支持向量机

蔡少伟

中国科学院大学

2016

概述

支持向量机(Support Vector Machine，SVM)

- 由AT&T Bell实验室研究小组Corinna Cortes和Vapnik于1995年首先提出的分类技术。
- 有坚实的统计学理论基础。
- 由于在许多领域(生物信息学，手写识别和文本分类等)显示出卓越性能，很快成为机器学习的主流技术，并直接掀起了“统计学习”在2000年前后的热潮。
- 针对二分类任务设计的，对多分类任务要进行专门的推广。
- 可以很好地用于分类高维数据，避免了维度灾难问题。
- 基于判别式分类。
- 使用最大边缘原理。

划分超平面

给定训练样本集$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_m, y_m)\}$, $y_i \in \{-1, +1\}$，分类学习最基本的思想就是基于训练集D在样本空间找到一个划分超平面，将不同类的样本分开。

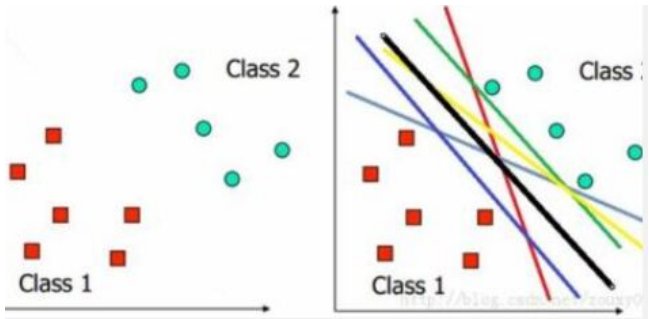

Figure: 存在多个划分超平面把两类训练样本分开

能将训练样本分开的划分超平面可能有很多，我们应该找哪一个呢？（假设样本是线性可分的）

划分超平面

划分超平面可用如下线性方程来描述

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

其中，$\mathbf{w}$为法向量，决定了超平面的方向；b为位移项，决定了超平面与原点之间的距离。我们把法向量$\mathbf{w}$和位移项b决定的超平面记为$(\mathbf{w}, b)$。

- 样本空间中任意点$\mathbf{x}$到超平面$(\mathbf{w}, b)$的距离为

$$r = \frac{|\mathbf{w}^T\mathbf{x} + b|}{\| \mathbf{w} \|} \quad (\| \mathbf{w} \| = \sqrt{\sum |w_i|^2})$$

- 假设超平面$(\mathbf{w}, b)$能将训练样本分类正确，即对于$(\mathbf{x}_i, y_i) \in D$，若$y_i = +1$，则$\mathbf{w}^T\mathbf{x}_i + b > 0$; 若$y_i = -1$，则$\mathbf{w}^T\mathbf{x}_i + b < 0$。
我们总可以通过缩放变换使得下面式子成立

$$f(x) = \begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq +1 & , y_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 & , y_i = -1 \end{cases}$$

支持向量和最大化间隔

距离超平面最近的几个训练样本点使得式子的等号成立，它们被称为支持向量(support vector) 。

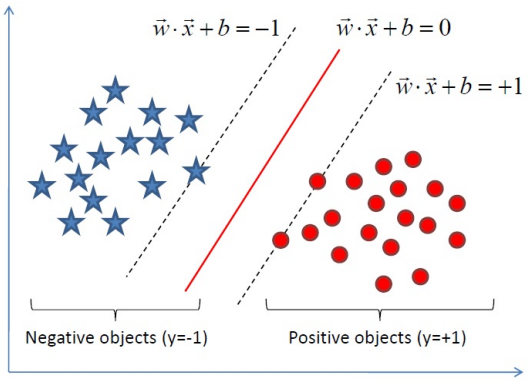

Figure: 支持向量

两个异类支持向量到超平面的距离之和称为间隔(margin) ，为

$$\gamma = \frac{2}{\| w \|}$$

最大间隔原理：找到具有最大间隔的划分超平面，也即寻找参数$\mathbf{w}$和b，使得

$$max_{\mathbf{w},b} \frac{2}{\| \mathbf{w} \|}$$

$$s.t. \;\; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, 2, ..., m.$$

SVM基本型

显然，为了最大化间隔，只需要最大化$\frac{1}{\|\mathbf{w}\|}$，这等价于最小化$\| \mathbf{w} \|^2$。于是，上式可以重写为

