支持向量机通俗导论（理解 SVM 的三层境界）

作者：July、pluskid ；致谢：白石、JerryLead
出处：结构之法算法之道 blog。

前言

动笔写这个支持向量机 (support vector machine) 是费了不少劲和困难的，原因很简单，一者这个东西本身就并不好懂，要深入学习和研究下去需花费不少时间和精力，二者这个东西也不好讲清楚，尽管网上已经有朋友写得不错了 (见文末参考链接)，但在描述数学公式的时候还是显得不够。得益于同学白石的数学证明，我还是想尝试写一下，希望本文在兼顾通俗易懂的基础上，真真正正能足以成为一篇完整概括和介绍支持向量机的导论性的文章。

本文在写的过程中，参考了不少资料，包括《支持向量机导论》、《统计学习方法》及网友 pluskid 的支持向量机系列等等，于此，还是一篇学习笔记，只是加入了自己的理解和总结，有任何不妥之处，还望海涵。全文宏观上整体认识支持向量机的概念和用处，微观上深究部分定理的来龙去脉，证明及原理细节，力保逻辑清晰 & 通俗易懂。

同时，阅读本文时建议大家尽量使用 chrome 等浏览器，如此公式才能更好的显示，再者，阅读时可拿张纸和笔出来，把本文所有定理. 公式都亲自推导一遍或者直接打印下来（可直接打印网页版或本文文末附的 PDF，享受随时随地思考、演算的极致快感），在文稿上演算。

Ok，还是那句原话，有任何问题，欢迎任何人随时不吝指正 & 赐教，感谢。

1 第一层、了解 SVM

1.1 什么是支持向量机 SVM

要明白什么是 SVM，便得从分类说起。

分类作为数据挖掘领域中一项非常重要的任务，它的目的是学会一个分类函数或分类模型 (或者叫做分类器)，而支持向量机本身便是一种监督式学习的方法 (至于具体什么是监督学习与非监督学习，请参见此系列 Machine L&Data Mining 第一篇)，它广泛的应用于统计分类以及回归分析中。

支持向量机（SVM）是 90 年代中期发展起来的基于统计学习理论的一种机器学习方法，通过寻求结构化风险最小来提高学习机泛化能力，实现经验风险和置信范围的最小化，从而达到在统计样本量较少的情况下，亦能获得良好统计规律的目的。

通俗来讲，它是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，即支持向量机的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。

对于不想深究 SVM 原理的同学或比如就只想看看 SVM 是干嘛的，那么，了解到这里便足够了，不需上层。而对于那些喜欢深入研究一个东西的同学，甚至究其本质的，咱们则还有很长的一段路要走，万里长征，咱们开始迈第一步吧，相信你能走完。

1.2 线性分类

OK，在讲 SVM 之前，咱们必须先弄清楚一个概念：线性分类器 (也可以叫做感知机，这里的机表示的是一种算法，本文第三部分、证明 SVM 中会详细阐述)。

1.2.1 分类标准

这里我们考虑的是一个两类的分类问题，数据点用 x 来表示，这是一个 n 维向量，w^T 中的 T 代表转置，而类别用 y 来表示，可以取 1 或者 -1 ，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 n 维的数据空间中找到一个分类超平面，其方程

可以表示为：

$$w^T x + b = 0 \tag{1}$$

上面给出了线性分类的定义描述，但或许读者没有想过：为何用 y 取 1 或者 -1 来表示两个不同的类别呢？其实，这个 1 或 -1 的分类标准起源于 logistic 回归，为了完整和过渡的自然性，咱们就再来看看这个 logistic 回归。

1.2.2 1 或 -1 分类标准的起源：logistic 回归

Logistic 回归目的是从特征学习出一个 0/1 分类模型，而这个模型是将特性的线性组合作为自变量，由于自变量的取值范围是负无穷到正无穷。因此，使用 logistic 函数（或称作 sigmoid 函数）将自变量映射到 $(0,1)$ 上，映射后的值被认为是属于 y=1 的概率。

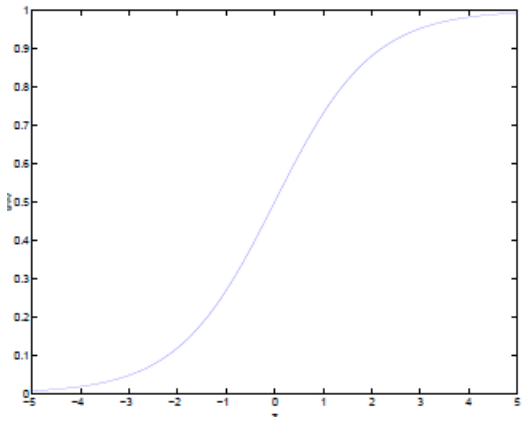
形式化表示就是

假设函数

$$h_0(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}} \tag{2}$$

其中 x 是 n 维特征向量，函数 g 就是 logistic 函数。

而 $g(z) = \frac{1}{1+e^{-z}}$ 的图像是



可以看到，将无穷映射到了 $(0,1)$。

而假设函数就是特征属于 $y=1$ 的概率。

$$P(y=1|x;\theta) = h_\theta(x) P(y=0|x;\theta) = 1 - h_\theta(x) \tag{3}$$

当我们要判别一个新来的特征属于哪个类时，只需求，若大于 0.5 就是 $y=1$ 的类，反之属于 $y=0$ 类。

再审视一下 $h_\theta(x)$，发现 $h_\theta(x)$ 只和 θ^T 有关，$\theta^T x > 0$，那么 $h_\theta(x) > 0.5$，$g(z)$ 只不过是用来映射，真实的类别决定权还在 $\theta^T x$。还有当时 $\theta^T x >> 0$，$h_\theta(x) = 1$，反之 $h_\theta(x) = 0$。如果我们只从 $\theta^T x$ 出发，希望模型达到的目标无非就是让训练数据中 $y=1$ 的特征

$$\theta^T x >> 0$$

，而是 $y=0$ 的特征 $\theta^T x << 0$。Logistic 回归就是要学习得到 θ，使得正例的特征远大于 0，负例的特征远小于 0，强调在全部训练实例上达到这个目标。

