【机器学习】支持向量机 SVM（非常详细）

阿泽

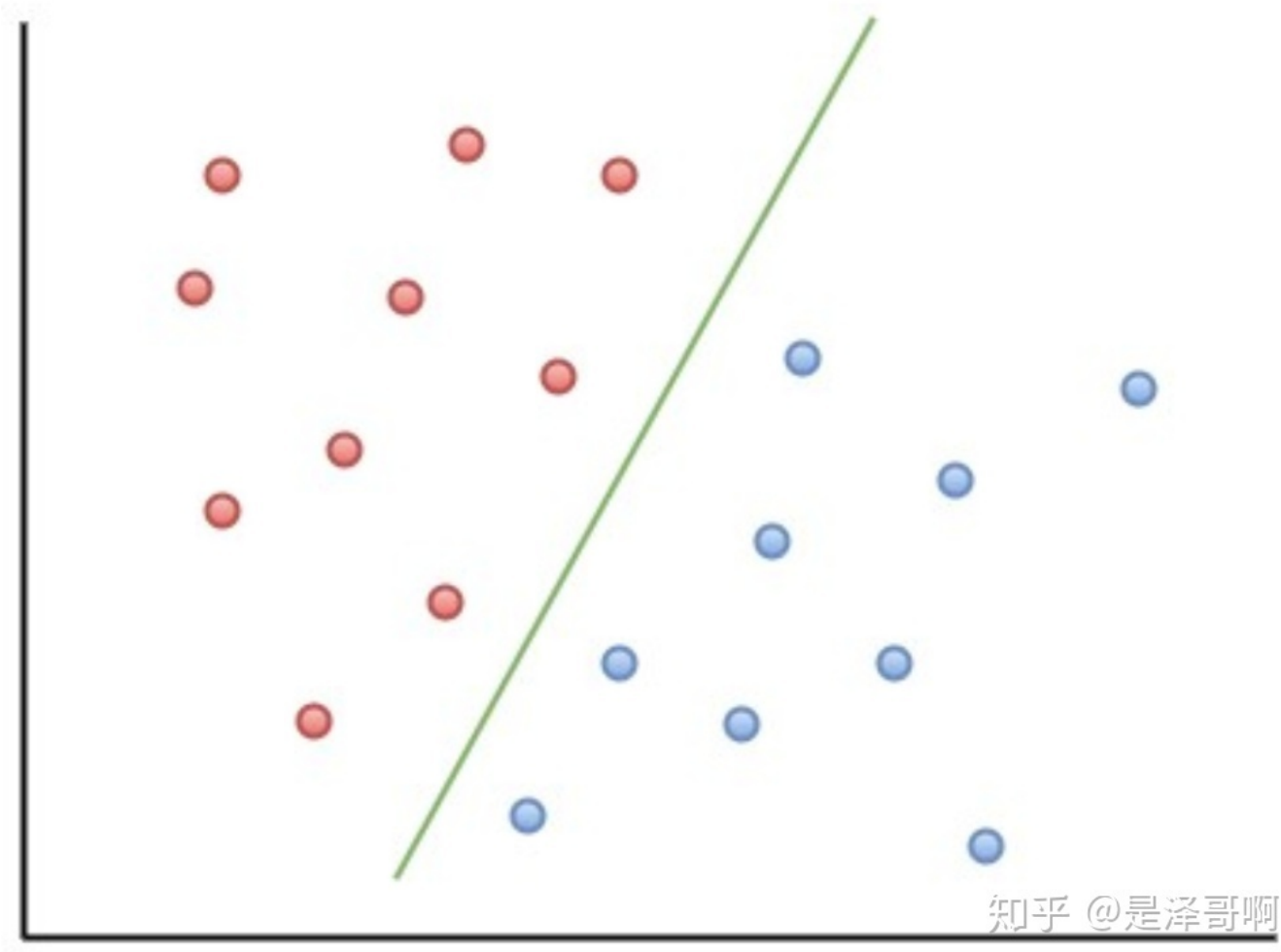
复旦大学 计算机技术硕士

SVM 是一个非常优雅的算法，具有完善的数学理论，虽然如今工业界用到的不多，但还是决定花点时间去写篇文章整理一下。

1. 支持向量

1.1 线性可分

首先我们先来了解下什么是线性可分。



在二维空间上，两类点被一条直线完全分开叫做线性可分。

严格的数学定义是：

D_0 和 D_1 是 n 维欧氏空间中的两个点集。如果存在 n 维向量 w 和实数 b，使得所有属于 D_0 的点 x_i 都有 $wx_i + b > 0$，而对于所有属于 D_1 的点 x_j 则有 $wx_j + b < 0$，则我们称 D_0 和 D_1 线性可分。

