SVM

支持向量机（Support Vector Machine-SVM）于1995年正式提出（Cortes and Vapnik, 1995），与logistics regression类似，最初SVM也是基于线性判别函数，并借助凸优化技术，以解决二分类问题，然而与逻辑回归不同的是，其输出结果为分类类别，并非类别概率。由于当时支持向量机在文本分类问题上显示出卓越的性能（AdaBoost+SVM），而很快成为机器学习领域的热点。直至2006年神经网络开始复兴（Hition在Science上发表文章指出在MINIST手写数字识别任务中神经网络的error rate达到1.2%，低于RBF核的SVM的1.4%error rate），12年深度学习"一鸣惊人"，这才使研究人员开始关注深度神经网络，SVM的研究逐渐淡去。但是不得不说的是SVM作为一种十分完美的分类器，其仍是有监督学习的经典代表。

Margin & Support Vector

考虑线性可分的二分类问题如下所示：

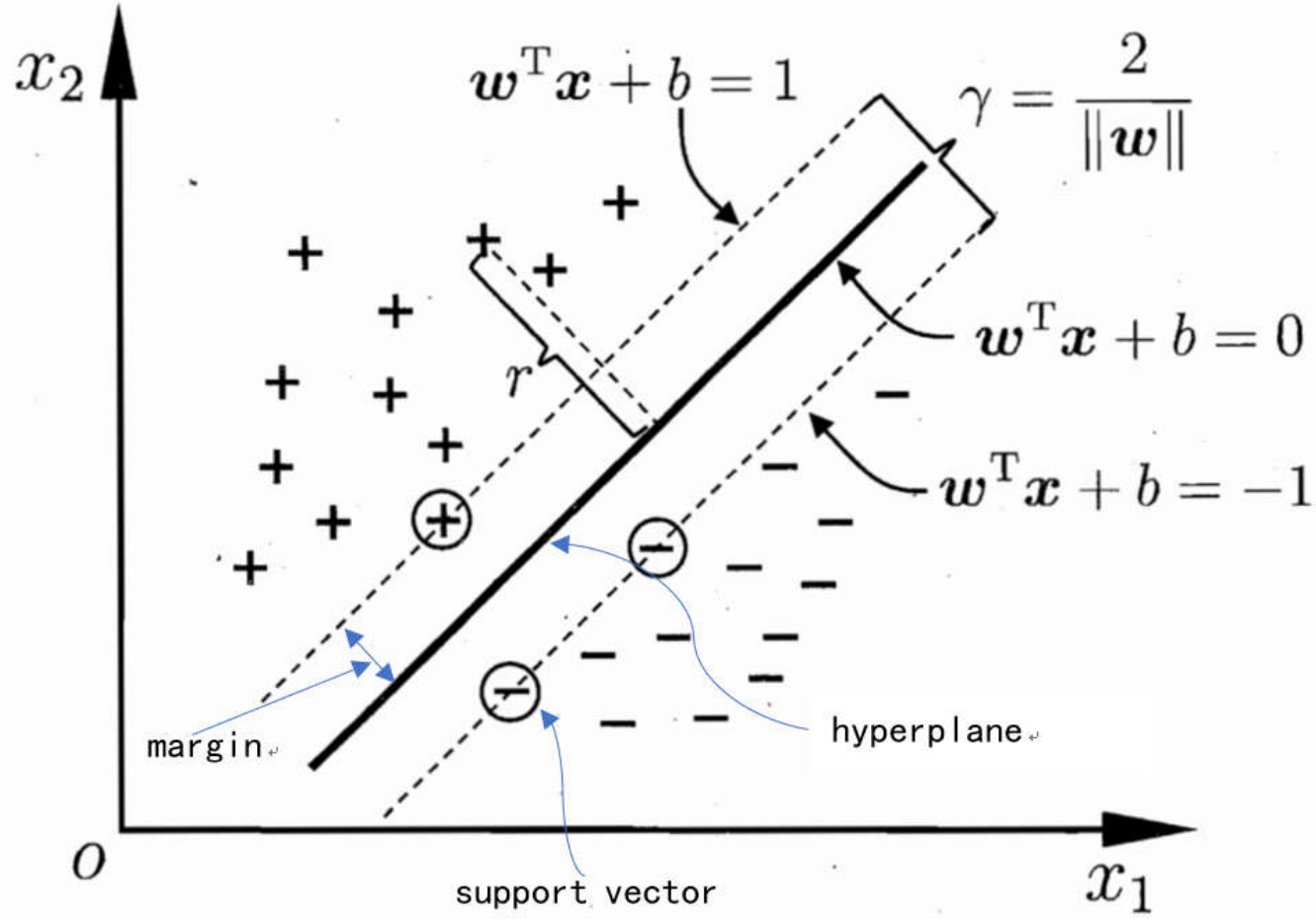

图1. 支持向量与间隔

在样本空间中中存在"正负"两类样本，这里我们期望能找到一个划分超平面（hyperplane）能更好的区分不同类别样本。在SVM中定义划分超平面由如下线性方程确定：

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

上式中，$W = (w_1, w_2, ..., w_n)$为超平面法向量（垂直超平面）；

b为超平面位移项，确定超平面与原点间的距离。

故超平面由W, b唯一确定。这里我们定义正负样本的标签$y_i = \pm 1$，且：

$$\begin{cases} W^T x_i + b \geq +1, & y_i = +1 \\ W^T x_i + b \leq -1, & y_i = -1 \end{cases} \tag{2}$$

同时定义使得$|W^T x_i + b = 1|$的x_i为支持向量（support vector）,对应图1中虚线上的point（hard margin要求虚线内无样本点）。显然支持向量即为距超平面最近的点，因此定义间隔（margin）γ为支持向量与超平面的距离，即其在超平面法线方向的投影。

$$\gamma = \frac{1}{2}\frac{(W^T x_+ + b) - (W^T x_- + b)}{||W||} = \frac{1}{2}\frac{(+1) - (-1)}{||W||} = \frac{1}{||W||} \tag{3}$$

