Dong Wang

现代机器学习技术导论

2018 年 3 月 20 日

Springer

Contents

Chapter 1
机器学习概述

Chapter 2
线性模型

Chapter 3
神经模型

Chapter 4
深度学习

Chapter 5
核方法

在第 2章我们讨论过线性回归模型$\mathbf{t} = \mathbf{W}\mathbf{x} + \boldsymbol{\varepsilon}$。如果$\mathbf{x}$和$\mathbf{t}$间有明显的线性关系，则该模型可取得较好效果；如果两者之间的线性关系不显著，则线性模型会出现较大偏差。同样，对于线性分类模型$y = \sigma(\mathbf{w}^T\mathbf{x})$，如果$\mathbf{x}$是线性可分的，则该模型可得到较好的分类效果，但当其线性不可分时，该模型将不再适用。一种方法是对$\mathbf{x}$做非线性映射$\boldsymbol{\phi}(\mathbf{x})$，如果$\boldsymbol{\phi}(\mathbf{x})$和$t$存在线性关系（回归任务）或线性可分（分类任务），则可在映射空间建立线性回归或分类模型，实现复杂数据的线性建模。如图 5.1所示，在原始二维空间中的两类点是线性不可分的，但当用一个非线性映射将数据投影到三维空间后，则可实现较好的线性分类。

然而，设计一个合理的映射$\boldsymbol{\phi}$并不容易，特别是当我们对任务本身的知识相对有限时。在第 3章中，我们通过一个参数化的神经网络来学习这一映射，这一方法也称为特征学习。这种基于学习的特映设计方法可以避免人为设计的困难，得到与目标任务相适应的映射函数（或特征）。然而，该方法存在几个缺点。首先，特征学习需要对原始数据有很明确的向量表达，但在很多实际应用中很难将每个对象表达成数值向量，如一个社交网络中的每个结点对应的成员，这些成员之间的关系是明确的，但要将每个成员单独表达成一个向量则比较困难。现实中这种关系明确，表达困难的任务有很多。对这类任务，特征学习存在较大困难[1]。第二，特征学习方法对特征空间的大小有严格限制，特征空间维度过高会导致学习困难，但一些复杂数据必须在较高维的特征空间上才能表现出线性。第三，特征学习，特别是基于复杂函数（如DNN）的特征学习是一个非凸问题，训练存在很大困难，容易发生过拟合或欠拟合。

[1] 最近兴起的嵌入化（Embedding）技术部分解决了对关系型问题的数据向量化问题，但这些方法的向量化都是近似的，基于这一向量建模会带来一定偏差。

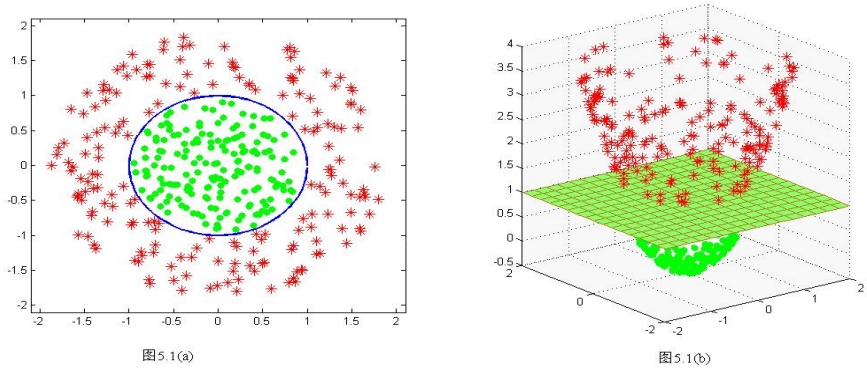

Fig. 5.1 利用非线性映射将二维空间中线性不可的数据（左图）映射到三维空间后变得线性可分（右图）。本例中，非线性变换取如下形式：$\phi(x_1,x_2)=(z_1,z_2,z_3)=(x_1,x_2,x_1^2+x_2^2)$。经过该变换后，平面$z_3=1$即可在映射空间中将数据线性分开。

核方法（Kernel Method）是另一种映射函数的生成方法。与特征学习不同，核方法不对特征映射函数$\boldsymbol{\phi}$做显示的表示或学习，而是通过数据间的相关性函数$k(\mathbf{x},\mathbf{x}')$对$\boldsymbol{\phi}$进行隐式定义，即：

$$k(\mathbf{x},\mathbf{x}')=\boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}').$$

相关性函数$k(\mathbf{x},\mathbf{x}')$称为核函数。研究表明，任何一个对称半正定的函数都是核函数，一个核函数隐式定义了一个映射函数$\boldsymbol{\phi}(\mathbf{x})$[2]。通过这一定义，我们可以在$\boldsymbol{\phi}$空间中完成拟合或分类任务，而且完成这一任务并不需要$\boldsymbol{\phi}$的显式表达，只需通过计算核函数$k(\mathbf{x},\mathbf{x}')$即可。这一方法具有若干明显优势：首先该方法只关注数据之间的关系，而不是数据本身，因此特别适合数据样本难以用向量明确表达的任务 [10]。其次，由$k(\mathbf{x},\mathbf{x}')$引导出来的特征空间$\boldsymbol{\phi}$可能具有非常高的维度，甚至是无限维，可满足对复杂数据分布的线性化要求。最后，特征空间中的模型是线性的，因此模型训练是一个凸优化问题，可保证得到全局最优解。

人们研究核方法已经有很长时间的历史了。著名的Mercer定理可以追溯到1909 年，再生核希尔伯特空间的研究早在20世纪40年代就开始了。1964年Aizermann等人在对势函数的研究中首次将核方法引入到机器学习领域，但并没有引起太大反响。1992年，Boser等人在研究最大边界分类器时，

[2] 严格来说，一个核函数可以对应多个映射函数。

将核方法和最大边界分类准则结合在一起，将线性支持向量机（SVM）推广到非线性支持向量机。自此，核方法的优势才被研究界充分认识和挖掘。近年来，核方法的一个重要发展是对核函数的扩展，使其得以处理更多实际问题，例如符号化的物体，复杂的序列或结构等，从而极大扩展了核方法的应用范围。

本章将对核方法进行介绍。我们将从简单的线性回归任务出发，引出其对偶表达。对偶表达是将参数模型转换成非参数模型的重要步骤，是核方法的基础。基于此，我们将给出核函数的概念，并给出Mercer定理，这一定理告诉我们如何构造一个合法的核函数，使之得以对数据进行有效映射。之后，我们将讨论一些有代表性的核函数，特别是一些复杂结构上的核函数，如集合、序列、图上的核函数。之后，我们将讨论四类具体核方法：Kernel PCA, 高斯过程（Gaussian Process），支持向量机（SVM），相关向量机（RVM）。

5.1 从线性回归到核方法

第 2章讨论了线性回归模型，该模型可定义如下：

$$y(\mathbf{x};\mathbf{w}) = \boldsymbol{\phi}(\mathbf{x})^T\mathbf{w}, \tag{5.1}$$

其中$\boldsymbol{\phi}$是一个确定的特征映射函数。给定一组训练数据$\{(\mathbf{x}_n,t_n) : n = 1,2,...,N\}$，该回归模型可写成如下矩阵形式：

$$\mathbf{y} = \boldsymbol{\Phi}^T\mathbf{w}, \tag{5.2}$$

其中，$\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1),\cdots,\boldsymbol{\phi}(\mathbf{x}_N)]$。定义回归模型的目标函数为：

$$L(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - y_n\}^2 = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2.$$

对$L(\mathbf{w})$取$\mathbf{w}$的梯度，有：

$$\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = \sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}\boldsymbol{\phi}(\mathbf{x}_n). \tag{5.3}$$

取上述梯度为零，整理可得:

$$\mathbf{w} = (\boldsymbol{\Phi}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{\Phi}\mathbf{t}, \tag{5.4}$$

其中，$\mathbf{t} = [t_1, \cdots, t_N]^T$是目标变量的观察值。

我们可以选择另一种方法对上述回归模型求解。将参数$\mathbf{w}$表示为训练集中所有样本的加权平均：

$$\mathbf{w} = \boldsymbol{\Phi}\boldsymbol{\alpha} \tag{5.5}$$

其中$\boldsymbol{\alpha} \in R^N$ 是每个样本的权重。如果我们求得了$\boldsymbol{\alpha}$，即可解得$\mathbf{w}$。因而，回归任务可写成如下形式：

$$\mathbf{y} = \boldsymbol{\Phi}^T\mathbf{w} = \boldsymbol{\Phi}^T\boldsymbol{\Phi}\alpha = \mathbf{K}\boldsymbol{\alpha} \tag{5.6}$$

其中$\mathbf{K} \in R^{N\times N}$ 定义了训练集中任意一对数据样本之间的内积，称为Gram矩阵，其元素k_{ij}定义为：

$$k_{ij} = \boldsymbol{\phi}(\mathbf{x}_i)^T\boldsymbol{\phi}(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j). \tag{5.7}$$

上式与式（5.2）具有类似的形式，只不过以参数$\boldsymbol{\alpha}$替换了参数$\mathbf{w}$，因此具有类似式（5.4）形式的解。注意$\mathbf{K}$是对称矩阵，故而有：

$$\boldsymbol{\alpha} = (\mathbf{K}\mathbf{K})^{-1}\mathbf{K}\mathbf{t} = \mathbf{K}^{-1}\mathbf{t}. \tag{5.8}$$

如果将式（5.2）称为原始问题（Primary Problem），则式（5.6）给出该问题的另一种表达方式，通常称为对偶问题（Dual Problem）。将原始问题变换成对偶问题是机器学习中的常见作法。通过这一变换，或简化问题的表达和求解，或寻求问题的另一种意义。后面讨论支持向量机（SVM）时我们会进一步看到对偶问题的价值。

将式（5.8）和式（5.5）代入回归模型（5.1），可得到对任一测试样本$\mathbf{x}$的预测：

$$y(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^T\mathbf{w} = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\Phi}\boldsymbol{\alpha} = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\Phi}\mathbf{K}^{-1}\mathbf{t} = \mathbf{k}(\mathbf{x})^T\mathbf{K}^{-1}\mathbf{t}. \tag{5.9}$$

其中,

$$\mathbf{k}(\mathbf{x}) = \boldsymbol{\Phi}^T\boldsymbol{\phi}(\mathbf{x}) = [\boldsymbol{\phi}(\mathbf{x}_1)^T\boldsymbol{\phi}(\mathbf{x}), \cdots, \boldsymbol{\phi}(\mathbf{x}_N)^T\boldsymbol{\phi}(\mathbf{x})]^T = [k(\mathbf{x}_1, \mathbf{x}), \cdots, k(\mathbf{x}_N, \mathbf{x})]^T.$$

