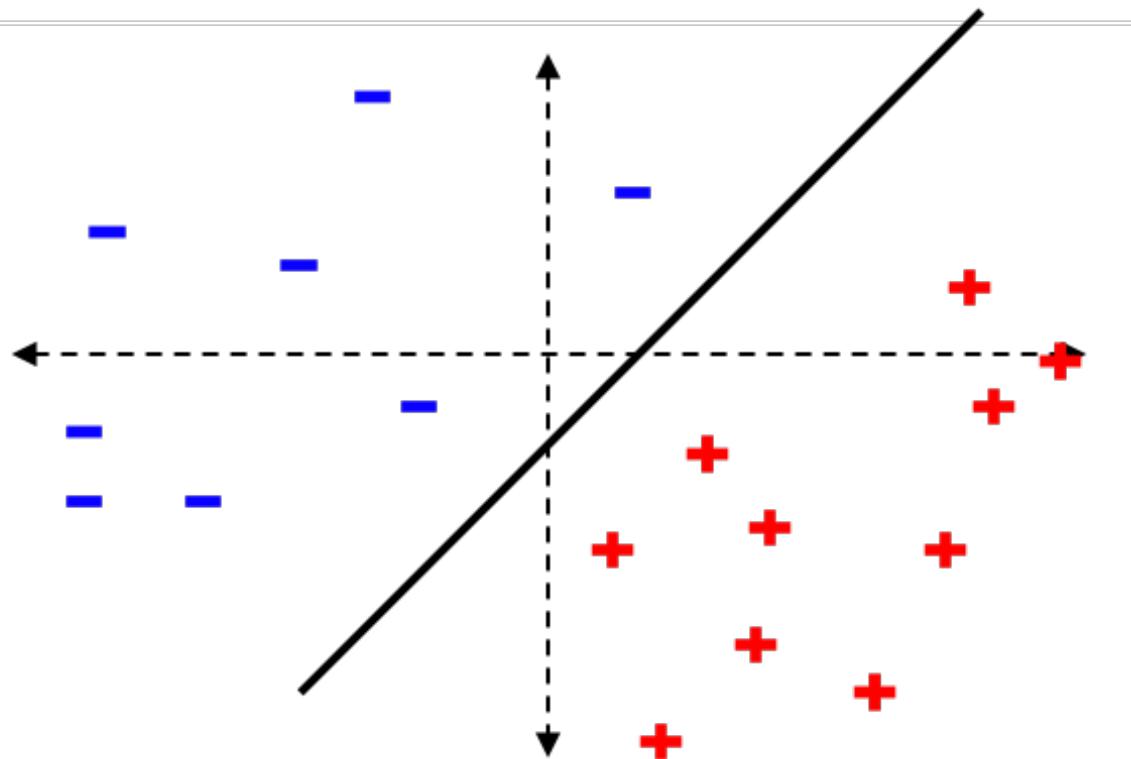
当 $a < 0$, $p(y=1|\mathbf{x}) < p(y=0|\mathbf{x})$, \mathbf{x} 的类别取 $y = 0$ ；

当 $a = 0$, $p(y=1|\mathbf{x}) = p(y=0|\mathbf{x})$, \mathbf{x} 的类别 $y = 0$ 和 $y = 1$ 的概率相等，此时 \mathbf{x} 位于决策面上。

所以Logistic回归是从概率出发的判别式分类器，SVM直接分类的判别函数出发的判别式分类器。在SVM中，直接根据判别函数的符号判断类别，所以用1和-1表示类别更方便。

例：二维空间中线性分类

假定现在有一个如下图所示的二维平面（每个样本有两维特征），平面上有两种不同的点，一种为黑色的-，另一种为黑色的+。我们要在这个二维平面上找到一个可行的超平面（在二维空间中，超平面就是一条直线），那么这个超平面可以是下图中那根黑色的线。从图中我们可以看出，以这条黑色的线为界，将红色+和黑色的一分开，在超平面一边的数据点所对应的y全是-1（黑色的-），而在另一边全是1（红色的+）。



令分类判别函数为

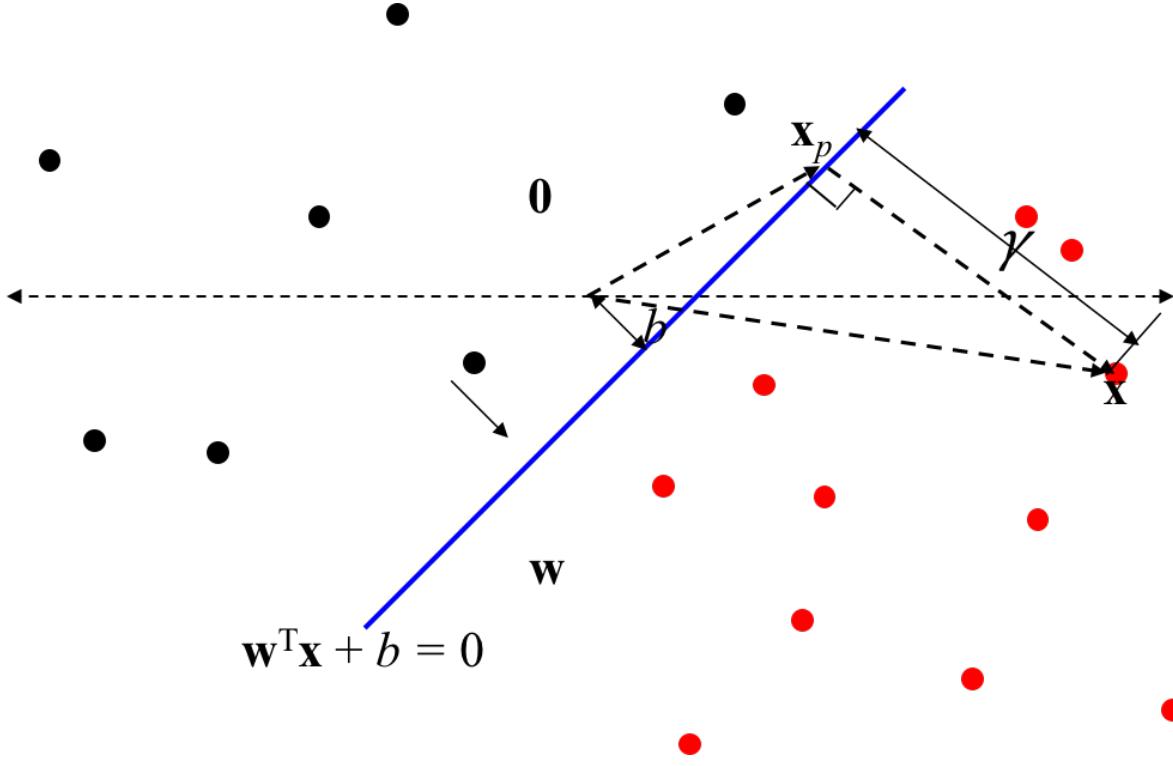
$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

显然，如果 $f(\mathbf{x}) = 0$ ，那么 \mathbf{x} 是位于超平面上的点。我们不妨要求对于所有满足 $f(\mathbf{x}) < 0$ 的点，其对应的 y 等于-1； $f(\mathbf{x}) > 0$ ，对应 $y = 1$ 的数据点。

下面我们先假设数据是线性可分的，即假设存在这样一个超平面，超平面的每一侧都是同一类样本。

一个点距离超平面的远近可以表示为分类预测的确信或准确程度。在超平面 $\mathbf{w}^T \mathbf{x} + b = 0$ 确定的情况下， $|\mathbf{w}^T \mathbf{x} + b|$ 可表示点 \mathbf{x} 到距离超平面的距离，而 $\mathbf{w}^T \mathbf{x} + b$ 的符号与类别标签 y 的符号是否一致表示分类是否正确，所以，可以用 $y(\mathbf{w}^T \mathbf{x} + b)$ 表示分类的正确性和确信度。

由此，我们引出样本到分类决策面的距离。



如上图所示，对于一个点 \mathbf{x} ，令其垂直投影到分类决策超平面上的对应的为 \mathbf{x}_p ， \mathbf{w} 是垂直于超平面的一个向量， γ 为样本 \mathbf{x} 到分类决策面的距离，则根据平面几何知识，有

$$\mathbf{x} = \mathbf{x}_p + \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|_2}$$

其中 $\|\mathbf{w}\|_2$ 表示 \mathbf{w} 的 L2 范数（模长）， $\frac{\mathbf{w}}{\|\mathbf{w}\|_2}$ 是单位向量（一个向量除以它的模称之为单位向量）。

又由于 \mathbf{x}_p 是超平面上的点，满足 $f(\mathbf{x}_p) = 0$ ，即

$$f(\mathbf{x}_p) = \mathbf{w}^T \mathbf{x}_p + b = 0$$

计算点 \mathbf{x} 对应的 $f(\mathbf{x})$ 为

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\ &= \mathbf{w}^T (\mathbf{x}_p + \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|_2}) + b \\ &= \mathbf{w}^T \mathbf{x}_p + b + \mathbf{w}^T \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|_2} \\ &= f(\mathbf{x}_p) + \gamma \frac{\mathbf{w}^T \mathbf{w}}{\|\mathbf{w}\|_2} \\ &= 0 + \gamma \frac{\|\mathbf{w}\|_2^2}{\|\mathbf{w}\|_2} \\ &= \gamma \|\mathbf{w}\|_2 \end{aligned}$$

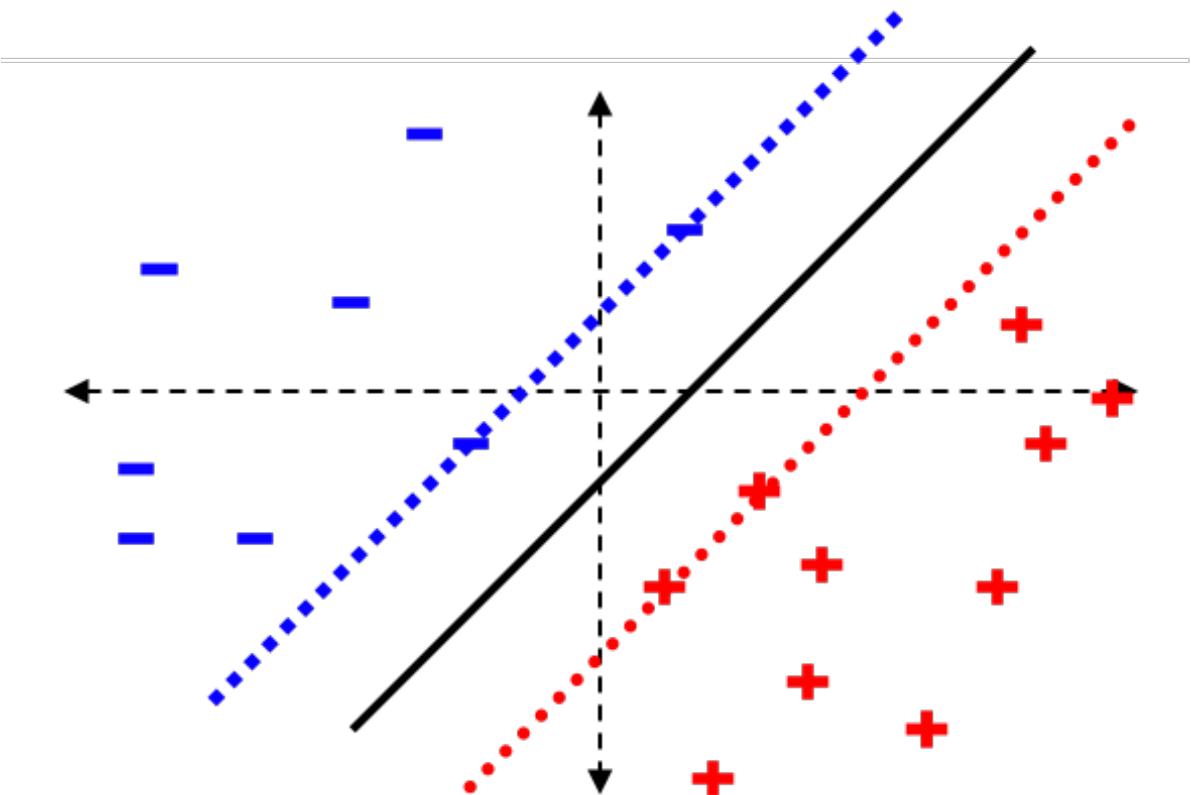
所以对于一个点 \mathbf{x} 到分类决策超平面 $\mathbf{w}^T \mathbf{x} + w_0 = 0$ 的距离 γ 为