1.2.3 形式化标示

我们这次使用的结果标签是 $y=-1,y=1$，替换在 logistic 回归中使用的 $y=0$ 和 $y=1$。同时将 θ 替换成 w 和 b。以前的 $\theta^T x=\theta_0+\theta_1 x_1+\theta_2 x_2+\cdots+\theta_n x_n$，其中认为 $x_0=1$。现在我们替换为 b，后面替换 $\theta_1 x_1+\theta_2 x_2+\cdots+\theta_n x_n$ 为 $w_1 x_1+w_2 x_2+\cdots+w_n x_n$（即 $x^T x$）。这样，我们让 $\theta^T x=w^T x+b$，进一步 $h_\theta(x)=g(\theta^T x)=g(w^T x+b)$。也就是说除了 y 由 $y=0$ 变为 $y=-1$，只是标记不同外，与 logistic 回归的形式化表示没区别。

再明确下假设函数

$$h_{w,b}=g(w^T x+b) \tag{4}$$

上面提到过我们只需考虑 $\theta^T x$ 的正负问题，而不用关心 $g(z)$，因此我们这里将 $g(z)$ 做一个简化，将其简单映射到 $y=-1$ 和 $y=1$ 上。映射关系如下：

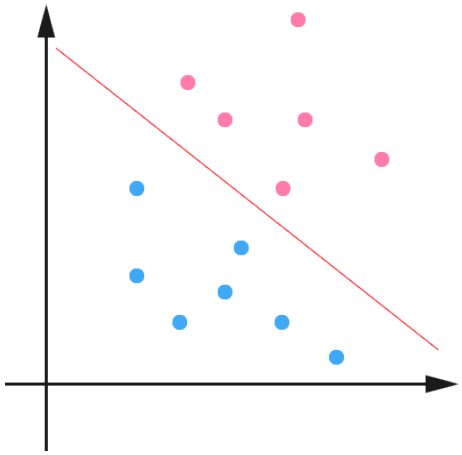
$$g(z)=\begin{cases}1, & z\geqslant 0\\ -1, & z<0\end{cases} \tag{5}$$

于此，想必已经解释明白了为何线性分类的标准一般用 1 或者 -1 来标示。

注：上小节来自 jerrylead 所作的斯坦福机器学习课程的笔记。

1.3 线性分类的一个例子

下面举个简单的例子，一个二维平面 (一个超平面，在二维空间中的例子就是一条直线)，如下图所示，平面上有两种不同的点，分别用两种不同的颜色表示，一种为红颜色的点，另一种则为蓝颜色的点，红颜色的线表示一个可行的超平面。

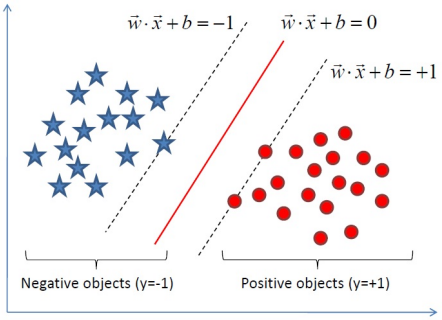


从上图中我们可以看出，这条红颜色的线把红颜色的点和蓝颜色的点分开来了。而这条红颜色的线就是我们上面所说的超平面，也就是说，这个所谓的超平面的的确确便把这两种不同颜色的数据点分隔开来，在超平面一边的数据点所对应的 y 全是 -1，而在另一边全是 1。

接着，我们可以令分类函数（提醒：下文很大篇幅都在讨论着这个分类函数）：

$$f(x)=w^T x+b \tag{6}$$

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



注：上图中，定义特征到结果的输出函数 $u=\vec{w}\cdot\vec{x}-b$，与我们之前定义的 $f(x)=w^Tx+b$ 实质是一样的。为什么？因为无论是 $u=\vec{w}\cdot\vec{x}-b$，还是 $f(x)=w^Tx+b$，不影响最终优化结果。下文你将看到，当我们转化到优化 $\max\frac{1}{\|w\|}, s.t., y_i(w^Tx_i+b)\geq 1, i=1,\cdots,n$ 的时候，为了求解方便，会把 $yf(x)$ 令为 1，即 $yf(x)$ 是 $y(w^x+b)$，还是 $y(w^x-b)$，对我们要优化的式子 $\max\frac{1}{\|w\|}$ 已无影响。

（有一朋友飞狗来自 Mare Desiderii，看了上面的定义之后，问道：请教一下 SVM functional margin 为 $\widehat{\gamma}=y(wTx+b)=yf(x)$ 中的 γ 是只取 1 和 -1 吗？y 的唯一作用就是确保 functional margin 的非负性？真是这样的么？当然不是，详情请见本文评论下第 43 楼）

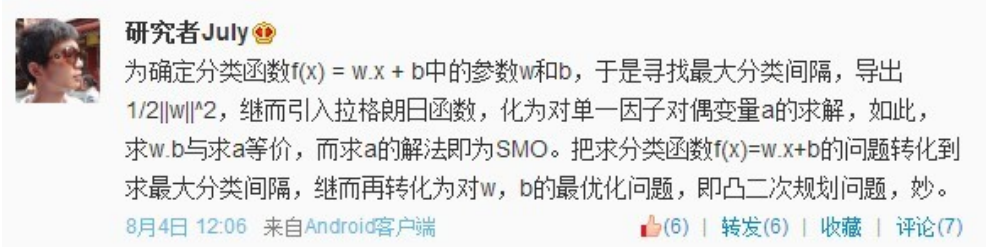
当然，有些时候，或者说大部分时候数据并不是线性可分的，这个时候满足这样条件的超平面就根本不存在 (不过关于如何处理这样的问题我们后面会讲)，这里先从最简单的情形开始推导，就假设数据都是线性可分的，亦即这样的超平面是存在的。

更进一步，我们在进行分类的时候，将数据点 x 代入 $f(x)$ 中，如果得到的结果小于 0，则赋予其类别 -1，如果大于 0 则赋予类别 1。如果 $f(x)=0$，则很难办了，分到哪一类都不是。

