六、支持向量机

主讲教师：周志华

支持向量机基本型

最大间隔：寻找参数 $\boldsymbol{w}$ 和 b , 使得 γ 最大

$$\begin{aligned}
\underset{\boldsymbol{w},b}{\arg\max} \quad & \frac{2}{\|\boldsymbol{w}\|} \\
\text{s.t.} \quad & y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \ i = 1, 2, \ldots, m.
\end{aligned}$$

$$\begin{aligned}
\underset{\boldsymbol{w},b}{\arg\min} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\
\text{s.t.} \quad & y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \ i = 1, 2, \ldots, m.
\end{aligned}$$

凸二次规划问题，能用优化计算包求解，但可以有更高效的办法

对偶问题

拉格朗日乘子法

- 第一步：引入拉格朗日乘子 $\alpha_i \geq 0$ 得到拉格朗日函数

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)\right)$$

- 第二步：令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 对 $\boldsymbol{w}$ 和 b 的偏导为零可得

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i \ , \quad 0 = \sum_{i=1}^{m} \alpha_i y_i$$

- 第三步：回代可得

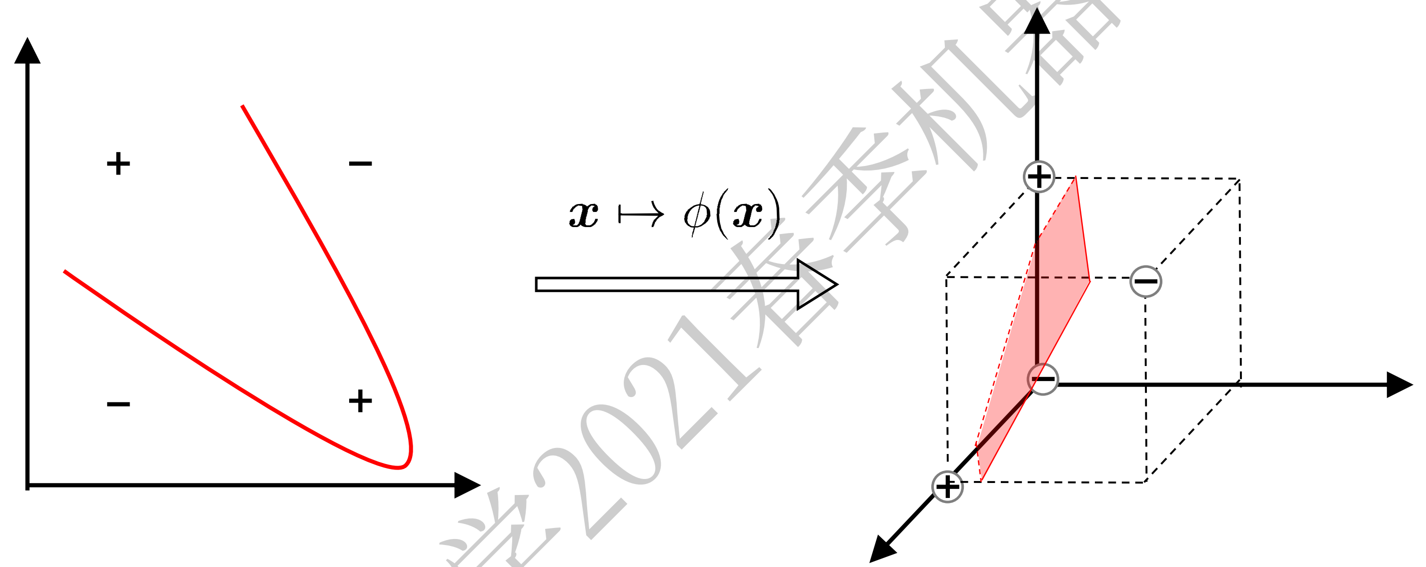
$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0 \ , \quad \alpha_i \geqslant 0 \ , \quad i = 1, 2, \ldots, m$$