上式中，$||W||$为W的二范数。对此，我们期望所求超平面能最大（好）的区分两类样本点（分的最开），即期望margin越大越好，则有：

$$\begin{aligned} & max_{W,b} \quad \frac{1}{||W||} \\ s.t. & \begin{cases} W^T x_i + b \geq +1, & if\ y_i = +1 \\ W^T x_i + b \leq -1, & if\ y_i = -1 \end{cases} \end{aligned} \tag{4}$$

注意到，$max_{W,b} \quad \frac{1}{||W||}$等价于$\min_{W,b} \quad \frac{1}{2}||W||^2$，且$\begin{cases} W^T x_i + b \geq +1, & if\ y_i = +1 \\ W^T x_i + b \leq -1, & if\ y_i = -1 \end{cases}$等价于$y_i(W^T x_i + b) \geq 1$，则式（4）可转化为：

$$\begin{aligned} & min_{W,b} \quad \frac{1}{2}||W||^2 \\ s.t. & \quad 1 - (y_i(W^T x_i + b)) \leq 0 \end{aligned} \tag{5}$$

式（5）即为标准二次规划（QP）问题，对此我们可以根据拉格朗日对偶性（Lagrange duality）将其转化为对偶问题求解。转化为对偶问题求解其好处主要有三：

- 对于二次规划的原问题其时间复杂度通常为$O(M^3)$，与样本维度M有关而与样本数目N$无关。将其转化为对偶问题后，其算法的复杂度将只与样本数目有关，而与维度无关。对于样本维度多于样本数目的数据，其复杂度将显著降低；
- 转化为对偶问题后，kernel的引入更加容易；
- 无论原问题的凹凸性如何，其对偶问题均为凸优化问题，因此我们总能通过梯度下降找到全局最优解（当然，对于二次规划其本身即为凸优化问题）。

Lagrange duality

假设$f_0(x), f_i(x), h_j(x)$在定义的$x \in R^n$空间内连续可微，对于原问题：

$$\begin{aligned} & min \quad f_0(x) \\ s.t & \quad f_i(x) \leq 0, i = 1, ..., m \\ & \quad h_j(x) = 0, j = 1, 2, ..., n \end{aligned} \tag{6}$$