请读者注意，下面的篇幅将按下述 3 点走：

1. 咱们就要确定上述分类函数 $f(x)=w\cdot x+b$（$w\cdot x$ 表示 w 与 x 的内积）中的两个参数 w 和 b，通俗理解的话 w 是法向量，b 是截距（再次说明：定义特征到结果的输出函数 $u=|w\cdot|x-b$，与我们最开始定义的 $f(x)=w^Tx+b$ 实质是一样的）；
2. 那如何确定 w 和 b 呢？答案是寻找两条边界端或极端划分直线中间的最大间隔（之所以要寻最大间隔是为了能更好的划分不同类的点，下文你将看到：为寻最大间隔，导出 $1/2||w||^2$!!!!!!!!!!!!!!!!!!!!，继而引入拉格朗日函数和对偶变量 a，化为对单一因数对偶变量 a 的求解，当然，这是后话），从而确定最终的最大间隔分类超平面 hyper plane 和分类函数；
3. 进而把寻求分类函数 $f(x)=w\cdot x+b$ 的问题转化为对 w，b 的最优化问题，最终化为对偶因子的求解。

总结成一句话即是：从最大间隔出发（目的本就是为了确定法向量 w），转化为求对变量 w 和 b 的凸二次规划问题。亦或如下图所示（有点需要注意，如读者 @ 酱爆小八爪所说：从最大分类间隔开始，就一直是凸优化问题）：



1.4 函数间隔 Functional margin 与几何间隔 Geometrical margin

一般而言，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 $w*x+b=0$ 确定的情况下，$|w*x+b|$ 能够相对的表示点 x 到距离超平面的远近，而 $w*x+b$ 的符号与类标记 y 的符号是否一致表示分类是否正确，所以，可以用量 $y*(w*x+b)$ 的正负性来判定或表示分类的正确性和确信度。

于此，我们便引出了定义样本到分类间隔距离的函数间隔 functional margin 的概念。

1.4.1 函数间隔 Functional margin

我们定义函数间隔 functional margin 为：

$$\widehat{\gamma} = y(w^T x + b) = yf(x) \tag{7}$$

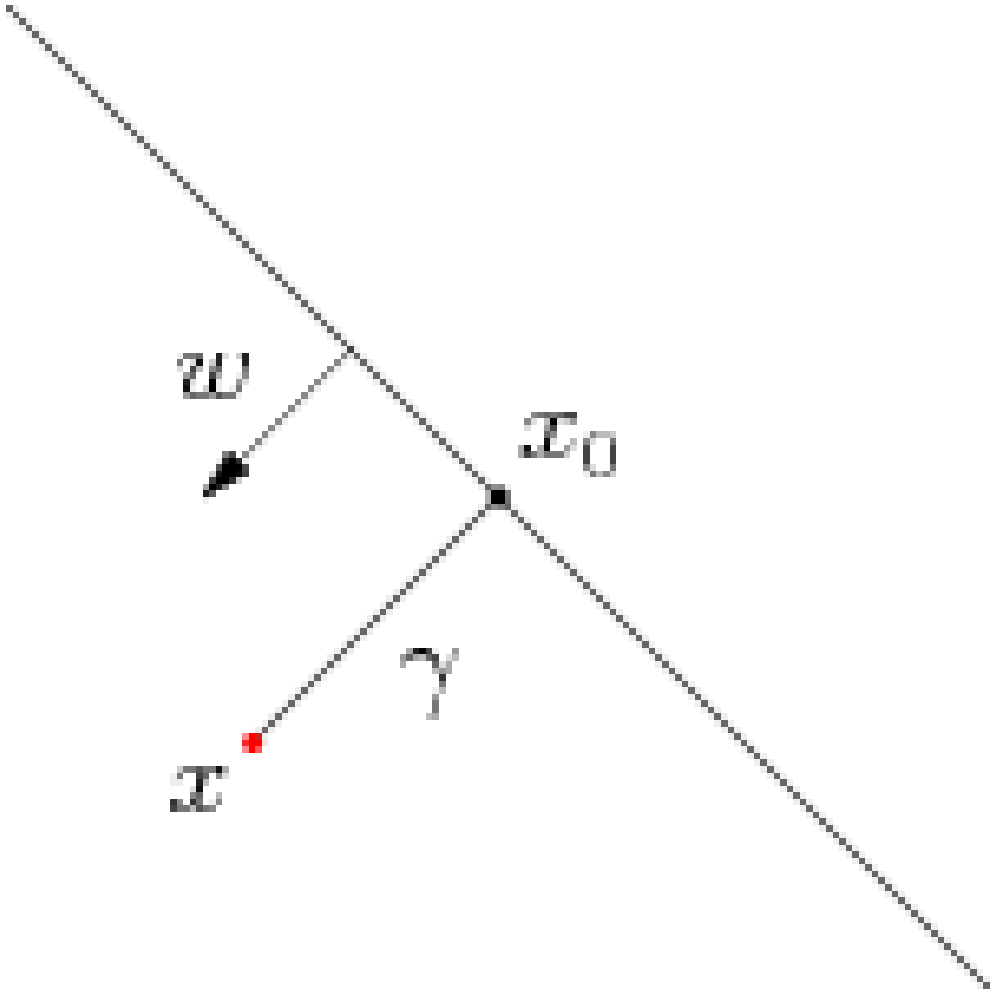
接着，我们定义超平面 (w,b) 关于训练数据集 T 的函数间隔为超平面 (w,b) 关于 T 中所有样本点 (x_i,y_i) 的函数间隔最小值，其中，x 是特征，y 是结果标签，i 表示第 i 个样本，有：

$$\widehat{\gamma} = \min \widehat{\gamma}_i (i=1,\cdots,n) \tag{8}$$

然与此同时，问题就出来了。上述定义的函数间隔虽然可以表示分类预测的正确性和确信度，但在选择分类超平面时，只有函数间隔还远远不够，因为如果成比例的改变 w 和 b，如将他们改变为 $2w$ 和 $2b$，虽然此时超平面没有改变，但函数间隔的值 $f(x)$ 却变成了原来的 2 倍。

其实，我们可以对法向量 w 加些约束条件，使其表面上看起来规范化，如此，我们很快又将引出真正定义点到超平面的距离 --几何间隔 geometrical margin 的概念（很快你将看到，几何间隔就是函数间隔除以个 $\|w\|$，即 $yf(x)/\|w\|$）。