特征空间映射

若不存在一个能正确划分两类样本的超平面，怎么办？

将样本从原始空间映射到一个更高维的特征空间，使样本在这个特征空间内线性可分



如果原始空间是有限维(属性数有限)，那么一定存在一个高维特征空间使样本可分

在特征空间中

设样本 $\boldsymbol{x}$ 映射后的向量为 $\phi(\boldsymbol{x})$，划分超平面为 $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$

原始问题

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{s.t.} \quad y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) \geq 1, \; i = 1, 2, \ldots, m.$$

对偶问题

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boxed{\phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j)}$$
$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0 \,, \quad \alpha_i \geqslant 0 \,, \quad i = 1, 2, \ldots, m$$

预测

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x}) + b = \sum_{i=1}^{m} \alpha_i y_i \boxed{\phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x})} + b$$

只以内积形式出现

核函数 (kernel function)

基本思路：设计核函数

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

绕过显式考虑特征映射、以及计算高维内积的困难

Mercer 定理：若一个对称函数所对应的核矩阵半正定，则它就能作为核函数来使用

任何一个核函数，都隐式地定义了一个RKHS (Reproducing Kernel Hilbert Space, 再生核希尔伯特空间)

"核函数选择"成为决定支持向量机性能的关键！

核函数

常用核函数

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	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\lVert \boldsymbol{x}_i - \boldsymbol{x}_j \rVert^2}{2\sigma^2}\right)$	$\sigma > 0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\lVert \boldsymbol{x}_i - \boldsymbol{x}_j \rVert}{\sigma}\right)$	$\sigma > 0$
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, $\beta > 0$, $\theta < 0$

基本经验：文本数据常用线性核，
情况不明时可先尝试高斯核

可通过函数组合得到：

若 κ_1和 κ_2 是核函数，则对任意正数 γ_1、γ_2 和任意函数 $g(\boldsymbol{x})$，

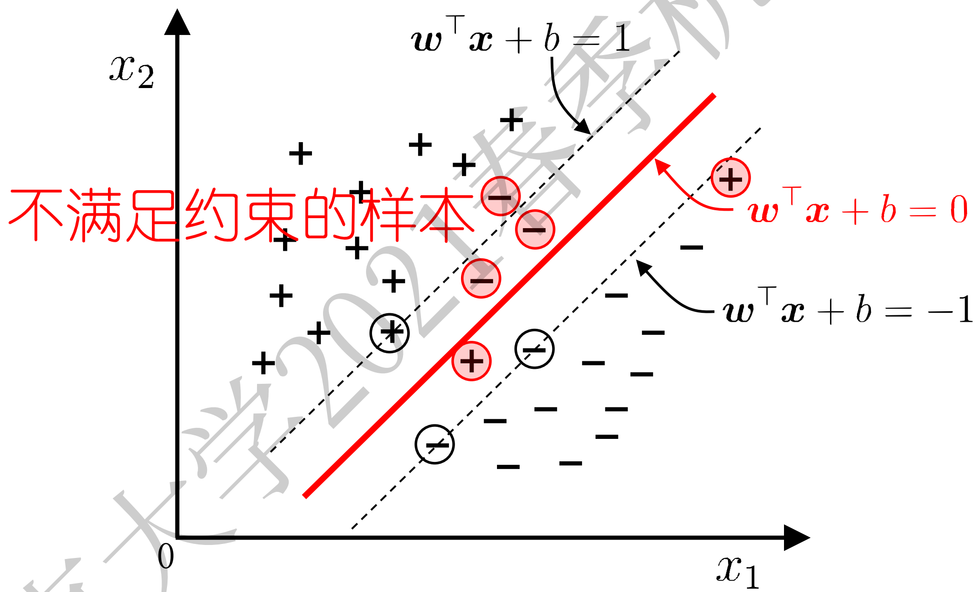
均为核函数 $\left\{\begin{array}{l} \gamma_1\kappa_1 + \gamma_2\kappa_2 \\ \kappa_1 \otimes \kappa_2(\boldsymbol{x}, \boldsymbol{z}) = \kappa_1(\boldsymbol{x}, \boldsymbol{z})\kappa_2(\boldsymbol{x}, \boldsymbol{z}) \\ \kappa(\boldsymbol{x}, \boldsymbol{z}) = g(\boldsymbol{x})\kappa_1(\boldsymbol{x}, \boldsymbol{z})g(\boldsymbol{z}) \end{array}\right.$

