第 4 章 核方法

4.1 基本思想

在前一章的结尾，我们提到可以通过一组基函数将输入 $\boldsymbol{x}$ 映射到一个新的特征空间 $\boldsymbol{\phi}(\boldsymbol{x})$。在原始输入空间中线性不可分的问题，经过适当的基函数变换之后，在新的特征空间中就可能变得线性可分了。然而，为特定任务选择合适的一组基函数并不容易。它通常依赖大量的人工设计与经验判断，并需要通过反复试验来筛选出有效的特征变换方式。另一种构建模型（或者构造特征）的思想是，不显式定义基函数，而是计算输入样本与训练集中各样本之间的相似度，再通过学习一组权重对这些相似度进行加权组合，从而实现拟合或分类。在第一章中简要介绍过的 K 近邻方法就是基于类似思想下所构建的一种模型，该方法先找出 K 个与输入 $\boldsymbol{x}$ 最相似（或距离最近）的样本，然后将它们的输出取平均作为预测结果。类似地，部分核函数（Kernel Function）也可以基于样本之间的距离构造，例如高斯核函数，其函数值随着样本间距离的增大而减小，因此可以用于反映样本之间的相似程度。不过，核函数与距离并不是同一个概念。

核方法属于基于记忆的方法（Memory-based Method），因为在预测时通常需要保留全部或部分训练样本，以便计算样本间的相似度并基于这些相似度进行预测。这与上一章介绍的线性模型不同：在传统线性模型中，一旦完成参数估计，原始训练样本所包含的信息便已被压缩并融入模型参数中，训练样本随即被舍弃，在预测时不再直接使用。因此，与传统线性模型相比，核方法由于需要构造和处理核矩阵，其训练和预测过程通常均相对较慢。

在讨论核方法的独特优势之前，我们需要了解前一章所介绍的用于回归和分类的线性模型都可以通过其对偶形式（Dual Representation）重新表述，而在这一过程中，核函数的形式会自然地被推导出来。这里我们仅以带 ℓ_2 正则项的线性回归模型为例，其损失函数可以表示为：

$$\mathcal{L}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w}\right)^2 + \frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w} \tag{4.1}$$

其中 $\lambda > 0$，$\boldsymbol{\phi}(\boldsymbol{x})$ 表示将输入 $\boldsymbol{x}$ 通过一组基函数变换后得到的特征向量。对上述损失函数关于权重参数 $\boldsymbol{w}$ 求偏导，令导数为零并整理可得：

$$\boldsymbol{w} = \frac{1}{\lambda}\sum_{i=1}^{N}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w}\right)\boldsymbol{\phi}(\boldsymbol{x}_i) \tag{4.2}$$

为便于后续推导，我们引入以下记号：

$$\alpha_i = \frac{1}{\lambda}\left(y_i - \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w}\right) \tag{4.3}$$

则公式4.2可改写为：

$$\boldsymbol{w} = \sum_{i=1}^{N}\alpha_i \boldsymbol{\phi}(\boldsymbol{x}_i) = \boldsymbol{\Phi}^\top \boldsymbol{\alpha} \tag{4.4}$$

其中 $\boldsymbol{\alpha}=(\alpha_1,\alpha_2,\ldots,\alpha_N)^\top$，矩阵 $\boldsymbol{\Phi}$ 的形式为：

$$\boldsymbol{\Phi}=\begin{bmatrix}\phi_0(\boldsymbol{x}_1) & \phi_1(\boldsymbol{x}_1) & \cdots & \phi_{M-1}(\boldsymbol{x}_1)\\ \phi_0(\boldsymbol{x}_2) & \phi_1(\boldsymbol{x}_2) & \cdots & \phi_{M-1}(\boldsymbol{x}_2)\\ \vdots & \vdots & \ddots & \vdots\\ \phi_0(\boldsymbol{x}_N) & \phi_1(\boldsymbol{x}_N) & \cdots & \phi_{M-1}(\boldsymbol{x}_N)\end{bmatrix} \tag{4.5}$$

矩阵 $\boldsymbol{\Phi}$ 在第3.1节已经出现过，通常称为设计矩阵（Design Matrix）或特征矩阵。将公式4.1所示的损失函数视为参数 $\boldsymbol{\alpha}$ 的函数，并将等式 $\boldsymbol{w}=\boldsymbol{\Phi}^\top\boldsymbol{\alpha}$ 代入该损失函数后，我们可得：