1.4.2 点到超平面的距离定义：几何间隔 Geometrical margin



在给出几何间隔的定义之前，咱们首先来看下，如上图所示，对于一个点 x，令其垂直投影到超平面上的对应的为 x_0，由于 w 是垂直于超平面的一个向量，γ 为样本 x 到分类间隔的距离，我们有

$$x = x_0 + \gamma \frac{w}{\|w\|} \tag{9}$$

又由于 x_0 是超平面上的点，满足 $f(x_0) = 0$，代入超平面的方程即可算出：

$$\gamma = \frac{w^T x + b}{\|w\|} = \frac{f(x)}{\|w\|} \tag{10}$$

（有的书上会写成把 $\|w\|$ 分开相除的形式，如本文参考文献及推荐阅读条目 11，其中，$\|w\|$ 为 w 的二阶泛数）

不过这里的 γ 是带符号的，我们需要的只是它的绝对值，因此类似地，也乘上对应的类别 y 即可，因此实际上我们定义几何间隔 geometrical margin 为 (注：别忘了，上面 $\widehat{\gamma}$ 的定义，$\widehat{\gamma} = y(w^T x + b) = yf(x)$)：

$$\widetilde{\gamma} = y\gamma = \frac{\widehat{\gamma}}{\|w\|} \tag{11}$$

（代入相关式子可以得出：yi*(w/||w|| + b/||w||)）

正如本文评论下读者 popol1991 留言：函数间隔 $y*(wx+b)=y*f(x)$ 实际上就是 $|f(x)|$，只是人为定义的一个间隔度量；而几何间隔 $|f(x)|/||w||$ 才是直观上的点到超平面距离。

想想二维空间里的点到直线公式：假设一条直线的方程为 $ax+by+c=0$, 点 P 的坐标是 (x_0,y_0)，则点到直线距离为 $|ax_0+by_0+c|/sqrt(a^2+b^2)$。如下图所示：

点到平面的距离

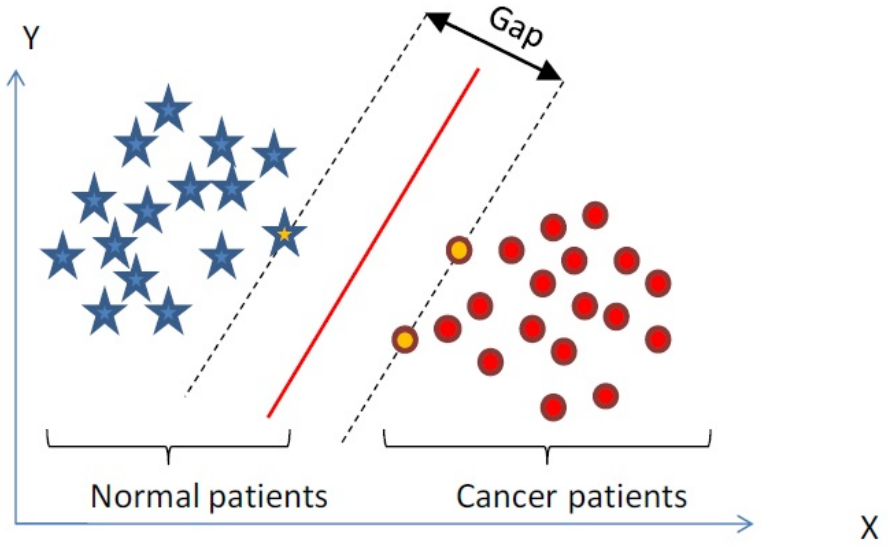
若点坐标为 $(x_0.y_0.z_0)$，平面为 $Ax+By+Cz+D=0$，则点到平面的距离为：

$$d=\left|\frac{Ax_0+By_0+Cz_0+D}{\sqrt{A^2+B^2+C^2}}\right| \tag{12}$$

那么如果用向量表示，设 $w=(a,b)$,$f(x)=wx+c$, 那么这个距离正是 $|f(p)|/||w||$。

1.5 最大间隔分类器 Maximum Margin Classifier 的定义

于此，我们已经很明显的看出，函数间隔 functional margin 和几何间隔 geometrical margin 相差一个的缩放因子。按照我们前面的分析，对一个数据点进行分类，当它的 margin 越大的时候，分类的 confidence 越大。对于一个包含 n 个点的数据集，我们可以很自然地定义它的 margin 为所有这 n 个点的 margin 值中最小的那个。于是，为了使得分类的 confidence 高，我们希望所选择的超平面 hyper plane 能够最大化这个 margin 值。



通过上节，我们已经知道：

1、functional margin 明显是不太适合用来最大化的一个量，因为在 hyper plane 固定以后，我们可以等比例地缩放 w 的长度和 b 的值，这样可以使得 $f(x)=w^Tx+b$ 的值任意大，亦即 functional margin$\hat{\gamma}$ 可以在 hyper plane 保持不变的情况下被取得任意大，

2、而 geometrical margin 则没有这个问题，因为除上了 $||w||$ 这个分母，所以缩放 w 和 b 的时候的值是不会改变的，它只随着 hyper plane 的变动而变动，因此，这是更加合适的一个 margin 。