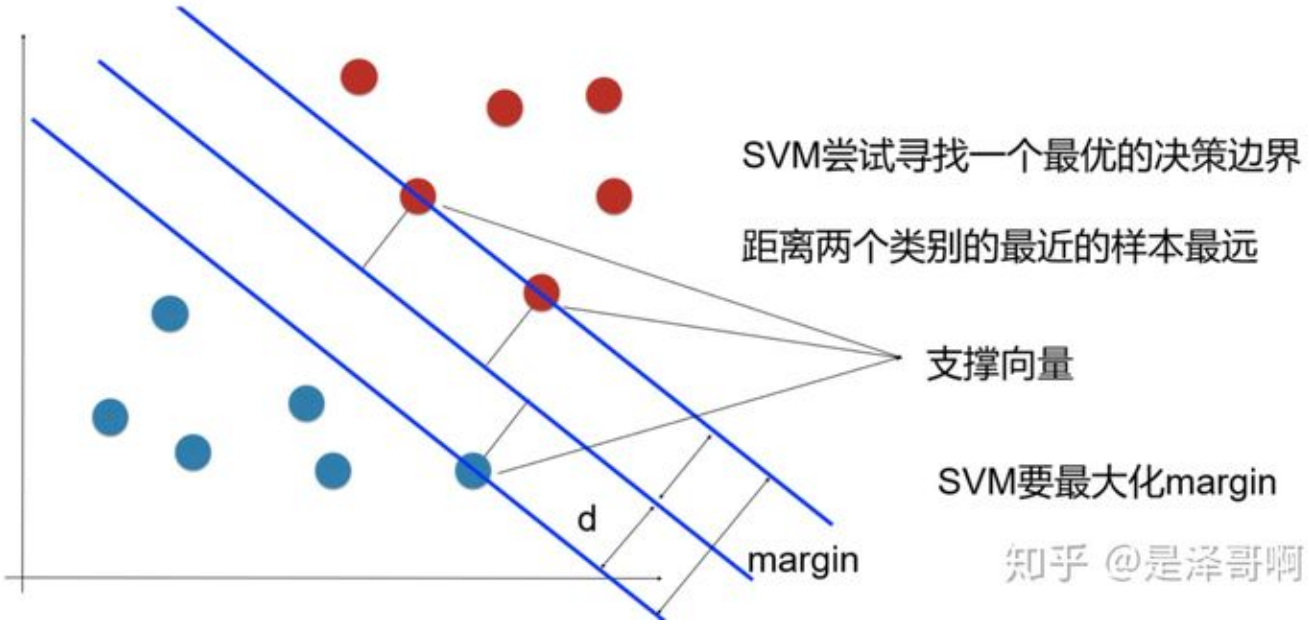
1.2 最大间隔超平面

从二维扩展到多维空间中时，将 D_0 和 D_1 完全正确地划分开的 $wx + b = 0$ 就成了一个超平面。

为了使这个超平面更具鲁棒性，我们会去找最佳超平面，以最大间隔把两类样本分开的超平面，也称之为最大间隔超平面。

- 两类样本分别分割在该超平面的两侧；
- 两侧距离超平面最近的样本点到超平面的距离被最大化了。

1.3 支持向量



样本中距离超平面最近的一些点，这些点叫做支持向量。

1.4 SVM 最优化问题

SVM 想要的就是找到各类样本点到超平面的距离最远，也就是找到最大间隔超平面。任意超平面可以用下面这个线性方程来描述：

$$w^T x + b = 0$$

二维空间点 (x, y) 到直线 $Ax + By + C = 0$ 的距离公式是：

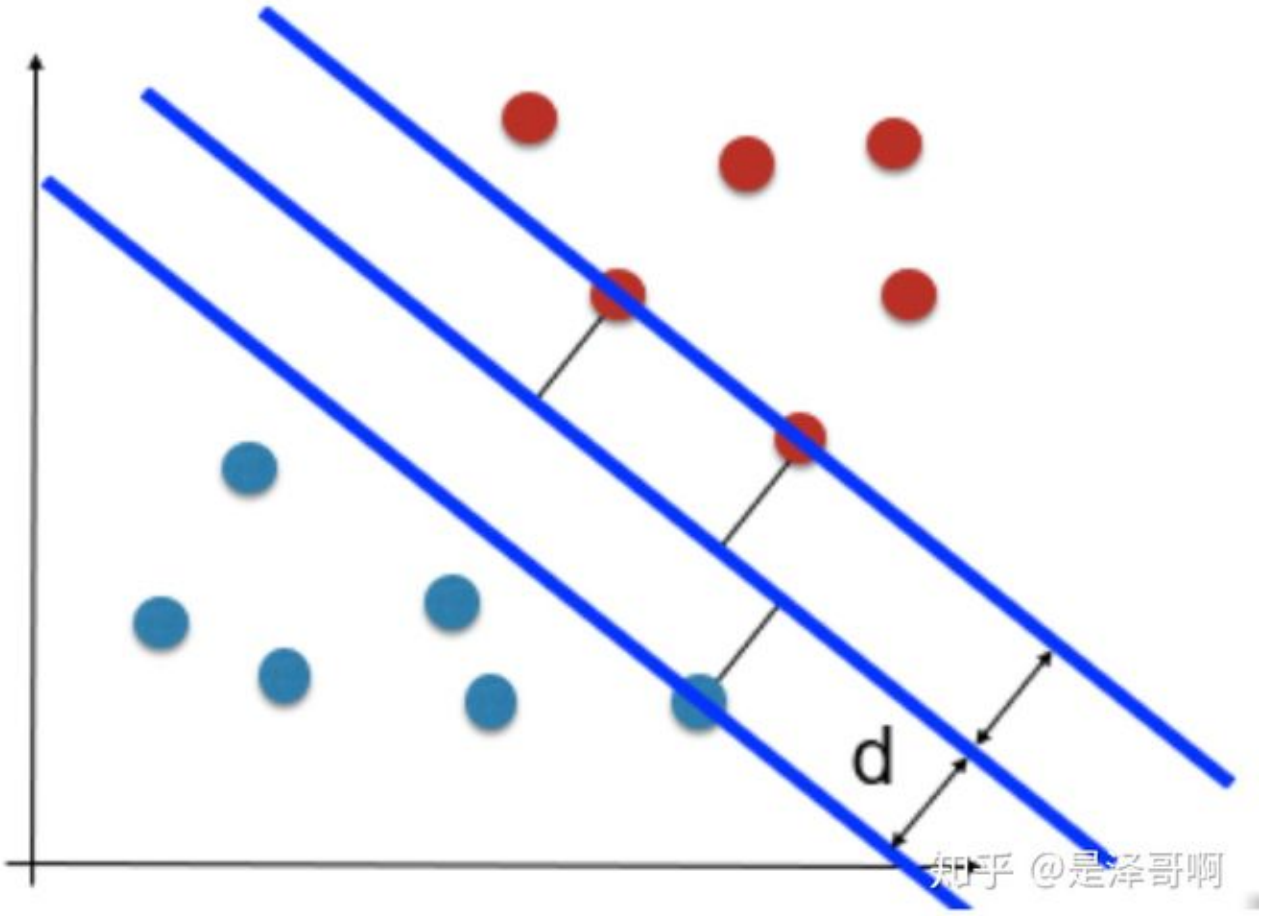
$$\frac{|Ax + By + C|}{\sqrt{A^2 + B^2}}$$

扩展到 n 维空间后，点 $x = (x_1, x_2 \dots x_n)$ 到直线 $w^T x + b = 0$ 的距离为：

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

其中 $||w|| = \sqrt{w_1^2 + \dots w_n^2}$ 。

如图所示，根据支持向量的定义我们知道，支持向量到超平面的距离为 d，其他点到超平面的距离大于 d。



于是我们有这样的一个公式：

$$\begin{cases} \dfrac{w^T x+b}{||w||} \geq d \quad y=1 \\ \dfrac{w^T x+b}{||w||} \leq -d \quad y=-1 \end{cases}$$

稍作转化可以得到：

$$\begin{cases} \dfrac{w^T x+b}{||w||d} \geq 1 \quad y=1 \\ \dfrac{w^T x+b}{||w||d} \leq -1 \quad y=-1 \end{cases}$$

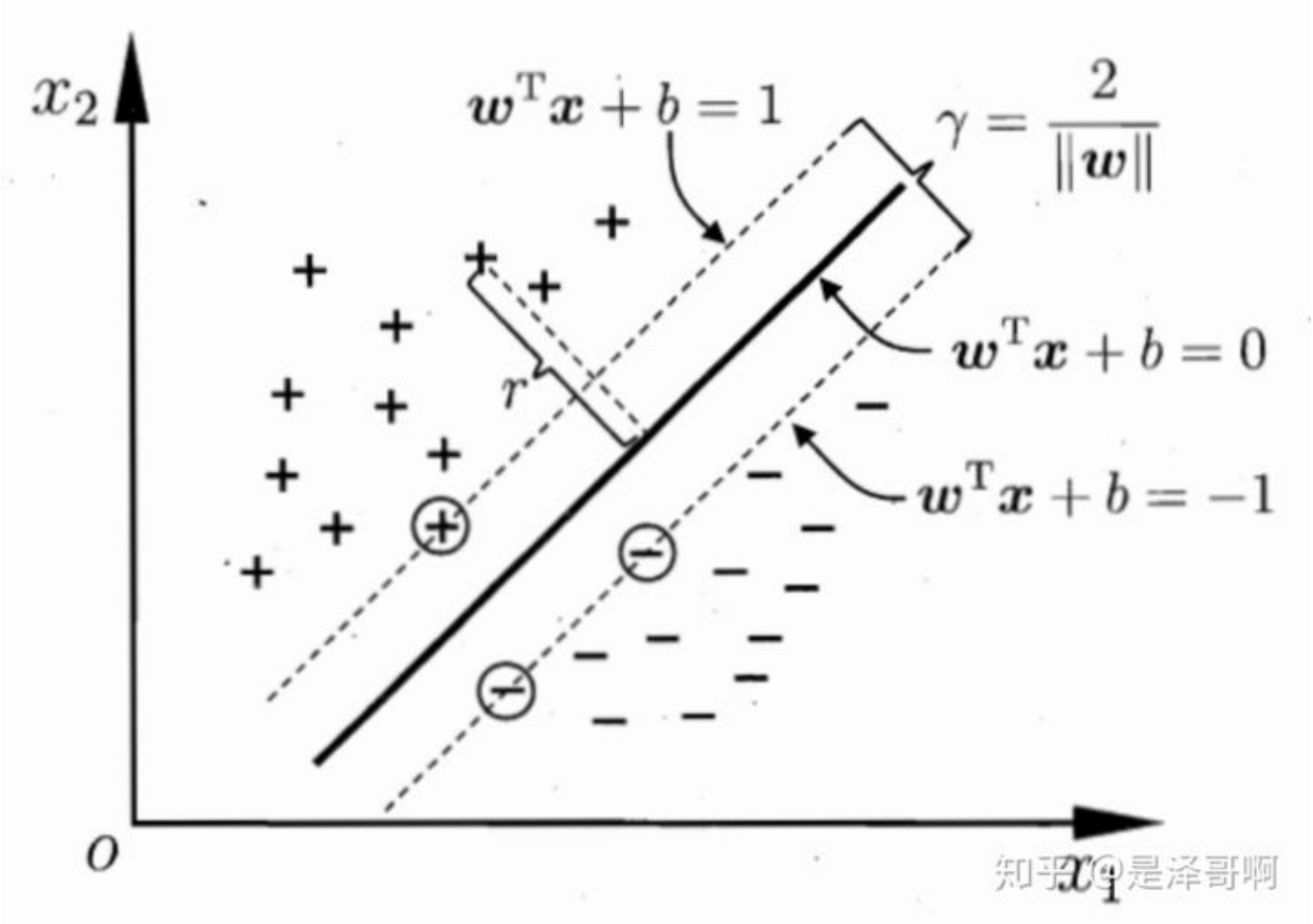
$||w||d$ 是正数，我们暂且令它为 1（之所以令它等于 1，是为了方便推导和优化，且这样做对目标函数的优化没有影响），故：