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha})&=\frac{1}{2}\sum_{i=1}^{N}\left(y_i-\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\alpha}\right)\right)^2+\frac{\lambda}{2}\left(\boldsymbol{\Phi}^\top\boldsymbol{\alpha}\right)^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\alpha}\right)\\&=\frac{1}{2}\mathbf{Y}^\top\mathbf{Y}-\boldsymbol{\alpha}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\mathbf{Y}+\frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\alpha}+\frac{\lambda}{2}\boldsymbol{\alpha}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\alpha}\end{aligned} \tag{4.6}$$

其中 $\mathbf{Y}=(y_1,y_2,\ldots,y_N)^\top$ 是所有训练样本的实际观测值所构成的列向量。将上式中多次出现的 $\boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ 记作 $\mathbf{K}$，则上式可以表示为：

$$\mathcal{L}(\boldsymbol{\alpha})=\frac{1}{2}\mathbf{Y}^\top\mathbf{Y}-\boldsymbol{\alpha}^\top\mathbf{K}\mathbf{Y}+\frac{1}{2}\boldsymbol{\alpha}^\top\mathbf{K}\mathbf{K}\boldsymbol{\alpha}+\frac{\lambda}{2}\boldsymbol{\alpha}^\top\mathbf{K}\boldsymbol{\alpha} \tag{4.7}$$

其中 $\mathbf{K}=\boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ 被称为格拉姆矩阵（Gram Matrix）或内积矩阵，其具体形式为：

$$\mathbf{K}=\begin{bmatrix}\boldsymbol{\phi}(\boldsymbol{x}_1)^\top\boldsymbol{\phi}(\boldsymbol{x}_1) & \boldsymbol{\phi}(\boldsymbol{x}_1)^\top\boldsymbol{\phi}(\boldsymbol{x}_2) & \cdots & \boldsymbol{\phi}(\boldsymbol{x}_1)^\top\boldsymbol{\phi}(\boldsymbol{x}_N)\\ \boldsymbol{\phi}(\boldsymbol{x}_2)^\top\boldsymbol{\phi}(\boldsymbol{x}_1) & \boldsymbol{\phi}(\boldsymbol{x}_2)^\top\boldsymbol{\phi}(\boldsymbol{x}_2) & \cdots & \boldsymbol{\phi}(\boldsymbol{x}_2)^\top\boldsymbol{\phi}(\boldsymbol{x}_N)\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{\phi}(\boldsymbol{x}_N)^\top\boldsymbol{\phi}(\boldsymbol{x}_1) & \boldsymbol{\phi}(\boldsymbol{x}_N)^\top\boldsymbol{\phi}(\boldsymbol{x}_2) & \cdots & \boldsymbol{\phi}(\boldsymbol{x}_N)^\top\boldsymbol{\phi}(\boldsymbol{x}_N)\end{bmatrix} \tag{4.8}$$

矩阵 $\mathbf{K}$ 是 $N\times N$ 的方阵，它的每个元素是两个样本特征向量之间的内积：

$$k_{ij}=\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{\phi}(\boldsymbol{x}_j)=\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{4.9}$$

其中 $\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$ 称为**核函数**。

将公式4.7对 $\boldsymbol{\alpha}$ 求偏导，并令导数为零，我们可以求解出：

$$\boldsymbol{\alpha}=(\mathbf{K}+\lambda\mathbf{I})^{-1}\mathbf{Y} \tag{4.10}$$

将上式代入线性回归模型中，就可以得到对新输入样本 $\boldsymbol{x}$ 的预测表达式：

$$\begin{aligned}f(\boldsymbol{x})&=\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{w}\\&=\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{\Phi}^\top\boldsymbol{\alpha} && \text{（利用公式4.4）}\\&=\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{\Phi}^\top(\mathbf{K}+\lambda\mathbf{I})^{-1}\mathbf{Y} && \text{（利用公式4.10）}\\&=(\kappa(\boldsymbol{x},\boldsymbol{x}_1),\kappa(\boldsymbol{x},\boldsymbol{x}_2),\ldots,\kappa(\boldsymbol{x},\boldsymbol{x}_N))(\mathbf{K}+\lambda\mathbf{I})^{-1}\mathbf{Y} && \text{（核函数的定义）}\end{aligned}$$

由上式可知，新样本 $\boldsymbol{x}$ 的预测值可以完全通过核函数 $\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$ 来表示（矩阵 $\mathbf{K}$ 的每一元素也是核函数的值），并且该预测值是所有训练样本的实际观测值的线性组合。