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

这就是支持向量机（SVM）的基本型。

- 通过求解上式可以得到最大间隔超平面对于的模型$f(\mathbf{x}) = \mathbf{w}^T \mathbf{w} + b$
- SVM基本型是一个凸优化问题，或者更具体地说，它是一个二次规划问题— 目标函数是二次的，约束条件是线性的。这个问题可以用任何现成的QP (Quadratic Programming)的优化包进行求解
- 通过拉格朗日对偶（Lagrange Duality）变换到对偶变量(dual variable)的优化问题进行求解。
  - 对偶问题存在更高效的解法；
  - 可以自然的引入核函数，进而推广到非线性分类问题。

拉格朗日对偶

先考虑只有一个等式约束的优化问题

$$\begin{aligned} & min_{\mathbf{x}} f(\mathbf{x}) \\ & s.t.\ \ h(\mathbf{x}) = 0, \quad i = 1, 2, ..., m. \end{aligned}$$

从几何角度看，我们的目标就是要在有方程$h(\mathbf{x}) = 0$确定的$m-1$维曲面上寻找使得目标函数$f(\mathbf{x})$最小的点。
从几何上不难观察以下结论

- 约束曲面上的任意一点$\mathbf{x}$的梯度$\Delta h(\mathbf{x})$正交于约束曲面；
- $d\mathbf{x}$的变化方向与$f(\mathbf{x})$的梯度垂直时才能获得最优值,也就是说，在最优点处，目标函数的梯度$\Delta f(\mathbf{x})$正交于约束曲面。

通过引入拉格朗日算子γ，得到拉格朗日函数

$$L(\mathbf{x}, \gamma) = f(\mathbf{x}) + \gamma h(\mathbf{x})$$

分别对$\mathbf{x}$和γ求偏导，使得偏导数等于0，解出$\mathbf{x}$和γ,得到对偶问题的最优（最大）值。这个时候，所求得的$\mathbf{x}$也是原问题的最优解。

拉格朗日对偶

扩展到多个等式约束的优化问题

$$\begin{aligned} & min_{\mathbf{x}} f(\mathbf{x}) \\ & s.t.\ \ h_i(\mathbf{x}) = 0, \quad i = 1, 2, ..., m. \end{aligned}$$

通过引入拉格朗日算子$\gamma = (\gamma_1, \gamma_2, ..., \gamma_m)^T$，得到拉格朗日函数

$$L(\mathbf{x}, \gamma) = f(\mathbf{x}) + \sum_{i=1}^{m} \gamma_i \cdot h_i(\mathbf{x})$$

分别对$\mathbf{x}$和γ求偏导，使得偏导数等于0，解出$\mathbf{x}$和γ,得到对偶问题的最优（最大）值。这个时候，所求得的$\mathbf{x}$也是原问题的最优解。

拉格朗日对偶

现在考虑包含不等式约束的优化问题

$$\begin{aligned} & min_{\mathbf{x}} f(\mathbf{x}) \\ s.t.\ & h_i(\mathbf{x}) = 0, \quad i = 1, 2, ..., m, \\ & g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, ..., n, \end{aligned}$$

通过引入拉格朗日算子$\gamma = (\gamma_1, \gamma_2, ..., \gamma_m)^T$和$\mu = (\mu_1, \mu_2, ..., \mu_n)^T$，得到拉格朗日函数

$$L(\mathbf{x}, \gamma, \mu) = f(\mathbf{x}) + \sum_{i=1}^{m} \gamma_i \cdot h_i(\mathbf{x}) + \sum_{j=1}^{n} \mu_j \cdot g_j(\mathbf{x})$$

为了处理不等式约束，还需要引入Karush-Kuhn-Tucker(KKT)条件:

$$\begin{cases} g_j(\mathbf{x}) \leq 0; \\ \mu_j \geq 0; \\ \mu_j \cdot g_j(\mathbf{x}) = 0. \end{cases}$$