软间隔

现实中很难确定合适的核函数，使训练样本在特征空间中线性可分

即便貌似线性可分，也很难断定是否是因过拟合造成的

引入软间隔 (soft margin), 允许在一些样本上不满足约束



优化目标

基本思路：最大化间隔的同时，

让不满足约束 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1$ 的样本尽可能少

$$\boxed{\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\ell_{0/1}\left(y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b\right)-1\right)}$$

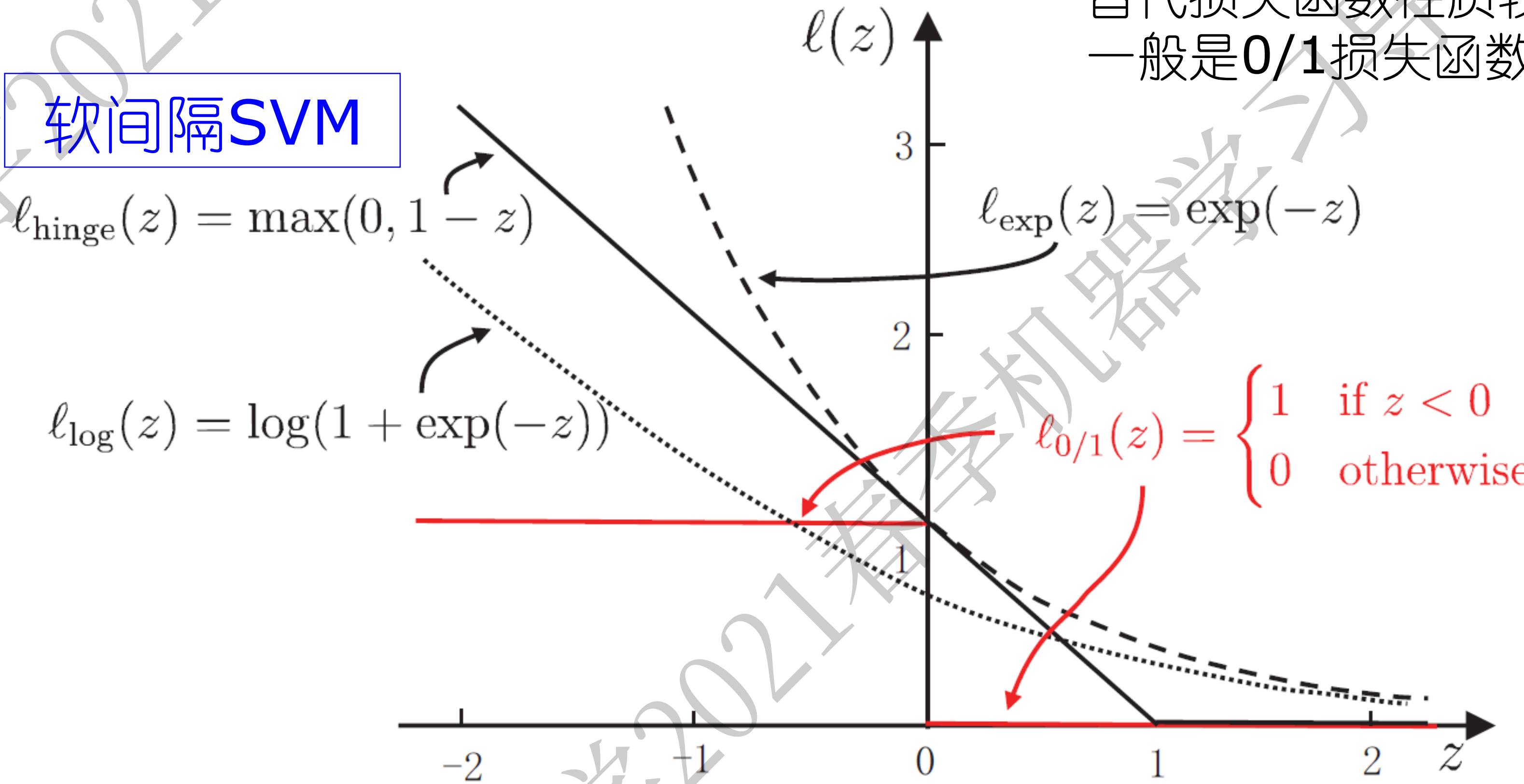
其中 $l_{0/1}$ 是 0/1损失函数 (0/1 loss function):