我们引入拉格朗日函数：

$$L(x, \lambda, v) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{n} v_j h_j(x) \tag{7}$$

上式中，λ, v_j均为拉格朗日乘子，且$\lambda_i \geq 0$。

则式（6）将等价于：

$$g(\lambda, v) = min_{x \in R^n} \quad max L(x, \lambda, v)_{\lambda \geq 0, v} = min\ max(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{n} v_j h_j(x)) \tag{8}$$

记原问题最优解$min f_0(x) = p^*$，可以看出若x不满足约束条件$\begin{cases} f_i(x) \leq 0, i = 1, ..., m \\ h_j(x) = 0, j = 1, 2, ..., n \end{cases}$时，则有$max L(x, \lambda, v)_{\lambda \geq 0, v} \rightarrow +\infty$（$\lambda, v \rightarrow +\infty$），此时$g(\lambda, v) \rightarrow +\infty$而没有最小值$min$，原问题$f_0(x) \neq p^*$，而当$x$满足约束条件时，则有$g(\lambda, v) \leq min\ f_0(x) = p^*$，即：

$$\begin{cases} g(\lambda, v) \rightarrow \infty, f_0(x) \neq P^*, & \text{不满足约束条件} \\ g(\lambda, v) \leq p^*, & \text{满足约束条件} \end{cases} \tag{9}$$

从式（8）中可以看出原问题为$minmax$问题。

定义原问题的对偶问题为：

$$d(\lambda, v) = max_{\lambda \geq 0, v} \quad min_x\ L(x, \lambda, v)$$

可以看出原问题的对偶问题为$maxmin$问题。记对偶问题$d(\lambda, v)$的最优解为d^*，若原问题和其对偶问题均有最优解，则当对偶问题取得最优解d^*时，x必满足约束条件：

$$\begin{cases} f_i(x) \leq 0 \\ h_j(x) = 0 \\ min_x\ L(x, \lambda, v) \leq p^* \\ d^* = max_{\lambda \geq 0, v} \quad min_x\ L(x, \lambda, v) \leq p^* \end{cases} \tag{10}$$

这里我们称$d^* \leq p^*$为弱对偶性。若$f_0(x), f_i(x)$均为凸函数，$h_j(x)$为仿射函数（多项式形式），且不等式约束（$f_i(x) \leq 0$）严格成立（不能取等），此时有$d^* = p^*$，我们称其为强对偶性。在强对偶性的条件下有对偶问题的最优解即为原问题的最优解。

综上所述，当x满足下式时：

$$\begin{cases} \lambda \geq 0 \\ \lambda f_i(x) = 0, i = 1, 2, .., n \\ f_i(x) \leq 0，\ h_j(x) = 0 \\ \nabla f_0(x) + \sum_{i=1}^{m} \lambda_i \nabla f_i(x) + \sum_{j=1}^{n} v_j \nabla h_j(x) = 0 \end{cases} \tag{11}$$

此时强对偶性成立，且原问题与其对偶问题同时取到相同最优解。式（11）即为著名的KKT条件（Karush-Kuhn-Tucker Conditions）。

SVM(hard margin)

回到式（5），这里我们引入拉格朗日乘子，有：

$$\begin{aligned} L(W, b, \alpha) &= \frac{1}{2}||W||^2 + \sum_{i=1}^{n} \alpha_i [1 - y_i(W^T x_i + b)] \\ &= \frac{1}{2} W^T W - W^T \sum_{i=1}^{n} \alpha_i y_i x_i - b \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i \\ & s.t \quad \alpha \geq 0 \end{aligned} \tag{12}$$

对比式（12）与Lasso回归，这里我们可以将$\frac{1}{2}||W||^2$看做目标函数，而在Lasso回归中其即为正则化项二范数，而约束条件$\sum_{i=1}^{n} \alpha_i [1 - y_i(W^T x_i + b)]$则可以看做铰链损失函数（hinge lossfunction），而Lasso回归中其损失函数则为最小二乘。