仔细观察式（5.9），可见线性回归模型存在另一种截然不同的解法。在这一解法中，我们并不需要显式地求出模型参数$\mathbf{w}$，也不需要明确定义特征映射函数$\boldsymbol{\phi}$，只需知道训练数据之间的关系$\mathbf{K}$和测试数据与训练数据的关系$\mathbf{k}(\mathbf{x})$。不论是$\mathbf{K}$还是$\mathbf{k}(\mathbf{x})$，都基于式 5.7所定义的关系函数$k(\cdot,\cdot)$。该函数称为*核函数*（Kernel Function），相应的方法称为*核方法*(Kernel Method)。

不论是$\mathbf{K}$还是$\mathbf{k}(\mathbf{x})$，都包含对训练集中所有N个数据的计算。当N远大于特征的维度时，核方法在计算量和内存开销上都大于原始参数方法。然而，这一方法提供了一种全新的学习思路：在这种学习中，用训练数据集合代替参数模型，用数据间的关系代替数据本身，前者使得该方法是一种非参数方法，后者使得该方法具有相似性学习的特征。事实上这两者是相关的，因为有数据间的关系，才能在预测时依赖训练数据而非参数模型。后面我们会看到，描述数据关系的核函数$k(\cdot,\cdot)$具有重要意义，它事实上隐性定义了映射函数$\boldsymbol{\phi}(\cdot)$，而这一定义的复杂度可能远超过人为定义可能达到的复杂度。

5.2 核函数的性质

5.2.1 再生核希尔伯特空间与Mercer定理

核函数定义为在映射空间中的内积，即：

$$k(x,x') = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}'). \tag{5.10}$$

由上式可知，$k(x,x')$显然是对称的。同时，$k(x,x')$具有半正定性，即对任何一个由$k(x,x')$定义的Gram矩阵$\mathbf{K} \in R^{N\times N}$，取任意一向量$\mathbf{c} \in R^N$，都有如下性质：

$$\begin{aligned}
\mathbf{c}^T\mathbf{K}\mathbf{c} &= \sum_{i,j=1}^{N} c_i c_j K_{ij} \\
&= \sum_{i,j=1}^{N} \langle c_i\boldsymbol{\phi}(\mathbf{x}_i), c_j\boldsymbol{\phi}(\mathbf{x}_j)\rangle \\
&= \left\langle \sum_{i=1}^{N} c_i\boldsymbol{\phi}(\mathbf{x}_i), \sum_{j=1}^{N} c_j\boldsymbol{\phi}(\mathbf{x}_j) \right\rangle \\
&= \| \sum_{i=1}^{N} c_i\boldsymbol{\phi}(\mathbf{x}_i)\|^2 \geq 0.
\end{aligned} \tag{5.11}$$

上式说明任意一个特征变换$\boldsymbol{\phi}(\mathbf{x})$都定义了一个对称半正定的核函数。反过来，如果给定一个对称半正定的二元函数$k(\cdot,\cdot)$，是否可以定义一个特征变换$\boldsymbol{\phi}(\mathbf{x})$，使得式（5.10）得以满足？如果能找到这样一个$\boldsymbol{\phi}(\mathbf{x})$，则$k(\cdot,\cdot)$一定是一个核函数。答案是肯定的：我们不仅可以找到这样的$\boldsymbol{\phi}(\mathbf{x})$，而且可能会找到多个。

这涉及到再生核希尔伯特空间（Reproducing Kernel Hilbert Space, RKHS）的概念。简单地说，希尔伯特空间是指一个完备的内积空间，可认为是欧式空间的扩展。再生希尔伯特空间是指一个实值函数的希尔伯特空间$\mathscr{H}$，对其中的任一个函数$f \in \mathscr{H}$，都可以通过如下方式生成：

$$f(\mathbf{x}) =< f, K_{\mathbf{x}} >$$

其中$\mathbf{x} \in \mathscr{X}$是定义域$\mathscr{X}$中的任一取值，$K_{\mathbf{x}}$是被$\mathbf{x}$定义的希尔伯特空间$\mathscr{H}$中的一个函数。对定义域中的另一取值$\mathbf{x}'$所定义的函数$K_{\mathbf{x}'}$，同样可由上述方式生成。用$K_{\mathbf{x}'}$代替上式中的$f$，则有：

$$K_{\mathbf{x}'}(\mathbf{x}) =< K_{\mathbf{x}'}, K_{\mathbf{x}} > .$$

依内积的对称性，显然有$K_{\mathbf{x}'}(\mathbf{x}) = K_{\mathbf{x}}(\mathbf{x}')$。由此，一个RKHS中的所有函数都可由如下核函数生成:

$$k(\mathbf{x}, \mathbf{x}') =< K_{\mathbf{x}}, K_{\mathbf{x}'} >,$$

这也是再生核希尔伯特空间这一名称的由来。另一方面，Moore－Aronszajn定理 [2]表明，任何一个对称半正定的二元函数对应唯一一个RKHS。

给定一个对称半正定的二元函数$k(\cdot,\cdot)$，我们至少可取其对应的RKHS作为映射空间，即：

$$\boldsymbol{\phi}(\mathbf{x}) = k(\mathbf{x}, \cdot) = K_{\mathbf{x}}(\cdot).$$

则有：

$$k(\mathbf{x}, \mathbf{x}') =< K_{\mathbf{x}}, K_{\mathbf{x}'} >=< \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}') > .$$

上式说明$k(\mathbf{x}, \mathbf{x}')$是一个有效核函数。注意，映射$\boldsymbol{\phi}(\mathbf{x})$将原始空间$\mathbf{x}$映射到了一个函数空间$\mathscr{H}$，而$\mathscr{H}$中的函数通常可看作是无限维向量，这相当于通过核函数$k(\cdot,\cdot)$将原始数据映射到了一个无限维空间中。

值得一提的是，$K_{\mathbf{x}}$并不是$k(\cdot,\cdot)$对应的唯一映射，可能有多个$\boldsymbol{\phi}(\mathbf{x})$对应同一个核函数。例如核函数$k(\mathbf{x},\mathbf{x}') =< \mathbf{x},\mathbf{x}' >^2$，其中$\mathbf{x} = (x_1,x_2)$。可以证明如下函数都是$k(\mathbf{x},\mathbf{x}')$对应的映射函数。

$$\boldsymbol{\phi}_1(\mathbf{x}) = [{x_1}^2, \sqrt{2}x_1x_2, {x_2}^2],$$

$$\boldsymbol{\phi}_2(\mathbf{x}) = \frac{1}{\sqrt{2}}[{x_1}^2 - {x_2}^2, 2x_1x_2, {x_1}^2 + {x_2}^2],$$

$$\boldsymbol{\phi}_3(\mathbf{x}) = [{x_1}^2, x_1x_2, x_1x_2, {x_2}^2].$$

综上所述，我们看到一个函数$k(\mathbf{x},\mathbf{x}')$是合法核函数的充分必要条件是该函数是对称且半正定的；或者，对于任意N个$\{\mathbf{x}_n; n = 1,..,N\}$，由$k(\cdot,\cdot)$导出的Gram矩阵是对称半正定矩阵。这一结论称为Mercer 定理，发表于1909年[21]。在构建核函数时，我们可以通过Mercer定理判断一个核函数是否合法，这一点在构造复杂核函数时非常有用。

5.2.2 核函数的基本性质

设$k_1(\cdot,\cdot)$，$k_2(\cdot,\cdot)$是合法的核函数，α 是一个非负数，$f(\cdot)$是任意一个函数，ϕ是从X 到R^N的映射，$k_3(\cdot,\cdot)$是定义在$R_N \times R_N$上的核函数，B是一个对称半正定矩阵。可以证明通过如下操作生成的函数都是合法的核函数：

$$k(\mathbf{x},\mathbf{x}') = k_1(\mathbf{x},\mathbf{x}') + k_2(\mathbf{x},\mathbf{x}') \tag{5.12}$$

$$k(\mathbf{x},\mathbf{x}') = \alpha k_1(\mathbf{x},\mathbf{x}') \tag{5.13}$$

$$k(\mathbf{x},\mathbf{x}') = k_1(\mathbf{x},\mathbf{x}')k_2(\mathbf{x},\mathbf{x}') \tag{5.14}$$

$$k(\mathbf{x},\mathbf{x}') = f(\mathbf{x})f(\mathbf{x}') \tag{5.15}$$

$$k(\mathbf{x},\mathbf{x}') = k_3(\phi(\mathbf{x}),\phi(\mathbf{x}')) \tag{5.16}$$

$$k(\mathbf{x},\mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x},\mathbf{x}')f(\mathbf{x}') \tag{5.17}$$

$$k(\mathbf{x},\mathbf{x}') = \mathbf{x}'\mathbf{B}\mathbf{x}' \tag{5.18}$$

此外，通过以下方式生成的核函数也是合法的：

$$k(\mathbf{x},\mathbf{x}') = \exp(k_1(\mathbf{x},\mathbf{x}')) \tag{5.19}$$

$$k(\mathbf{x},\mathbf{x}') = P(k_1(\mathbf{x},\mathbf{x}')) \tag{5.20}$$

$$k(\mathbf{x},\mathbf{x}') = \exp\left(\frac{-||\mathbf{x}-\mathbf{x}'||^2}{2\sigma^2}\right). \tag{5.21}$$

其中，$k_1(\mathbf{x},\mathbf{x}')$是一个合法的核函数，$P(\mathbf{x})$是一个具有正系数的多项式，$\sigma$是一个任意常数。通过核函数的这些基本性质，我们可以从简单核函数生成复杂核函数，这比直接构造复杂核函数要容易的多。

5.3 常用核函数

由前面讨论可知，核函数的形式决定了映射函数的属性，不同的核函数将数据映射到不同的特征空间。在解决实际问题时，当然希望数据在特征空间的性质越简单越好（如线性可预测性，线性可区分性、高斯分布等），因此对不同任务需要设计不同的核函数。本章首先介绍一些常用的核函数，之后介绍复杂核函数的构造方法。关于核函数更详细的说明，可参考文献[12]。

5.3.1 简单核函数

线性核 线性核是最简单的核，定义如下：

$$k(\mathbf{x},\mathbf{x}') = \mathbf{x}\cdot\mathbf{x}' + c.$$

线性核对应等值映射，不能提高表示性，但在很多实际问题中可取得较好的效果。注意核方法不仅包含特征映射，同时包含非参数建模，因此，即使是线性核在很多实用场景中也好于传统参数方法，典型的如线性SVM。同时，线性核可以用来验证算法的正确性。例如选择线性核的kernel PCA (KPCA）等价于传统PCA，因此可以用传统PCA结果来验证Kernal PCA实现是否正确。

多项式核 根据上一节所述的核函数性质可知，如果$k_1(\mathbf{x},\mathbf{x}')$是一个核函数，那么在该核基础上衍生的多项式扩展$k(\mathbf{x},\mathbf{x}') = P(k_1(\mathbf{x},\mathbf{x}'))$同样是一个合法核函数，其中$P(\cdot)$是任意一个具有正系数的多项式。设$k(\mathbf{x},\mathbf{x}') = (k_1(\mathbf{x},\mathbf{x}')+c)^d$，且$k_1(\mathbf{x},\mathbf{x}') = \mathbf{x}\cdot\mathbf{x}'$，则得到多项式核函数如下：

$$k(\mathbf{x},\mathbf{x}') = (\alpha\mathbf{x}\cdot\mathbf{x}' + c)^d \quad \alpha > 0, c \geq 0, d \in Z_+.$$