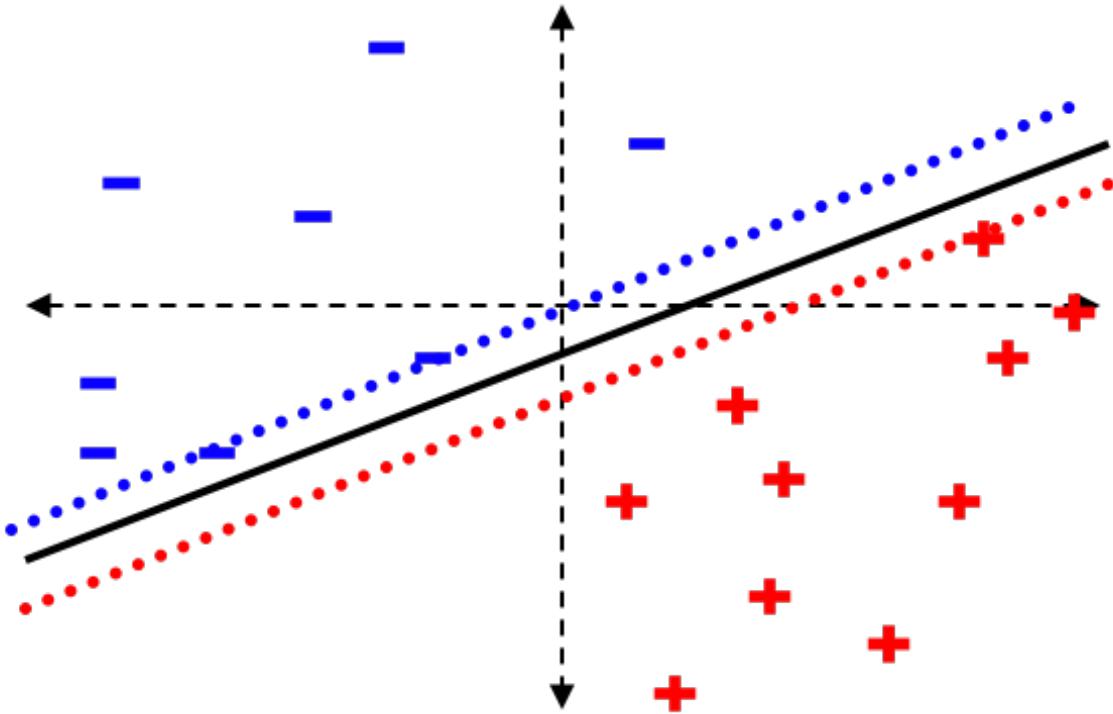
$$\gamma = \frac{f(\mathbf{x})}{\|\mathbf{w}\|_2}$$

原点 $\mathbf{x} = \mathbf{0}$ 到超平面的距离为

$$\gamma_0 = \frac{f(\mathbf{0})}{\|\mathbf{w}\|_2} = \frac{b}{\|\mathbf{w}\|_2}$$

由上，我们已经知道，一个点 \mathbf{x} 到分类决策超平面 $\mathbf{w}^T \mathbf{x} + b = 0$ 的距离 γ 的绝对值（或 $\frac{y f(\mathbf{x})}{\|\mathbf{w}\|_2}$ ）越大，分类的置信度越大。对于一个包含 N 个点的数据集，我们很自然地定义它的希望每个类别到决策超平面的距离越大越好。如下图中两个不同的决策超平面，每个类别到第一个超平面的最小距离比到第二个超平面的最小距离大，我们期望第一个超平面的分类效果更好。因为第一个超平面能将两类分得更开，这样即使测试样本在训练样本的基础上有一些小的变化，这个超平面仍能将两类分开，即泛化能力好。





因此我们定义间隔 (margin) 为所有 N 个点中到分类决策面的最小距离为

$$\tilde{\gamma} = \min(\gamma_i), \\ \text{where } \gamma_i = \frac{f(\mathbf{x}_i)}{\|\mathbf{w}\|_2}, i = 1, \dots, N$$

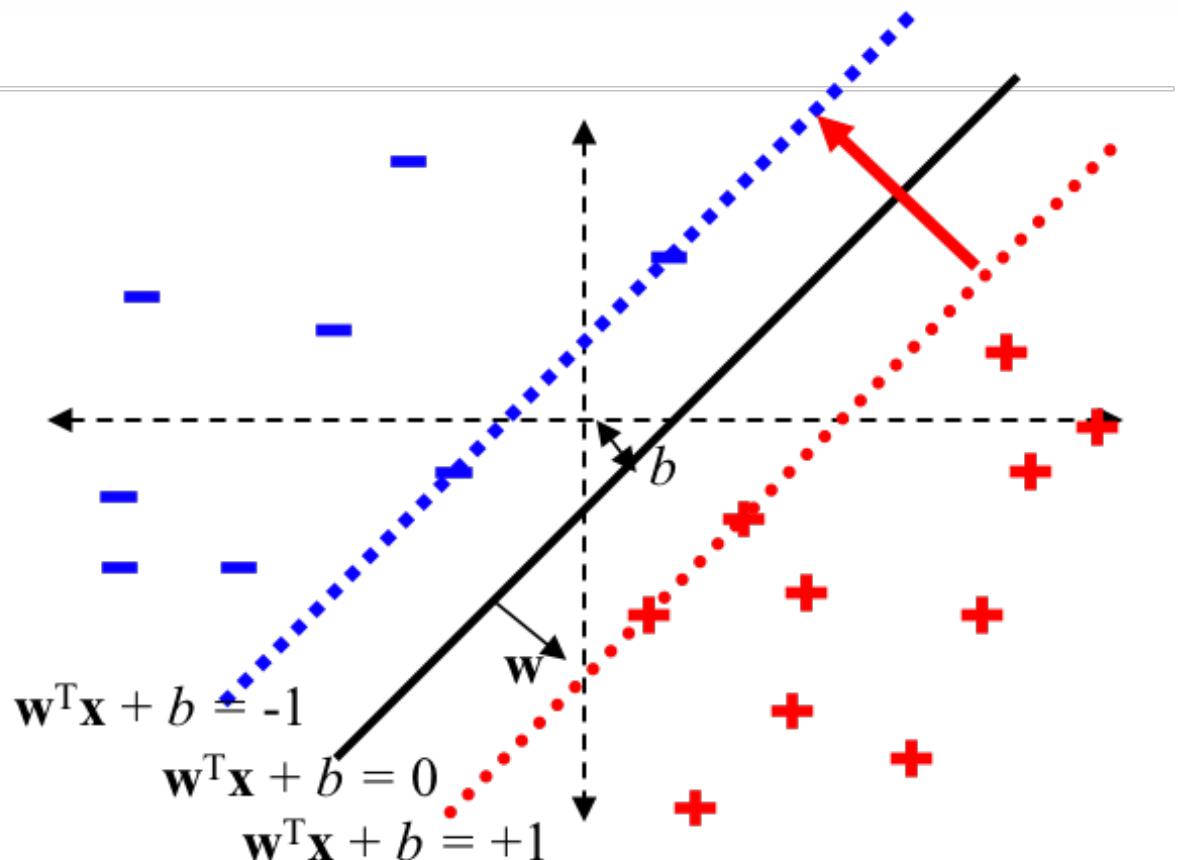
为了使得分类的置信度高，我们希望所选择的超平面能够最大化 $\tilde{\gamma}$ (margin)。为了使得 margin 最大，分类超平面应该位于两类的正中间。由于 $y \in \{-1, 1\}$ ，我们令离超平面最近点的 $f(\mathbf{x}) = 1$ (正类，对应下面中红色虚线上的点) 或 $f(\mathbf{x}) = -1$ (负类，对应下面中蓝色虚线上的点)，这些点被称为支持向量 (Support Vectors)。关于为什么叫支持向量请参见SVM求解的KKT条件。

将 $f(\mathbf{x}) = 1$ 代入距离计算公式，得到

$$\tilde{\gamma} = \frac{f(\mathbf{x})}{\|\mathbf{w}\|_2} = \frac{1}{\|\mathbf{w}\|_2}$$

考虑正负两类样本，我们用正类样本到超平面的最短距离与负类样本到超平面的最短距离之和 (红色虚线和蓝色虚线之间的距离，红色实线段长度) 表示分类超平面的margin，即

$$\tilde{\gamma} = \frac{2}{\|\mathbf{w}\|_2}$$



因此我们最大间隔分类器（maximum margin classifier）的目标函数为：

$$\begin{aligned} & \max_{\mathbf{w}, b} \frac{2}{\|\mathbf{w}\|_2} \\ \text{s.t. } & y_i(\mathbf{w}^T + b) \geq 1, i = 1, \dots, N \end{aligned}$$

| s.t., 即subject to, 表示约束条件。

上述最大化优化问题等价于下面的最小化问题

$$\begin{aligned} & \min_{\mathbf{w}, w_0} \frac{\|\mathbf{w}\|_2}{2} \\ \text{s.t. } & y_i(\mathbf{w}^T + b) \geq 1, i = 1, \dots, N \end{aligned}$$

上述问题进一步等价于

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t. } & y_i(\mathbf{w}^T + b) \geq 1, i = 1, \dots, N \end{aligned}$$

这就是SVM的基本型。

1.2 SVM的对偶问题

SVM基本型的目标函数

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ s.t. \quad & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, i = 1, \dots, N \end{aligned}$$

是一个凸二次规划 (convex quadratic programming) 问题，可以用先吃的优化计算包求解。不过我们可以有更高效的方法。

一个有 n 个变数与 m 个限制的二次规划问题可以用以下的形式描述。首先给定：

- 一个 n 维的向量 \mathbf{c}
- 一个 $n \times n$ 维的对称矩阵 \mathbf{Q}
- 一个 $m \times n$ 维的矩阵 \mathbf{A}
- 一个 m 维的向量 \mathbf{b}

则此二次规划问题的目标即是在限制条件为

$$\mathbf{Ax} \leq \mathbf{b}$$

的条件下，找一个 n 维的向量 \mathbf{x} ，使得

$$f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x}$$

最小。

根据不同的参数特性，可以得到对问题不同的结论

- 如果 \mathbf{Q} 是半正定矩阵，那么 $f(\mathbf{x})$ 是一个凸函数。
- 如果 \mathbf{Q} 是正定矩阵，那么全局最小值就是唯一的。
- 如果 $\mathbf{Q} = \mathbf{0}$ ，二次规划问题就变成线性规划问题。
- 如果有至少一个向量 \mathbf{x} 满足约束而且 $f(\mathbf{x})$ 在可行域有下界，二次规划问题就有一个全局最小值 \mathbf{x} 。