为求式（5）的对偶问题最优解，我们将$L(W, b, \alpha)$分别对W, b求偏导，即求解$minL(W, b, \alpha)_x$，有：

$$\begin{aligned} \frac{\partial L(W, b, \alpha)}{\partial W} &= W - \sum_{i=1}^{n} \alpha y_i x_i = 0 \\ \frac{\partial L(W, b, \alpha)}{\partial b} &= -\sum_{i=1}^{n} \alpha y_i = 0 \end{aligned} \tag{13}$$

将式（13）带入式（12）有：

$$\begin{aligned} L(W, b, \alpha) &= \frac{1}{2} W^T W - W^T \sum_{i=1}^{n} \alpha_i y_i x_i - b \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i \\ &= \frac{1}{2} W^T W - W^T W + \sum_{i=1}^{n} \alpha_i = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} W^T W \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} (\alpha_i y_i x_i)^T \sum_{i=1}^{n} \alpha_i y_i x_i = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \alpha_i y_i x_i^T \sum_{i=1}^{n} \alpha_i y_i x_i \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j \end{aligned} \tag{14}$$

故式（5）的对偶问题即为：

$$\begin{aligned} max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ s.t. & \begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ \alpha_i \geq 0, i = 1, 2, ..., n \end{cases} \end{aligned} \tag{15}$$

观察上式，其仍为二次规划问题。若采用二次规划算法求解，起时间复杂度和空间复杂度将分别为：$O(N^3), O(N^2)$，与样本数目有关。当样本数目过多时这显然是不可行的，对此我们需要采用一种更高效的方法及SMO算法。

SMO算法的简单来说就是，每次选择一对α_i, α_j然后固定其它所有参数，同时求解式（15）。不断反复这一过程直至收敛。具体来说，当我们仅考虑α_i, α_j时，有：

$$\begin{aligned} & \alpha_i y_i + \alpha_j y_i + \sum_{k \neq i,j}^{n} \alpha_k y_k = \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 记 \alpha_i y_i + \alpha_j y_i = c, 则有 \sum_{k \neq i,j}^{n} \alpha_k y_k = -c \end{aligned} \tag{16}$$

由于$\alpha_k(k \neq i,j)$为固定值，因此，c为一常数。此时我们用α_i表示$\alpha_i y_i + \alpha_j y_i = c$中的$\alpha_j$，则式（15）即转化为单变量的二次规划问题，其约束仅为$\alpha_i \geq 0$，此时将可直接求出最优解。通过SMO算法将大大降低问题的复杂度。

考虑式（15）为原问题的对偶问题，因此其必满足KKT条件，则有$\alpha_i(y_i f(x_i) - 1) = 0$，即$\alpha_i = 0$或$y_i f(x_i) = 1$。当$y_i f(x_i) = 1$时其为支持向量，而当其不等于1时必有$\alpha_i = 0$。因此最终的模型$f(x) = W^T x + b = \sum_{i=1}^{n} \alpha_i y_i x_i^T x_i + b$中将只考虑$\alpha_i$不等于0的样本$(x_i, y_i)$，也即支持向量（这就说明SVM模型将只与支持向量有关，而忽视不在边界上的样本，这也是为何SVM在大量的样本条件下效果不如神经网络的一个原因）。

由式（12）知：

$$y_i(\sum_{i=1}^{n} \alpha_i y_i x_i^T x_i + b) = 1 \tag{17}$$

理论上我们可以根据任一(x_i, y_i, α_i)求解偏移量b，但是为使模型更加鲁棒，我们一般利用所用支持向量求其平均值:

$$b = \frac{1}{|SV|} \sum_{s \in S} (\frac{1}{y_s} - \sum_{i \in S} \alpha_i y_i x_i^T x_i) \tag{18}$$

则最终SVM的模型即为：

$$g(x) = W^T x + b = (\sum_{s \in S} \alpha_s y_s x_s)^T x + \frac{1}{|SV|} \sum_{s \in S} (\frac{1}{y_s} - \sum_{i \in S} \alpha_i y_i x_i^T x_i) \tag{19}$$