这样一来，我们的 maximum margin classifier 的目标函数可以定义为：

$$\max \tilde{\gamma} \tag{13}$$

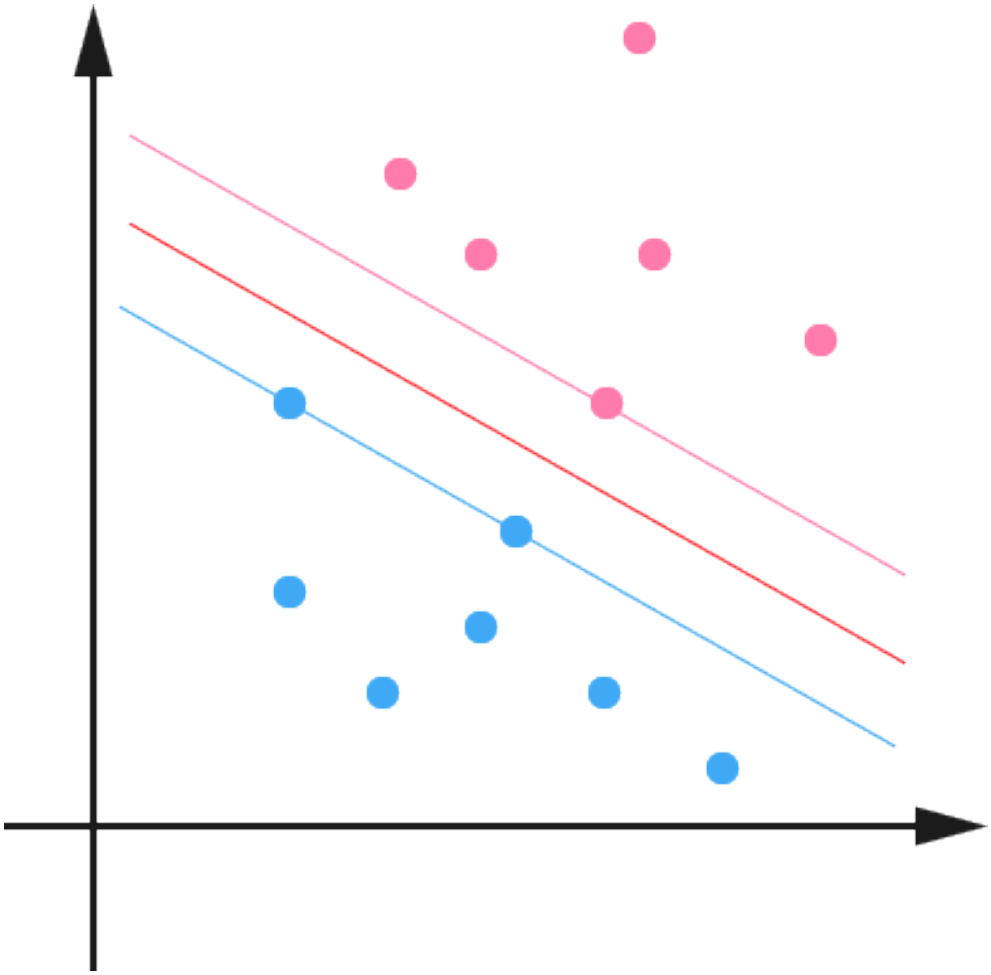
当然，还需要满足一些条件，根据 margin 的定义，我们有

$$y_i(w^T x_i + b) = \hat{\gamma}_i \geq \hat{\gamma}, i = 1, \cdots, n \tag{14}$$

其中 $\hat{\gamma} = \tilde{\gamma}\|w\|$ (等价于 $\tilde{\gamma} = \gamma/\|w\|$，故有稍后的 $\hat{\gamma} = 1$ 时，$\tilde{\gamma} = 1/\|w\|$)，处于方便推导和优化的目的，我们可以令 $\hat{\gamma} = 1$(对目标函数的优化没有影响，至于为什么，请见本文评论下第 42 楼回复)，此时，上述的目标函数 $\tilde{\gamma}$ 转化为 (其中，s.t.，即 subject to 的意思，它导出的是约束条件)：

$$\max \frac{1}{\|w\|}, w.t., y_i(w^T x_i + b) \geq 1, i = 1, \cdots, n \tag{15}$$

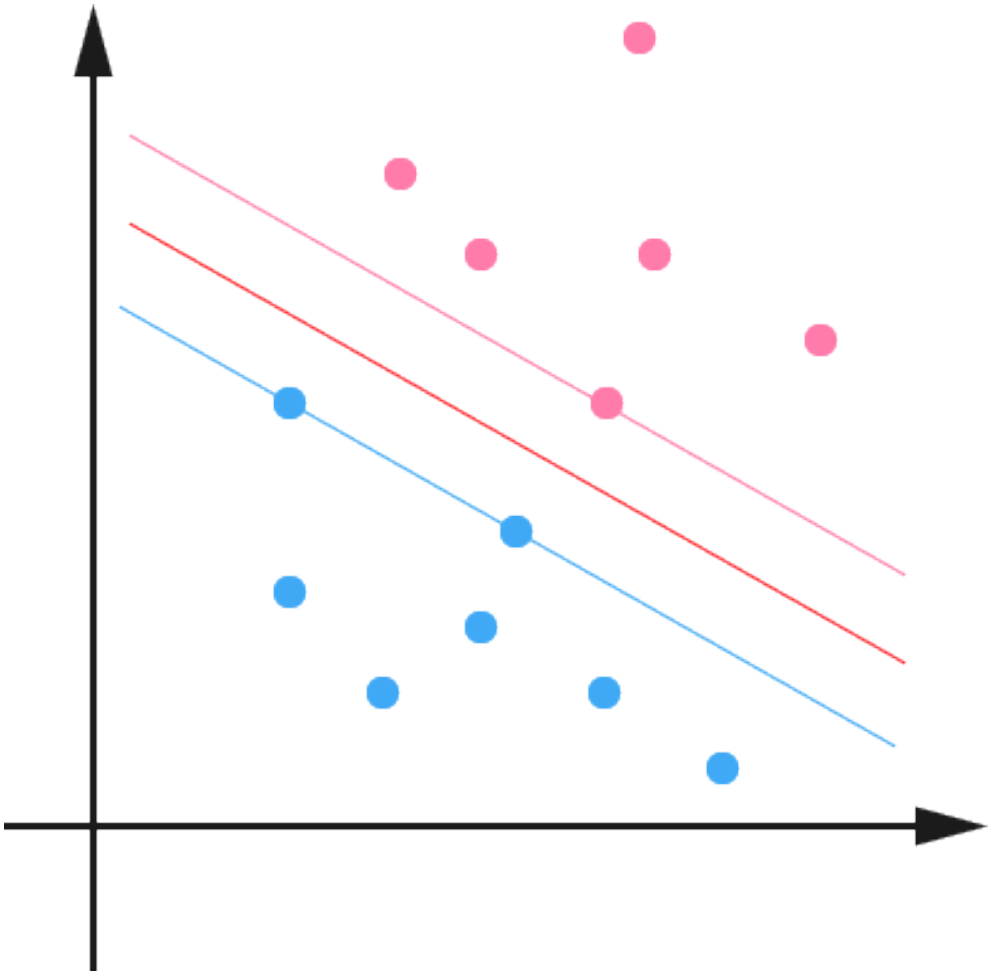
通过求解这个问题，我们就可以找到一个 margin 最大的 classifier，如下图所示，中间的红色线条是 Optimal Hyper Plane，另外两条线到红线的距离都是等于 $\tilde{\gamma}$ 的 ($\tilde{\gamma}$ 便是上文所定义的 geometrical margin，当令 $\hat{\gamma} = 1$ 时，$\tilde{\gamma}$ 便为 $1/\|w\|$，而我们上面得到的目标函数便是在相应的约束条件下，要最大化这个 $1/\|w\|$ 值)：