拉格朗日对偶

带不等式约束的优化问题的拉格朗日对偶

- 不管主问题的凸性如何，对偶问题始终是凸优化问题。
- 对偶问题$maxL(\mathbf{x}, \gamma, \mu)$的最优值$d^*$给出了主问题最优值$p^*$的下界，即$d^* \leq g^*$。这称为弱对偶性。
- 若$d^* = g^*$，这称为强对偶性成立。若主问题是凸优化问题（如f和g均为凸函数，h为仿射函数，且可行域中至少有一点使得不等式严格成立），则此时强对偶性成立。

值得注意的是，在强对偶性成立时，将拉格朗日函数分别对原变量和对偶变量求导并令其导数为0，即可得到原变量与对偶变量的数值关系。于是，对偶问题解决了，主问题也就解决了。

SVM基本型的拉格朗日对偶

回到SVM基本型，把"≥"不等式描述为"≤"不等式，得

$$min_{\mathbf{w},b}\frac{1}{2}\parallel \mathbf{w} \parallel^2$$

$$s.t.\ \ 1-y_i(\mathbf{w}^T\mathbf{x}_i+b)\leq 0,\quad i=1,2,...,m.$$

对每条约束添加拉格朗日乘子$\alpha_i \geq 0$，该问题的拉格朗日函数为

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

令$L(\mathbf{w},b,\alpha)$对$\mathbf{w}$和b的偏导为0，得$\mathbf{w}=\sum_{i=1}^{m}\alpha_i y_i\mathbf{x}_i$,和$0=\sum_{i=1}^{m}\alpha_i y_i$，
代入$L(\mathbf{w},b,\alpha)$消去$\mathbf{w}$和b，得到对偶问题

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

$$s.t.\ \ \sum_{i=1}^{m}\alpha_i y_i=0,$$

$$\alpha_i\geq 0,\quad i=1,2,...,m.$$

解出α之后，求出$\mathbf{w}$和b即可得到模型

$$f(\mathbf{x})=\mathbf{w}^T\mathbf{x}+b=\sum_{i=1}^{m}\alpha_i y_i\mathbf{x}_i^T\mathbf{x}+b.$$

SVM基本型的拉格朗日对偶

通过对偶问题的求解得到模型

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b.$$

为了给一个新样本分类，要与所有的训练样本都做运算是不是太耗时了？
幸运的是，SVM基本型主问题中的不等式约束带来的KKT条件使得运算大大简化。

$$\begin{cases} 1 - y_i f(\mathbf{x}_i) \le 0 \iff y_i f(\mathbf{x}_i) - 1 \ge 0; \\ \alpha_i \ge 0; \\ \alpha_i \cdot (1 - y_i f(\mathbf{x}_i)) = 0. \end{cases}$$

由上述KKT条件，对任意训练样本$(\mathbf{x}_i, y_i)$，总有$\alpha_i = 0$或$y_i f(\mathbf{x}_i) = 1$。也就是，只有$y_i f(\mathbf{x}_i) = 1$成立的样本会对模型产生影响，而根据定义这些样本都是支持向量。

- 训练完成之后，大部分的训练样本都不需要保留，最终模型只与支持向量有关。
- 支持向量机这个名字强调了此类学习器的关键是如何从支持向量构建出解；也意味着其复杂度主要和支持向量的数目有关。

求解SVM基本型的对偶问题

回到SVM基本型对偶问题

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

$$s.t. \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \quad i = 1, 2, ..., m.$$

对偶问题求解之后可得模型

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b.$$

如何求出参数α和b呢？

- 直接用二次规划算法求解，则问题规模正比于训练样本数，开销太大
- 利用问题本身的特性提出高效算法，一个著名的代表是SMO(Sequential Minimal Optimization)算法。

SMO算法

SMO算法是一个迭代优化算法，在参数初始化之后，SMO迭代地每次更新两个变量α_i和α_j，直到收敛为止。通常来说，获得精确的最优值是不太现实的，因此需要定义近似最优条件，因此后面提及优化一词时，将代表近似最优解。