上式中，S即为支持向量构成的集合，$|SV|$即为支持向量的数目。

上式即可完成二分类问题，对于多分类问题，简单的做法即是构造多个二分类问题求解。

SVM(soft margin)

在某些情况下，由于存在噪声数据或数据标签缺失或数据部分重叠，我们将很难找到最优的超平面将数据完全分开或使数据全部落到边界之外（± 1）。或者有时即使我们能够找到超平面完成分类，然而由于过拟合的存在，其也并非是最优的。因此我们有必要引入松弛变量以解决上述问题。

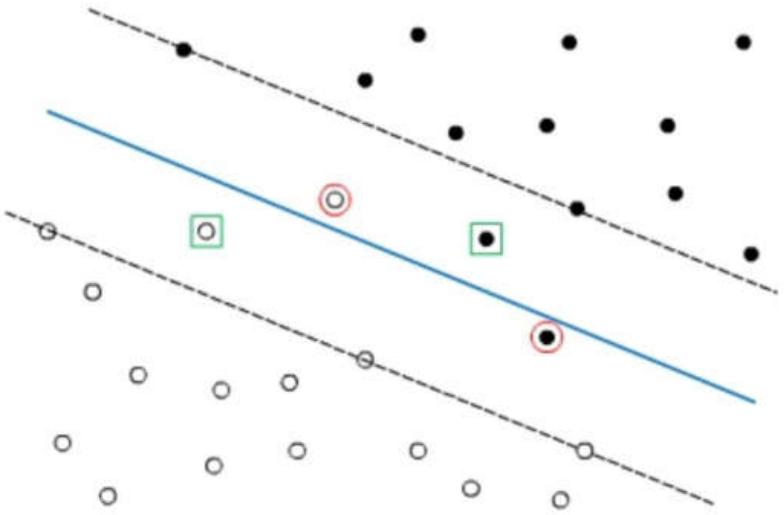

图2. 软间隔

考虑hard margin时其约束为：

$$y_i(W^T x_i + b) \geq 1$$

这里引入松弛变量$\xi_i \geq 0, i = 1, 2, , ...n$，则上述约束即变为：

$$y_i(W^T x_i + b) \geq 1 - \xi_i \tag{20}$$

很明显，当ξ很大以至$\rightarrow +\infty$时，式（20）将永远成立。因此为避免trivial solution，我们必须在目标函数中引入惩罚项，使得不满足约束的样本尽可能的小。则式（5）即变为：

$$min\frac{1}{2}||W||^2+C\sum_{i=1}^{n}\xi_i$$
$$s.t.\begin{cases}y_i(W^Tx_i+b)\geq 1-\xi_i\\ \xi_i\geq 0, i=1,2,...n\end{cases} \tag{21}$$

上式中，$C\sum_{i=1}^{n}\xi_i$即为惩罚项，C即为惩罚因子（该参数较为敏感，有关C的确定也有较多研究）。可以看出，若松弛变量越大则惩罚越大，若松弛变量为0时，惩罚也将为0。此外我们还可以将惩罚项理解为正则操作，而C即为调节偏差-方差的变量α。

同理，引入拉格朗日函数，则有：

$$L(W,b,\alpha,\xi,\mu)=\frac{1}{2}||W||^2+C\sum_{i=1}^{n}\xi_i$$
$$=\frac{1}{2}||W||^2+\sum_{i=1}^{n}\alpha_i[1-\xi_i-y_i(W^Tx_i+b)]-\sum_{i=1}^{n}\mu_i\xi_i \tag{22}$$
$$s.t\quad \alpha\geq 0,\mu\geq 0$$

其中α,μ为拉格朗日乘子。

式（22）分别对w,b,μ_i球偏导为0，则有：

$$\begin{cases}W=\sum_{i=1}^{n}\alpha_iy_ix_i\\ \sum_{i=1}^{n}\alpha_iy_i=0\\ C=\alpha_i+\mu_i\end{cases} \tag{23}$$