根据优化理论，一个点 \mathbf{x} 成为全局最小值的必要条件是满足 [Karush-Kuhn-Tucker 条件](#) (KKT)。当 $f(\mathbf{x})$ 是凸函数时，KKT 条件也是充分条件。

当二次规划问题只有等式约束时，二次规划可以用线性方程求解。否则的话，常用的二次规划解法有：内点法(interior point)、active set 和共轭梯度法等。

在 SVM 中，未知变量为向量 \mathbf{w} 和 w_0 ， $\mathbf{c} = \mathbf{0}$ ， $\mathbf{Q} = \mathbf{I}$ 为正定矩阵， $f(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w}$ ，

约束项为 $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, i = 1, \dots, N$ ，即 $-y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq -1, i = 1, \dots, N$

所以 $\mathbf{b} = -\mathbf{1}$ ，

$$\mathbf{A} = \begin{bmatrix} -y_1 x_{11} & -y_1 x_{12} & \cdots & -y_1 x_{1D} & -y_1 \\ -y_2 x_{21} & -y_2 x_{22} & \cdots & -y_2 x_{2D} & -y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -y_N x_{N1} & -y_N x_{N2} & \cdots & -y_N x_{ND} & -y_N \end{bmatrix}$$

SVM的目标函数为带约束的优化问题，可采用拉格朗日乘子法变成非约束的优化问题，再进一步变成对偶问题（dual problem）。通过求解与原问题等价的对偶问题(dual problem)得到原始问题的最优解，这样做的优点在于：一者对偶问题往往更容易求解；二者可以自然的引入核函数，进而推广到非线性分类问题。

SVM的原问题

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ s.t. \quad & 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 0, i = 1, \dots, N \end{aligned}$$

对应的拉格朗日函数为

$$\begin{aligned} L(\mathbf{w}, b, \alpha) &= \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^N \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) \\ s.t. \quad & \alpha_i \geq 0, i = 1, \dots, N \end{aligned}$$

等价于

$$\begin{aligned} L(\mathbf{w}, b, \alpha) &= \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1) \\ s.t. \quad & \alpha_i \geq 0, i = 1, \dots, N \end{aligned}$$

在求有约束条件的优化问题时，**拉格朗日乘子法** (Lagrange Multiplier) 和**KKT条件**是非常重要的两个求取方法。对于等式约束的优化问题，可以应用拉格朗日乘子法去求取最值；如果含有不等式约束，可以应用KKT条件去求取。当然，这两个方法求得的结果只是必要条件，只有当是凸函数的情况下，才能保证是充分必要条件。

拉格朗日乘子法：

<https://zh.wikipedia.org/wiki/%E6%8B%89%E6%A0%BC%E6%9C%97%E6%97%A5%E4%B9%98%E6%95%BD>

http://guoshangshu.com/2016/11/07/machine_learning/lagrange/

拉格朗日乘子法是一种在最优化的问题中寻找多元函数在其变量受到一个或多个条件的相等约束时的求局部极值的方法。这种方法可以将一个有 d 个变量和 n 个约束条件的最优化问题转换为一个解有 $d + n$ 个变量的方程组的解的问题

考虑一个最优化问题

$$\begin{aligned} & \max_{\mathbf{x}} f(\mathbf{x}) \\ s.t. \quad & g(\mathbf{x}) = 0 \end{aligned}$$

为了求 \mathbf{x} ，引入一个新的变量 λ ， λ 称为**拉格朗日乘数**，问题转化为求朗格朗日函数

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$$

的极值。

朗格朗日函数对 \mathbf{x} 的偏导 $\nabla_{\mathbf{x}} L(\mathbf{x}, \lambda)$ 置零

$$\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0$$

朗格朗日函数对 $\nabla_{\lambda} L(\mathbf{x}, \lambda)$ 的偏导置零

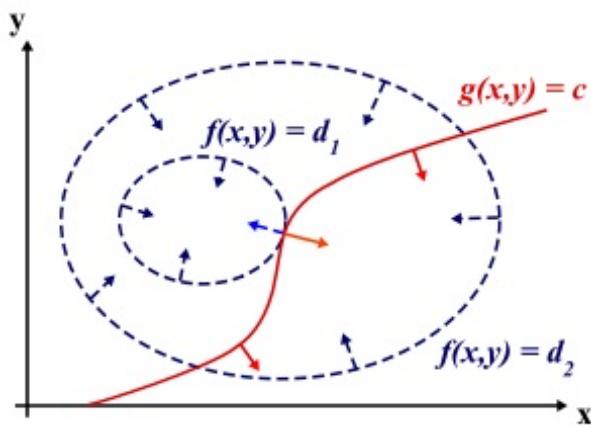
$$g(\mathbf{x}) = 0$$

因此，原约束优化问题可转化为对拉格朗日函数的无约束优化问题。

根据约束条件 $\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0$ ，在最优点 \mathbf{x}^* ，梯度 $\nabla g(\mathbf{x}^*)$ 和 $\nabla f(\mathbf{x}^*)$ 的方向必相同或相反。

** 为什么拉格朗日乘子法 (Lagrange Multiplier) 能够得到最优值? **

设想我们的目标函数 $z = f(\mathbf{x})$, z 取不同的值，相当于投影在 \mathbf{x} 构成的平面（曲面）上的等高线。如下图，目标函数是 $f(x, y)$ ，虚线是等高线，现在假设我们的约束 $g(x, y) = c$ 。假设 $g(x, y)$ 与等高线相交，交点就是同时满足等式约束条件和目标函数的可行域的值，但肯定不是最优值，因为相交意味着肯定还存在其它的等高线在该条等高线的内部或者外部，使得新的等高线与目标函数的交点的值更大或者更小，只有到等高线与目标函数的曲线相切的时候，可能取得最优值。如下图所示，等高线和目标函数的曲线在该点的法向量必须有相同方向，所以最优值必须满足：梯度 $\nabla f(\mathbf{x}^*) = \lambda \nabla g(\mathbf{x}^*)$ ， λ 是常数，表示左右两边同向或反向。



KKT 条件：

对含有不等式约束的优化问题

$$\begin{aligned} & \min f(\mathbf{x}), \\ & s.t. \quad g(\mathbf{x}) \leq 0; \end{aligned}$$

我们将不等式约束与 $f(\mathbf{x})$ 写为一个式子，也称为拉格朗日函数

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}) \quad s.t. \quad \lambda \geq 0$$

系数 λ 也称拉格朗日乘子。

原带不等式越是的优化问题转化为在如下约束下最小化拉格朗日函数：

$$\begin{cases} g(\mathbf{x}) \leq 0 \\ \lambda \geq 0 \\ \lambda g(\mathbf{x}) = 0 \end{cases}$$

这些约束条件称为 KKT 条件。

其中第三个式子非常有趣，因为 $g(\mathbf{x}) \leq 0$ ，如果要满足这个等式，必须 $\lambda = 0$ 或者 $g(\mathbf{x}) = 0$ 。这是 SVM 的很多重要性质的来源，如支持向量的概念。

最优点 \mathbf{x}^* 或在 $g(\mathbf{x}) < 0$ 的区域中，或在边界 $g(\mathbf{x}) = 0$ 上。

- $g(\mathbf{x}) < 0$ 时，约束 $g(\mathbf{x}) \leq 0$ 不起作用，可直接通过条件 $\nabla f(\mathbf{x}) = 0$ 来获得最优点；这等价于将朗格朗日函数中的 λ 置零，然后对 $\nabla_{\mathbf{x}} L(\mathbf{x}, \lambda)$ 置零得到最优点。
- $g(\mathbf{x}) = 0$ 时，就是等式约束问题。需要注意的是，此时梯度 $\nabla g(\mathbf{x}^*)$ 和 $\nabla f(\mathbf{x}^*)$ 的方向必须相反，即 KKT 条件中 $\lambda > 0$ 。

整合上面两种情况，可以得到 KKT 条件的约束。

然后我们令

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

容易验证，当某个约束条件不满足时，例如 $y_i(\mathbf{w}^T \mathbf{x}_i + b) < 1$ ，那么我们显然有 $\theta(\mathbf{w}) = \infty$ （只要令 $\alpha_i = \infty$ 即可）。而当所有约束条件都满足时，则有 $\theta(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|_2^2$ ，亦即我们最初要最小化的量。因此，在要求约束条件得到满足的情况下最小化 $\frac{1}{2} \|\mathbf{w}\|_2^2$ 实际上等价于直接最小化 $\theta(\mathbf{w})$ （当然也有约束条件 $\alpha_i \geq 0, i = 1, \dots, N$ ），因为如果约束条件没有得到满足， $\theta(\mathbf{w})$ 会等于无穷大，自然不会是我们所要求的最小值。

目标函数变成了

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

这里用 p^* 表示这个问题的最优值，这个问题和我们最初的问题是等价的，称为原问题（primary）。将最小和最大的位置交换一下：

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

交换以后的问题称为对偶问题（dual），不再等价于原问题，这个新问题的最优值用 d^* 表示，并且有 $d^* \leq p^*$ 。

这在直观上不难理解，最大值中最小的一个总也比最小值中最大的一个要大。第二个问题的最优值 d^* 提供了一个第一个问题的最优值 p^* 的一个下界。在满足某些条件（KKT 条件）的情况下，这两者相等，这个时候我们就可以通过求解第二个问题来间接地求解第一个问题。具体来说，就是要满足 KKT 条件。

首先要让 L 关于 \mathbf{w}, b 最小化，我们分别令 $\partial L / \partial \mathbf{w}$ 和 $\partial L / \partial b$ 等于零：

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

代入 L 得到：

$$\begin{aligned}
L(\mathbf{w}, b, \alpha) &= \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1) \\
&= \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - \sum_{i=1}^N \alpha_i y_i b + \sum_{i=1}^N \alpha_i \\
&= \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - b \sum_{i=1}^N \alpha_i y_i + \sum_{i=1}^N \alpha_i \\
&= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j
\end{aligned}$$

此时我们得到关于对偶变量 α 的优化问题：

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

此时的拉格朗日函数只包含了一个变量 α ，方便求解（由于该问题结构的特殊性，有高效的求解算法 SMO）。

α 求出了便能求出 \mathbf{w}, b ，从而得到分类判别函数

$$\begin{aligned}
f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\
&= \left(\sum_{i=1}^N \alpha_i y_i \mathbf{x}_i \right)^T \mathbf{x} + b \\
&= \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b \\
&= \sum_{i=1}^N \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b
\end{aligned}$$

这里的形式的有趣之处在于，对于新点 \mathbf{x} 的预测，只需要计算它与训练数据点的内积即可（这里 $\langle \cdot, \cdot \rangle$ 表示向量内积）。这一点至关重要，稍后我们使用核方法（Kernel）进行非线性推广的基本前提。此外，所谓支持向量（Supporting Vector）也在这里显示出来：所有非支持向量所对应的系数 α 都是等于零，因此对于新点的内积计算实际上只要针对少量的“支持向量”而不是所有的训练数据。

为什么非支持向量对应的 α 等于零呢？直观上来理解的话，就是这些“后方”的点——正如我们之前分析过的一样，对超平面是没有影响的。由于分类完全由超平面决定，所以这些无关的点并不会参与分类问题的计算，因而也就不会产生任何影响了。这个结论也可由刚才的推导中得出，回忆一下我们刚才通过 Lagrange multiplier 得到的目标函数：

$$\begin{aligned}
L(\mathbf{w}, b, \alpha) &= \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1) \\
&s.t. \quad \alpha_i \geq 0, i = 1, \dots, N
\end{aligned}$$