$$\ell_{0/1}(z)=\begin{cases}1, & \text{if } z<0;\\ 0, & \text{otherwise.}\end{cases}$$

障碍：0/1损失函数非凸、非连续，不易优化！

替代损失 (surrogate loss)

替代损失函数性质较好，
一般是0/1损失函数的上界



- 采用替代损失函数，是在解决困难问题时的常见技巧
- 求解替代函数得到的解是否仍是原问题的解？理论上称为替代损失的"一致性" (consistency)问题

软间隔支持向量机

原始问题

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m} \max\left(0, 1 - y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b\right)\right)$$

引入“松弛变量”(slack variables) ξ_i

$$\begin{aligned} \min_{\boldsymbol{w},b,\xi_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i \ , \ \xi_i \geqslant 0 \ , \ i = 1, 2, \ldots, m. \end{aligned}$$

对偶问题

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \ , \ \boxed{0 \leqslant \alpha_i \leqslant C} \ , \quad i = 1, 2, \ldots, m \ . \end{aligned}$$

与“硬间隔SVM”的区别

根据 KKT 条件可知，最终模型仅与支持向量有关，也即采用hinge 损失函数后仍保持了 SVM 解的稀疏性

正则化 (regularization)

统计学习模型（例如 SVM）的更一般形式

正则化项

$$\min_f \underbrace{\Omega(f)}_{\text{结构风险}} + C \sum_{i=1}^{m} \underbrace{l(f(\boldsymbol{x}_i), y_i)}_{\text{经验风险}}$$