直观上看，多项式核等价于对原始数据进特征扩展，不仅考虑原始特征，同时考虑不同特征之间的相关性。多项式核在自然语言处理任务中有广泛应用[13]。多项式核的一个缺点是当阶数d比较大时，在取值容易出现数值上的不稳定，可能出现过大或过小值。

高斯核 高斯核是应用最广泛的核函数，其式如下：

$$k(\mathbf{x},\mathbf{x}') = exp(-\alpha||\mathbf{x}-\mathbf{x}'||^2),$$

其中α是控制核函数宽度的参数。上式与高斯分布具有类似形式，故而称为高斯核。高斯核对应无限维特征空间。高斯核是距离的函数，具有位置不变性。由第 3章可知，这一形式与径向基函数（RBF）一致，因此高斯核也常称为RBF核。

高斯核有多种扩展形式。例如在一个特征空间$\boldsymbol{\phi}(\mathbf{x})$中计算该核函数，则有：

$$||\boldsymbol{\phi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x}')||^2 = k_1(\mathbf{x},\mathbf{x}) - 2k_1(\mathbf{x},\mathbf{x}') + k_1(\mathbf{x}',\mathbf{x}'),$$

其中

$$k_1(\mathbf{x},\mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}').$$

因此，该特征空间中的高斯核可以表示为如下原始空间中的核函数形式：

$$k(\mathbf{x},\mathbf{x}') = exp(-\alpha(k_1(\mathbf{x},\mathbf{x}) - 2k_1(\mathbf{x},\mathbf{x}') + k_1(\mathbf{x}',\mathbf{x}'))).$$

与高斯核具有类似形式的是指数核（有时也称为拉普拉斯核），具有如下形式：

$$k(\mathbf{x},\mathbf{x}') = exp(-\alpha||\mathbf{x}-\mathbf{x}'||_1).$$

和高斯核相比，指数核具有更明显的长尾效应，可用于描述较大范围内的相关性。

概率核 上述核函数计算数据样本点之间的距离，不具有概率意义，对噪音比较敏感。一种可能的处理方法是设计一个概率生成模型$p(\mathbf{x})$，并定义如下核函数：

$$k(\mathbf{x},\mathbf{x}') = p(\mathbf{x})p(\mathbf{x}'). \tag{5.22}$$

上述定义意味着当两个输入$\mathbf{x}$和$\mathbf{x}'$都具有较大的概率时，则两者是相似的，这在很多任务中是合理的。对上述核函数可进行扩展，用混合分布描述更复杂的相似性：

$$k(\mathbf{x},\mathbf{x}') = \sum_{z_i} p(\mathbf{x}|z_i)p(\mathbf{x}'|z_i)p(z_i),$$

其中z_i是一个隐变量，$p(z_i) > 0$是z_i的先验概率。当隐变量是连续值时，上式可以写成

$$k(\mathbf{x},\mathbf{x}') = \int p(\mathbf{x}|z)p(x'|z)\mathrm{d}z,$$

其中z是连续隐变量。

另一种基于概率模型的核函数是Fisher核，由Jaakkola 和Haussler 于1999年定义 [16]。设以$\boldsymbol{\theta}$为参数的概率生成模型$p(\mathbf{x}|\boldsymbol{\theta})$，对任一个$\mathbf{x}$，考虑在该点处关于$\boldsymbol{\theta}$ 的梯度向量如下：

$$g(\mathbf{x};\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \ln\ p(\mathbf{x}|\boldsymbol{\theta}). \tag{5.23}$$

$g(\mathbf{x};\boldsymbol{\theta})$ 通常称为Fisher Score，其维度与$\boldsymbol{\theta}$维度一致。Fisher 核定义为Fisher Score的内积：

$$k(\mathbf{x},\mathbf{x}';\boldsymbol{\theta}) = g(\mathbf{x};\boldsymbol{\theta})^T \mathbf{I}(\boldsymbol{\theta})^{-1} g(\mathbf{x}';\boldsymbol{\theta}),$$

其中$\mathbf{I}(\boldsymbol{\theta})$为Fisher信息矩阵，定义为：

$$\mathbf{I}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}}[g(\mathbf{x};\boldsymbol{\theta})g(\mathbf{x};\boldsymbol{\theta})^T]. \tag{5.24}$$

直观上，Fisher信息矩阵定义了一个Riemannian流形（非线性空间），在该流形上$\boldsymbol{\theta}$处的距离定义为：

$$D(\boldsymbol{\theta},\boldsymbol{\theta}+\boldsymbol{\delta}) = \frac{1}{2}\boldsymbol{\delta}^T \mathbf{I}\boldsymbol{\delta}.$$

仔细观察Fisher核，可以发现它事实上定义了一个$\mathbf{x}$到某一梯度空间的映射，该梯度是$\boldsymbol{\theta}$沿Riemannian流形变动时，$p(\mathbf{x}|\boldsymbol{\theta})$取最大值的方向。这一方向称为自然梯度，在第4章中讨论自然梯度下降算法时提到过。Fisher核假设当$\mathbf{x}$与$\mathbf{x}'$接近时，其自然梯度是接近的，即要求对$\boldsymbol{\theta}$进行相似的调整使得$p(\mathbf{x}|\boldsymbol{\theta})$和$p(\mathbf{x}|\boldsymbol{\theta})$都实现最大化。这一基于梯度空间的距离计算方法具有一定抗噪能力。例如在图象识别中，如果两幅图片很近似，但其中一幅被噪声干扰，基于原始空间的距离计算方法通常会产生误判；如果基于梯度计算，

并假设$p(\mathbf{x}|\boldsymbol{\theta})$对图片内容进行建模，则任意调整模型参数$\boldsymbol{\theta}$通常不能增加带噪图片的概率（因为内容没有改变），因此两幅图片在梯度空间中依然具有相似性。

核函数的组合 前面所述的核函数定义简单，计算方便，应用范围广泛，在各种应用问题中应首先予以考虑。更多简单核函数可参见相关文献[3]。基于上述简单核，可利用上节讨论的核函数的性质组合生成更复杂的核。如果只考虑乘法和加法，则对核函数的组合可表示为一幅有向图G，图中每一条边e_i代表一个基础核函数k_i，边与边的连接代表核函数相乘，则图中每条路径代表一个基于乘法生成的组合核函数:

$$k(\xi) = \prod_{e_i \in \xi} k_i.$$

图中所有路径代表的核函数相加之和为该图所代表的核函数$k(G)$:

$$k(G) = \sum_{\xi;\xi \in G} k(\xi).$$

5.3.2 复杂核函数

上一节所述的简单核函数对于简单数据对象很适用，这些对象可表达为维度不高的向量，可用简单核函数或其组合来描述对象间的距离。然而，当对象比较复杂，有较强的结构特性时，这些简单核函数就不太适用了。例如集合、序列、图结构等，对这些对象进行简单向量化往往需要很高维度，而且各维之间具有很强相关性，用简单核函数对这些数据进行建模往往不能取得很好的效果。一种方法是利用数据的结构化特性，设计与数据相匹配的核函数。

集合上的核 给定一个集合$\mathscr{S}$，这个集合的所有可能的子集构成一个集合空间$2^{\mathscr{S}}$。如果A和A'是这个空间上的两个元素（即$\mathscr{S}$的两个子集），那么一个简单的核可以定义为：

$$k(A,A') = |A \bigcap A'|,$$

[3] http://crsouza.com/2010/03/17/kernel-functions-for-machine-learning-applications/

其中$A \bigcap A'$表示集合A和A'的交集，$|A|$表示集合A 的元素的数量。另一种常见方法是将集合A表示为概率分布$P(A)$，将集合间的距离转化为概率间的距离：

$$k(A,A') = D(P(A),P(A')),$$

其中$D(\cdot,\cdot)$是两个概率分布之间的距离。常用的距离包括Histogram intersection，Kullback－Leibler散度，Itakura－Saito 距离，Hellinger 距离，Chi-squared 统计量，Kolmogorov-Smirnov 距离，Jensen－Shannon散度，Earth movert距离等 [30]。在原始数据空间的概率分布距离度量可能存在较强的噪声，如在图象处理中的光照、角度等带来的影响。一种方法是将原始数据点映射到一个特征空间H，再通过该特征空间上的概率分布来计算集合间的距离。这里的特征映射可以表示为一个核函数$k(\mathbf{x},\mathbf{x}')$。注意$k(\mathbf{x},\mathbf{x}')$和$k(A,A')$是两个不同的核，分别用于对数据进行特征映射和对集合做距离计算 [19]。

序列上的核 如果数据对象$\mathbf{x}$是一个序列，如文本串、DNA等，则距离计算更加复杂。因为不同序列的长度不同，序列与序列之间可能有复杂的包含关系，这为设计序列核带来一定困难。

一种简单的处理方法是忽略序列中元素的顺序，这时一个序列退化为一个集合，序列上的核函数退化为集合上的核函数。文本处理中常用的词袋模型（Bag of Word Model）即是这种方法。另一种方法是忽略全局顺序，只考虑局部顺序。例如在文本处理中的N-gram词袋模型，词袋中的元素是局部词序列（即N-gram）。这种方法可兼顾计算复杂性与序列的时序性，在实践中被广泛应用。

如果考虑全局顺序，则可用各种序列对齐方法计算序列间的距离。具体来说，给定两个序列$\mathbf{x}$和$\mathbf{x}'$, 其元素分别用$x(i)$和$x'(j)$表示。如果任意一对元素x_i和x'_j之间的距离可计算，则总可搜索得到一种将$\mathbf{x}$和$\mathbf{x}'$对齐的方式ξ，使得依此方式得到的元素间距离的和最小，形式化为：

$$k(\mathbf{x},\mathbf{x}') = \arg\min_{\xi} \sum_{i} \sum_{j \in \xi(i)} d(x_i,x'_j),$$

其中$d(a,b)$为两个元素a和b间的距离，$\xi(i)$表示依ξ，序列$\mathbf{x}$中的元素$x(i)$对应的序列$\mathbf{x}'$中的元素序号。这一优化任务可用动态规划方法求解。文本处理中常用的编辑距离即是这一方法的特例，其中$d(a,b)$当且仅当$a=b$时为0，否则为1。这种序列对齐方法得到的距离并不能保证$k(\mathbf{x},\mathbf{x}')$是一个合法的核函数。

另一种考虑全局顺序的方法是将序列表示成一个时序概率模型，进而将序列间的距离度量转变为概率模型间的距离。隐马尔可夫模型（HMM）是最常见也是最简单的时序模型。设观测序列$\mathbf{x} = [x_1, \cdots, x_T]$，隐含状态序列是$\mathbf{z} = [z_1, \cdots, z_T]$，其中$z_t$是$t$时刻的状态，取值为某一离散状态空间。状态序列的先验概率表示为$p(\mathbf{z})$，则$\mathbf{x}$的概率表示为：