通过最大化 margin，我们使得该分类器对数据进行分类时具有了最大的 confidence，从而设计决策最优分类超平面。

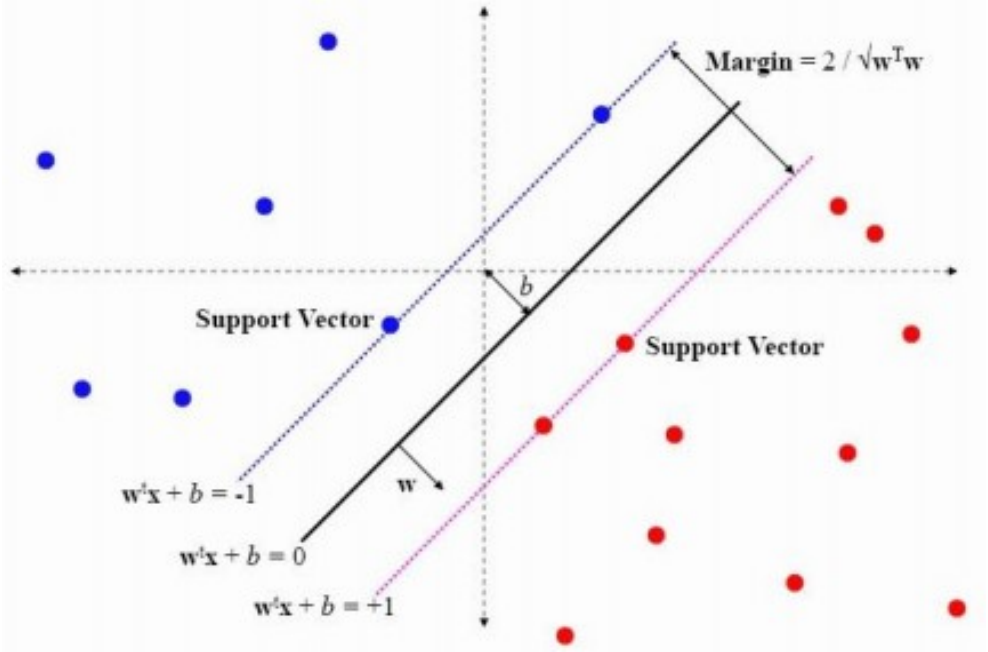
1.6 到底什么是 Support Vector

上节，我们介绍了 Maximum Margin Classifier，但并没有具体阐述到底什么是 Support Vector，本节，咱们来重点阐述这个概念。咱们不妨先来回忆一下上节 1.4 节最后一张图：



可以看到两个支撑着中间的 gap 的超平面，它们到中间的纯红线 separating hyper plane 的距离相等，即我们所能得到的最大的 geometrical margin$\tilde{\gamma}$，而“支撑”这两个超平面的必定会有一些点，而这些“支撑”的点便叫做支持向量 Support Vector。

或亦可看下来自此 PPT 中的一张图，Support Vector 便是那蓝色虚线和粉红色虚线上的点：



很显然，由于这些 supporting vector 刚好在边界上，所以它们满足 $y(w^Tx+b)=1$（还记得我们把 functional margin 定为 1 了吗？上节中：“处于方便推导和优化的目的，我们可以令 $\hat{\gamma}=1$”），而对于所有不是支持向量的点，也就是在“阵地后方”的点，则显然 $y(w^Tx+b)>1$ 有。当然，除了从几何直观上之外，支持向量的概念也可以从下文优化过程的推导中得到。

OK，到此为止，算是了解到了 SVM 的第一层，对于那些只关心怎么用 SVM 的朋友便已足够，不必再更进一层深究其更深的原理。

2 深入 SVM

2.1 从线性可分到线性不可分

2.1.1 从原始问题到对偶问题的求解

虽然上文 1.4 节给出了目标函数，却没有讲怎么来求解。现在就让我们来处理这个问题。回忆一下之前得到的目标函数（subject to 导出的则是约束条件）：

$$\max \frac{1}{\|w\|} \; s.t., y_i(w^Tx_i+b)\geq 1, i=1,\cdots,n \tag{16}$$

由于求 $\frac{1}{\|w\|}$ 的最大值相当于求 $\frac{1}{2}\|w\|^2$ 的最小值，所以上述问题等价于（w 由分母变成分子，从而也有原来的 max 问题变为 min 问题，很明显，两者问题等价）：

$$\min \frac{1}{2}\|w\|^2 \; s.t., y_i(w^Tx_i+b)\geq 1, i=1,\cdots,n \tag{17}$$

- 转化到这个形式后，我们的问题成为了一个凸优化问题，或者更具体的说，因为现在的目标函数是二次的，约束条件是线性的，所以它是一个凸二次规划问题。这个问题可以用任何现成的 QP (Quadratic Programming) 的优化包进行求解，归结为一句话即是：在一定的约束条件下，目标最优，损失最小；
- 但虽然这个问题确实是一个标准的 QP 问题，但是它也有它的特殊结构，通过 Lagrange Duality 变换到对偶变量 (dual variable) 的优化问题之后，可以找到一种更

加有效的方法来进行求解，而且通常情况下这种方法比直接使用通用的 QP 优化包进行优化要高效得多。

也就说，除了用解决 QP 问题的常规方法之外，还可以通过求解对偶问题得到最优解，这就是线性可分条件下支持向量机的对偶算法，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。

至于上述提到，关于什么是 Lagrange duality？简单地来说，通过给每一个约束条件加上一个 Lagrange multiplier(拉格朗日乘值)，即引入拉格朗日乘子 α 如此我们便可以通过拉格朗日函数将约束条件融和到目标函数里去 (也就是说把条件融合到一个函数里头，现在只用一个函数表达式便能清楚的表达出我们的问题)：

$$\mathcal{L}(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i\left(y_i(w^Tx_i+b)-1\right) \tag{18}$$