式（22）的对偶问题即为：

$$max_{\alpha}\quad \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_jy_iy_jx_i^Tx_j$$
$$s.t.\begin{cases}\sum_{i=1}^{n}\alpha_iy_i=0\\ 0\leq\alpha_i\leq C, i=1,2,...,n\end{cases} \tag{24}$$

对比式（24）与式（15），可以发现其唯一的区别即为约束条件不同。其对偶问题需满足的KKT条件即为：

$$\begin{cases}\alpha_i\geq 0,\mu_i\geq 0\\ y_if(x_i)-1+\xi_i\geq 0\\ \alpha_i(y_if(x_i)-1+\xi_i)=0\\ \xi_i\geq 0,\mu_i\xi_i=0\end{cases} \tag{25}$$

Kernel function

上文讨论的问题中，我们均假设，原始样本空间线性可分，然而在实际中大量的情况为线性不可分问题，一种可行的做法是将原始样本特征空间通过某种非线性变换映射至更高维的特征空间使其线性可分。这里我们引入kernel trick。

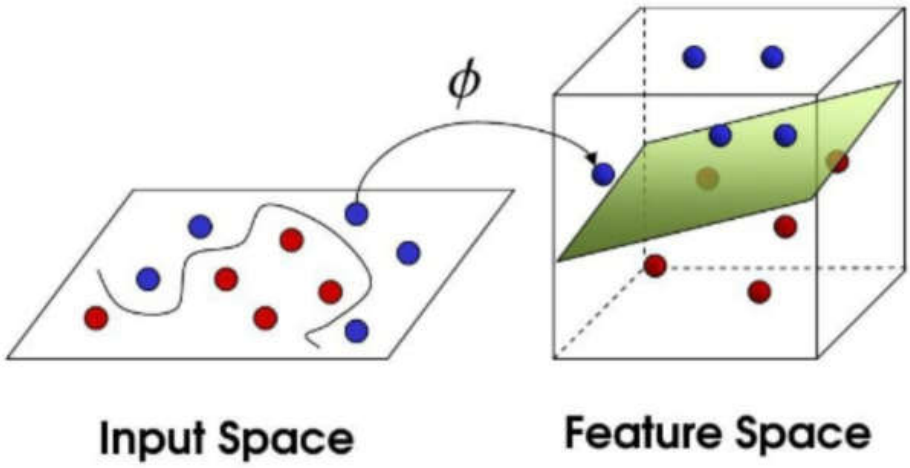

图3. 高维映射

这里我们记$\phi(x)$为映射函数，则此时支持向量模型变为$g(x)=W^T\phi(x)+b$，式（5）即变为：

$$\begin{aligned} & min_{W,b} \quad \frac{1}{2}||W||^2 \\ & s.t. \quad 1-(y_i(W^T\phi(x_i)+b)) \leq 0 \end{aligned} \tag{26}$$

其对偶问题则变为：

$$\begin{aligned} max_{\alpha} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \phi(x_i)^T\phi(x_j) \\ s.t. & \begin{cases} \sum_{i=1}^{n}\alpha_i y_i = 0 \\ \alpha_i \geq 0, i=1,2,...,n \end{cases} \end{aligned} \tag{27}$$

这里我们记：

$$k(x_i,x_j) =< \phi(x_i)\phi(x_j) >= \phi(x_i)\phi(x_j) \tag{28}$$

$k(x_i,x_j)$即为核函数。在很多情况下由于样本的高维映射$\phi(x_i)^T\phi(x_j)$计算困难，通过核函数的引入，我们可以在低维空间求解问题。此外，核函数的引入同时也确保了我们可以利用有效收敛的凸优化技术学习非线性模型。长见的的kernel function主要包括：

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j)^d$	$d\geqslant 1$ 为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma^2}\right)$	$\sigma>0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|}{\sigma}\right)$	$\sigma>0$
Sigmoid 核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\tanh(\beta\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j+\theta)$	tanh 为双曲正切函数, $\beta>0,\theta<0$