$$\begin{cases} w^T x+b \geq 1 \quad y=1 \\ w^T x+b \leq -1 \quad y=-1 \end{cases}$$

将两个方程合并，我们可以简写为：

$$y(w^T x+b) \geq 1$$

至此我们就可以得到最大间隔超平面的上下两个超平面：



每个支持向量到超平面的距离可以写为：

$$d=\frac{|w^Tx+b|}{||w||}$$

由上述 $y(w^Tx+b)>1>0$ 可以得到 $y(w^Tx+b)=|w^Tx+b|$ ，所以我们得到：

$$d=\frac{y(w^Tx+b)}{||w||}$$

最大化这个距离：

$$\max 2*\frac{y(w^Tx+b)}{||w||}$$

这里乘上 2 倍也是为了后面推导，对目标函数没有影响。刚刚我们得到支持向量 $y(w^Tx+b)=1$ ，所以我们得到：

$$\max \frac{2}{||w||}$$

再做一个转换：

$$\min \frac{1}{2}||w||$$

为了方便计算（去除 $||w||$ 的根号），我们有：

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

所以得到的最优化问题是：

$$\min \frac{1}{2}||w||^2 \ s.t. \quad y_i（w^T x_i + b） \geq 1$$

2. 对偶问题

2.1 拉格朗日乘数法

2.1.1 等式约束优化问题

本科高等数学学的拉格朗日程数法是等式约束优化问题：

$$\min f(x_1, x_2, \ldots, x_n)$$
$$s.t. \quad h_k(x_1, x_2, \ldots, x_n) = 0 \quad k = 1, 2, \ldots, l$$

我们令 $L(x, \lambda) = f(x) + \sum_{k=1}^{l} \lambda_k h_k(x)$ ，函数 $L(x, y)$ 称为 Lagrange 函数，参数 λ 称为 Lagrange 乘子**没有非负要求。**

利用必要条件找到可能的极值点：

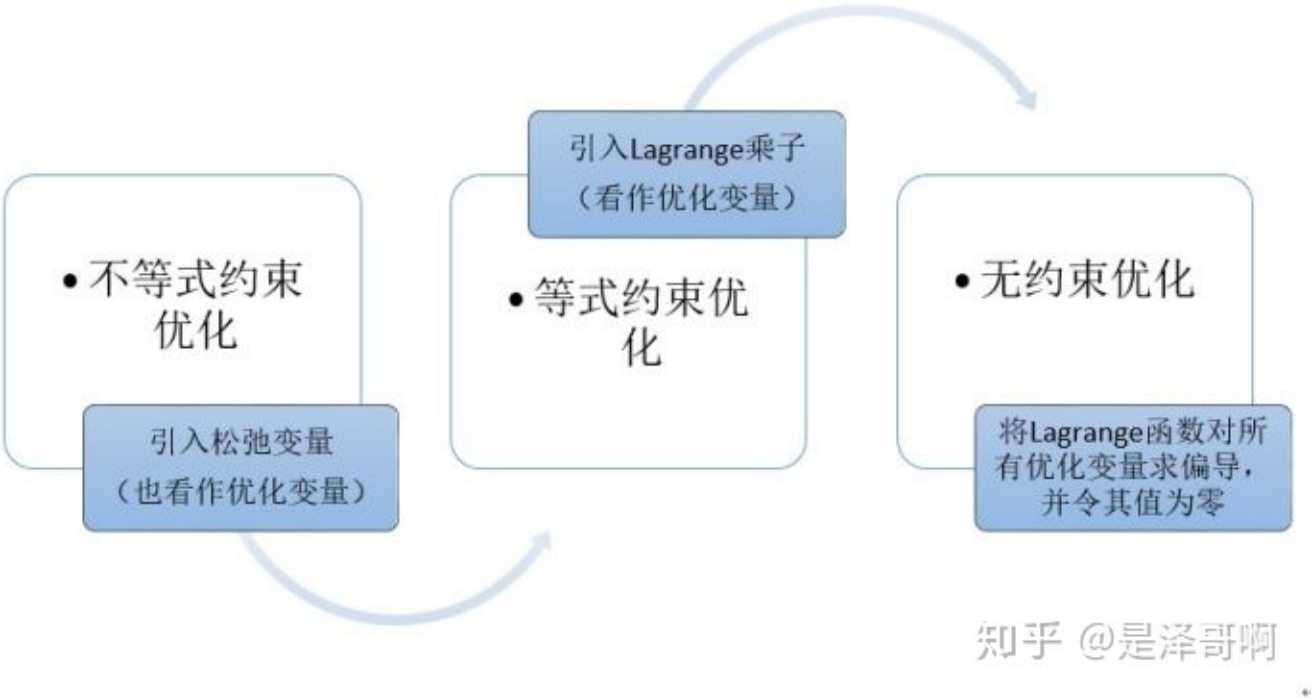
$$\begin{cases} \dfrac{\partial L}{\partial x_i} = 0 \quad i = 1, 2, \ldots, n \\ \dfrac{\partial L}{\partial \lambda_k} = 0 \quad k = 1, 2, \ldots, l \end{cases}$$

具体是否为极值点需根据问题本身的具体情况检验。这个方程组称为等式约束的极值必要条件。

等式约束下的 Lagrange 乘数法引入了 l 个 Lagrange 乘子，我们将 x_i 与 λ_k 一视同仁，把 λ_k 也看作优化变量，共有 $(n+l)$ 个优化变量。