结构风险 (structural risk)
描述模型本身的某些性质

经验风险 (empirical risk)
描述模型与训练数据的契合程度

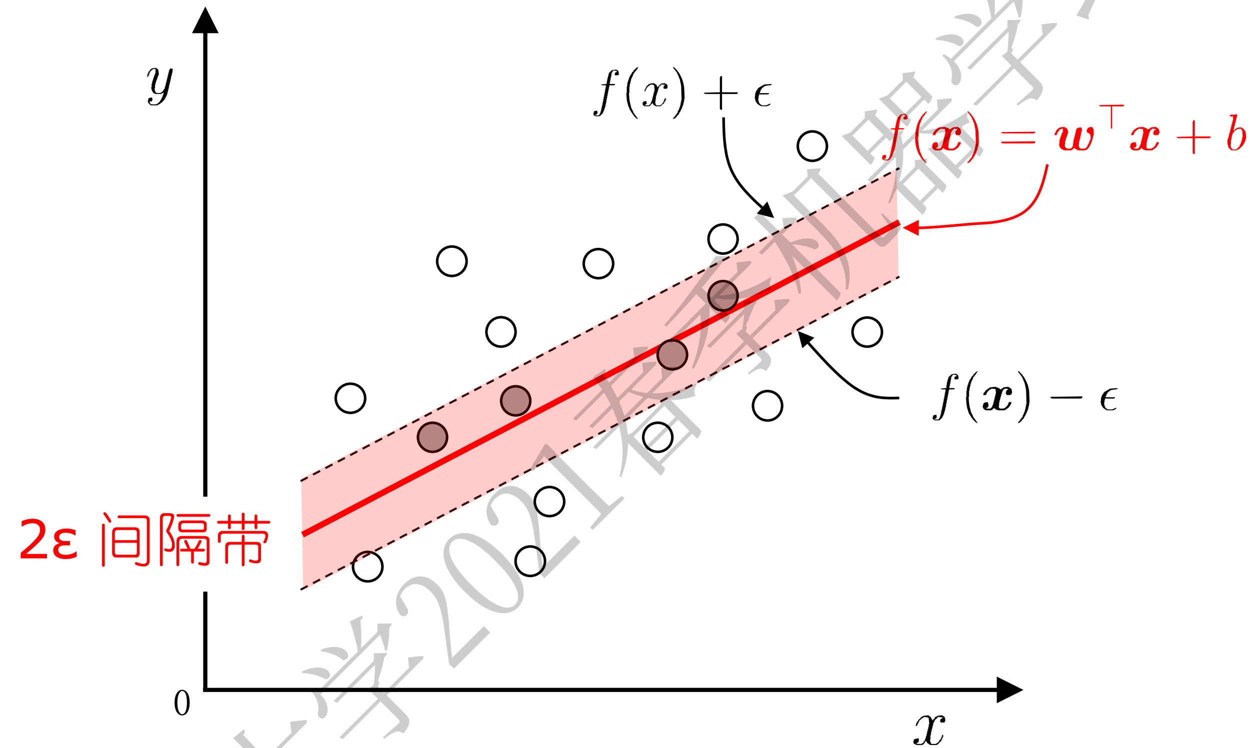
归纳偏好

- 正则化可理解为“罚函数法”
  通过对不希望的结果施以惩罚，使得优化过程趋向于希望目标
- 从贝叶斯估计的角度，则可认为是提供了模型的先验概率

如何使用SVM
解决自己特定的任务？

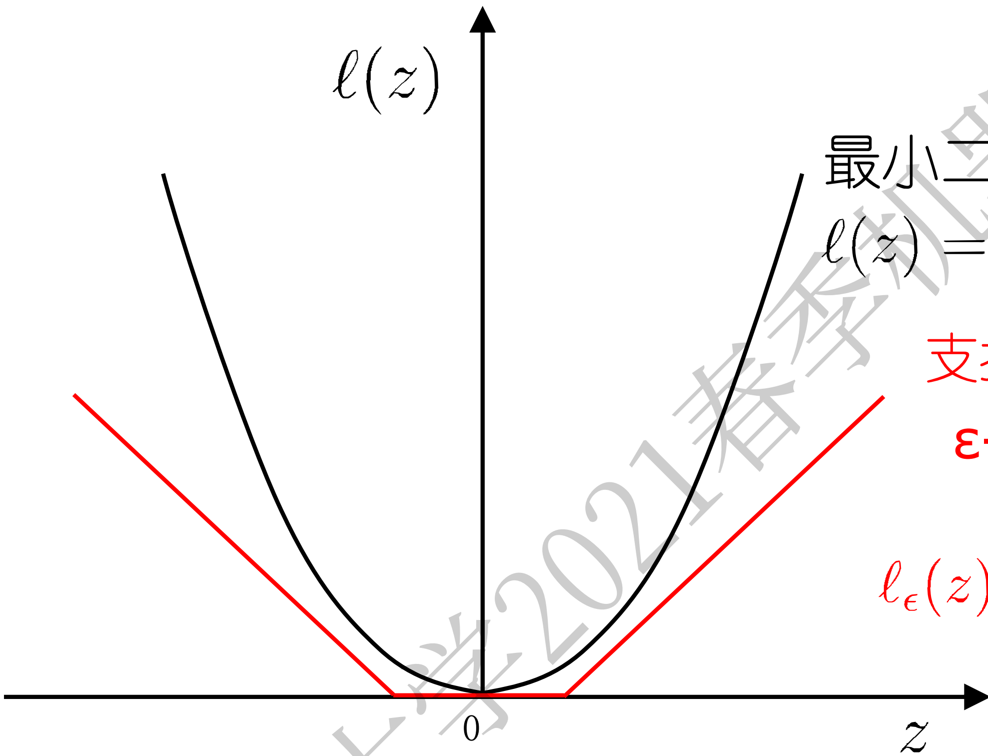
以回归学习为例

基本思路：允许模型输出与实际输出间存在 2ε 的差别



ε-不敏感(insensitive)损失函数

落入 2ε 间隔带的样本不计算损失



最小二乘损失函数

$\ell(z) = z^2$

支持向量回归使用的

ε-不敏感损失函数

$$\ell_\epsilon(z) = \begin{cases} 0 & \text{if } |z| \le \epsilon \\ |z| - \epsilon & \text{otherwise} \end{cases}$$

支持向量回归 (SVR)

原始问题

$$\begin{aligned}
\min_{\boldsymbol{w},b,\xi_i,\hat{\xi}_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \hat{\xi}_i) \\
\text{s.t.} \quad & f(\boldsymbol{x}_i) - y_i \leqslant \epsilon + \xi_i \ , \\
& y_i - f(\boldsymbol{x}_i) \leqslant \epsilon + \hat{\xi}_i \ , \\
& \xi_i \geqslant 0, \ \hat{\xi}_i \geqslant 0 \ , \ i = 1, 2, \ldots, m
\end{aligned}$$

对偶问题

$$\begin{aligned}
\max_{\boldsymbol{\alpha},\hat{\boldsymbol{\alpha}}} \quad & \sum_{i=1}^{m} y_i(\hat{\alpha}_i - \alpha_i) - \epsilon(\hat{\alpha}_i + \alpha_i) - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j \\
\text{s.t.} \quad & \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i) = 0 \ , \ \ 0 \leqslant \alpha_i, \hat{\alpha}_i \leqslant C
\end{aligned}$$

预测 $$f(\boldsymbol{x}) = \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x} + b$$

现实应用中……

如何使用SVM？

- 入门级—— 实现并使用各种版本SVM
- 专业级—— 尝试、组合核函数
- 专家级—— 根据问题而设计目标函数、替代损失、进而……

根据当前任务“度身定制”是关键

表示定理 (Representer Theorem)

观察
- 核 SVM: $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b \; = \sum_{i=1}^{m} \alpha_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b$