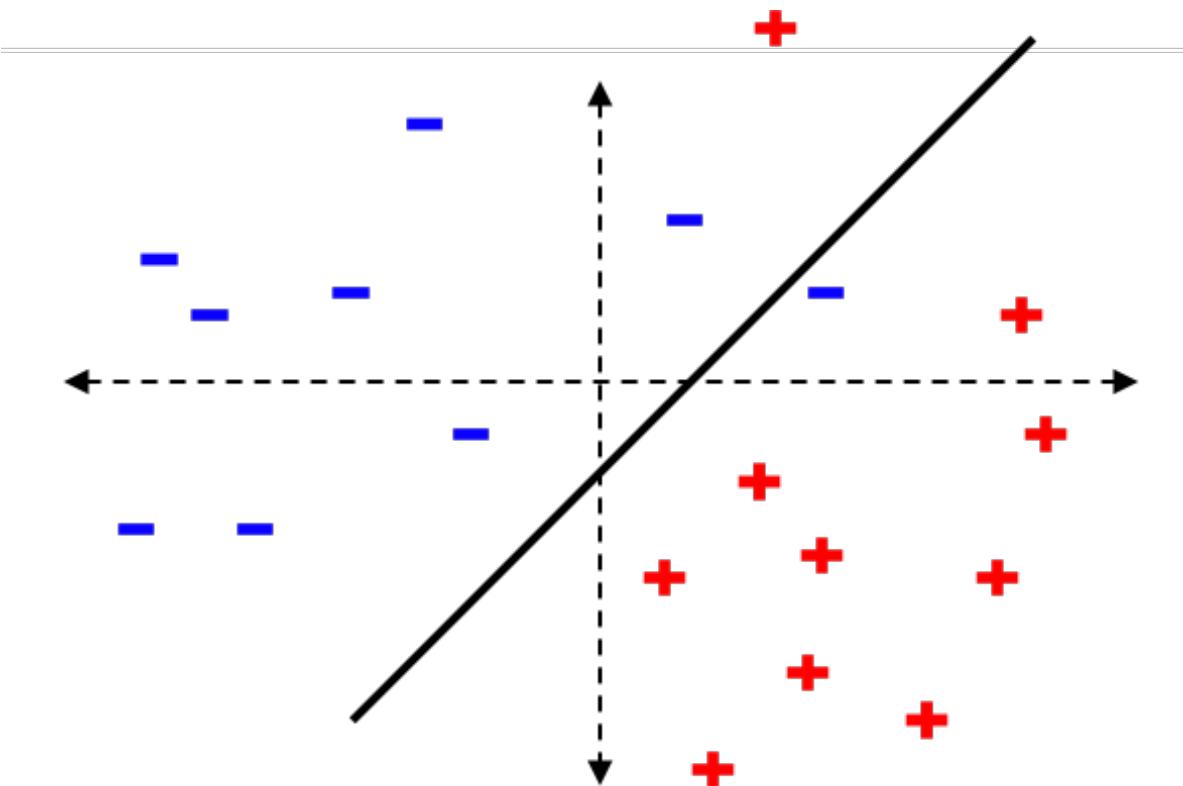
对应的 KKT 条件：

$$\begin{cases} y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \geq 0 \\ \alpha_i \geq 0 \\ \xi_i \geq 0 \\ \alpha_i(1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0 \\ \mu_i \geq 0 \\ \mu_i \xi_i = 0 \end{cases}$$

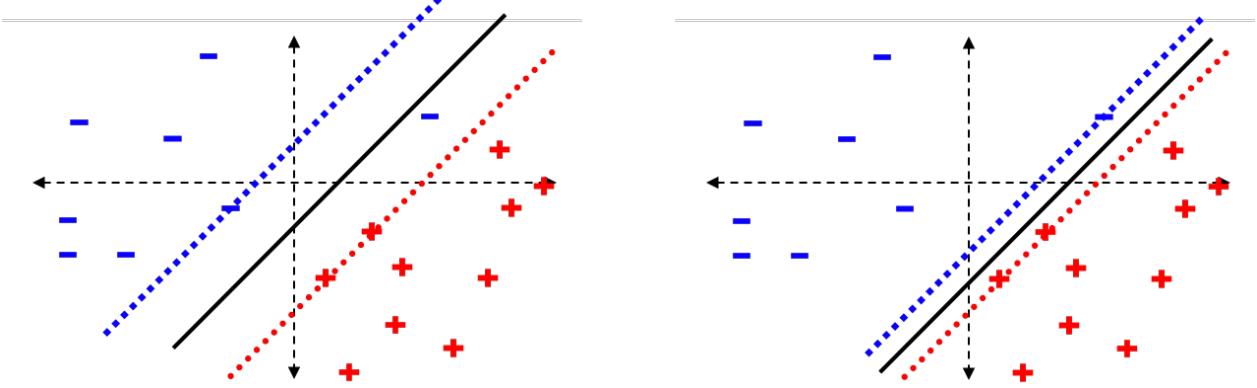
注意到如果 \mathbf{x}_i 是支持向量的话，不等式约束 $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \geq 0$ 变成等式约束 $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0$ (因为支持向量到超平面的距离最小，等于 1)；而对于非支持向量来说，该点到超平面的距离会大于 1， $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \geq 0$ ，而 α_i 又是非负的，为了满足最大化， α_i 必须等于 0。

2 带松弛因子的SVM (C-SVM)

第1节中我们讨论的是数据是线性可分的。但有时因为混入了异常点，导致不能线性可分，如下图，本来数据是可以按下面的实线来做超平面分离的，可以由于一个红色和一个蓝色的异常点导致我们没法用下面的实现进行完美分类。



或者数据仍然可以分开，但是会严重影响模型的泛化预测效果。如下图所示，如果我们不考虑异常点，SVM 的超平面应该是左图中的黑色线所示；但是由于有一个蓝色的异常点，导致我们学习到的超平面是右图中的黑线所示，这样会严重影响我们的分类模型预测效果（右边分类面对应的margin小）。



缓解上述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，引入“软间隔（soft margin）”的概念。所谓的软间隔，是相对于第1节中的硬间隔而言。

回顾一下硬间隔最大化的条件：

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t. } & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, i = 1, \dots, N \end{aligned}$$

在硬间隔最大化中，要求所有样本都必须被正确分类。而软间隔允许某些样本不满足约束

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, i = 1, \dots, N$$

SVM对训练集里面的每个样本 (\mathbf{x}_i, y_i) 引入了一个松弛变量 $\xi_i \geq 0$ ，使函数间隔加上松弛变量大于等于1，即：

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, \dots, N$$

对比硬间隔最大化，可以看到我们对样本到超平面的函数距离的要求放松了，之前是一定要大于等于1，现在只需要加上一个大于等于0的松弛变量能大于等于1就可以了。当然，松弛变量是有成本的，每一个松弛变量 ξ_i ，对应了一个代价 ξ_i ，这个就得到了我们的软间隔最大化的SVM的最小化优化目标可写为

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t. } & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, \dots, N \\ & \xi_i \geq 0, i = 1, \dots, N \end{aligned}$$

也就是说，我们希望 $\frac{1}{2} \|\mathbf{w}\|_2^2$ 尽量小，同时误分类的点尽可能的少。其中 $C > 0$ 为惩罚参数，可以理解为我们回归和分类问题中的正则化参数（的倒数）。显然， C 越大，对误分类的惩罚越大， $C = \infty$ 时，迫使所有样本满足硬约束（松弛变量均为0）； C 越小，对误分类的惩罚越小。

这个目标函数的优化和第1节中线性可分SVM的优化方式类似，通过朗格朗日乘子法变成七对偶问题求解。

带松弛变量的SVM的原问题

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t. } & 1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 0, i = 1, \dots, N \\ & -\xi_i \leq 0, i = 1, \dots, N \end{aligned}$$

对应的拉格朗日函数为

$$\begin{aligned}
L(\mathbf{w}, b, \alpha, \xi, \mu) &= \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i + \sum_{i=1}^N \alpha_i (1 - \xi_i - y_i (\mathbf{w}^T \mathbf{x}_i + b)) - \sum_{i=1}^N \mu_i \xi_i \\
s.t. \quad &\alpha_i \geq 0, i = 1, \dots, N \\
&\mu_i \geq 0, i = 1, \dots, N \\
&\xi_i \geq 0, i = 1, \dots, N
\end{aligned}$$

等价于

$$\begin{aligned}
L(\mathbf{w}, b, \alpha, \xi, \mu) &= \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^N \mu_i \xi_i \\
s.t. \quad &\alpha_i \geq 0, i = 1, \dots, N \\
&\mu_i \geq 0, i = 1, \dots, N \\
&\xi_i \geq 0, i = 1, \dots, N
\end{aligned}$$

首先要让 $L(\mathbf{w}, b, \alpha, \xi, \mu)$ 关于 \mathbf{w}, b, ξ 最小化，我们分别令 $\partial L / \partial \mathbf{w}$, $\partial L / \partial b$ 和 $\partial L / \partial \xi_i$ 等于零：

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{w}} = 0 &\Rightarrow \mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i \\
\frac{\partial L}{\partial b} = 0 &\Rightarrow \sum_{i=1}^N \alpha_i y_i = 0 \\
\frac{\partial L}{\partial \xi_i} = 0 &\Rightarrow C = \alpha_i + \mu_i
\end{aligned}$$

代入 $L(\mathbf{w}, b, \alpha, \xi, \mu)$ 得到：

$$\begin{aligned}
L(\mathbf{w}, b, \alpha, \xi, \mu) &= \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^N \mu_i \xi_i \\
&= \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i) + \sum_{i=1}^N \alpha_i \xi_i \\
&= \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1) \\
&= \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - \sum_{i=1}^N \alpha_i y_i b + \sum_{i=1}^N \alpha_i \\
&= \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - b \sum_{i=1}^N \alpha_i y_i + \sum_{i=1}^N \alpha_i \\
&= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j
\end{aligned}$$

此时我们得到关于对偶变量 α 的优化问题：

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

$L(\mathbf{w}, b, \alpha, \xi, \mu)$ 的形式与硬间隔的SVM相同，这是约束条件有所不同：硬间隔SVM的约束条件为 $0 \leq \alpha_i$ ，而软间隔SVM的约束条件为 $0 \leq \alpha_i \leq C$ 。

软间隔SVM的约束条件为：

$$\begin{aligned}\alpha_i &\geq 0, i = 1, \dots, N \\ \mu_i &\geq 0, i = 1, \dots, N \\ \xi_i &\geq 0, i = 1, \dots, N\end{aligned}$$

再加上

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

对于 $C = \alpha_i + \mu_i = 0$, $\alpha_i \geq 0, \mu_i \geq 0$, 这3个式子，我们可以消去 μ_i , 只留下 α_i , 也就是说 $0 \leq \alpha_i \leq C$ 。

类似硬间隔的SVM, α 求出了便能求出 \mathbf{w}, b , 从而得到分类判别函数

$$\begin{aligned}f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\ &= \left(\sum_{i=1}^N \alpha_i y_i \mathbf{x}_i \right)^T \mathbf{x} + b \\ &= \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b \\ &= \sum_{i=1}^N \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b\end{aligned}$$

对偶问题的目标函数

$$\begin{aligned}L(\mathbf{w}, b, \alpha, \xi, \mu) &= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ s.t. \quad 0 \leq \alpha_i &\leq C, \quad i = 1, \dots, N \\ \sum_{i=1}^N \alpha_i y_i &= 0\end{aligned}$$

对应的KKT条件：

$$\begin{cases} y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 - \xi_i \geq 0 \\ \xi_i \geq 0 \\ \alpha_i \geq 0 \\ \mu_i \geq 0 \\ \alpha_i (1 - \xi_i - y_i (\mathbf{w}^T \mathbf{x}_i + b)) = 0 \\ \mu_i \xi_i = 0 \end{cases}$$