2.1.2 不等式约束优化问题

而我们现在面对的是不等式优化问题，针对这种情况其主要思想是将不等式约束条件转变为等式约束条件，引入松弛变量，将松弛变量也是为优化变量。



以我们的例子为例：

$$\begin{aligned} & min f(w) = min \frac{1}{2} ||w||^2 \\ s.t. \quad & g_i(w) = 1 - y_i(w^T x_i + b) \leq 0 \end{aligned}$$

我们引入松弛变量 a_i^2 得到 $h_i(w, a_i) = g_i(w) + a_i^2 = 0$ 。这里加平方主要为了不再引入新的约束条件，如果只引入 a_i 那我们必须要保证 $a_i \geq 0$ 才能保证 $h_i(w, a_i) = 0$ ，这不符合我们的意愿。

由此我们将不等式约束转化为了等式约束，并得到 Lagrange 函数：

$$\begin{aligned} L(w,\lambda,a) &= f(w)+\sum_{i=1}^{n}\lambda_i h_i(w) \\ &= f(w)+\sum_{i=1}^{n}\lambda_i[g_i(w)+a_i^2] \quad \lambda_i \geq 0 \end{aligned}$$

由等式约束优化问题极值的必要条件对其求解，联立方程：

$$\begin{cases} \dfrac{\partial L}{\partial w_i} = \dfrac{\partial f}{\partial w_i}+\sum\limits_{i=1}^{n}\lambda_i\dfrac{\partial g_i}{\partial w_i}=0, \\ \dfrac{\partial L}{\partial a_i} = 2\lambda_i a_i = 0, \\ \dfrac{\partial L}{\partial \lambda_i} = g_i(w)+a_i^2 = 0, \\ \lambda_i \geq 0 \end{cases}$$

（为什么取 $\lambda_i \geq 0$ ，可以通过几何性质来解释，有兴趣的同学可以查下 KKT 的证明）。

针对 $\lambda_i a_i = 0$ 我们有两种情况：

情形一： $\lambda_i = 0, a_i \neq 0$

由于 $\lambda_i = 0$ ，因此约束条件 $g_i(w)$ 不起作用，且 $g_i(w) < 0$

情形二： $\lambda_i \neq 0, a_i = 0$

此时 $g_i(w) = 0$ 且 $\lambda_i > 0$ ，可以理解为约束条件 $g_i(w)$ 起作用了，且 $g_i(w) = 0$

综合可得： $\lambda_i g_i(w) = 0$ ，且在约束条件起作用时 $\lambda_i > 0, g_i(w) = 0$ ；约束不起作用时 $\lambda_i = 0, g_i(w) < 0$

由此方程组转换为：

$$\begin{cases} \dfrac{\partial L}{\partial w_i} = \dfrac{\partial f}{\partial w_i} + \sum\limits_{j=1}^{n} \lambda_j \dfrac{\partial g_j}{\partial w_i} = 0, \\ \lambda_i g_i(w) = 0, \\ g_i(w) \le 0 \\ \lambda_i \ge 0 \end{cases}$$

以上便是不等式约束优化优化问题的 **KKT(Karush-Kuhn-Tucker) 条件**， λ_i 称为 KKT 乘子。

这个式子告诉了我们什么事情呢?

直观来讲就是，支持向量 $g_i(w) = 0$ ，所以 $\lambda_i > 0$ 即可。而其他向量 $g_i(w) < 0, \lambda_i = 0$ 。

我们原本问题时要求： $min\frac{1}{2}||w||^2$ ，即求 $minL(w, \lambda, a)$

$$\begin{aligned} L(w, \lambda, a) &= f(w) + \sum_{i=1}^{n} \lambda_i [g_i(w) + a_i^2] \\ &= f(w) + \sum_{i=1}^{n} \lambda_i g_i(w) + \sum_{i=1}^{n} \lambda_i a_i^2 \end{aligned}$$

由于 $\sum\limits_{i=1}^{n} \lambda_i a_i^2 \ge 0$ ，故我们将问题转换为： $minL(w, \lambda)$ ：

$$L(w, \lambda) = f(w) + \sum_{i=1}^{n} \lambda_i g_i(w)$$

假设找到了最佳参数是的目标函数取得了最小值 p。即 $\frac{1}{2}||w||^2 = p$ 。而根据 $\lambda_i \ge 0$ ，可知 $\sum\limits_{i=1}^{n} \lambda_i g_i(w) \le 0$ ，因此 $L(w, \lambda) \le p$ ，为了找到最优的参数 λ ，使得 $L(w, \lambda)$ 接近 p，故问题转换为出 $\max\limits_{\lambda} L(w, \lambda)$ 。

故我们的最优化问题转换为:

$$\min_{w}\max_{\lambda} L(w,\lambda)$$
$$s.t. \quad \lambda_i \geq 0$$

出了上面的理解方式，我们还可以有另一种理解方式： 由于 $\lambda_i \geq 0$ ，

$$\max_{\lambda} L(w,\lambda) = \begin{cases} \infty & g_i(w) \geq 0 \\ \frac{1}{2}||w||^2 & g_i(w) \leq 0 \end{cases}$$

所以 $\min(\infty, \frac{1}{2}||w||^2) = \frac{1}{2}||w||^2$ ，所以转化后的式子和原来的式子也是一样的。

2.2 强对偶性

对偶问题其实就是将：

$$\min_{w}\max_{\lambda} L(w,\lambda)$$
$$s.t. \quad \lambda_i \geq 0$$

变成了：

$$\max_{\lambda}\min_{w} L(w,\lambda)$$
$$s.t. \quad \lambda_i \geq 0$$

假设有个函数 f 我们有：

$$\min\max f \geq \max\min f$$

也就是说，最大的里面挑出来的最小的也要比最小的里面挑出来的最大的要大。这关系实际上就是弱对偶关系，而强对偶关系是当等号成立时，即：

$$\min\max f = \max\min f$$

如果 f 是凸优化问题，强对偶性成立。而我们之前求的 KKT 条件是强对偶性的**充要条件**。

3. SVM 优化

我们已知 SVM 优化的主问题是：

$$\min_{w} \frac{1}{2}||w||^2$$
$$s.t. \quad g_i(w,b) = 1 - y_i（w^T x_i + b） \leq 0, \quad i = 1, 2, \ldots, n$$

那么求解线性可分的 SVM 的步骤为：

步骤 1：

构造拉格朗日函数：

$$\min_{w,b} \max_{\lambda} L(w,b,\lambda) = \frac{1}{2}||w||^2 + \sum_{i=1}^{n} \lambda_i [1 - y_i(w^T x_i + b)]$$
$$s.t. \quad \lambda_i \geq 0$$

步骤 2：

利用强对偶性转化：

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

现对参数 w 和 b 求偏导数：

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \lambda_i x_i y_i = 0$$
$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \lambda_i y_i = 0$$

得到：

$$\sum_{i=1}^{n}\lambda_i x_i y_i = w$$

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

我们将这个结果带回到函数中可得：

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

也就是说：

$$\min_{w,b} L(w,b,\lambda) = \sum_{j=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j (x_i \cdot x_j)$$

步骤 3：

由步骤 2 得：

$$\max_{\lambda}[\sum_{j=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j (x_i \cdot x_j)]$$

$$s.t. \quad \sum_{i=1}^{n}\lambda_i y_i = 0 \quad \lambda_i \geq 0$$

我们可以看出来这是一个二次规划问题，问题规模正比于训练样本数，我们常用 SMO(Sequential Minimal Optimization) 算法求解。

SMO(Sequential Minimal Optimization)，序列最小优化算法，其核心思想非常简单：每次只优化一个参数，其他参数先固定住，仅求当前这个优化参数的极值。我们来看一下 SMO 算法在 SVM 中的应用。