- 核 SVR: $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b \; = \sum_{i=1}^{m} (\hat{\alpha}_i - \alpha_i) \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b$

无论SVM还是SVR, 学得模型总能表示成核函数的线性组合

更一般的结论(表示定理): 对于任意单调递增函数 $\Omega : [0, \infty] \mapsto \mathbb{R}$ 和任意非负损失函数 $\ell : \mathbb{R}^m \mapsto [0, \infty]$, 优化问题

$$\min_{h \in \mathbb{H}} \; F(h) \; = \; \Omega(\|h\|_{\mathbb{H}}) + \ell\big(h(\boldsymbol{x}_1), h(\boldsymbol{x}_2), \ldots, h(\boldsymbol{x}_m)\big)$$

的解总可写为 $h^*(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i)$

核方法 (Kernel methods)

基于表示定理能得到很多线性模型的“核化”(kernelized)版本

例如 KLDA (Kernelized LDA):

将样本映射到高维特征空间，然后在此特征空间中做线性判别分析

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = \frac{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_b^{\phi} \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_w^{\phi} \boldsymbol{w}}$$

$$h(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i)$$

$$\max_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^{\mathrm{T}} \mathbf{M} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^{\mathrm{T}} \mathbf{N} \boldsymbol{\alpha}}$$

“核技巧” (kernel trick)
是机器学习中处理非线性问题的基本技术之一

SVM 与统计学习简史

V. Vapnik (1936-)

1963：Vapnik 提出支持向量的概念

1968：Vapnik 和 Chervonenkis 提出 VC 维

1974：提出结构风险最小化原则

… …　苏联解体前一年(1990), Vapnik 来到美国

1995：Support Vector Network 文章发表

1995: 《The Nature of Statistical Learning》出版

1998: SVM 在文本分类上取得巨大成功

1998: 《Statistical Learning Theory》出版

… …

“ Nothing is more practical than a good theory ”