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z}).$$

基于该模型可对两个观测序列$\mathbf{x}$，$\mathbf{x}'$计算相似性。一种简单计算方法是假设任一状态序列$\mathbf{z}$，该状态序列可以独立产生两个观测序列$\mathbf{x}$和$\mathbf{x}'$。依基于概率的核函数公式（5.22）计算在该序列假设下两个序列的概率距离，再对所有可能的状态序列求和，有：

$$k(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{x}'|\mathbf{z})p(\mathbf{z}).$$

上式要求$\mathbf{x}$和$\mathbf{x}'$是等长序列（与$\mathbf{z}$的长度相等），稍加扩展后，可以用来计算不等长序列的距离。设$\mathbf{z}$中的任一状态$z(t)$ 可生成两个独立元素$(x(t), x'(t))$，其中每个元素既可以是观测序列中的元素，也可以是一个特别指定的空元素。这相当于对$\mathbf{x}$和$\mathbf{x}'$ 进行对齐，再对所有可能的对齐方式进行求和。研究表明，这种距离计算方式是一个合法核函数，称为动态对齐核（Dynamic Alignment Kernel），对应的HMM模型称为Pair HMM [33, 26]。

利用HMM模型对序列的建模能力，可对序列数据定义Fisher核 [16, 32]。在Fisher核函数中，我们首先建立一个HMM模型，并基于此得到在观测数据$\mathbf{x}$处的Fisher Score $g(\mathbf{x}) = \nabla_\theta \ln p(\mathbf{x}; \boldsymbol{\theta})$，其中$p(\mathbf{x}; \boldsymbol{\theta})$即为预先建立的HMM模型。依Fisher核定义，有：

$$k(\mathbf{x}, \mathbf{x}') = g(\mathbf{x})^T \mathbf{I}^{-1} g(\mathbf{x}')$$

其中$\mathbf{I}$ 为Fisher信息矩阵。

图上的核函数 图是比序列更复杂的结构，如社交网络中每个人组成的网络，科学文献中引用关系组成的网络。这些网络具有各异结构，如何定义两幅图之间的相似性具有相当挑战性。研究者提出了一些解决方法，一个基本思路是同时在两幅图上进行随机游走，计算生成路径的相似性，并对可能的路径进行相似性求和 [17]。

具体来说，一个无向图G可由节点集合V和边集合E表示，如果V中的两个节点v_j和v_k相邻，则二者被E中的一条边e_{jk}相连，记为$j \sim k$。设图G中节

点间的跳转概率由矩阵P定义，其中P_{jk}表示由节点j到节点k的跳转概率。当在G上进行t步随机游走时，经过节点的下标序列为$i_1, i_2, ..., i_{t+1}$，对应边上的标记序列为$h_1, h_2, .., h_t$。再设节点的初始概率为p，结束概率为q，则路径h 的概率为:

$$p(h|G) = q_{i_{t+1}} \prod_{j=1}^{t} P_{i_j, i_{j+1}} p_{i_1}.$$

如果两个标记序列h和h'长度不等，则二者相似性为零；若长度同为t，其相似性由定义在标记间的核函数$k(h_i, h'_i)$确定如下：

$$k(h, h') = \prod_{i=1}^{t} k(h_i, h'_i).$$

则两幅图G和G'间的距离可由下式确定：

$$k(G, G') = \sum_{h} \sum_{h'} k(h, h') p(h|G) p(h'|G').$$

Gartner[11]等提出类似的思路，不同的是以随机游走时匹配的路径数作为距离度量标准。

另一种常见的图结构是有限状态机（Automaton）。例如一个语音识别系统的输出通常是一组不等长的N-best识别结果，这些结果可以表示为一个有限状态机。计算有限状态机之间的距离通常用到Rational核 [8]。Vishwanathan [31]等证明，图上基于随机游走的核函数和有限状态机上的Rational 核具有密切联系，后者可以认为是随机游走核函数在有限状态机上的扩展。

不论是随机游走核还是Rational核都需要大量计算。Vishwanathan对各种图核函数进行了仔细研究，提出一个基于矩阵Kronecker乘法的统一框架，并给出一系列快速计算方法，可使图核计算量由$O(n^6)$下降到$O(n^3)$，其中n为图中节点个数 [31]。

基于局部距离度量的核函数 上述各种核函数在距离表达上都是完全的，即目标间的度量是确定的，可知的。如果目标间的度量是间接的，则会给核函数的设计带来较大困难。典型的如在社交网络中，每个人作为网络中的一个节点只跟部分人发生直接联系，从而获得距离度量，对没有直接联系的人其距离度量是缺失的。然而，核函数设计要求在任意两人间的相关性都可以直接计算出来，因此需要一种方法对缺失的距离度量进行补全。

如果我们用图来表示这种局部度量关系，则上述问题转化为如何依靠局部连接性来计算全局连接性。常见的启发式方法，如对象间的最短路径长

度，可以部分解决这一问题，但这种方式不能保证度量矩阵的半正定性，因此无法确认为合法的核函数。

一种可用的核函数称为散射核函数 [20]。同图核函数的定义类似，设无向图G的边集合E和顶点集合V及节点间的近邻关系$j \sim k$。定义图的相邻矩阵如下：

$$H_{jk} = \begin{cases} 1 & \text{if } j \sim k \\ -d_j & \text{if } j = k \\ 0 & otherwise, \end{cases}$$

其中d_j为和节点j连接的边数。注意$-\mathbf{H}$即是该图的拉普拉斯矩阵，在谱图理论（Spectral Graph Theory）中占有重要地位 [7]。可以证明对任一向量$\mathbf{w}$，$\mathbf{H}$具有如下性质：

$$\mathbf{w}^T \mathbf{H} \mathbf{w} = -\sum_{\{j,k\} \in E} (w_j - w_k)^2,$$

因此$\mathbf{H}$是负定矩阵。定义$\mathbf{H}$的指数矩阵如下：

$$\mathbf{K}_\beta = e^{\beta \mathbf{H}} = \mathbf{I} + \beta \mathbf{H} + \frac{\beta^2}{2!} \mathbf{H}^2 + ...,$$

由$\mathbf{H}$的对称性，可知$\mathbf{K}$是一个对称正定矩阵，因此是一个有效核函数。注意H_{jk}^m描述了结点j通过m步到达结点k的路径总数，因此$\mathbf{K}_\beta$事实上描述了节点j通过各种途径到达节点k的可能路径，即j到k之间的连接性。

如果求$\mathbf{K}$对β的导数，有：

$$\frac{\mathrm{d}}{\mathrm{d}\beta} \mathbf{K}_\beta = \mathbf{H} \mathbf{K}_\beta,$$

这一公式和热力学中描述热散射的热平衡方程具有相同形式，因此$\mathbf{K}$ 称为散射核（Diffusion Kernel） [20]。

5.4 Kernel PCA

从本节开始，我们将讨论基于核方法的几种重要模型。我们依然从线性模型开始。在第 5.1节我们推导了线性回归问题的核函数表示，类似的，可

以得到线性概率模型的核版本。我们以PCA为例，推导Kernel版的PCA，亦称Kernel PCA (KPCA)。

在第 2章我们提到过，PCA是一个线性高斯模型，其基本假设是数据由一个符合正态分布的隐变量通过线性映射得到，因此可很好描述高斯分布的数据。然而，在很多实际应用中数据的高斯性并不能保证，这时用PCA建模通常会产生较大偏差。如图 5.2 所示，原始数据的样本点呈现明显的非高斯性，这时用传统PCA很难找到一个合适的主成分方向。为解决这一问题，我们可以设计一个合理的非线性映射，将非线性数据映射到特征空间，使之具有合理的高斯性，即可进行有效的PCA建模。

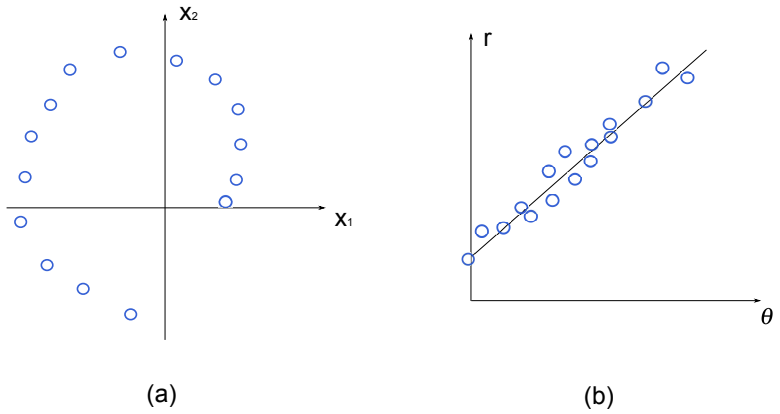

Fig. 5.2 不符合高斯分布的数据（左图）无法用PCA有效描述。选择合适的特征映射，在变换空间中表现出更明显的高斯性，可用PCA较好描述。图中所选的特征为样本的角度和辐值。

定义原始数据空间样本为$\{\mathbf{x}_n\}$, 非线性映射为$\boldsymbol{\phi}(\mathbf{x})$，且在原始空间和映射空间满足如下归一化条件，

$$\sum_n \mathbf{x}_n = \mathbf{0} \quad \sum_n \boldsymbol{\phi}(\mathbf{x}_n) = \mathbf{0}.$$

则在映射空间的协方差矩阵可写作：

$$\mathbf{S}^{\phi} = \frac{1}{N}\sum_n \boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^T = \frac{1}{N}\boldsymbol{\Phi}\boldsymbol{\Phi}^T$$

其中, $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), ..., \boldsymbol{\phi}(\mathbf{x}_N)]$。由第 2章对PCA的介绍可知，在映射空间求主成份$\mathbf{v}$等价于求$\mathbf{S}^{\phi}$的特征向量，即：

$$\mathbf{S}^{\phi}\mathbf{v} = \lambda\mathbf{v}. \tag{5.25}$$

整理可得：

$$\mathbf{v} = \frac{1}{N\lambda}\boldsymbol{\Phi}\boldsymbol{\Phi}^T\mathbf{v} = \boldsymbol{\Phi}(\frac{1}{N\lambda}\boldsymbol{\Phi}^T\mathbf{v}) = \boldsymbol{\Phi}\boldsymbol{\alpha}. \tag{5.26}$$

其中，

$$\boldsymbol{\alpha} = \frac{1}{N\lambda}\boldsymbol{\Phi}^T\mathbf{v}.$$

注意，$\boldsymbol{\alpha}$ 是一个N维向量，其中每一维对应一个数据点与特征向量$\mathbf{v}$的内积。同时，式（5.26）说明在映射空间的特征向量$\mathbf{v}$由所有数据样本的向量加权平均得到，权重为$\boldsymbol{\alpha}$。因此，求特征向量$\mathbf{v}$转化为求权重$\boldsymbol{\alpha}$，即由原始问题转化为对偶问题。