图5. kernel function

其中高斯核又被称为RBF kernel，而Sigmoi核可以等价为神经网络中的两隐层激活函数。因此SVM模型将变为：

$$\begin{aligned} g(x) = W^Tx+b &= (\sum_{s\in S}\alpha_s y_s \phi(x_s)^T\phi(x) + \frac{1}{|SV|}\sum_{s\in S}(\frac{1}{y_s} - \sum_{i\in S}\alpha_i y_i \phi(x_i)^T\phi(x_i)) \\ &= \sum_{s\in S}\alpha_s y_s k(x,x_s) + \frac{1}{|SV|}\sum_{s\in S}(\frac{1}{y_s} - \sum_{i\in S}\alpha_i y_i k(x_i,x_i)) \end{aligned} \tag{29}$$

因此从上文分析可知，此时SVM模型的关键问题将变为核函数的设计和选择（对比神将网络模型，其映射函数有学习而得，而SVM核函数则需要手工设计）。2004-2005年kernel learning曾是一个hot point，有关该问题的研究主要包括两方面：①learning kernel matrix（适用范围仅限该问题）；②learning kernel function。同时通过kernrl function求解映射函数也是一个研究方向，通常的做法包括kernel-PCA。这里我们以二次kernel和Gaussian kernel为例求解映射函数。

(1)Polynomial kernel(k=2)

$$k(x,x') = (x^Tx'+1)^2 = (x_1x_1'+x_2x_2'+1)^2 = (x_1)^2(x_1')^2 + (x_2)^2(x_2')^2 + 2x_1x_1'x_2x_2' + 2x_1x_1' + 2x_2x_2' + 1 \tag{30}$$

则：

$$\phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, (x_1)^2, (x_2)^2)^T \tag{31}$$

(2)Gaussian kernel

$$\begin{aligned} k(x,x') = exp(-||x-x'||^2/2\sigma^2) &= exp(-(x^Tx - 2x^Tx' + (x')^Tx')/2\sigma^2) = exp(-x^Tx/2\sigma^2)exp(x^Tx'/\sigma^2)exp(-(x')^Tx'/2\sigma^2) \\ &= exp(-(x_1x_1+x_2x_2)/2\sigma^2)exp((x_1x_1'+x_2x_2')/\sigma^2)exp(-(x_1'x_1'+x_2'x_2')/2\sigma^2) \end{aligned} \tag{32}$$

则：

$$\phi(x) = (exp(-x_1x_1/\sqrt{2}\sigma), exp(-x_2x_2/\sqrt{2}\sigma), exp(x_1)/\sigma, exp(x_2)/\sigma) \tag{33}$$

Loss function & Regularization

最后我们来讨论SVM中的loss function与regularization。观察式（21）有：

$$min\frac{1}{2}||W||^2 + C\sum_{i=1}^{n}\xi_i = \frac{1}{2}||W||^2 + C\sum_{i=1}^{n}max(0, 1-y_i(W^Tx_x+b)) = \frac{1}{2}||W||^2 + C\sum_{i=1}^{n}l_{0/1}(y_i(W^Tx_x+b)) \tag{34}$$

式（34）即为hinge损失。常用的损失函数主要包括：

- hinge loss：$l_{hinge}(z) = max(0, 1-z)$；
- exponential loss；$l_{exp}(z) = exp(-z)$；
- logistic loss；$l_{log}(z) = log(1+exp(-z))$；