我们刚说了 SMO 算法每次只优化一个参数，但我们的优化目标有约束条件： $\sum_{i=1}^{n}\lambda_i y_i = 0$ ，没法一次只变动一个参数。所以我们选择了一次选择两个参数。具体步骤为：

1. 选择两个需要更新的参数 λ_i 和 λ_j ，固定其他参数。于是我们有以下约束：

这样约束就变成了：

$$\lambda_i y_i + \lambda_j y_j = c \quad \lambda_i \geq 0, \lambda_j \geq 0$$

其中 $c = -\sum_{k\neq i,j}\lambda_k y_k$ ，由此可以得出 $\lambda_j = \frac{c - \lambda_i y_i}{y_j}$ ，也就是说我们可以用 λ_i 的表达式代替 λ_j 。这样就相当于把目标问题转化成了仅有一个约束条件的最优化问题，仅有的约束是 $\lambda_i \geq 0$ 。

2. 对于仅有一个约束条件的最优化问题，我们完全可以在 λ_i 上对优化目标求偏导，令导数为零，从而求出变量值 $\lambda_{i_{new}}$ ，然后根据 $\lambda_{i_{new}}$ 求出 $\lambda_{j_{new}}$ 。

3. 多次迭代直至收敛。

通过 SMO 求得最优解 λ^* 。

步骤 4：

我们求偏导数时得到：

$$w = \sum_{i=1}^{m}\lambda_i y_i x_i$$

由上式可求得 w。

我们知道所有 $\lambda_i > 0$ 对应的点都是支持向量，我们可以随便找个支持向量，然后带入： $y_s(wx_s + b) = 1$ ，求出 b 即可，

两边同乘 y_s ，得 $y_s^2(wx_s + b) = y_s$

因为 $y_s^2 = 1$ ，所以： $b = y_s - wx_s$

为了更具鲁棒性，我们可以求得支持向量的均值：

$$b = \frac{1}{|S|} \sum_{s \in S}(y_s - wx_s)$$

步骤 5： w 和 b 都求出来了，我们就能构造出最大分割超平面： $w^T x + b = 0$

分类决策函数： $f(x) = sign(w^T x + b)$

其中 $sign(\cdot)$ 为阶跃函数：

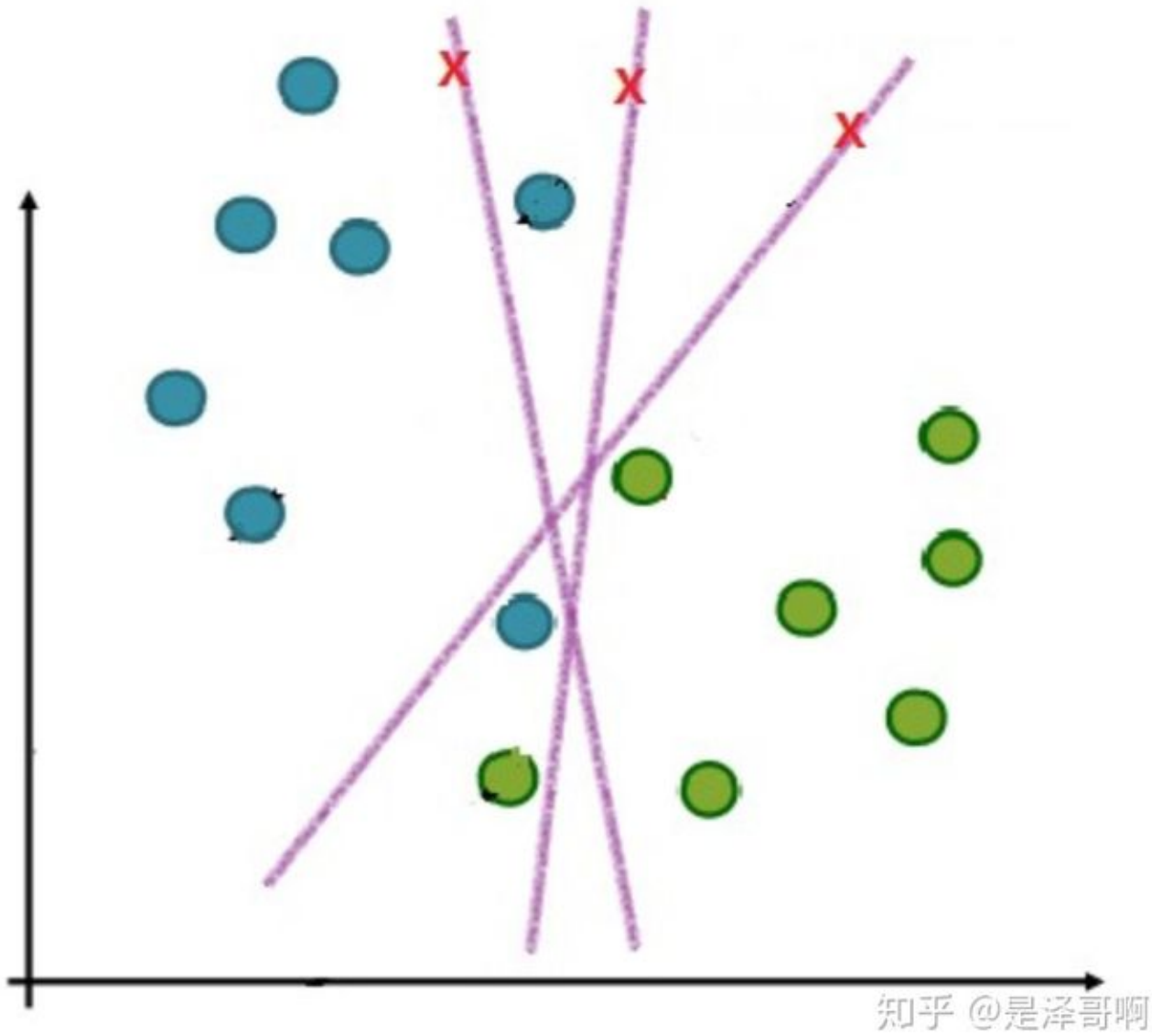
$$sign(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