将$\mathbf{v} = \boldsymbol{\Phi}\boldsymbol{\alpha}$代回式（5.25），有:

$$\mathbf{S}^{\phi}\boldsymbol{\Phi}\boldsymbol{\alpha} = \lambda N\boldsymbol{\Phi}\boldsymbol{\alpha} \tag{5.27}$$

$$\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\alpha} = \lambda N\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\alpha} \tag{5.28}$$

$$\mathbf{K}^2\boldsymbol{\alpha} = \lambda N\mathbf{K}\boldsymbol{\alpha} \tag{5.29}$$

其中$K_{ij} = \boldsymbol{\phi}(\mathbf{x}_i)^T\boldsymbol{\phi}(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j)$。可证明，上式成立的必要条件是：

$$\mathbf{K}\boldsymbol{\alpha} = \lambda N\boldsymbol{\alpha}. \tag{5.30}$$

考虑特征向量$\mathbf{v}$应满足$\mathbf{v}^T\mathbf{v} = 1$，将$\mathbf{v} = \boldsymbol{\Phi}\boldsymbol{\alpha}$代入，有：

$$\boldsymbol{\alpha}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\alpha} = \boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} = 1$$

由式（5.30），上述限制条件可化简为：

$$\lambda N\boldsymbol{\alpha}^T\boldsymbol{\alpha} = 1.$$

由此，$\boldsymbol{\alpha}$可通过下式求解：

$$\mathbf{K}\boldsymbol{\alpha} = \lambda N\boldsymbol{\alpha} \quad s.t. \quad \boldsymbol{\alpha}^T\boldsymbol{\alpha} = \frac{1}{\lambda N}.$$

注意这一方程和传统PCA类似，其中$\boldsymbol{\alpha}$是$\mathbf{K}$的特征向量。解出$\boldsymbol{\alpha}$后，即可基于式（5.26）得到映射空间的主成份向量$\mathbf{v}$。和标准PCA 类似，我们可以求得多个主成份，组成主成份向量集$\{\mathbf{v}_i\}$

基于$\{\mathbf{v}_i\}$可对任一测试样本$\mathbf{x}$降维，这等价于在映射空间中计算$\boldsymbol{\phi}(\mathbf{x})$ 在各个主成份$\mathbf{v}_i$上的投影，计算如下：

$$\boldsymbol{\phi}(\mathbf{x})^T\mathbf{v}_i = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\Phi}\boldsymbol{\alpha} = \sum_n \alpha_n k(\mathbf{x}, \mathbf{x}_n).$$

由上式可知，虽然我们的目的是在映射空间中进行主成份提取并基于得到的主成份对数据进行降维，但并不需要在映射空间进行任何操作，所有计算都在原始空间中以核函数方式进行，计算得到的结果等价于在映射空间中进行计算。由此，我们得以在非常复杂的映射空间中对数据进行PCA建模，以解决在原始数据空间中的非高斯化问题，具有极大的灵活性和可扩展性。

5.5 高斯过程

在第 5.1节中，我们介绍了基于核方法的线性回归模型。和传统线性回归模型一样，该方法可对未知数据$\mathbf{x}$进行预测，但不能确定预测的可信度。在第 2章中我们知道，基于贝叶斯方法可以实现对未知数据的依概率预测，进而可得到预测的可信度。在这一方法中，通过对模型参数$\mathbf{w}$引入先验概率$p(\mathbf{w})$，通过学习可得到该参数的后验概率$p(\mathbf{w}|D)$，并以此对$\mathbf{x}$进行依概率预测，形式化如下:

$$p(t|\mathbf{x}) = \int p(t|\mathbf{x};\mathbf{w})p(\mathbf{w}|D)\mathrm{d}\mathbf{w},$$

其中$p(t|\mathbf{x};\mathbf{w})$是生成模型，$p(\mathbf{w}|D)$是基于训练数据$D$得到的对$\mathbf{w}$的后验估计，计算如下：

$$p(\mathbf{w}|D) \propto p(D|\mathbf{w})p(\mathbf{w}).$$

值得注意的是，上式中我们通过对$\mathbf{w}$赋予先验概率来实现对每个具体模型$p(t|\mathbf{x};\mathbf{w})$赋予先验概率。在核方法中，由于不存在一个显式的$\mathbf{w}$，上述通过参数引入先验的方法无法适用。高斯过程是在非参数模型中引入随机性的一种方法。

高斯过程是随机过程中的一种。一个随机过程可以认为是随机变量的扩展：随机变量是独立变量$\mathbf{x}$的分布特性，随机过程是一个变量集合$\mathscr{X}$的分布

特性。这里的变量集合在很多情况下是可数无穷集合（如自然数）或不可数集合（如实数）。如果我们对$\mathscr{X}$中的所有变量进行一次采样，则得到一个定义在$\mathscr{X}$上的函数f。因此，随机过程也可以认为是以函数f为变量的概率分布。

任何一个随机过程必须满足一致性和对称性。所谓一致性，是指从$\mathscr{X}$中任选的一个子集其分布形式是一致的。更严格地说，如果存在两个子集$\mathscr{X}_1$和$\mathscr{X}_2$，$X_1 \cap X_2 \neq \emptyset$，则由$X_1$或$X_2$通过边缘化其它变量导出的P($X_1 \cap X_2$)应该是一致的 [24]。所谓对称性，是指从$\mathscr{X}$中任选的一个子集，当对该子集中的变量调换位置时，其概率分布形式不变，只需对其中随机变量进行相应置换。Kolmogorov定理表明 [18]，如果满足这种一致性和对称性，则可保证该随机过程存在，且该随机过程可以由$\mathscr{X}$上任一子集的分布形式（称为finite-dimensional distribution, f.f.d.）描述。

高斯过程是f.f.d.为高斯分布的一种随机过程，即任取一个有限点集组成的矩阵$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]$，其目标变量取值组成的向量$\mathbf{y} = [y_1, y_2, ..., y_N]$ 满足高斯分布$N(\mathbf{y}; \boldsymbol{\mu}(\mathbf{X}), \mathbf{K}(\mathbf{X}))$。设$\boldsymbol{\mu}(\mathbf{X}) = 0$，则该高斯过程完全由协方差矩阵$\mathbf{K}(\mathbf{X})$ 确定，其中$K(\mathbf{X})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$，$k(\cdot, \cdot)$为任意核函数。直观上，我们希望距离相近的点具有较强的相关性，从而得到相似的取值y。一种常用的核函数有如下形式 [34]：

$$k(\mathbf{x}, \mathbf{x}') = v_0 exp\{-\frac{1}{2}\sum_{l=1}^{d} w_l (x_l - x_l')^2\} + a_0 + a_1 \sum_{l=1}^{d} x_l x_l',$$

其中d为x的维度，$\boldsymbol{\theta} = \{v_0, w_1, ..., w_d, a_0, a_1\}$ 为模型参数。

值得注意的是，上述分布规律对任意一个子集X都适用。现在假设训练数据为X，对任一测试数据$\mathbf{x}_*$，则$\hat{X} = X \cup \{\mathbf{x}_*\}$的观察值$\hat{\mathbf{y}}$同样符合高斯分布，即:

$$p(\hat{\mathbf{y}}) = N(\hat{\mathbf{y}}; 0, \hat{\mathbf{K}}),$$

其中:

$$\hat{\mathbf{K}} = \begin{pmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^T & v \end{pmatrix},$$

其中，$\mathbf{K}$是训练集X的Gram矩阵，$k_n = k(\mathbf{x}_*, \mathbf{x}_n)$，$v = k(\mathbf{x}_*, \mathbf{x}_*)$。由高斯分布的性质，可知其条件分布也是高斯的，即：

$$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = N(t_*; m(\mathbf{x}_*, \mathbf{X}, \mathbf{y}), \sigma^2(\mathbf{x}_*, \mathbf{X}, \mathbf{y})), \tag{5.31}$$

其中:

$$m(\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = \mathbf{k}^T \mathbf{K}^{-1} \mathbf{y} \tag{5.32}$$

$$\sigma^2(\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = v - \mathbf{k}^T \mathbf{K}^{-1} \mathbf{k}. \tag{5.33}$$

实际应用中，我们无法得到$\mathbf{y}$，而是$\mathbf{y}$的带噪声观察值$\mathbf{t}$，即：

$$\mathbf{t} = \mathbf{y} + \boldsymbol{\varepsilon},$$

其中$\boldsymbol{\varepsilon} \sim N(\mathbf{0}, \beta^{-1}\mathbf{I})$，则有：

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{y}) p(\mathbf{y}) \mathrm{d}\mathbf{y}.$$

由于$p(\mathbf{t}|\mathbf{y})$和$p(\mathbf{y})$都是高斯的，有：

$$p(\mathbf{t}) = N(\mathbf{t}; \mathbf{0}, \mathbf{C}),$$

其中:

$$\mathbf{C} = \mathbf{K} + \beta^{-1}\mathbf{I}.$$

基于式（5.31）类似的推导过程，可得：

$$p(t_*|\mathbf{x}_*, \mathbf{X}, \mathbf{t}) = N(t_*; m(\mathbf{x}_*, \mathbf{X}, \mathbf{t}), \sigma^2(\mathbf{x}_*, \mathbf{X}, \mathbf{t})),$$

其中:

$$m(\mathbf{x}_*, \mathbf{X}, \mathbf{t}) = \mathbf{k}^T \mathbf{C}^{-1} \mathbf{t} \tag{5.34}$$

$$\sigma^2(\mathbf{x}_*, \mathbf{X}, \mathbf{t}) = v - \mathbf{k}^T \mathbf{C}^{-1} \mathbf{k}. \tag{5.35}$$

回顾上述推导过程，可以发现我们并没有定义一个类似线性回归的显式预测函数，而是通过定义数据间的相关性来描述数据的整体分布属性，从而隐式定义了从$\mathbf{x}$到y 的随机预测函数$y(\mathbf{x})$，即高斯过程。和第 5.1节中基于核方法的线性回归模型相比，高斯过程不仅引入了数据间的距离，而且通过该距离定义了一个联合概率分布，从而引入了预测模型的随机性。引入这一随机性事实上给出了预测过程的可信度。比较式（5.32）和式（5.9），可以看到基于高斯过程预测的期望值和传统核方法得到的预测值是一致的，但高斯过程

给出了形如式(5.33)的估计方差。因此，高斯过程可以认为是传统核方法的随机版本。

另一方面，如果我们考虑对线性回归的系数$\mathbf{w}$引入先验概率$N(\mathbf{0},\mathbf{I})$，可以看到预测值y的联合概率分布正是以$\mathbf{K}=\boldsymbol{\Phi}^T\boldsymbol{\Phi}$为协方差距阵的高斯分布，即$y$是一个高斯过程。因此，高斯过程可以认为是贝叶斯线性回归方法的核函数版本。

5.6 支持向量机

不论是基于核函数的线性回归还是基于高斯过程的非参数模型，都需要计算Gram 矩阵$\mathbf{K}$及其逆矩阵。当训练集中的数据量较大时，这显然会带来非常高的计算复杂度和内存开销。同时，在预测过程中，测试样本要和训练集中的所有样本做核函数计算，同样带来较高的计算量。一种有效的解决方法是仅保留部分较重要的训练数据来进行预测，而将那些不重要的数据丢弃。这些保留下来的训练样本称为*支持向量*，相应的模型称为*支持向量机*，即SVM。