类似硬间隔SVM, 对任意训练样本 (\mathbf{x}_i, y_i) , $\alpha_i (1 - \xi_i - y_i (\mathbf{w}^T \mathbf{x}_i + b)) = 0$, 所以

- $\alpha_i = 0$: (\mathbf{x}_i, y_i) 为非支持向量;

- 或 $\alpha_i > 0$, $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \xi_i$: (\mathbf{x}_i, y_i) 是支持向量:
 - $\alpha_i < C$: 则 $\mu_i > 0$, 进而根据 $\mu_i \xi_i = 0 \Rightarrow \xi_i = 0$, 则 (\mathbf{x}_i, y_i) 恰好在最大间隔边界 $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \xi_i = 1$ 上;
 - $\alpha_i = C$: 则 $\mu_i = 0$, 此时 $\xi_i \geq 0$:
 - 若 $\xi_i \leq 1$, 则 (\mathbf{x}_i, y_i) 落在在最大间隔边界内部 ($y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \xi_i \geq 0$), 仍能被正确分类;
 - 若 $\xi_i \leq 1$, 则 (\mathbf{x}_i, y_i) 落在在最大间隔边界外部 ($y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \xi_i \leq 0$), 样本被错误分类。

由此可以看出, 软间隔SVM仍有支持向量, 能保持稀疏性。

KKT条件:

对含有不等式约束的优化问题

$$\begin{aligned} & \min f(\mathbf{x}), \\ & s.t. \quad g(\mathbf{x}) \leq 0; \end{aligned}$$

我们将不等式约束与 $f(\mathbf{x})$ 写为一个式子, 也称为拉格朗日函数

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}) \quad s.t. \quad \lambda \geq 0$$

系数 λ 也称拉格朗日乘子。

原带不等式越是的优化问题转化为在如下约束下最小化拉格朗日函数:

$$\begin{cases} g(\mathbf{x}) \leq 0 \\ \lambda \geq 0 \\ \lambda g(\mathbf{x}) = 0 \end{cases}$$

这些约束条件称为KKT条件。

带松弛变量的SVM的原问题

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i \\ & s.t. \quad 1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 0, i = 1, \dots, N \\ & \quad -\xi_i \leq 0, i = 1, \dots, N \end{aligned}$$

对应的拉格朗日函数为

$$\begin{aligned} L(\mathbf{w}, b, \alpha, \xi, \mu) &= \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i + \sum_{i=1}^N \alpha_i (1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b)) - \sum_{i=1}^N \mu_i \xi_i \\ & s.t. \quad \alpha_i \geq 0, i = 1, \dots, N \\ & \quad \mu_i \geq 0, i = 1, \dots, N \\ & \quad \xi_i \geq 0, i = 1, \dots, N \end{aligned}$$

对应的KKT条件:

$$\begin{cases} y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 - \xi_i \geq 0 \\ \xi_i \geq 0 \\ \alpha_i \geq 0 \\ \mu_i \geq 0 \\ \alpha_i(1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0 \\ \mu_i \xi_i = 0 \end{cases}$$

3. 合页损失函数 (Hinge Loss)

C-SVM中,

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t. } & 1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 0, i = 1, \dots, N \end{aligned}$$

若当 $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ 时, 定义 $\xi_i = 0$; 否 $\xi_i = 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)$, 即

$$\xi_i = [1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)]_+$$

其中

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

这样C-SVM的目标函数可写为

$$\min_{\mathbf{w}, b} \sum_{i=1}^N \xi_i + \lambda \|\mathbf{w}\|_2^2$$

或者将 $\xi_i = [1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)]_+$ 代入, 得到最小化目标函数为

$$J(\mathbf{w}, b) = \sum_{i=1}^N [1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)]_+ + \lambda \|\mathbf{w}\|_2^2$$

将上式与Logistic回归的目标函数

$$J(\mathbf{w}) = - \sum_{i=1}^N [y_i \log(\mu(\mathbf{x}_i)) + (1 - y_i) \log(1 - \mu(\mathbf{x}_i))] + \lambda \|\mathbf{w}\|_2^2$$

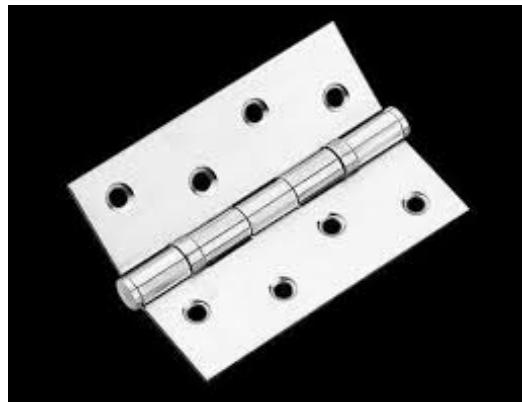
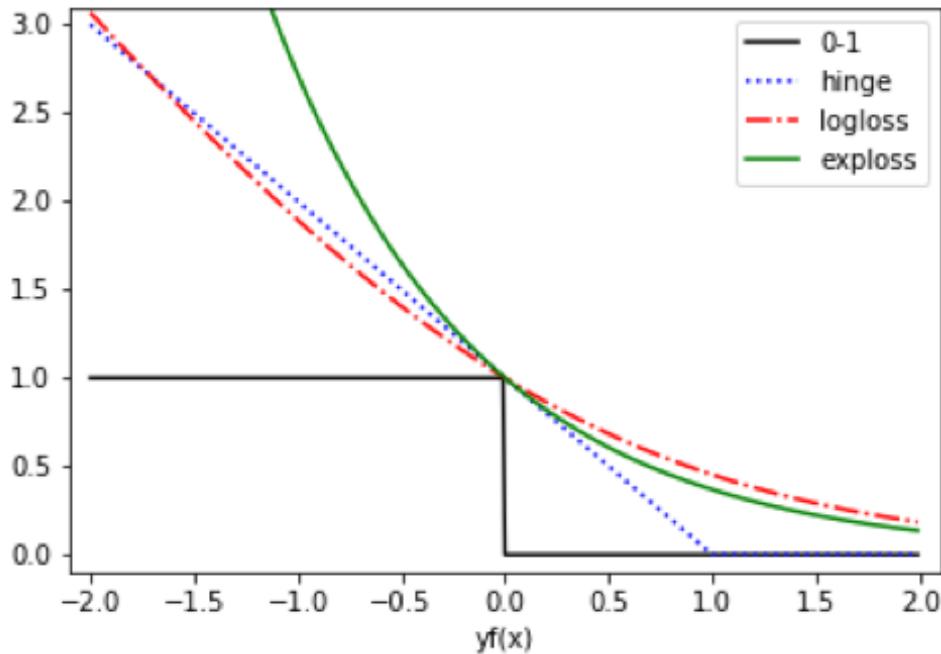
对比, 可以发现二者的形式类似, 只是第一项损失函数不同。Logistic回归取得是负log似然损失 (logloss)

$$l_{log}(y_i, f(\mathbf{x}_i)) = y_i \log(f(\mathbf{x}_i)) + (1 - y_i) \log(1 - f(\mathbf{x}_i)) = \log(1 + \exp(-y_i f(\mathbf{x}_i)))$$

而SVM取得是合页损失 (Hinge Loss) :

$$l_{hinge}(y_i, f(\mathbf{x}_i)) = [1 - y_i f(\mathbf{x}_i)]_+$$

集中分类中常用的损失函数如下图所示。SVM中的合页损失函数取名为“合页”是因为其形状如合页。



事实上，负log似然损失和合页损失都可以看成是0/1损失

$$l_{0/1} = \begin{cases} 0 & y_i = f(\mathbf{x}_i) \\ 1 & y_i \neq f(\mathbf{x}_i) \end{cases}$$

的近似，被称为代理损失函数（Surrogate loss function）。因为0/1损失函数非凸、非连续，数学性质不太好，从而优化计算不方便。

SVM正是因为采用合页损失函数保持了支持向量机解的稀疏性，因为合页损失函数的零区域对应的正是非支持向量的普通样本，从而所有的普通样本都不参与最终超平面的决定。

采用合页损失的方式解释SVM模型，使得我们对SVM的理解可以放到机器学习模型的一般框架

$$J(\theta) = - \sum_{i=1}^N L(y_i, f(\mathbf{x}_i; \theta)) + R(\theta)$$

即SVM的最佳模型为与训练数据拟合得最好的最简单的模型，亦被称为结构风险最小化，一直对应的训练集上的损失函数值和被称为经验风险。

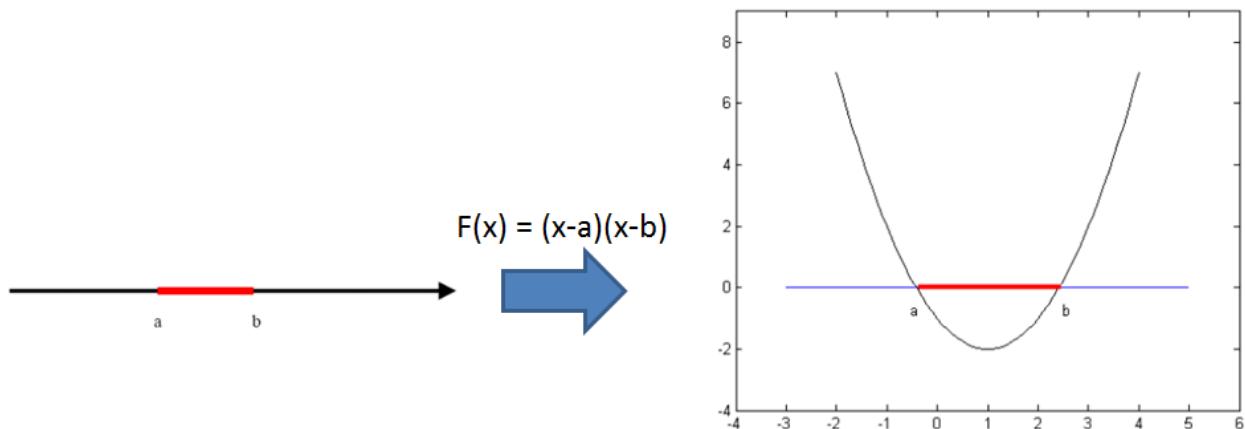
4 核方法

前面我们讲到了线性可分SVM的硬间隔最大化和软间隔最大化的算法，当数据不完全线性可分时，线性模型表现不好。本节我们探讨SVM如何处理线性不可分的数据，重点讲述核方法将线性模型非线性化，从而能处理线性不可分数据。

4.1 核技巧

例：下图左边的原始数据是一维，我们用表示。黑色区域的数据点和红色区域的数据点通过线性模型不能完全分开（在一维空间中，线性模型为一个点，也就是说我们在数据轴上任取一点，都不能将红色部分与黑色部分完全分开，因为黑色区域分列在红色区域两侧）。

但将原始数据变换到如右图所示的二维空间 $\mathbf{z} = (z_1, z_2) = (\mathbf{x}, (\mathbf{x} - \mathbf{a})(\mathbf{x} - \mathbf{b}))$ ，我们用线性模型 $(z_2 = 0)$ 就可以将黑色区域和红色区域分开，即 $z_2 > 0$ 为黑色，否则为红色。



也就是说，对于一维的不是线性的数据，我们将其映射到了二维以后，就变成了线性可分的数据。这给了我们启发，也就是说对于在低维线性不可分的数据，在映射到了高维以后，就变成线性可分的了。

假设我们用映射 $\phi(\mathbf{x})$ 将 \mathbf{x} 映射成高维特征向量，则在特征空间中分类超平面对应的模型可表示为

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

由于该模型在高维特征空间仍是线性模型，所以我们可以完全照搬第2节中线性SVM模型的推导过程，只需将原来的 \mathbf{x} 换成 $\phi(\mathbf{x})$ 即可。