将新样本点导入到决策函数中既可得到样本的分类。

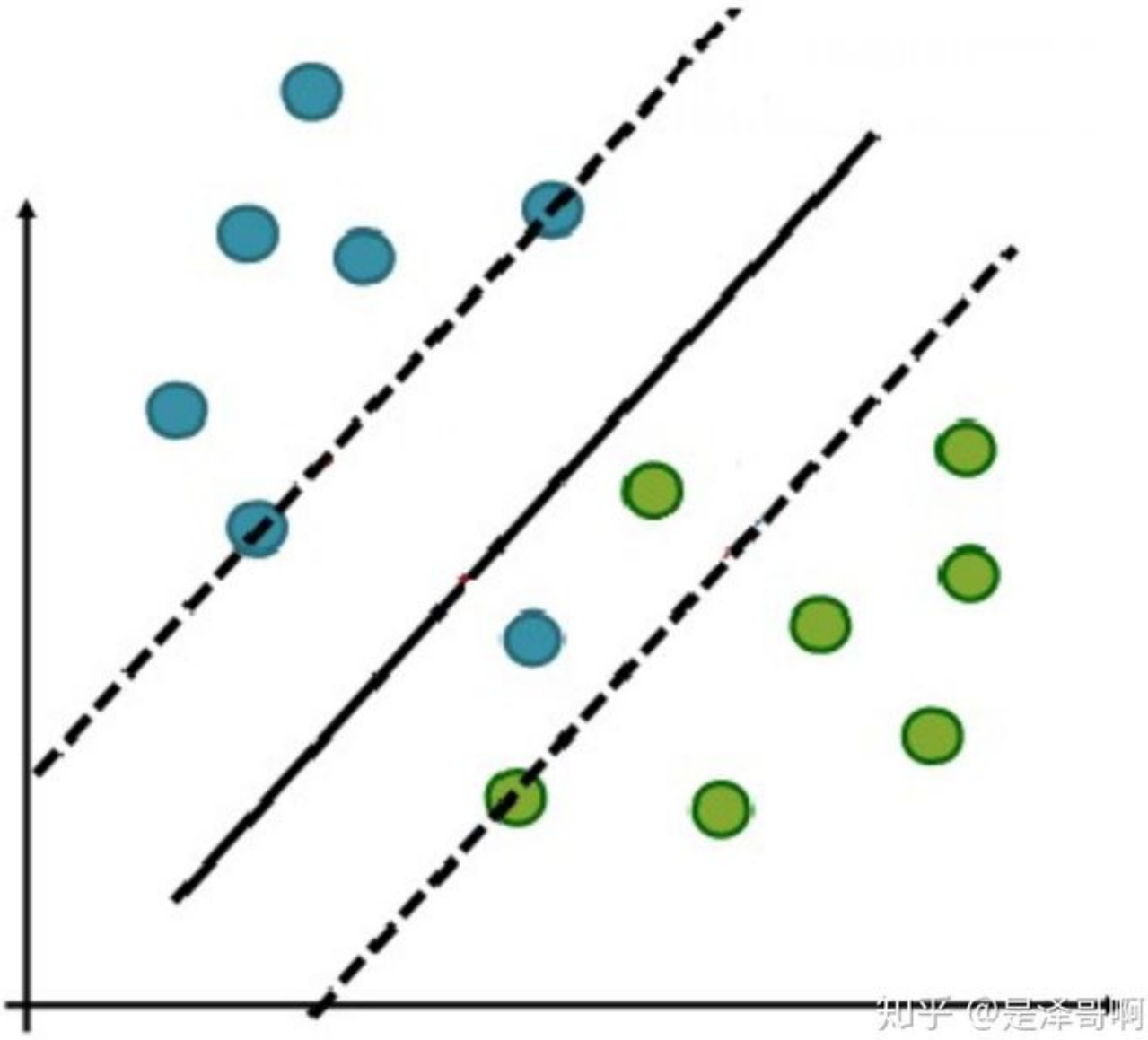
4. 软间隔

4.1 解决问题

在实际应用中，完全线性可分的样本是很少的，如果遇到了不能够完全线性可分的样本，我们应该怎么办？比如下面这个：



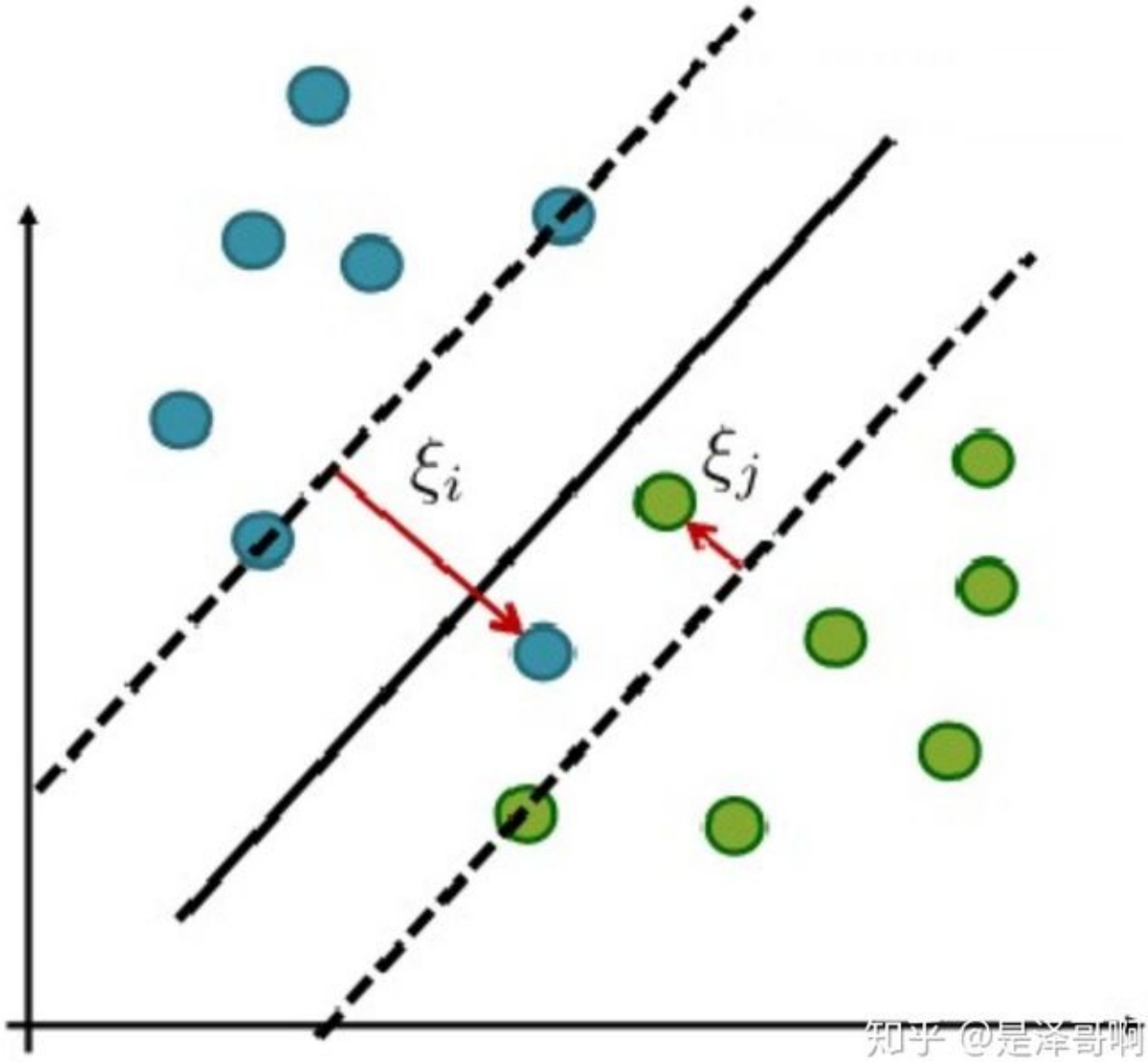
于是我们就有了软间隔，相比于硬间隔的苛刻条件，我们允许个别样本点出现在间隔带里面，比如：



我们允许部分样本点不满足约束条件：

$$1-y_i(w^Tx_i+b)\leq 0$$

为了度量这个间隔软到何种程度，我们为每个样本引入一个松弛变量 ξ_i ，令 $\xi_i\geq 0$ ，且 $1-y_i(w^Tx_i+b)-\xi_i\leq 0$ 。对应如下图所示：



4.2 优化目标及求解

增加软间隔后我们的优化目标变成了：

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

$$s.t. \quad g_i(w,b) = 1 - y_i(w^T x_i + b) - \xi_i \leq 0, \quad \xi_i \geq 0, \quad i = 1, 2, \ldots, n$$