5.6.1 线性可分的SVM

我们以二分类问题来讨论SVM的基本概念。考虑两类数据C_1和C_2，并假设这两类数据线性可分。对这类问题，我们可以找到多个分类面对C_1和C_2进行完美划分，但我们希望得到的线性分类面$L: y(\mathbf{x})=\mathbf{w}^T\mathbf{x}+b=0$具有最大边界属性。具体而言，首先找到$C_1$和$C_2$两类数据样本中距离$L$最近的样本集合$S(C_1)$和$S(C_2)$，这两个样本集合可称为对应类别的边界样本集，每个边界样本集中的样本到分类面L的距离是相等的，两个边界样本集到分类面间的距离之和称为边界（Margin）。我们希望找到这样的分类面L，使得$S(C_1)$和$S(C_2)$两个边界样本集中的数据到分类面的距离相等，且该距离在所有分类面中最大化。这一分类面称为*最大边界分类面*（Max-Margin Hyperplane），相应的分类器称为*最大边界分类器*（Max-Margin Classifier）。上述最大边界分类面的确定仅与边界样本集相关，因此边界样本集中的训练样本称为支持向量（Support Vector）, 该分类器称为支持向量机（Support Vector Machine, SVM）。支持向量如图 5.3所示。

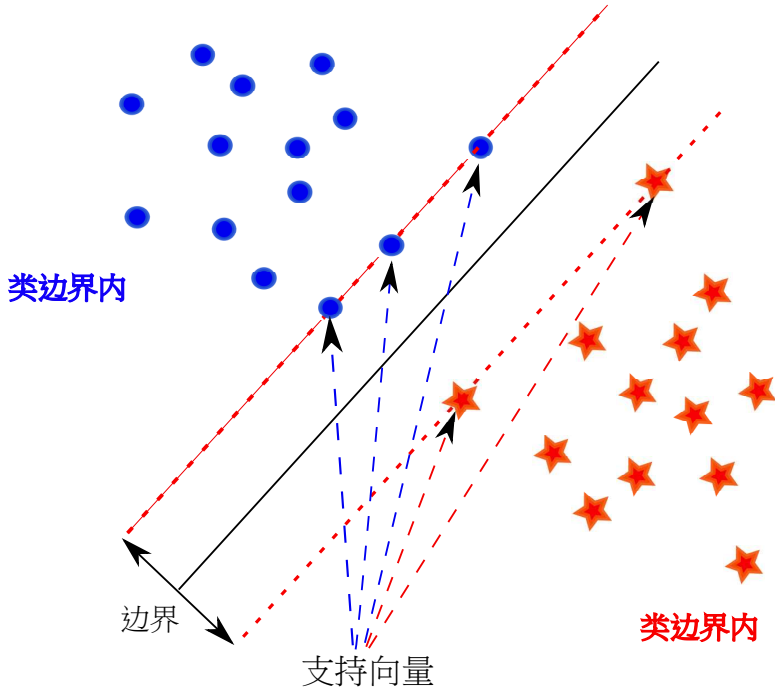

Fig. 5.3 线性可分的二分类问题中的支持向量。虚线上的点为支持向量（Support Vector），虚线间的距离为基于当前分类面的边界（Margin）。

我们简单推导一下如何从最大边界这一优化目标得到一个SVM。设训练数据$\{(\mathbf{x}_n, t_n)\}$，其中$t_n \in \{-1, 1\}$，代表样本$\mathbf{x}_n$的类别。定义：

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

设$y = 0$是分类面，则样本$\mathbf{x}_n$到分类面之间的距离可表示为$|\frac{y_n}{||\mathbf{w}||^2}|$。考虑$t_n$的取值，这一距离可表示为：

$$t_n \frac{y_n}{||\mathbf{w}||^2} = t_n \frac{\mathbf{w}^T \mathbf{x}_n + b}{||\mathbf{w}||^2}.$$

则最大边界分类准则可表示如下：

$$\arg\max_{\mathbf{w},b} \frac{1}{||\mathbf{w}||^2} \min_n [t^{(n)} y^{(n)}]. \tag{5.36}$$

注意当$\mathbf{w}$和b同时乘以相同的尺度常数κ时，分类面不会发生改变。利用这一点，可通过选择合适大小尺度，使得任意一个支持向量$\mathbf{x}_n$都满足$t_n y_n = 1$。注

意，引入上述限制并不会改变$\mathbf{x}_n$到分类面的距离。由于所有非支持向量到分类面的距离都大于支持向量到分类面的距离，因此对所有训练数据$\mathbf{x}_n$都满足如下限制条件：

$$t_n y_n \geq 1. \tag{5.37}$$

因此，式（5.36）定义的优化任务可改写成如下格式：

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 \quad s.t. \;\; t_n y_n \geq 1 \;\; \forall n = 1,2,...,N. \tag{5.38}$$

这是个典型的二次规划问题（Quadratic Programming），即在若干线性约束下对一个二阶函数进行优化。显然，这一问题的局部最优解即为全局最优解。可得到全局最优解是SVM相对神经网络和很多其它模型的重要优势，也是SVM广为应用的原因之一。

应用拉格朗日乘子法可将式（5.38）所示的受限优化问题写成如下形式：

$$\arg\min_{\mathbf{w},b,\{a_n\}} \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} a_n\{t_n y_n - 1\}. \tag{5.39}$$

注意上式需满足Karush-Kuhn-Tucker（KKT）条件：

$$a_n > 0$$

$$t_n y_n - 1 \geq 0$$

$$a_n\{t_n y_n - 1\} = 0.$$

对式(5.39)进行优化。首先求对b的偏导为零，有：

$$\sum_n a_n t_n = 0.$$

再求对$\mathbf{w}$的偏导为零，有：

$$\mathbf{w} = \sum_n a_n t_n \mathbf{x}_n.$$

代回式（5.39），则得到以a_n为变量的优化问题：

$$\arg\max_{\mathbf{a}} \left\{ \sum_n a_n - \frac{1}{2} \sum_n \sum_m a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \right\}. \tag{5.40}$$

其中$k(\mathbf{x},\mathbf{x}')=\mathbf{x}^T\mathbf{x}'$。这一问题是原问题（5.36）的对偶问题。对这一任务求解，可得到优化的$\{a_n\}$。注意上式中训练数据点仅以核函数的形式出现。我们可以扩展$k(\mathbf{x},\mathbf{x}')$为任意核函数，从而将原始空间的最大边界线性分类问题转化为映射空间的最大边界线性分类问题。通过这一扩展，原始空间中线性不可分问题可在映射空间中解决。为表示方便，我们依然以原始空间线性可分问题为例来讨论，但相关结论可直接扩展到映射空间线性可分的任务中。

基于上述优化过程得到$\{a_n\}$之后，即得到一个非参数SVM模型。对一个新样本$\mathbf{x}_*$进行分类时，我们计算该数据到分类面的距离，并依距离的符号确定该数据的类别，这同样可以表达成核函数形式，计算如下：

$$\mathbf{w}^T\mathbf{x}_*+b=\sum_{n=1}^{N}a_nt_n\mathbf{x}_*\mathbf{x}_n+b=\sum_{n=1}^{N}a_nt_nk(\mathbf{x}_*,\mathbf{x}_n)+b. \tag{5.41}$$

对任一支持向量$\mathbf{x}_n$，总有$t_ny_n=t_n(\mathbf{w}^T\mathbf{x}_n+b)=1$，由此可求得参数$b$。上式说明对数据$\mathbf{x}_*$的预测由所有训练数据通过$k(\mathbf{x},\mathbf{x}_n)$实现，每个训练数据依$a_nk(\mathbf{x}_*,\mathbf{x}_n)$作为权重，贡献其类别标记$t_n$，得到的平均值即为对该数据类别的预测。由于核函数描述了样本间的相似性，因此SVM的预测可认为是一种联想预测，用与测式样本相近的训练样本的类别来预测测试样本的类别。

值得注意的是，由KKT条件可知，当$t_ny_n\geq 1$时，$a_n=0$。这意味着，如果训练数据$\mathbf{x}_n$不是支持向量，则该数据对预测将不产生影向。因此，该模型中仅有支持向量才对分类预测起作用，其余数据都可以丢弃。这极大减小了模型规模，大幅减少了预测时的计算量。之所以产生这样的稀疏效果，是因为我们在寻找最大边界分类面时，只有支持向量会影响分类面的形状，所有类内点，不论如何变化，都不会对分界面的确定产生影响，因此，这些点与得到的分界面无关，也就不会对预测产生影响。

5.6.2 线性不可分的*SVM*

到目前为止，我们对SVM的推导都假设两类数据在特征空间是线性可分的。如果这一条件不能满足，则对任一个分类面，都会有一些点越过边界，我们称这些点为所属类别的边界外点，如图 5.4所示。这时限制条件（5.37）无法得到满足。为了得到一个合理的限制条件，可以对每个训练数据样本引入一个松驰变量ξ_n，使得加上该松驰变量后满足（5.37）所示的限制条件，即：

$$t_n y_n \geq 1-\xi_n \quad s.t. \quad \xi_n \geq 0.$$

可见，对在类边界上和边界内的$\mathbf{x}_n$，设$\xi=0$即可满足约束；如果$\mathbf{x}_n$在类边界外且被正确分类，则有$\xi_n \leq 1$; 如果$\mathbf{x}_n$被分类面错误分类，则需要$\xi_n > 1$。在优化过程中，我们希望ξ_n越小越好，即希望找到的分类面使得不需要引入太大的松驰量即可满足限制条件。因此，线性不可分的SVM优化问题可形式化为：

$$\arg\max_{\mathbf{w},b,\{\xi_n\}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n \quad s.t. \ t_n y_n \geq 1-\xi_n \ ; \ \xi_n \geq 0 \quad \forall n=1,2,...,N, \tag{5.42}$$

其中C是平衡边界大小和分类错误的超参数。同样用拉格朗日乘子法解上述优化问题，将其写成如下形式：

$$\arg\max_{\mathbf{w},b,\{\xi_n\}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\{t_n y_n - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n. \tag{5.43}$$

其对应的KKT条件为：

$$a_n(t_n y_n - 1 + \xi_n) = 0, \tag{5.44}$$

$$\mu_n \xi_n = 0. \tag{5.45}$$

基于式(5.43)所示的优化函数求对w,b,ξ_n的偏导并取0，可得：

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n, \tag{5.46}$$

$$\sum_{n=1}^{N} a_n t_n = 0, \tag{5.47}$$

$$a_n = C - \mu_n. \tag{5.48}$$

代入式（5.43）有：

$$\arg\max_{a} \left\{ \sum_n a_n - \frac{1}{2}\sum_n\sum_m a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \right\}. \tag{5.49}$$