因此SVM原问题的最小化优化目标可写为

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t. } & y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, i = 1, \dots, N \\ & \xi_i \geq 0, i = 1, \dots, N \end{aligned}$$

对偶变量 α 的优化问题：

$$\begin{aligned}
& \max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j < \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) > \\
& \text{s.t.}, \quad 0 \leq \alpha_i \leq C, \quad i = 1, \dots, N \\
& \quad \sum_{i=1}^N \alpha_i y_i = 0
\end{aligned}$$

计算出 α 便能求出 \mathbf{w}, b ，从而得到分类判别函数

$$\begin{aligned}
f(\mathbf{x}) &= \mathbf{w}^T \phi(\mathbf{x}) + b \\
&= \left(\sum_{i=1}^N \alpha_i y_i \phi(\mathbf{x}_i) \right)^T \phi(\mathbf{x}) + b \\
&= \sum_{i=1}^N \alpha_i y_i \phi^T(\mathbf{x}_i) \phi(\mathbf{x}) + b \\
&= \sum_{i=1}^N \alpha_i y_i < \phi(\mathbf{x}_i), \phi(\mathbf{x}) > + b
\end{aligned}$$

从上述推导过程可以看出，我们将线性模型中的原始特征的内积换成映射后特征的内积，就完成了将线性模型非线性化。看起来似乎这样我们就已经完美解决了线性不可分SVM的问题了。但是特征空间维数可能很高（甚至无穷维）才能将数据变得先行可分，而在高维空间直接计算特征空间的点积通常是困难的。

为了避开这个障碍，核函数隆重出场了！

假设 ϕ 是一个从低维的输入空间 \mathcal{X} （欧式空间的子集或者离散集合）到高维的希尔伯特空间的 \mathcal{H} 映射。那么如果存在函数 $\kappa(\mathbf{x}, \mathbf{z})$ ，对于任意 $\mathbf{x}, \mathbf{z} \in \mathcal{X}$ ，都有：

$$\kappa(\mathbf{x}, \mathbf{z}) = < \phi(\mathbf{x}), \phi(\mathbf{z}) >$$

那么我们就称 $\kappa(\mathbf{x}, \mathbf{z})$ 为核函数。

这样我们在推导SVM过程中高维特征内积 $< \phi(\mathbf{x}_i), \phi(\mathbf{x}_i) >$ 、 $< \phi(\mathbf{x}_i), \phi(\mathbf{x}) >$ 分别用核函数 $\kappa(\mathbf{x}_i, \mathbf{x}_i)$ 、 $\kappa(\mathbf{x}_i, \mathbf{x})$ 替代即可。核函数 $\kappa(\mathbf{x}, \mathbf{z})$ 的计算是在低维特征空间来计算的，避免了上面提到的在高维维度空间计算内积的计算量大的问题。也就是说，我们可以享受在高维特征空间线性可分的红利，却避免了高维特征空间繁重的内积计算量。这种技巧被称为核技巧（Kernel trick）。

4.2 核函数构造

我们还需要解决一个问题，即核函数的存在性判断和如何构造？既然我们不关心高维度空间的表达形式，那么怎么才能判断一个函数是否是核函数呢？

定理：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任意数据 $\mathcal{D} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$ ，核矩阵 \mathbf{K} 总是半正定的：

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \kappa(\mathbf{x}_1, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) & \kappa(\mathbf{x}_2, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) & \kappa(\mathbf{x}_N, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

事实上，只要一个函数对应的矩阵是半正定，总可以找到一个与之对应的映射 ϕ 。也就是说任何一个核函数都隐式定义了一个再生核希尔伯特空间（Reproducing Kernel Hilbert Space, RKHS）。

通过前面讨论可知，我们希望样本映射到高维特征空间后是线性可分的，因此特征空间的好坏对支持向量机的性能至关重要。在核方法中，特征空间由核函数决定，因此核函数的选择对支持向量机的性能很重要。

常用的核函数有：

- 线性核函数

线性核函数（Linear Kernel）其实就是我们前两篇的线性可分SVM，表达式为：

$$\kappa(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$$

这样我们可以将线性SVM和核SVM统一起来，区别仅仅在于线性可分SVM用的是线性核函数。

- 多项式核函数

多项式核函数（Polynomial Kernel）是线性不可分SVM常用的核函数之一，表达式为：

$$\kappa(\mathbf{x}, \mathbf{z}) = (\gamma \langle \mathbf{x}, \mathbf{z} \rangle + r)^M$$

其中 γ 、 r 、 M 为多项式和函数的参数。 M 越大，多项式阶数越高，决策边界越不平滑。

- 高斯核函数

高斯核函数（Gaussian Kernel），在SVM中也称为径向基核函数（Radial Basis Function, RBF），它是非线性分类SVM最主流的核函数。libsvm默认的核函数就是高斯核函数。表达式为：

$$\kappa(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2)$$

其中 $\gamma > 0$ 为高斯核函数的宽度。 γ 越大，决策边界越不平滑。

此外，核函数还可以通过函数组合得到，例如：

如果 κ_1 和 κ_2 是核函数，则下列函数也是核函数：

$\gamma_1 \kappa_1 + \gamma_2 \kappa_2$, 其中 $\gamma_1 > 0$ 和 $\gamma_2 > 0$;

$\kappa_1 \kappa_2$ ；

5 SVM优化求解：SMO

<https://zhuanlan.zhihu.com/p/29212107>

5.1 SMO算法原理：获得没有修剪的原始解

SVM的对偶问题：

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

也是一个二次规划问题，可用通用的二次规划算法求解。然而该对偶问题的未知数规模与训练样本数目 N 相同，这会在样本数目比较多的实际任务中造成很大开销。为了避开这个障碍，人们通过利用问题本身特性，提出了很多高效的算法。John Platt于1998年提出序列最小化优化算法（Sequential minimal optimization, SMO），并成为最快的二次规划优化算法，特别针对线性SVM和数据稀疏时性能更优。关于SMO最好的资料是他本人写的《Sequential Minimal Optimization A Fast Algorithm for Training Support Vector Machines》。

SMO算法将原始问题的求解 N 个参数二次规划问题分解成多个二次规划问题求解，每个子问题只需要求解2个参数。根据坐标下降/上升算法，我们应该固定其他参数，每次更新一个参数 α_i 。但在SVM中， α 并不是完全独立，而是具有约束的：

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

因此一个 α_i 改变，另一个 α_j 也要随之变化以满足条件。因此在SMO算法中我们每次需要选择一对变量 (α_i, α_j) 一起修改。

SMO在整个二次规划的过程中：

- 选取一对参数 (α_i, α_j)
- 固定向量 α 的其他参数，对 (α_i, α_j) 求最优解，获得更新后的 (α_i, α_j)

SMO不断执行这两个步骤直至收敛。

将SVM对偶问题的最大化优化目标转化为最小化优化目标：

$$\begin{aligned} & \min_{\alpha} \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \sum_{i=1}^N \alpha_i \\ & \text{s.t.}, \quad 0 \leq \alpha_i \leq C, \quad i = 1, \dots, N \\ & \quad \sum_{i=1}^N \alpha_i y_i = 0 \end{aligned}$$

假设我们选取的两个需要优化的参数为 (α_1, α_2) ，剩下的 $(\alpha_3, \dots, \alpha_N)$ 固定，作为常数处理。将SVM优化问题进行展开就可以得到(去掉与 (α_1, α_2) 无关的项)

$$\begin{aligned} W(\alpha_1, \alpha_2) = & \frac{1}{2} \alpha_1^2 y_1^2 \mathbf{x}_1^T \mathbf{x}_1 + \frac{1}{2} \alpha_2^2 y_2^2 \mathbf{x}_2^T \mathbf{x}_2 + \alpha_1 \alpha_2 y_1 y_2 \mathbf{x}_1^T \mathbf{x}_2 \\ & + \alpha_1 y_1 \sum_{i=3}^N \alpha_i y_i \mathbf{x}_1^T \mathbf{x}_i + \alpha_2 y_2 \sum_{i=3}^N \alpha_i y_i \mathbf{x}_2^T \mathbf{x}_i \\ & - \alpha_1 - \alpha_2 \end{aligned}$$

令 $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ ，且

$$\begin{aligned} v_1 &= \sum_{i=3}^N \alpha_i y_i \mathbf{x}_1^T \mathbf{x}_i = \sum_{i=3}^N \alpha_i y_i K_{1i} \\ v_2 &= \sum_{i=3}^N \alpha_i y_i \mathbf{x}_2^T \mathbf{x}_i = \sum_{i=3}^N \alpha_i y_i K_{2i} \end{aligned}$$

则目标函数 $W(\alpha_1, \alpha_2)$ 可简写为

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

根据约束条件

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

可得到

$$\alpha_1 y_1 + \alpha_2 y_2 = - \sum_{i=3}^N \alpha_i y_i = \zeta$$

两边同乘以 y_1 , 且 $y_i^2 = 1$,

$$\begin{aligned}
\alpha_1 y_1^2 + \alpha_2 y_1 y_2 &= \zeta y_1 \\
\alpha_1 &= \zeta y_1 - \alpha_2 y_1 y_2
\end{aligned}$$

将 $\alpha_1 = \zeta y_1 - \alpha_2 y_1 y_2$ 且 $y_i^2 = 1$ 代入 $W(\alpha_1, \alpha_2)$, 消除 α_1 , 得到仅仅包含 α_2 的式子

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

我们需要对这个一元函数进行求极值, W 对 α_2 的一阶导数为 0 得到:

$$\begin{aligned}
\frac{\partial W(\alpha_2)}{\partial \alpha_2} = & -y_2 (\zeta - \alpha_2 y_2) K_{11} + \alpha_2 K_{22} + y_2 \zeta K_{12} - 2\alpha_2 K_{12} - y_2 v_1 + y_2 v_2 + y_1 y_2 - 1 \\
= & -y_2 (\zeta - \alpha_2 y_2) K_{11} + \alpha_2 K_{22} + y_2 \zeta K_{12} - 2\alpha_2 K_{12} - y_2 v_1 + y_2 v_2 + y_1 y_2 - y_2^2 \\
= & (K_{11} + K_{22} - 2K_{12}) \alpha_2 - y_2 (\zeta (K_{11} - K_{12}) + v_1 - v_2 - y_1 + y_2) \\
= & 0
\end{aligned}$$

得到:

$$(K_{11} + K_{22} - 2K_{12}) \alpha_2^{new} = y_2 (\zeta (K_{11} - K_{12}) + v_1 - v_2 - y_1 + y_2)$$

下面我们稍微对上式进行下变形, 使得 α_2^{new} 能够用更新前的 α_2^{old} 表示。

因为 SVM 对数据点的预测值为:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i=1}^N \alpha_i y_i < \mathbf{x}_i, \mathbf{x} > + b = \sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b,$$

则 v_1 以及 v_2 的值可以表示成:

$$v_1 = \sum_{i=3}^N \alpha_i y_i K_{1i} = f(\mathbf{x}_1) - \alpha_1 y_1 K_{11} - \alpha_2 y_2 K_{12} - b$$

$$v_2 = \sum_{i=3}^N \alpha_i y_i K_{2i} = f(\mathbf{x}_2) - \alpha_1 y_1 K_{12} - \alpha_2 y_2 K_{22} - b$$