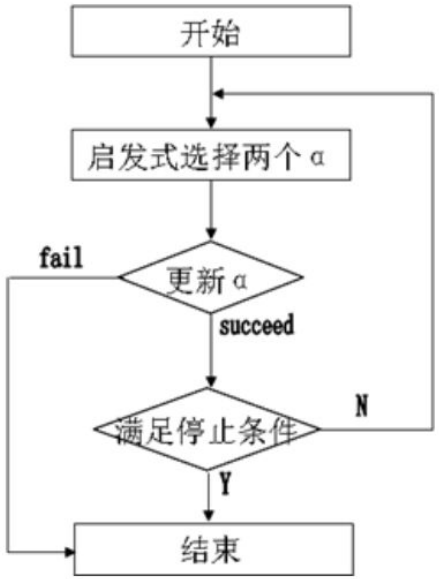

Figure: SMO算法框架

在每一个迭代步骤中，算法首先选取两个待更新的变量，在其他参数固定的前提下，找到这两个参数的最优值并更新，而这两个点乘子的优化可以获得解析解。最后再根据SVM的定义计算出偏移量b。

SMO算法

SMO算法的每一个迭代步

- 选取一对变量α_i和α_j;
- 固定α_i和α_j以外的变量，计算更新后的α_i和α_j。

选取变量

- 注意到对偶问题还需满足KKT条件，所以SMO先选取一个违背KKT条件程度最大的α_i；并且，Osuna定理告诉我们只要选择出来的两个α_i变量中有一个违背了KKT条件，那么目标函数在一步迭代后值会优化。
- 第二个变量α_j应选择一个可以使目标函数值改进最快的变量。但是因为比较所有变量对于的目标函数增幅复杂度过高，SMO采用了一个启发式：使得选取的两个变量对应的样本之间的间隔最大。一种直观解释是，这样选取的两个变量的更新，与对两个相似变量的更新相比，会带给目标函数数值更大的变化。

计算变量

计算α_i和α_j的更新值

记更新后的α_i和α_j的数值为α_i'和α_j' 考虑约束$\sum_{i=1}^{m}\alpha_i y_i = 0$，得到

$$\alpha_i' y_i + \alpha_j' y_j = -\sum_{k\neq i,j}\alpha_k y_k = \alpha_i y_i + \alpha_j y_j$$

由此消去其中一个变量，代入目标函数，则得到一个关于单变量α_i的二次规划问题，仅有的约束为$\alpha_i \geq 0$。这样的二次规划问题有闭式解。

计算偏移量b

模型为$f(\mathbf{x}) = \sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b = \sum_{i\in S}\alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b$,
其中$S = \{i|\alpha_i > 0, i = 1, 2, ..., m\}$为所有支持向量的下标集。
注意到对任意支持向量$(\mathbf{x}_s, y_s)$都有$y_s f(\mathbf{x}_s) = 1$，也即

$$y_s(\sum_{i\in S}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}_s + b) = 1$$

所以可选任意支持向量通过上式求得b。
现实任务中常用一种更鲁棒的做法，也即使用所有支持向量求解的平均值

$$b = \frac{1}{|S|}\sum_{s\in S}(\frac{1}{y_s} - \sum_{i\in S}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}_s)$$

注意到$y_s = +1$或-1，又可简化为$b = \frac{1}{|S|}\sum_{s\in S}(y_s - \sum_{i\in S}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}_s)$。

Remarks on SMO

- SMO算法之所以高效，就在于固定其他变量之后，仅优化两个变量的过程能做到非常高效。
- 尽管需要更多的迭代才收敛，每次迭代需要很少的操作，因此算法在整体上的速度上有数量级的提高。
- 算法没有矩阵操作，它不需要存储核矩阵，所需要的内存大小和训练数据集的大小线性增长，因此也有效降低了空间复杂度。