这一结果与式（5.40）具有相同形式，不同的是需满足不同的约束条件。同样，$k(\mathbf{x}_n, \mathbf{x}_m)$可以取任一有效核函数，实现在映射空间而非原始空间中的线性分类。对某一测试样本$\mathbf{x}_*$的预测具有式（5.41）相同的形式，即：

$$y(\mathbf{x}_*) = \mathbf{w}^T\mathbf{x}_* + b = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n^T \mathbf{x}_* + b = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}_*, \mathbf{x}_n) + b. \tag{5.50}$$

同样，仅有$a_n > 0$对应的训练数据对预测结果产生影响，这些训练数据组成支持向量集。由式（5.44）可知，当$a_n > 0$时有：

$$t_n y_n - 1 + \xi_n = 0. \tag{5.51}$$

如果$\mathbf{x}_n$在正确分类的边界内，则必然有$t_n(\mathbf{w}^T\mathbf{x}_n + b) < 1$，因此不能满足式（5.51）所示的条件。因此，在线性不可分条件下，所有在边界上和边界外的训练样本都是支持向量。

回到优化任务（5.42），可知所有$\xi_n > 0$的点都是支持向量（反之不成立），因而会对模型产生影响。同时，$\sum_n \xi_n$是分类错误的上界（注意，如果$\xi_n > 1$，代表产生了一个分类错误），因此当C越大时，在对（5.42）进行优化时对分类错误越重视，得到的模型的支持向量数越少，模型的复杂度越低。当$C \to 0$时，线性不可分的优化问题退化为线性可分条件下的SVM。另一方面，如果两类数据混淆度越大，则$\xi_n > 0$的点越多，模型的复杂度越高。

5.6.3 μ-SVM

上节所述的SVM由参数C来控制模型复杂度，通常称为C-SVM。由于C的取值是无限制的，在构造实际系统时不容易操作。μ-SVM是和C-SVM等价的另一种SVM实现方法，但选择更有物理意义的参数，使实现起来更加容易。μ-SVM定义目标函数为:

$$\arg\min_{\mathbf{w},b,\{\xi_n\},\rho} \frac{1}{2}||\mathbf{w}||^2 - \mu\rho + \frac{1}{N}\sum_n \xi_n,$$

其中$\mu \in [0,1]$ 为模型参数。限制条件为：

$$t_n y_n > \rho - \xi_n; \quad \xi_n > 0; \quad \rho > 0.$$

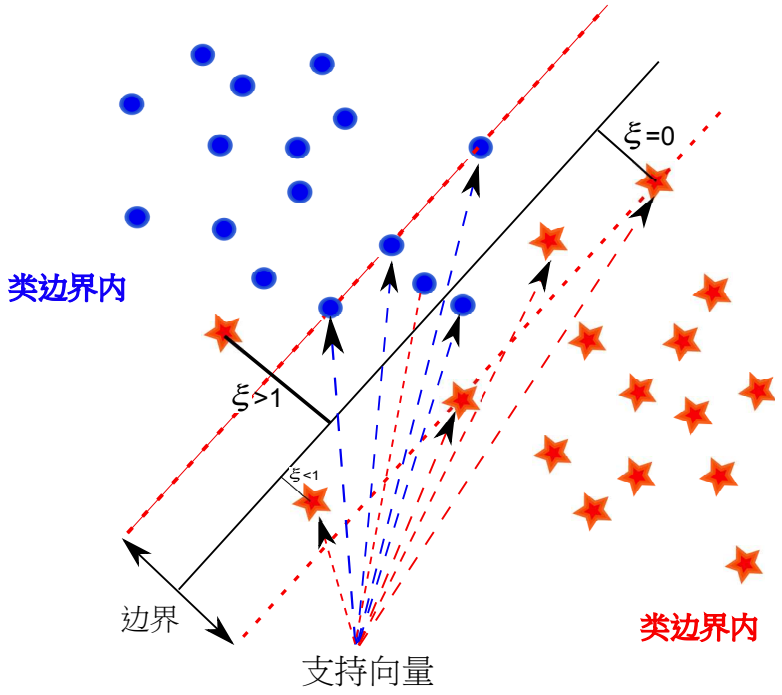

Fig. 5.4 非线性可分问题中的支持向量。对边界上和边界内的点，$\xi = 0$；对于边界外但分类正确的点，$0 < \xi < 1$；对其它点，$\xi \geq 1$。

和C-SVM相比，这一目标函数不需要调整$\mathbf{w}$和b的比例使边界上的点满足$y_n = 1$，而是由一个变量ρ来代表边界，并目标函数中对该边界最大化。

类似C-SVM，对上式应用拉格朗日法求解，可得如下对偶任务：

$$\arg\max_{\mathbf{a}} -\frac{1}{2}\sum_i\sum_j a_m a_n t_n t_m k(\mathbf{x}_n, \mathbf{x}_m),$$

约束条件为：

$$0 \geq \alpha_n \geq \frac{1}{N} \tag{5.52}$$

$$\sum_{n=1}^{N} \alpha_n t_n = 0 \tag{5.53}$$

$$\sum_{n=1}^{N} \alpha_n > \mu. \tag{5.54}$$

由式（5.54 ）可知，μ可以理解为训练数据中所有支持向量所占比例的下界。因此，通过设定合理的μ，可直观控制模型复杂度。同时，可以证明μ也是边界错误的上界。所谓边界错误，是指在边界外的所有点（即满足$\xi_n > 0$的点）在整个训练集中所占的比例 [5]。

5.6.4 SVM的若干讨论

SVM的特征

与其它分类器相比，SVM具有两个特征使得它具有强大的分类能力和广泛应用价值：最大边界分类准则和基于核方法的特征映射。

首先，最大边界分类准则使其关注最容易发生分类错误的样本，而非所有数据。这使得SVM在以减少分类错误为目标的分类任务中具有优势。虽然Logistic Regression 等分类器也有类似效果，即关注分类面附近易混淆的样本，但最大边界分类准则做的更为彻底，即没有发生分类错误的样本对分类面设计完全没有影响。这可以从模型的优化目标清楚看到：SVM的优化目标中，不是所有样本都会对误差函数产生贡献，只有在正确类别边界之外的数据其ξ值才是非零的，才会对优化任务的目标函数产生贡献。换句话说，只有满足$t(\mathbf{w}^T\mathbf{x}+b) < 1$的样本才会贡献误差，贡献度为$1-t(\mathbf{w}^T\mathbf{x}+b)$。这一事实可以写成误差函数如下：

$$\ell(\mathbf{x},t) = max\{0, 1-t(\mathbf{w}^T\mathbf{x}+b)\} = [1-t(\mathbf{w}^T\mathbf{x}+b)]_+, \tag{5.55}$$

其中$[\cdot]_+$表示取正值函数。式（5.55）称为Hinge Loss。基于这一表达，SVM的目标函数(5.42)可以写作：

$$\sum_{n=1}^{N} \ell(x_n,t_n) + \lambda||\mathbf{w}||^2.$$

因此，SVM可以看作是以Hinge Loss 为误差函数，并加入二阶规范的线性分类模型。

基于核方法的特征映射是SVM具有强大功能的另一个重要原因。最大边界分类模型本质上是线性的，这一线性性带来的一个优势是具有全局最优解，因此SVM的训练过程要比神经网络等模型简单的多，性能也更容易保证。但是，这种线性性对非线性可分数据很难处理。核方法通过设计恰当的

核函数将数据隐式地映射到特征空间，在特征空间中进行线性建模，从而极大解决了非线性数据的最大边界分类模型的建模问题。特别重要的是，最大边界分类准则极大简化了核方法的推理复杂性，因为只有支持向量才会对预测产生影响，而不像基于核方法的Logistic Regression模型那样需要保留所有训练数据。从这个角度上看，可以认为SVM 是前述基于核方法的线性回归模型的稀疏版本，这一稀疏性来源于最大边界的训练准则，而这一准则对应的是将传统基于Logistic function的误差函数改写为基于Hinge Loss的误差函数。

多分类SVM

SVM只可用于二分类任务。如果用到多分类任务上，一般常用*one-versus-the-rest*方式，为每个类设计一个二分类SVM，其正样本集为该类包含的训练数据，负样本集为所有其它类的训练数据。给定一个测试样本$\mathbf{x}_*$，利用所有分类器进行分类。如果有多个分类器同时认为该数据属于其对应的类，可比较每个SVM的输出值，取最大输出值的分类器对应的类为测试样本的分类。这种比较SVM输出值的方法并不能保证得到分类是正确的，因为不同SVM的输出值可能不具有可比性。另一种方法是对K类两两成对设计$K(K-1)$个分类器，然后采用投票法决定测试样本的类。这一方法计算量较大，且投票方法并不能保证得到正确分类。

Crammer等 [9]给出一个多分类SVM的设计方法，其基本思路是同时设计K个分类器，使得对任一训练样本$\mathbf{x}$，其到正确分类所对应的分类器输出的结果显著高于其它分类器的输出。具体地，当正确分类器输出的结果大于所有其它分类器一个边界值δ时，即不产生误差，否则依输出值差距的大小计算误差。这事实上即是Hinge Loss误差函数。

用于回归任务的SVM

SVM本身是用于分类任务的，但Hinge Loss的思路同样可用于回归任务。与SVM仅关注分类面附近产生混淆的训练样本点类似，在回归任务中，我们可以更加关注远离回归中心的点，这些点对回归错误贡献最大，而对那些在回归曲线附近的点可不予考虑。

以标准线性回归任务为例，其二阶约束的误差函数可定义为：

$$\frac{1}{2}\sum_{n=1}^{N}\{y_n - t_n\}^2 + \frac{\lambda}{2}||\mathbf{w}||^2,$$

其中$y_n = \mathbf{w}^T\mathbf{x}_n$为模型对$\mathbf{x}_n$的回归值。可以将上式中的二阶误差用Hinge Loss替换:

$$\sum_{n=1}^{N}[|y_n - t_n| - \varepsilon]_+ + \frac{\lambda}{2}||\mathbf{w}||^2,$$

其中ε为不计入误差的边界大小。对上式进行优化，即可得到基于Hing Loss的回归模型。与标准线性模型类似，该模型同样可以写成核函数形式，从而对非线性数据在映射空间里进行线性建模。

5.7 相关向量机

前面提到过，SVM可以认为是基于核方法的Logistic Regression的Hinge Loss 版本。这种基于Hinge Loss的稀疏建模方法本质上是以距离为标准选择支持向量，因此缺少概率意义。另一种获得稀疏向量的方法是基于贝叶斯框架的自动相关性检测（Automatic Relevance Detection，ARD）。通过ARD得到的稀疏核模型称为相关向量机（Relevance Vector Machine，RVM）。