已知 $\alpha_1 = \zeta y_1 - \alpha_2 y_1 y_2$, $\alpha_1 y_1 = \zeta - \alpha_2 y_2$,

$$\begin{aligned} v_1 - v_2 &= f(\mathbf{x}_1) - \alpha_1 y_1 K_{11} - \alpha_2 y_2 K_{12} - w_0 - (f(\mathbf{x}_2) - \alpha_1 y_1 K_{12} - \alpha_2 y_2 K_{22} - w_0) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - \alpha_1 y_1 (K_{11} - K_{12}) - \alpha_2 y_2 (K_{12} - K_{22}) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - (\zeta - \alpha_2 y_2) (K_{11} - K_{12}) - \alpha_2 y_2 (K_{12} - K_{22}) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - \zeta (K_{11} - K_{12}) + \alpha_2 y_2 (K_{11} - K_{12}) - \alpha_2 y_2 (K_{12} - K_{22}) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - \zeta (K_{11} - K_{12}) + \alpha_2 y_2 (K_{11} - K_{12} - K_{12} + K_{22}) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - \zeta (K_{11} - K_{12}) + \alpha_2 y_2 (K_{11} - 2K_{12} + K_{22}) \\ &= f(\mathbf{x}_1) - f(\mathbf{x}_2) - \zeta (K_{11} - K_{12}) + (K_{11} + K_{22} - 2KK_{12}) \alpha_2 y_2 \end{aligned}$$

将 $v_1 - v_2$ 代入 $\frac{\partial W(\alpha_2)}{\partial \alpha_2} = 0$, 得到

$$\begin{aligned} (K_{11} + K_{22} - 2K_{12}) \alpha_2^{new} &= y_2 (\zeta (K_{11} - K_{12}) + v_1 - v_2 - y_1 + y_2) \\ &= y_2 (\zeta (K_{11} - K_{12}) + f(\mathbf{x}_1) - f(\mathbf{x}_2) - \zeta (K_{11} - K_{12}) + (K_{11} + K_{22} - 2K_{12}) \alpha_2 y_2 - y_1 + y_2) \\ &= y_2 (f(\mathbf{x}_1) - f(\mathbf{x}_2) + (K_{11} + K_{22} - 2KK_{12}) \alpha_2 y_2 - y_1 + y_2) \\ &= y_2 (f(\mathbf{x}_1) - y_1 - (f(\mathbf{x}_2) - y_2)) + (K_{11} + K_{22} - 2KK_{12}) \alpha_2 \end{aligned}$$

等式右边的 α_2 是更新前的值, 记为 α_2^{old} , 并且记预测值 $f(\mathbf{x}_i)$ 和真值 y_i 之间的差异为 $E_i = f(\mathbf{x}_i) - y_i$, 则上式可写成

$$(K_{11} + K_{22} - 2K_{12}) \alpha_2^{new} = (K_{11} + K_{22} - 2K_{12}) \alpha_2^{old} + y_2 (E_1 - E_2)$$

令 $\eta = K_{11} + K_{22} - 2K_{12}$, 得到 α_2 的更新公式为

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

这样我们就得到了通过旧的 α_2 获取新的 α_2 的表达式, α_1^{new} 便可以通过 α_2^{new} 得到。

SMO这种解析求解方法避免了二次规划数值解法的复杂迭代过程。这不但大大节省了计算时间, 而且不会牵涉到迭代法造成的误差积累。

5.2 对原始解进行修剪

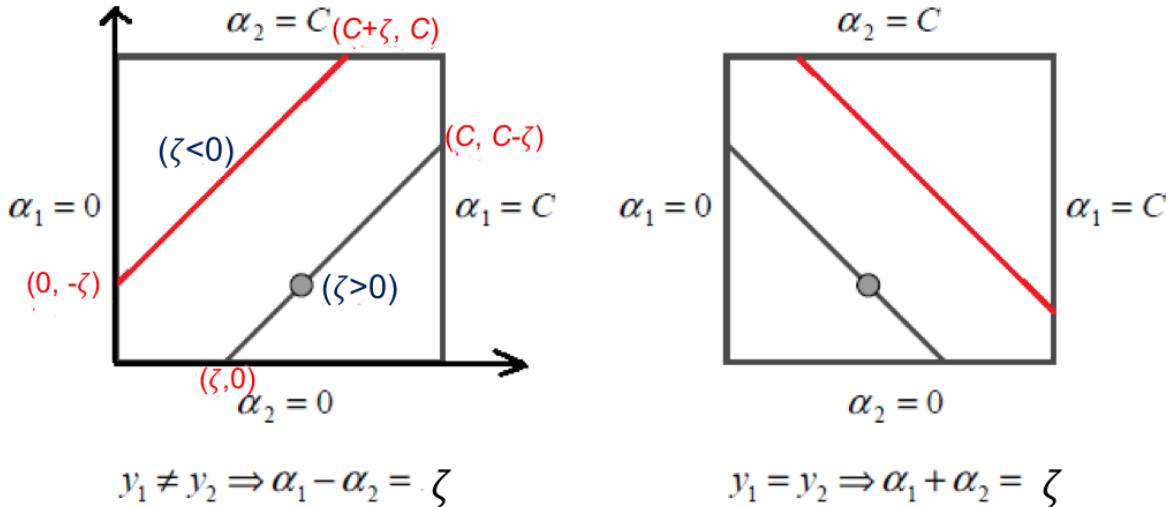
上面我们通过对一元函数求极值的方式得到的最优 α_i, α_j 是未考虑约束条件下的最优解。下面我们将1.3.1节中的 α_2^{new} 记为 $\alpha_2^{new, unclipped}$, 即

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

但是在SVM中的 α_i 是有约束的, 即:

$$\begin{aligned} \alpha_1 y_1 + \alpha_2 y_2 &= \zeta \\ 0 \leq \alpha_i &\leq C, i = 1, 2 \end{aligned}$$

又由于 y_1, y_2 只能取值1或者-1, 这样 α_1, α_2 在 $[0, C]$ 和 $[0, C]$ 形成的盒子里面, 并且两者的关系直线的斜率只能为1或者-1, 也就是说 α_1, α_2 的关系直线平行于 $[0, C]$ 和 $[0, C]$ 形成的盒子的对角线, 如下图所示:



由于 α_1, α_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 α_2 的优化问题。

(如左图) 当 $y_1 \neq y_2$ 时, 线性限制条件可以写成: $\alpha_1 - \alpha_2 = \zeta$, 根据 ζ 的正负可以得到不同的上下界, 因此统一表示成:

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

(如右图) 当 $y_1 = y_2$ 时, 限制条件可写成: $\alpha_1 + \alpha_2 = \zeta$, 上下界表示成:

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

根据得到的上下界, 我们可以得到修剪后的 α_2^{new} :

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

得到了 α_2^{new} 我们便可以根据 $\alpha_1^{old}y_1 + \alpha_2^{old}y_2 = \alpha_1^{new}y_1 + \alpha_2^{new}y_2$ 得到 α_1^{new} :

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

5.3 α 选择

SMO算法需要选择合适的两个变量做迭代, 其余的变量做常量来进行优化, 那么怎么选择这两个变量呢?

在SMO迭代的两个步骤中, 只要 (α_1, α_2) 中有一个违背了KKT条件, 这一轮迭代完成后, 目标函数的值必然会减小。通常而言, KKT条件违背的程度越大, 迭代后的优化效果越明显, 降幅越大。

和梯度下降类似, 我们要找到使之优化程度最大的方向 (变量) 进行优化。所以SMO先选取违背KKT条件程度最大的变量, 那么第二个变量应该选择使目标函数值减少最快的变量。SMO使用了一个启发式的方法, 当确定了第一个变量后, 选择使两个变量对应样本之间最大的变量作为第二个变量。直观来说, 更新两个差别很大的变量, 比起相似的变量, 会带给目标函数更大的变化。

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。对于每个样本点，要满足的KKT条件（请见1.2节）为：

$$\begin{cases} y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \geq 0 \\ 0 \leq \alpha_i \leq C \\ \alpha_i(1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0 \end{cases}$$

所以

$$\begin{cases} \alpha_i = 0 & \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \\ 0 < \alpha_i < C & \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \\ \alpha_i = C & \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 1 \end{cases}$$

一般来说，我们首先选择违反 $0 < \alpha_i < C \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$ 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 $\alpha_i = 0 \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) > 1$ 的点和 $\alpha_i = C \Rightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) < 1$ 的点。

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 α_1 ，第二个变量 α_2 的选择标准是让 $|E_1 - E_2|$ 有足够大的变化 (recall: $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$)。由于 α_1 确定时， 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 α_2 ，直到目标函数有足够的下降，如果所有的支持向量做 α_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 α_1 。

5.4 更新截距项 b

当我们更新了一对 α_i, α_j 之后都需要重新计算阈值 b ，因为 b 关系到 $f(\mathbf{x})$ 的计算，关系到下次优化的时候误差 E_i 的计算。

为了使得被优化的样本都满足KKT条件，

当 α_1^{new} 不在边界，即 $\alpha_1^{new} > 0$ ，根据KKT条件可知相应的数据点为支持向量，满足

$$y_1(\mathbf{w}^T \mathbf{x}_1 + b) = 1$$

其中 $f(\mathbf{x}_1) = \mathbf{w}^T \mathbf{x}_1 + b = \sum_{i=1}^N \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_1 \rangle + b = \sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_1) + b$ 。

上式两边同时乘上 y_1 ，由于 $y_1^2 = 1$ ，得到 $\sum_{i=1}^N \alpha_i y_i K_{i1} + b = y_1$ ，进而得到 b_1^{new} 的值：

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

其中上式的前两项可以写成：

$$y_1 - \sum_{i=3}^N \alpha_i y_i K_{i1} = -E_1 + \alpha_1^{old} y_1 K_{11} + \alpha_2^{old} y_2 K_{21} + w_0^{old}$$

当 $\alpha_2^{new} > 0$ ，可以得到 b_2^{new} 的表达式(推导同上)：

$$w_{0,2}^{new} = -E_2 - y_1 K_{12} (\alpha_1^{new} - \alpha_1^{old}) - y_2 K_{22} (\alpha_2^{new} - \alpha_2^{old}) + b_0^{old}$$

当 b_1^{new} 和 b_2^{new} 都有效的时候他们是相等的，即 $b^{new} = b_1^{new} = b_2^{new}$ 。

当两个乘子 α_1, α_2 都在边界上，且 $L \neq H$ 时， b_1 和 b_2 之间的值就是和KKT条件一的阈值一致，SMO选择他们的中点作为新的阈值：

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

得到了 b^{new} 我们需要更新 E_i ：

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

其中 S 是所有支持向量 x_j 的集合。

5.5 SMO小结

SMO算法是一个迭代优化算法。在每一个迭代步骤中，算法首先选取两个待更新的向量，此后分别计算它们的误差项，并根据上述结果计算出 α_2^{new} 和 α_1^{new} 。最后再根据SVM的定义计算出偏移量 w_0 。对于误差项而言，可以根据 α_1^{new} 、 α_2^{new} 和 w_0 的增量进行调整，而无需每次重新计算。具体的算法如下：

- 1 随机数初始化向量权重 α^0 , $t = 0$, 并计算偏移 b^0
- 2 初始化误差项 E_i
- 3 选取两个向量作为需要调整的点 4 令 $\alpha_2^{t+1} = \alpha_2^t + \frac{y_2(E_1 - E_2)}{\eta}$, 其中 $\eta = K_{11} + K_{22} - 2K_{12}$
- 5 如果 $\alpha_2^{t+1} > H$, 令 $\alpha_2^{t+1} = H$;
- 如果 $\alpha_2^t + 1 < L$, 令 $\alpha_2^{t+1} = L$ 。
- 6 令 $\alpha_1^{t+1} = \alpha_1^t + y_1 y_2 (\alpha_2^t - \alpha_2^t + 1)$
- 7 利用更新的 α_1^{t+1} 和 α_2^{t+1} , 修改 E_i 和 b^{t+1} 的值。
- 8 如果达到终止条件, 则停止算法, 否则 $t = t + 1$, 转3