其图像如下：

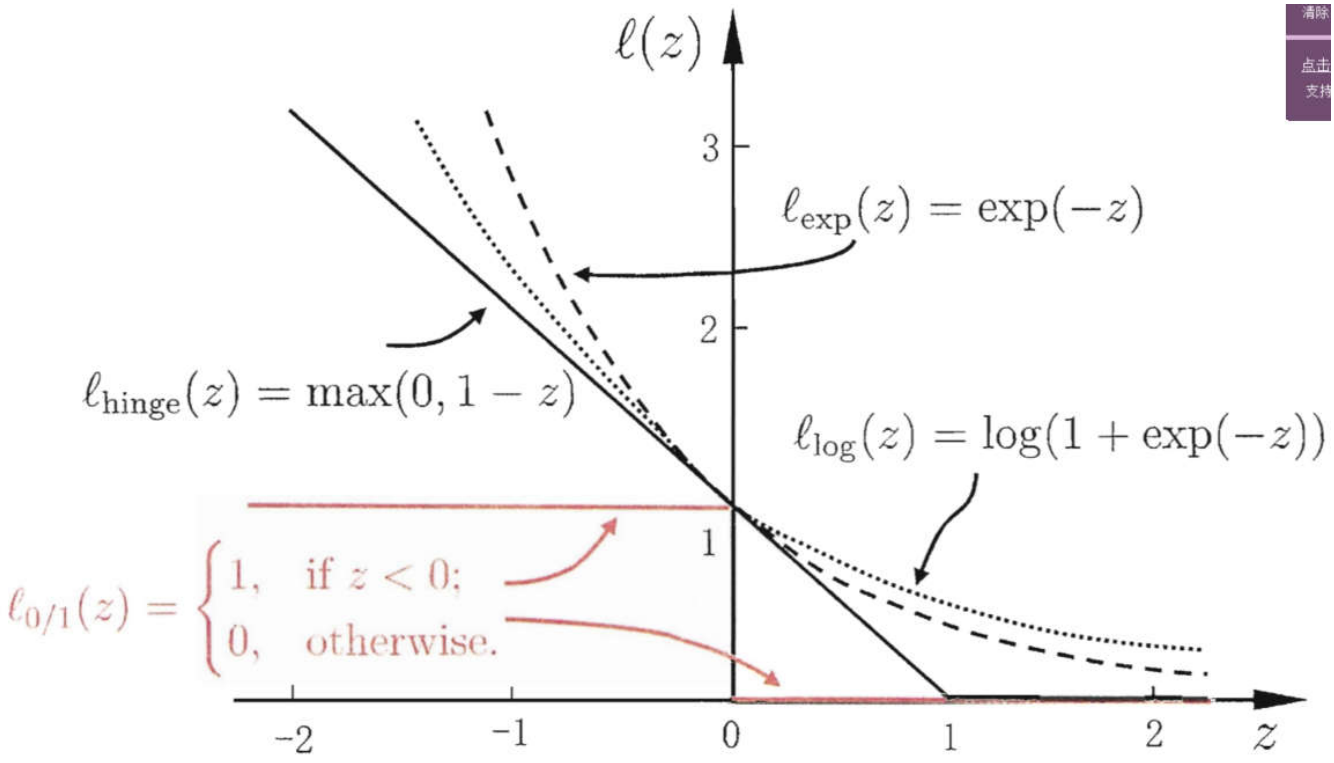

图6. loss function

上文讨论可知，当loss function为hinge loss时，SVM model类似于Lasso regression，而当loss function为logistic loss时，SVM model类似于logistics regression，所不同的是logistics regression的结果为类别概率，而SVM为类别，且logistics regression能直接用于多分类问题，而SVM需要进行推广。比较hinge loss与logistic loss，由于hinge loss具有一段为0的区域，因而更容易得到稀疏解。更一般的我们可以将式（34）写成如下形式：

$$min_g \quad \underbrace{C\sum_{i}^{n} l(g(x_i), y_i)}_{(1)} + \underbrace{\Omega(g)}_{(2)} \tag{35}$$

上式中的第一项即被我们常称为的经验风险（empirical risk），用于反映偏差，即各种损失函数。第二项则为结构风险（structural risk），反映模型的方差，即过拟合程度，通常为各种范数。而C可以认为 bias-variance trade-off调节参数。

Reference

[1] Bishop C M, 박원석. Pattern Recognition and Machine Learning, 2006[M]. Academic Press, 2006.

[2] 周志华. 机器学习[M]. 清华大学出版社, 2016.

[3] 李航. 统计学习方法[M]. 清华大学出版社, 2012.

[4] Cortes C , Vapnik V . Support-Vector Networks[J]. 1995.

[5] Platt J C. Fast training of support vector machines using sequential minimal optimization[M]// Advances in kernel methods. MIT Press, 1999:185-208.