其中 C 是一个大于 0 的常数，可以理解为错误样本的惩罚程度，若 C 为无穷大， ξ_i 必然无穷小，如此一来线性 SVM 就又变成了线性可分 SVM；当 C 为有限值的时候，才会允许部分样本不遵循约束条件。

接下来我们将针对新的优化目标求解最优化问题：

步骤 1：

构造拉格朗日函数：

$$\min_{w,b,\xi}\max_{\lambda,\mu} L(w,b,\xi,\lambda,\mu)=\frac{1}{2}||w||^2+C\sum_{i=1}^{m}\xi_i+\sum_{i=1}^{n}\lambda_i[1-\xi_i-y_i(w^Tx_i+b)]-\sum_{i=1}^{n}\mu_i\xi_i$$
$$s.t.\quad \lambda_i\geq 0\quad \mu_i\geq 0$$

其中 λ_i 和 μ_i 是拉格朗日乘子，w、b 和 ξ_i 是主问题参数。

根据强对偶性，将对偶问题转换为：

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

步骤 2：

分别对主问题参数w、b 和 ξ_i 求偏导数，并令偏导数为 0，得出如下关系：

$$w=\sum_{i=1}^{m}\lambda_i y_i x_i$$
$$0=\sum_{i=1}^{m}\lambda_i y_i$$
$$C=\lambda_i+\mu_i$$

将这些关系带入拉格朗日函数中，得到：

$$\min_{w,b,\xi} L(w,b,\xi,\lambda,\mu)=\sum_{j=1}^{n}\lambda_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j(x_i\cdot x_j)$$

最小化结果只有 λ 而没有 μ ，所以现在只需要最大化 λ 就好：

$$\max_{\lambda}[\sum_{j=1}^{n}\lambda_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j(x_i\cdot x_j)]$$
$$s.t.\quad \sum_{i=1}^{n}\lambda_i y_i=0,\quad \lambda_i\geq 0,\quad C-\lambda_i-\mu_i=0$$

我们可以看到这个和硬间隔的一样，只是多了个约束条件。

然后我们利用 SMO 算法求解得到拉格朗日乘子 λ^* 。

步骤 3：

$$w = \sum_{i=1}^{m} \lambda_i y_i x_i$$

$$b = \frac{1}{|S|} \sum_{s \in S} (y_s - w x_s)$$

然后我们通过上面两个式子求出 w 和 b，最终求得超平面 $w^T x + b = 0$ ，

这边要注意一个问题，在间隔内的那部分样本点是不是支持向量?

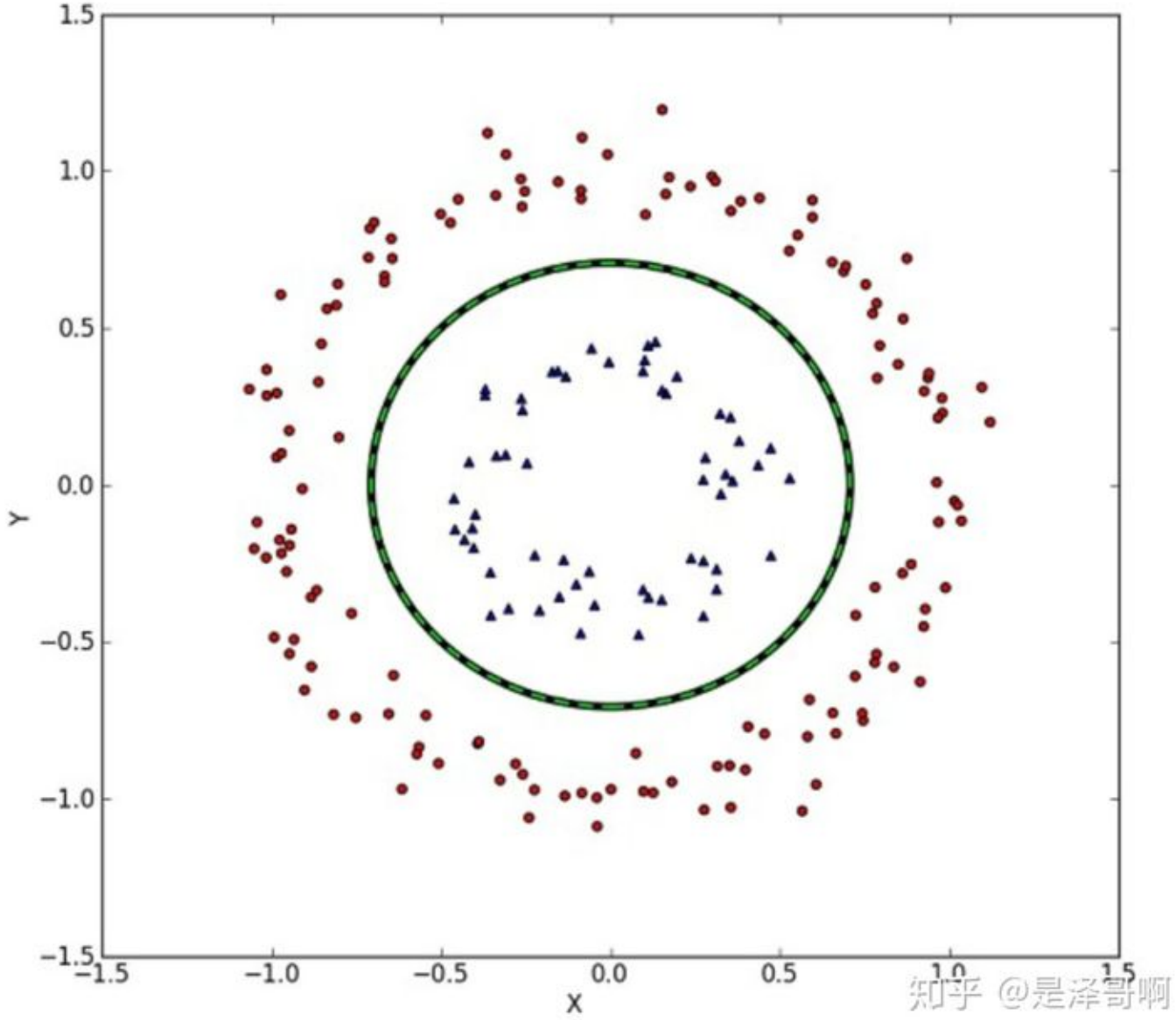
我们可以由求参数 w 的那个式子可看出，只要 $\lambda_i > 0$ 的点都能够影响我们的超平面，因此都是支持向量。