SMO算法的终止条件可以所有向量均满足KKT条件：

$$\begin{aligned} \sum_{i=1}^N \alpha_i^{t+1} y_i &= 0, \\ 0 \leq \alpha_i^{t+1} \leq C, \quad i &= 1, 2 \dots N \\ \alpha_i^{t+1} = 0 \Rightarrow y_i f(\mathbf{x}_i) &\geq 1 \\ 0 \leq \alpha_i^{t+1} \leq C \Rightarrow y_i f(\mathbf{x}_i) &= 1 \\ \alpha_i^{t+1} = C \Rightarrow y_i f(\mathbf{x}_i) &\leq 1 \end{aligned}$$

或者目标函数 $W(\alpha)$ 增长率小于某个阈值, 即

$$\frac{W(\alpha^{t+1}) - W(\alpha^t)}{W(\alpha^t)} < \epsilon$$

与通常的分解算法比较, 尽管SMO可能需要更多的迭代次数, 但每次迭代的计算量比较小, 所以该算法表现出整理的快速收敛性, 且不需要存储核矩阵, 也没有矩阵运算。

SMO算法的Python实现：

台湾的林智仁教授写了一个封装SVM算法的[libsvm](#)库，当然scikit Learn你也集成了SVM模型。

6 支持向量回归 (SVR)

6.1 ϵ 不敏感损失函数

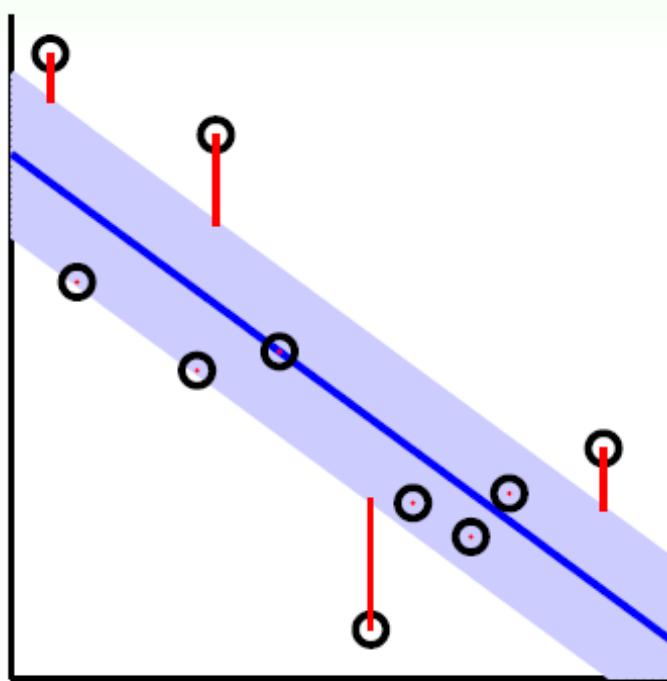
在SVM分类模型中，我们的目标函数是让 $\frac{1}{2} \|\mathbf{w}\|_2^2$ 最小，同时每个训练样本尽量远离自己类别一边的支持向量，即 $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ 。如果是加入一个松弛变量 $\xi_i \geq 0$ ，则目标函数是 $\frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N \xi_i$ ，对应的约束条件变成： $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$ 。

但在回归模型中，优化目标函数可以继续和SVM分类模型保持一致为 $\frac{1}{2} \|\mathbf{w}\|_2^2$ ，但是约束条件是让训练集中的每个点 (\mathbf{x}_i, y_i) 尽量拟合到一个线性模型 $y_i = \mathbf{w}^T \mathbf{x}_i + b$ 。对于一般的回归模型，我们是用均方差作为损失函数，但在SVM中我们定义一个常量 $\epsilon > 0$ ，对于某一个点 (\mathbf{x}_i, y_i) ，

如果 $|y_i - \mathbf{w}^T \mathbf{x}_i - b| \leq \epsilon$ ，则完全没有损失；

如果 $|y_i - \mathbf{w}^T \mathbf{x}_i - b| > \epsilon$ ，则对应的损失为 $|y_i - \mathbf{w}^T \mathbf{x}_i - b| - \epsilon$ 。

这个损失函数称为 ϵ 不敏感损失如下图所示。在蓝色条带里面的点都是没有损失的，但是外面的点的是有损失的，损失大小为红色线的长度。这个损失函数均方差损失函数不同。如果是均方差，那么只要 $y_i - \mathbf{w}^T \mathbf{x}_i - b$ ，那么就会有损失。



6.2 SVR

根据6.1小节定义的 ϵ 不敏感损失函数，得到SVR的目标函数：

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 \\ s.t. \quad & |y_i - \mathbf{w}^T \phi(\mathbf{x}_i) - b| \leq \epsilon, \quad i = 1, 2, \dots, N \end{aligned}$$

和SVM分类模型相似，回归模型也可以对每个样本 (\mathbf{x}_i, y_i) 加入松弛变量 $\xi_i \geq 0$ 。但是由于我们这里用的是绝对值，实际上是两个不等式，也就是说两边都需要松弛变量，我们定义为 ξ_i^\vee, ξ_i^\wedge ，则我们SVM回归模型的损失函数度量在加入松弛变量之后变为：

$$\begin{aligned} & \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N (\xi_i^\vee + \xi_i^\wedge) \\ s.t. \quad & -\epsilon - \xi_i^\vee \leq y_i - \mathbf{w}^T \mathbf{x}_i - b \leq \epsilon + \xi_i^\wedge, \quad i = 1, 2, \dots, N \\ & \xi_i^\vee \geq 0, \quad \xi_i^\wedge \geq 0, \quad i = 1, 2, \dots, N \end{aligned}$$

依然和SVM分类模型相似，我们可以用拉格朗日乘子法得到拉格朗日函数：

$$\begin{aligned} L(\mathbf{w}, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge) = & \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^N (\xi_i^\vee + \xi_i^\wedge) \\ & + \sum_{i=1}^N \alpha_i^\vee (-\epsilon - \xi_i^\vee - y_i + \mathbf{w}^T \mathbf{x}_i + b) \\ & + \sum_{i=1}^N \alpha_i^\wedge (y_i - \mathbf{w}^T \mathbf{x}_i - b - \epsilon - \xi_i^\wedge) \\ & - \sum_{i=1}^N \mu_i^\vee \xi_i^\vee - \sum_{i=1}^m \mu_i^\wedge \xi_i^\wedge \end{aligned}$$

首先我们拉格朗日函数 L 对于 $\mathbf{w}, b, \xi_i^\vee, \xi_i^\wedge$ 的偏导数：

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^\vee) \mathbf{x}_i$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^\vee) = 0$$

$$\frac{\partial L}{\partial \xi_i^\vee} = 0 \Rightarrow C - \alpha_i^\vee - \mu_i^\vee = 0$$

$$\frac{\partial L}{\partial \xi_i^\wedge} = 0 \Rightarrow C - \alpha_i^\wedge - \mu_i^\wedge = 0$$

将上面4个式子带入 $L(\mathbf{w}, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge)$ 进行消元，最终得到的对偶形式为：

$$\begin{aligned}
& \max_{\alpha^V, \alpha^\wedge} - \sum_{i=1}^N (\epsilon - y_i) \alpha_i^\wedge + (\epsilon + y_i) \alpha_i^V - \sum_{j=1}^N \frac{1}{2} (\alpha_j^\wedge - \alpha_j^V)(\alpha_j^\wedge - \alpha_j^V) \mathbf{x}_i^T \mathbf{x}_j \\
s.t., \quad & \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^V) = 0 \\
& 0 < \alpha_i^V < C, \quad i = 1, 2, \dots, N \\
& 0 < \alpha_i^\wedge < C, \quad i = 1, 2, \dots, N
\end{aligned}$$

对目标函数取负号，求最小值可以得到和SVM分类模型类似的求极小值的目标函数如下：

$$\begin{aligned}
& \min_{\alpha^V, \alpha^\wedge} \sum_{j=1}^N \frac{1}{2} (\alpha_j^\wedge - \alpha_j^V)(\alpha_j^\wedge - \alpha_j^V) \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^N (\epsilon - y_i) \alpha_i^\wedge + (\epsilon + y_i) \alpha_i^V - \\
s.t., \quad & \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^V) = 0 \\
& 0 \leq \alpha_i^V \leq C, \quad i = 1, 2, \dots, N \\
& 0 \leq \alpha_i^\wedge \leq C, \quad i = 1, 2, \dots, N
\end{aligned}$$

对应的KKT条件为：

$$\left\{
\begin{array}{l}
-\epsilon - \xi_i^V \leq y_i - \mathbf{w}^T \mathbf{x}_i - b \leq \epsilon + \xi_i^\wedge, \quad i = 1, 2, \dots, N \\
\xi_i^V \geq 0 \\
\xi_i^\wedge \geq 0 \\
(C - \alpha_i^V) \xi_i^V = 0 \\
(C - \alpha_i^\wedge) \xi_i^\wedge = 0
\end{array}
\right.$$

根据 $\alpha_i^V (\epsilon + \xi_i^V + y_i - \mathbf{w}^T \mathbf{x}_i - b) = 0$ 可以看出，只有 $\epsilon + \xi_i^V + y_i - \mathbf{w}^T \mathbf{x}_i - b = 0$ 时， α_i^V 才能取非0值。换句话说，仅当样本 (\mathbf{x}_i, y_i) 落在 ϵ -间隔中， $\alpha_i^V, \alpha_i^\wedge$ 才能取非0值。此外， $\epsilon + \xi_i^V + y_i - \mathbf{w}^T \mathbf{x}_i - b = 0$ 和 $\epsilon + \xi_i^\wedge + y_i + \mathbf{w}^T \mathbf{x}_i + b = 0$ 不能同时成立，所以 $\alpha_i^V, \alpha_i^\wedge$ 至少一个为0。这些样本我们称为支持向量。

求出 $\alpha_i^V, \alpha_i^\wedge$ 后，将 $\mathbf{w} = \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^V) \mathbf{x}_i$ 代入 $f(\mathbf{x})$ ，得到回归函数为

$$\begin{aligned}
f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\
&= \left(\sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^V) \mathbf{x}_i \right)^T \mathbf{x} + b
\end{aligned}$$

如果采用核函数，回归函数为

$$f(\mathbf{x}) = \sum_{i=1}^N (\alpha_i^\wedge - \alpha_i^\vee) \kappa(\mathbf{x}_i, \mathbf{x}) + b$$

7 Scikit learn 中的SVM实现

略。

8 SVM 算法小结

SVM算法是一个很优秀的算法，在集成学习和深度神经网络算法没有表现出优越性能前，SVM基本占据了分类模型的统治地位。目前则是在大数据时代的大样本背景下,SVM由于其在大样本时超级大的计算量，热度有所下降，但是仍然是一个常用的机器学习算法。

SVM算法的主要优点有：

1. 解决高维特征的分类问题和回归问题很有效,在特征维度大于样本数时依然有很好的效果。
2. 仅仅使用一部分支持向量来做超平面的决策，无需依赖全部数据。
3. 有大量的核函数可以使用，从而可以很灵活的来解决各种非线性的分类回归问题。
4. 样本量不是海量数据的时候，分类准确率高，泛化能力强。

SVM算法的主要缺点有：

1. SVM在样本量非常大，核函数映射维度非常高时，计算量过大，不太适合使用。
2. 非线性问题的核函数的选择没有通用标准，难以选择一个合适的核函数。
3. SVM对缺失数据敏感。