然后我们令

$$\theta(w)=\max_{\alpha_i\geq 0}\mathcal{L}(w,b,\alpha) \tag{19}$$

容易验证，当某个约束条件不满足时，例如 $y_i(w^Tx_i+b)<1$，那么我们显然有 $\theta(w)=\inf$（只要令 $\alpha_i=\inf$ 即可）。而当所有约束条件都满足时，则有 $\theta(w)=\frac{1}{2}\|w\|^2$，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 $\frac{1}{2}\|w\|^2$，实际上等价于直接最小化 $\theta(w)$（当然，这里也有约束条件，就是 $\alpha_i\geq 0,i=1,\cdots,n$），因为如果约束条件没有得到满足，$\theta w$ 等于无穷大，自然不会是我们所要求的最小值。具体写出来，我们现在的目标函数变成了：

$$\min_{w,b}\theta(w)=\min_{w,b}\max_{\alpha_i\geq 0}\mathcal{L}(w,b,\alpha)=p^* \tag{20}$$

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。不过，现在我们来把最小和最大的位置交换一下（稍后，你将看到，当下面式子满足了一定的条件之后，这个式子 d 便是上式 P 的对偶形式表示）：

$$\max_{\alpha_i\geq 0}\min_{w,b}\mathcal{L}(w,b,\alpha)=d^* \tag{21}$$

当然，交换以后的问题不再等价于原问题，这个新问题的最优值用 d^* 来表示。并且，我们有 $d^*\leq p^*$，这在直观上也不难理解，最大值中最小的一个总也比最小值中最大的一个要大吧！总之，第二个问题的最优值 d^* 在这里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。

也就是说，下面我们将先求 L 对 w、b 的极小，再求 L 对 α 的极大。而且，之所以从 min max 的原始问题 p^*，转化为 max min 的对偶问题 d^*，一者因为 d^* 是 p^* 的近似解，二者，转化为对偶问题后，更容易求解。

2.1.2 KKT 条件

与此同时，上段说“在满足某些条件的情况下”，这所谓的“满足某些条件”就是要满足 KKT 条件。那 KKT 条件的表现形式是什么呢？据维基百科：KKT 条件的介绍，一般地，一个最优化数学模型能够表示成下列标准形式：

$$\begin{aligned}&\min .f(x)\\ &s.t.h_j(x)=0,j=1,\cdots,p,\\ &\quad g_k(x)\leq 0,k=1,\cdots,q,\\ &x\in X\subset\Re^n\end{aligned} \tag{22}$$

其中，$f(x)$ 是需要最小化的函数，$h(x)$ 是等式约束，$g(x)$ 是不等式约束，p 和 q 分别为等式约束和不等式约束的数量。同时，我们得明白以下两个定理：

- 凸优化的概念：$\mathcal{X} \subset \mathbb{R}^n$ 为一凸集，$f : \mathcal{X} \to \mathbb{R}$ 为一凸函数。凸优化就是要找出一点 $x^* \in \mathcal{X}$ ，使得每一 $x \in \mathcal{X}$ 满足 $f(x^*) \leq f(x)$。
- KKT 条件的意义：它是一个非线性规划（Nonlinear Programming）问题能有最优化解法的必要和充分条件。

那到底什么是所谓 Karush-Kuhn-Tucker 条件呢？KKT 条件就是指上面最优化数学模型的标准形式中的最小点 x^* 必须满足下面的条件：

$$\begin{aligned} &1.\ h_j(x_*) = 0, j = 1, \cdots, p, g_k(x_*) \leq 0, k = 1, \cdots, q, \\ &2.\ \nabla f(x_*) + \sum_{j=1}^{p} \lambda_j \nabla h_j(x_*) + \sum_{k=1}^{q} \mu_k \nabla g_k(x_*) = 0, \\ &\qquad\qquad \lambda \neq 0 m \mu_k \geq 0, \mu_k g_k(x_*) = 0 \end{aligned} \tag{23}$$

经过论证，我们这里的问题是满足 KKT 条件的（首先已经满足 Slater condition，再者 f 和 g_i 也都是可微的，即 L 对 w 和 b 都可导），因此现在我们便转化为求解第二个问题。也就是说，现在，咱们的原问题通过满足一定的条件，已经转化成了对偶问题。而求解这个对偶学习问题，分为 3 个步骤，首先要让 L(w，b，a) 关于 w 和 b 最小化，然后求对 α 的极大，最后利用 SMO 算法求解对偶因子。

2.1.3 对偶问题求解的 3 个步骤

（1）、首先固定，要让 L 关于 w 和 b 最小化，我们分别对 w，b 求偏导数，即令 $\partial\mathcal{L}/\partial w$ 和 $\partial\mathcal{L}/\partial b$ 等于零（对 w 求导结果的解释请看本文评论下第 45 楼回复）：

$$\begin{aligned} \frac{\mathcal{L}}{\partial w} &= \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \frac{\mathcal{L}}{\partial b} &= \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{24}$$

以上结果代回上述的 $\mathcal{L}$：

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1\right) \tag{25}$$

得到：

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i \\ &= \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \end{aligned} \tag{26}$$