5. 核函数

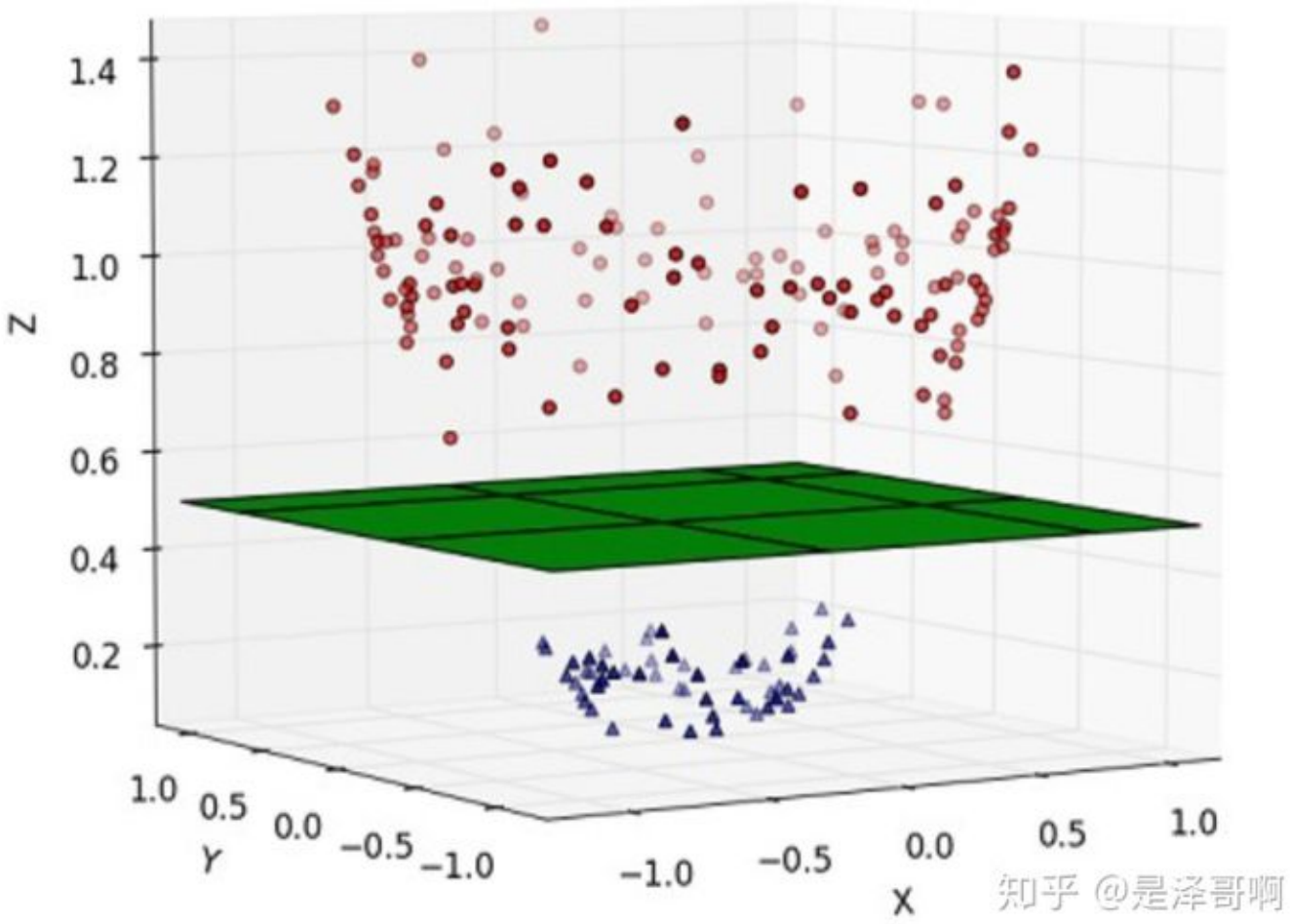
5.1 线性不可分

我们刚刚讨论的硬间隔和软间隔都是在说样本的完全线性可分或者大部分样本点的线性可分。

但我们可能会碰到的一种情况是样本点不是线性可分的，比如：



这种情况的解决方法就是：将二维线性不可分样本映射到高维空间中，让样本点在高维空间线性可分，比如：



对于在有限维度向量空间中线性不可分的样本，我们将其映射到更高维度的向量空间里，再通过间隔最大化的方式，学习得到支持向量机，就是非线性 SVM。

我们用 x 表示原来的样本点，用 $\phi(x)$ 表示 x 映射到特征新的特征空间后到新向量。那么分割超平面可以表示为： $f(x) = w\phi(x) + b$ 。

对于非线性 SVM 的对偶问题就变成了：

$$\min_{\lambda}[\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j(\phi(x_i)\cdot\phi(x_j)) - \sum_{j=1}^{n}\lambda_i]$$

$$s.t. \quad \sum_{i=1}^{n}\lambda_i y_i = 0, \quad \lambda_i \geq 0, \quad C - \lambda_i - \mu_i = 0$$

可以看到与线性 SVM 唯一的不同就是：之前的 $(x_i \cdot x_j)$ 变成了 $(\phi(x_i)\cdot\phi(x_j))$ 。

5.2 核函数的作用

我们不禁有个疑问：只是做个内积运算，为什么要有核函数的呢？

这是因为低维空间映射到高维空间后维度可能会很大，如果将全部样本的点乘全部计算好，这样的计算量太大了。

但如果我们有这样的一核函数 $k(x,y)=(\phi(x),\phi(y))$ ， x_i 与 x_j 在特征空间的内积等于它们在原始样本空间中通过函数 $k(x,y)$ 计算的结果，我们就不需要计算高维甚至无穷维空间的内积了。

举个例子：假设我们有一个多项式核函数：

$$k(x,y)=(x\cdot y+1)^2$$

带进样本点的后：

$$k(x,y)=(\sum_{i=1}^{n}(x_i\cdot y_i)+1)^2$$

而它的展开项是：

$$\sum_{i=1}^{n}x_i^2y_i^2+\sum_{i=2}^{n}\sum_{j=1}^{i-1}(\sqrt{2}x_ix_j)(\sqrt{2}y_iy_j)+\sum_{i=1}^{n}n(\sqrt{2}x_i)(\sqrt{2}y_i)+1$$

如果没有核函数，我们则需要把向量映射成：

$$x^{'}=(x_1^2,\dots,x_n^2,\dots\sqrt{2}x_1,\dots,\sqrt{2}x_n,1)$$

然后在进行内积计算，才能与多项式核函数达到相同的效果。

可见核函数的引入一方面减少了我们计算量，另一方面也减少了我们存储数据的内存使用量。

5.3 常见核函数

我们常用核函数有：

线性核函数

$$k(x_i,x_j)=x_i^Tx_j$$

多项式核函数

$$k(x_i, x_j) = (x_i^T x_j)^d$$

高斯核函数

这三个常用的核函数中只有高斯核函数是需要调参的。

6. 优缺点

6.1 优点

- 有严格的数学理论支持，可解释性强，不依靠统计方法，从而简化了通常的分类和回归问题；
- 能找出对任务至关重要的关键样本（即：支持向量）；
- 采用核技巧之后，可以处理非线性分类/回归任务；
- 最终决策函数只由少数的支持向量所确定，计算的复杂性取决于支持向量的数目，而不是样本空间的维数，这在某种意义上避免了“维数灾难”。

6.2 缺点

- 训练时间长。当采用 SMO 算法时，由于每次都需要挑选一对参数，因此时间复杂度为 $O(N^2)$ ，其中 N 为训练样本的数量；
- 当采用核技巧时，如果需要存储核矩阵，则空间复杂度为 $O(N^2)$ ；
- 模型预测时，预测时间与支持向量的个数成正比。当支持向量的数量较大时，预测计算复杂度较高。

因此支持向量机目前只适合小批量样本的任务，无法适应百万甚至上亿样本的任务。

7. 参考

1. 《机器学习》 周志华
2. 最优化问题的KKT条件
3. 一文理解拉格朗日对偶和KKT条件
4. 支持向量机通俗导论（理解SVM的三层境界）

欢迎关注我的公众号，第一时间追踪高质量学习笔记：阿泽的学习笔记。

阿泽的学习笔记

编辑于 2020-06-13 14:01