高维映射

在我们前面的讨论中，我们假设训练样本是线性可分的，即存在一个划分超平面能将样本正确分类。
然而在现实任务中，原始样本空间也许并非线性可分的。

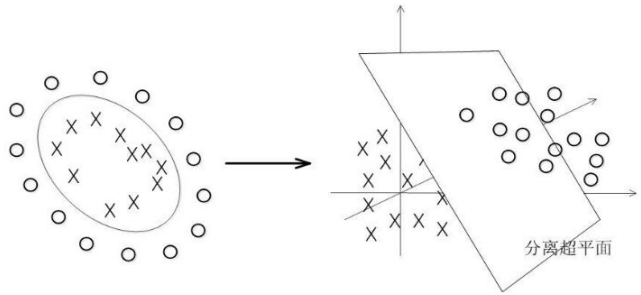

Figure: 非线性样本空间与空间映射

- 对于非线性可分的问题，可以将样本从原始空间映射到一个更高维的特征空间，使得样本在新的特征空间内线性可分。
- 如果原始空间是有限维，那么一定存在一个高维特征空间使得样本可分。

高维映射

记原向量$\mathbf{x}$映射得到的向量为$\phi(\mathbf{x})$，于是在映射后的特征空间中划分超平面对应的模型为

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

目标是找一个划分超平面$(\mathbf{w}, b)$, 满足

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

其对偶问题是

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

对偶问题求解之后可得模型

$$f(\mathbf{x}) = \sum_{i=1}^{m}\alpha_i y_i \phi(\mathbf{x}_i)^T\phi(\mathbf{x}) + b.$$

核函数

Now we have a problem:

- 上式的对偶问题的求解涉及到计算$\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$，这是样本$x_i$ 和x_j映射到高维空间之后的内积。
- 由于映射后的特征空间维数可能很高，甚至是无穷维，因此直接计算$\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$是困难的。

引入核函数来避开高维障碍！

设想这样一个函数

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

即$\mathbf{x}_i$与$\mathbf{x}_j$在特征空间的内积$\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$等于他们在原始样本空间中通过函数$\kappa(\cdot,\cdot)$计算的结果。这样的函数称为核函数。

有了核函数，我们就不必直接计算高维甚至无穷维特征空间中的内积。

核函数

引入核函数之后，对偶问题可以写成

$$max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

$$s.t. \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \quad i = 1, 2, ..., m.$$

求解之后可得模型

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \kappa(\mathbf{x}_i, \mathbf{x}_j) + b.$$

核函数

若已知映射$\phi(\cdot)$的具体形式，则可写出核函数$\kappa(\cdot,\cdot)$。
但是现实任务中我们一般不知道$\phi(\cdot)$是什么形式。那么什么样的函数能做核函数呢？

Theorem 1

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$\kappa$是核函数当且仅当对于任意数据$D=\{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_m\}$，“核矩阵”$K$总是半正定的。

$$\begin{bmatrix} \kappa(\mathbf{x}_1,\mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_1,\mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_1,\mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_i,\mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_i,\mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_i,\mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_m,\mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_m,\mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_m,\mathbf{x}_m) \end{bmatrix}$$

关于半正定矩阵

- 定义：如果$A\in R^{n\times n}$是实对称矩阵，且对任意的实非零向量$X\in R^n$有$X^TAX\geq 0$，就称A为半正定矩阵。
- 等价于A所有特征值非负
- 等价于A所有主子式非负
- ...

核函数

- 特征空间的选择对SVM的性能至关重要，核函数也隐式地定义了特征空间。
- 于是核函数选择就成为SVM的最大变数。
- 如果核函数选择不合适，意味着将样本空间映射到了一个不合适的特征空间，很可能导致性能不佳。

Table: 常见的核函数

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

一些经验：对文本数据通常采用线性核；情况不明的时候先尝试高斯核。

核函数

如果κ_1 和κ_2为核函数，则以下函数也是核函数：

- $\gamma_1\kappa_2 + \gamma_2\kappa_2$，$\gamma_1$和$\gamma_2$ 为任意正整数(线性组合)
- $\kappa_1 \otimes \kappa_2(\mathbf{x}, \mathbf{z}) = \kappa_1(\mathbf{x}, \mathbf{z})\kappa_2(\mathbf{x}, \mathbf{z})$ (直积)
- $g(\mathbf{x})\kappa_1(\mathbf{x}, \mathbf{z})g(\mathbf{x})$，其中$g(\cdot)$为任意函数