提醒：有读者可能会问上述推导过程如何而来？说实话，其具体推导过程是比较复杂的，如下图所示：

$$
\begin{aligned}
\mathcal{L}(w,b,\alpha) &= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m}\alpha_i[y^{(i)}(w^Tx^{(i)}+b)-1] \\
&= \frac{1}{2}w^Tw - \sum_{i=1}^{m}\alpha_i y^{(i)}w^Tx^{(i)} - \sum_{i=1}^{m}\alpha_i y^{(i)}b + \sum_{i=1}^{m}\alpha_i \\
&= \frac{1}{2}w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - \sum_{i=1}^{m}\alpha_i y^{(i)}w^Tx^{(i)} - \sum_{i=1}^{m}\alpha_i y^{(i)}b + \sum_{i=1}^{m}\alpha_i \\
&= \frac{1}{2}w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - \sum_{i=1}^{m}\alpha_i y^{(i)}b + \sum_{i=1}^{m}\alpha_i \\
&= -\frac{1}{2}w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - \sum_{i=1}^{m}\alpha_i y^{(i)}b + \sum_{i=1}^{m}\alpha_i \\
&= -\frac{1}{2}w^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - b\sum_{i=1}^{m}\alpha_i y^{(i)} + \sum_{i=1}^{m}\alpha_i \\
&= -\frac{1}{2}\left(\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}\right)^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - b\sum_{i=1}^{m}\alpha_i y^{(i)} + \sum_{i=1}^{m}\alpha_i \\
&= -\frac{1}{2}\sum_{i=1}^{m}\alpha_i y^{(i)}\left(x^{(i)}\right)^T\sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)} - b\sum_{i=1}^{m}\alpha_i y^{(i)} + \sum_{i=1}^{m}\alpha_i \\
&= -\frac{1}{2}\sum_{i,j=1}^{m}\alpha_i y^{(i)}\left(x^{(i)}\right)^T\alpha_j y^{(j)}x^{(j)} - b\sum_{i=1}^{m}\alpha_i y^{(i)} + \sum_{i=1}^{m}\alpha_i
\end{aligned}
\tag{27}
$$

最后，得到：

$$
\begin{aligned}
\mathcal{L}(w,b,\alpha) &= \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^Tx_j - \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^Tx_j - b\sum_{i=1}^{n}\alpha_i y_i + \sum_{i=1}^{n}\alpha_i \\
&= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^Tx_j
\end{aligned}
\tag{28}
$$

如 jerrylead 所说：“倒数第 4 步”推导到“倒数第 3 步”使用了线性代数的转置运算，由于 ai 和 yi 都是实数，因此转置后与自身一样。“倒数第 3 步”推导到“倒数第 2 步”使 $(a+b+c+\ldots)(a+b+c+\cdots) = aa+ab+ac+ba+bb+bc+\cdots$ 的乘法运算法则。最后一步是上一步的顺序调整。

从上面的最后一个式子，我们可以看出，此时的拉格朗日函数只包含了一个变量，那就是 α_i，然后下文的第 2 步，求出了 α_i 便能求出 w，和 b，由此可见，上文第 1.2 节提出来的核心问题：分类函数 $f(x) = w^Tx + b$ 也就可以轻而易举的求出来了。

（2）、求对 α 的极大，即是关于对偶问题的最优化问题，从上面的式子得到：

(不得不提醒下读者：经过上面第一个步骤的求 w 和 b，得到的拉格朗日函数式子已经没有了变量 w，b，只有 α，而反过来，求得的 α 将能导出 w，b 的解，最终得出分离超平面和分类决策函数。为何呢？因为如果求出了 α_i，根据 $w = \sum_{i=1}^{m}\alpha_i y^{(i)}x^{(i)}$，即

可求出 w。然后通过 $b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T}x^{(i)} + \min_{i:y^{(i)}=1} w^{*T}x^{(i)}}{2}$，即可求出 b)

$$\begin{aligned} \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ s.t, \alpha_i \geq 0, i = 1, \cdots, n \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{29}$$

如前面所说，这个问题有更加高效的优化算法，即我们常说的 SMO 算法。

2.1.4 序列最小最优化 SMO 算法

细心的读者读至上节末尾处，怎么拉格朗日乘子的值可能依然心存疑惑。实际上，关于的求解可以用一种快速学习算法即 SMO 算法，这里先简要介绍下。

OK，当：

$$\begin{aligned} \max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \\ s.t.\ 0 \leq \alpha_i \leq C, i = 1, \cdots, m \\ \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \end{aligned} \tag{30}$$

要解决的是在参数 $\{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ 上求最大值 W 的问题，至于 $x^{(i)}$ 和 $y^{(i)}$ 都是已知数（其中 C 是一个参数，用于控制目标函数中两项（“寻找 margin 最大的超平面”和“保证数据点偏差量最小”）之间的权重。和上文最后的式子对比一下，可以看到唯一的区别就是现在 dual variableiα 多了一个上限 C，关于 C 的具体由来请查看下文第 2.3 节）。

要了解这个 SMO 算法是如何推导的，请跳到下文第 3.5 节、SMO 算法。

2.1.5 线性不可分的情况

OK，为过渡到下节 2.2 节所介绍的核函数，让我们再来看看上述推导过程中得到的一些有趣的形式。首先就是关于我们的 hyper plane ，对于一个数据点 x 进行分类，实际上是通过把 x 带入到 $f(x) = w^T x + b$ 算出结果然后根据其正负号来进行类别划分的。而前面的推导中我们得到

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{31}$$

因此分类函数为：

$$\begin{aligned} f(x) &= \left(\sum_{i=1}^{n} \alpha_i y_i x_i \right)^T x + b \\ &= \sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + b \end{aligned} \tag{32}$$

这里的形式的有趣之处在于，对于新点 x 的预测，只需要计算它与训练数据点的内积即可（$\langle *, * \rangle$ 表示向量内积），这一点至关重要，是之后使用 Kernel 进行非线性推广的基本前提。此外，所谓 Supporting Vector 也在这里显示出来——事实上，所有非 Supporting

Vector 所对应的系 α 数都是等于零的，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即可。

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，由于分类完全有超平面决定，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。

回忆一下我们 2.1.1 节中通过 Lagrange multiplier 得到的目标函数：

$$\max_{\alpha_i \geq 0} \mathcal{L}(w, b, \alpha) = \max_{\alpha_i \geq 0} \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1\right) \tag{33}$$

注意到如果 x_i 是支持向量的话，上式中红颜色的部分是等于 0 的（因为支持向量的 functional margin 等于 1 ），而对于非支持向量来说，functional margin 会大于 1 ，因此红颜色部分是大于零的，而 α_i 又是非负的，为了满足最大化，α_i 必须等于 0 。这也就是这些非 Supporting Vector 的点的局限性。

从 1.5 节到上述所有这些东西，便得到了一个 maximum margin hyper plane classifier，这就是所谓的支持向量机（Support Vector Machine）。当然，到目前为止，我们的 SVM 还比较弱，只能处理线性的情况，不过，在得到了对偶 dual 形式之后，通过 Kernel 推广到非线性的情况就变成了一件非常容易的事情了 (相信，你还记得本节开头所说的：“通过求解对偶问题得到最优解，这就是线性可分条件下支持向量机的对偶算法，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”)。

2.2 核函数 Kernel