-- V. Vapnik

前往第六站……

五、神经网络

主讲教师：周志华

什么是神经网络？

“神经网络是由具有适应性的简单单元组成的广泛并行互连的网络，它的组织能够模拟生物神经系统对真实世界物体所作出的交互反应”

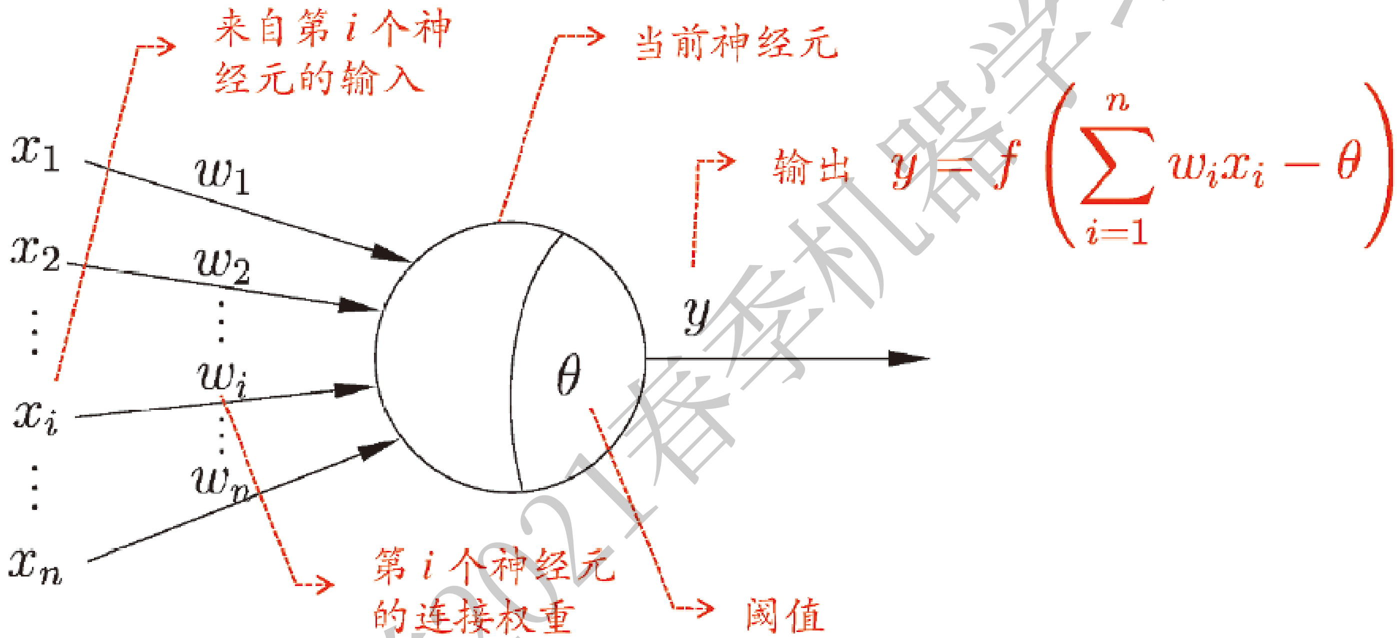
[T. Kohonen, 1988, *Neural Networks* 创刊号]

神经网络是一个很大的学科领域，本课程仅讨论神经网络与机器学习的交集，即“神经网络学习”