我们以回归任务为例来推导RVM。一个简单的线性回归模型可以写作：

$$t = \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}) + \varepsilon,$$

如果我们对参数$\mathbf{w}$引入高斯先验$p(\mathbf{w}) \sim N(\mathbf{0}, \alpha^{-1}\mathbf{I})$,则得到贝叶斯线性回归模型。如果我们对不同特征维度引入不同的高斯先验，即:

$$p(w_i) \sim N(0, \alpha_i^{-1}),$$

并对α_i做最大似然估计，则与任务无关的α_i将趋向无穷大。无穷大的α_i 意味着$\phi_i(\mathbf{x})$对应的参数w_i被强烈限制在0点附近，因此$\phi_i(\mathbf{x})$将不对预测产生影响。这一方法称为稀疏贝叶斯方法 [27]。通过这一方法，那些对预测无关的特征会被自动检测出来，因此称为自动相关性检测（ARD）。

如果我们将$\phi_i(\mathbf{x})$定义到所有训练样本上，即$\phi_i(\mathbf{x}) = k(\mathbf{x}_i, \mathbf{x})$，其中$k(\cdot,\cdot)$是任意一个二元函数，则通过上述稀疏贝叶斯方法，会自动选出与预测任务相关的训练样本，组成相关向量集，在进行预测时仅需与相关向量集中的样本进行核函数计算。这事实上与SVM 选择支持向量具有类似思路，只不过

用相关向量代替了支持向量，因此称为相关向量机（RVM）。RVM和SVM具有很多相似性，都是选择训练样本的子集来大幅减小预测时的计算复杂性，因此可以统称稀疏性核方法。RVM的模型大小通常小于SVM。

和回归任务类似，RVM也可以用于分类任务，其基本思路也是类似的：基于Logistic Regression 模型$y=\sigma(\sum_{i=1}^{N} w_i k(\mathbf{x}_i,\mathbf{x}))$，其中$k(\mathbf{x}_i,\mathbf{x})$为定义在训练样本$\mathbf{x}_i$处的特征函数。对不同$w_i$引入独立的高斯先验$p(w_i)=N(0,\alpha_i^{-1})$，通过最大似然准则对$\{\alpha_i\}$进行优化，与预测任务无关的$k(\mathbf{x}_i,\mathbf{x})$所对应的$\alpha_i$将被置为无穷大值，其所对应的$w_i$将被置零，因而$k(\mathbf{x}_i,\mathbf{x})$将自动从预测方程中去除。注意，该方法基于概率模型，因此输出自然具有概率意义（即$\mathbf{x}$属于某一类的后验概率）。同时，该方法可以方便扩展到多分类问题，只需用Softmax函数取代二分类问题中的Sigmoid函数。这是RVM与SVM相比的一个明显优势。

值得注意的是，RVM 中的二元函数$k(\mathbf{x}_n,\mathbf{x})$可以是任意函数，而SVM中的$k(\mathbf{x}_n,\mathbf{x})$必须是合法核函数。同时，稀疏贝叶斯方法可以灵活定义$\boldsymbol{\phi}(\mathbf{x})$，RVM中对$\boldsymbol{\phi}(\mathbf{x})$ 的定义仅是稀疏贝叶斯方法的一种特殊形式。

RVM与SVM的另一个不同是，RVM的目标函数不是凸函数，因此不能保证得到全局最优解。同时，RVM要求$N\times N$ 维Gram矩阵的逆矩阵，因此模型训练的计算量较大，但其优势是只需训练一次，而不必象SVM那样训练多次以选择合适的模型参数C或μ.

5.8 本章小结

本章我们从线性回归模型出发，推导出该模型的对偶表达，从而引出了核函数的概念。我们介绍了核函数的性质和构造方法，并讨论了一些常用核函数形式。我们进一步讨论了主成分分析（PCA）的核版本，这一扩展使得PCA得以处理非高斯数据。进一步，我们讨论了高斯过程，并推导出基于该过程进行贝叶斯线性回归的表达方式。最后，我们讨论了支持向量机和相关向量机，这是两种典型的稀疏性核方法，该方法在预测时仅考虑最有影响力的训练数据（支持向量或相关向量），因而可极大减小预测时的计算量。

核方法的一个基本特征是基于训练集中的数据对未知数据进行预测，这和其它机器学习方法有很大不同。传统方法都是假设一个参数模型，利用训练集对参数进行选择，再基于该参数模型进行预测。模型训练是一种抽象化过程，相当于知识积累。核方法本质上是一种非参数方法，即不存在一个抽象过程，而是用全体（如高斯过程）或部分（如SVM）训练数据对未知数据

进行预测，预测时不同数据依其与待测试数据的相似性参与贡献。如果传统参数模型方法称为抽象学习，核方法更接近一种基于相似性的联想学习，衡量相似性的方法即是核函数。

值得注意的是，核函数隐性定义了一个高维特征空间，但这并不意味着数据在这一特征空间中具有更多信息。事实上，这些高维特征空间各个维度之间是有相关性的，本质上其自由维度的个数和原始数据是一样的。然而，这种特征映射的确可以使数据在特征空间表现出更强的线性或高斯性，从而可以被线性模型更好建模。至于这一特征映射是否真的具有这样的效果，则完全依赖映射本身的性质。核方法的强大之处在于我们可以将特征映射的设计转化成距离度量的设计，从而极大降低了特征设计的难度。

如何设计合理的核函数在核方法中具有重要意义。在很多问题中，一些常见的核函数（如线性核，多项式核，RBF核等）即可取得较好的效果，但在更多情况下，我们需要设计与任务相关的核函数，如前面讨论过的在集合、序列上的核函数等。设计这些核函数时应尽量应用领域知识，使核函数可以有效反映数据点间的距离。应用领域知识是核方法相对神经网络方法的对比优势之一。

另一方面，必须应用领域知识对核函数进行定义也是核方法的一个缺陷。因为核函数必须人为定义，导致模型的学习能力较差，无法从数据中得到有效知识。神经网络和深度学习方法可以从大规模数据中学习知识，但对人为知识的表达能力有限。贝叶斯学派提供了一种对人为知识和数据知识进行有效组合的方式，由人来设计概率结构，通过数据对概率结构中的参数进行学习，从而得到比核方法更灵活，同时比神经网络更具扩展性的模型，即概率图模型。我们下一章将具体讨论概率图模型的基本概念和学习方法。

5.9 相关资源

- 本章关于核方法的讨论参考了Holfmann的文章 [15]。
- 本章关于SVM和RVM的讨论大量参考了Bishop的《Pattern Recognition and Machine Learning》一书的第7章 [3]。
- 关于核方法在聚类、回归、相关性分析等任务上的应用，请参考Shawe-Taylor 2004年的著作 [25]。
- Kung 2014年的著作《Kernel methods and machine learning》对核方法进行了较全面的介绍 [1]。
- 关于高斯过程的细节知识，可参考Seeger的综述文章 [24]。

- 关于SVM的更多知识，请参考相关文献 [28, 29, 4, 6, 22, 23, 14]

。

Chapter 6
图模型

Chapter 7
非监督学习

Chapter 8
非参数模型

Chapter 9
遗传学习

Chapter 10
强化学习

Chapter 11
优化方法

References

[1] (2014) Kernel methods and machine learning. Cambridge University Press

[2] Aronszajn N (1950) Theory of reproducing kernels. Transactions of the American mathematical society 68(3):337–404

[3] Bishop CM (2006) Pattern recognition and machine Learning. Springer

[4] Burges CJ (1998) A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery 2(2):121–167

[5] Chen PH, Lin CJ, Schölkopf B (2005) A tutorial on ν-support vector machines. Applied Stochastic Models in Business and Industry 21(2):111–136

[6] Christianini N, Shawe-Taylor J (2000) Support Vector Machines and other kernel-based learning methods. Cambridge University Press, New York

[7] Chung FR (1997) Spectral graph theory. 92, American Mathematical Soc.

[8] Cortes C, Haffner P, Mohri M (2004) Rational kernels: Theory and algorithms. Journal of Machine Learning Research 5(Aug):1035–1062

[9] Crammer K, Singer Y (2002) On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research 2(2):265–292

[10] Gärtner T (2003) A survey of kernels for structured data. ACM SIGKDD Explorations Newsletter 5(1):49–58

[11] Gärtner T, Flach P, Wrobel S (2003) On graph kernels: Hardness results and efficient alternatives. Learning Theory and Kernel Machines pp 129–143

[12] Genton MG (2001) Classes of kernels for machine learning: a statistics perspective. Journal of machine learning research 2(Dec):299–312

[13] Goldberg Y, Elhadad M (2008) splitsvm: fast, space-efficient, non-heuristic, polynomial kernel computation for nlp applications. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers, Association for Computational Linguistics, pp 237–240

[14] Herbrich R (2016) Learning kernel classifiers. Mit Press

[15] Hofmann T, Sch?lkopf B, Smola AJ (2008) Kernel methods in machine learning. Annals of Statistics 36(3):1171–1220

[16] Jaakkola T, Haussler D (1999) Exploiting generative models in discriminative classifiers. In: Advances in neural information processing systems, pp 487–493

[17] Kashima H, Tsuda K, Inokuchi A (2003) Marginalized kernels between labeled graphs. In: Proceedings of the 20th international conference on machine learning (ICML-03), pp 321–328

[18] Kolmogorov A (1956) Foundations of the theory of probability. Chelsea Pubishing Company, New York

[19] Kondor R, Jebara T (2003) A kernel between sets of vectors. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp 361–368

[20] Kondor RI, Lafferty J (2002) Diffusion kernels on graphs and other discrete input spaces. In: ICML, vol 2, pp 315–322

[21] Mercer J (1909) Functions of positive and negative type, and their connection with the theory of integral equations. Philosophical transactions of the royal society of London Series A, containing papers of a mathematical or physical character 209:415–446

[22] Muller KR, Mika S, Ratsch G, Tsuda K, Scholkopf B (2001) An introduction to kernel-based learning algorithms. IEEE transactions on neural networks 12(2):181–201

[23] Scholkopf B, Smola AJ (2001) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press

[24] Seeger MW (2004) Gaussian processes for machine learning. International Journal of Neural Systems 14(2):69–106

[25] Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge university press

[26] Shimodaira H, Noma Ki, Nakai M, Sagayama S (2002) Dynamic time-alignment kernel in support vector machine. In: Advances in neural information processing systems, pp 921–928

[27] Tipping ME, Faul AC, et al (2003) Fast marginal likelihood maximisation for sparse bayesian models. In: AISTATS

[28] Vapnik V (2013) The nature of statistical learning theory. Springer science & business media

[29] Vapnik VN, Vapnik V (1998) Statistical learning theory, vol 1. Wiley New York

[30] Venkatasubramanian S (2013) Moving heaven and earth: Distances between distributions. ACM SIGACT News 44(3):56–68

[31] Vishwanathan SVN, Schraudolph NN, Kondor R, Borgwardt KM (2010) Graph kernels. Journal of Machine Learning Research 11(Apr):1201–1242

[32] Wan V, Renals S (2002) Evaluation of kernel methods for speaker verification and identification. In: Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on, IEEE, vol 1, pp I–669

[33] Watkins C (1999) Dynamic alignment kernels. Advances in neural information processing systems pp 39–50

[34] Williams CKI, Rasmussen CE (1996) Gaussian processes for regression pp 514–520