亦称“连接主义(connectionism)” 学习

“简单单元”：神经元模型

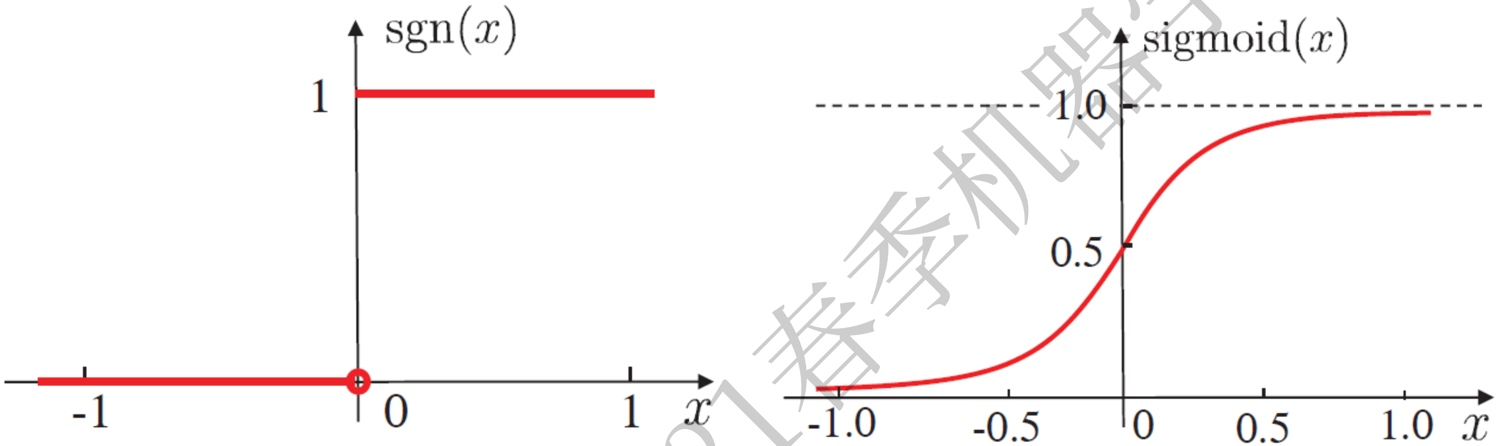
M-P 神经元模型 [McCulloch and Pitts，1943]



神经网络学得的知识蕴含在连接权与阈值中

神经元的“激活函数”

- 理想激活函数是阶跃函数，0表示抑制神经元而1表示激活神经元
- 阶跃函数具有不连续、不光滑等不好的性质，常用的是 Sigmoid 函数



$$\mathrm{sgn}(x) = \begin{cases} 1, & \text{if } x \geq 0; \\ 0, & \text{if } x < 0. \end{cases}$$

$$\mathrm{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

(a) 阶跃函数　　(b) Sigmoid 函数

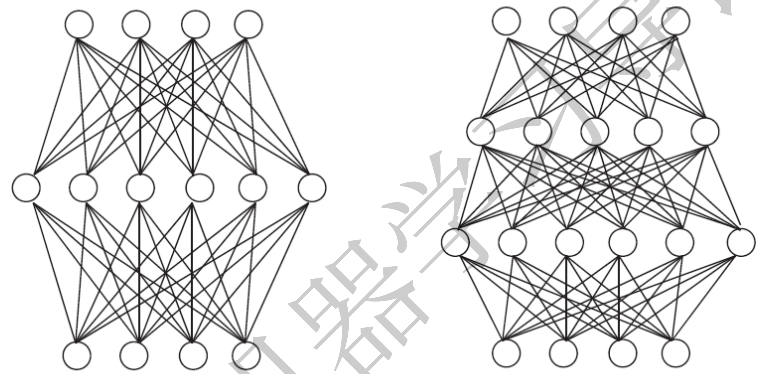
图 5.2　典型的神经元激活函数

多层前馈网络结构

多层网络：包含隐层的网络

前馈网络：神经元之间不存在同层连接也不存在跨层连接

隐层和输出层神经元亦称“功能单元”(functional unit)



(a) 单隐层前馈网络　　(b) 双隐层前馈网络

多层前馈网络有强大的表示能力（”万有逼近性”）

> 仅需一个包含足够多神经元的隐层，多层前馈神经网络就能以任意精度逼近任意复杂度的连续函数　　[Hornik et al., 1989]

但是，如何设置隐层神经元数是未决问题(Open Problem)．实际常用“试错法”