在核方法中，核函数的选择至关重要，因为它直接影响预测阶段如何整合训练样本的实际观测值。由公式4.9可知，核函数通常定义为两个样本特征向量之间的内积，而这些特征向量由

一组基函数所确定。假设样本 $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^2$（即输入空间为二维），我们可以定义如下核函数：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = (1 + x_1 x_1' + x_2 x_2')^2 \tag{4.11}$$

该核函数可以看成先对样本使用如下一组基函数进行特征变换：

$$\boldsymbol{\phi}(\boldsymbol{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)^\top \tag{4.12}$$

然后在变换后的特征空间中计算内积，其结果为：

$$\begin{aligned}
\boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{z}) &= (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)(1, \sqrt{2}x_1', \sqrt{2}x_2', x_1'^2, x_2'^2, \sqrt{2}x_1' x_2')^\top \\
&= 1 + 2x_1 x_1' + 2x_2 x_2' + x_1^2 x_1'^2 + x_2^2 x_2'^2 + 2x_1 x_2 x_1' x_2' \\
&= (1 + x_1 x_1' + x_2 x_2')^2
\end{aligned}$$

尽管如上例所示，核函数可以对应于基函数变换后的向量内积。但在实际应用中，核方法的一大优势在于：在满足特定条件（在第4.2节讨论）的情况下，我们并不需要显式地计算和存储基函数的值，而可以根据需要直接定义合适的核函数 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$。由于无需显式构造基函数进行特征变换，核函数可以是定义在原始输入空间中的一个标量函数，其值等于变换后高维空间中两个向量的内积。理论上，这一高维特征空间甚至可以扩展至无限维，而这在需要显式计算和存储基函数值的情况下是无法实现的。换言之，我们无需考虑样本在高维空间中的特征表示，只需专注于设计能够有效衡量两个样本之间相似性的距离度量函数。所谓的“核技巧”（Kernel Trick），又称“核替换”（Kernel Substitution）就是将方法中出现的向量内积替换为核函数。在无需显式构造高维或无限维特征映射 $\boldsymbol{\phi}$ 的情况下，等价地在高维或无限维特征空间中计算，从而实现方法的扩展或改进。

例如，在传统的主成分分析 PCA（Principal Component Analysis）中，我们先对数据矩阵 $\mathbf{X}$ 进行中心化处理（即减去各列的均值），再对中心化后的矩阵进行奇异值分解（参见公式3.36），从而找到方差最大的若干个投影方向。然而，当数据呈现同心圆或环状聚类等非线性分布时，传统 PCA 就会失效。此时，可以采用径向基函数（公式4.23）或高斯核（公式4.24）将数据映射到高维特征空间，然后再在该空间中进行主成分分析，从而克服传统 PCA 方法的局限性。值得注意的是，在这一过程中也无需显式构造特征变换的基函数，因为通过核函数就能够直接计算奇异值分解所需的格拉姆矩阵 $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$。类似地，在 K 近邻方法中，我们也可以使用核技巧，将样本间的距离度量替换为合适的核函数，以便更好地反映数据在高维特征空间中的相似性，从而进一步提升 K 近邻的分类或回归性能。

4.2 核函数

核函数需要满足以下两个条件：

- 对称性：即 $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x}', \boldsymbol{x})$；
- 半正定性：核函数所构造的格拉姆矩阵（Gram Matrix）必须是半正定的，即对于任意 N 维列向量 $\boldsymbol{\alpha}$，都有 $\boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \geq 0$，其中 $\mathbf{K} \in \mathbb{R}^{N \times N}$。

对称性条件相对容易理解。正如前文所述，核函数可以看作是两个样本之间相似度的度量，因此在交换两个输入变量时，其取值应保持不变。半正定性则与优化目标函数的凸性要求紧密相关。在许多核方法中（例如本章后续将介绍的支持向量机），其优化目标往往包含如下形式的二次型：

$$\frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} \tag{4.13}$$

对于包含上述二次项的目标损失函数 $\mathcal{L}(\boldsymbol{\alpha})$，其海森矩阵正好是 $\mathbf{K}$。根据二阶泰勒展开，对于任意扰动向量 $\Delta\boldsymbol{\alpha}$，可以将目标函数表示为：

$$\mathcal{L}(\boldsymbol{\alpha}+\Delta\boldsymbol{\alpha}) = \mathcal{L}(\boldsymbol{\alpha}) + \nabla\mathcal{L}(\boldsymbol{\alpha})^\top\Delta\boldsymbol{\alpha} + \frac{1}{2}\Delta\boldsymbol{\alpha}^\top\mathbf{K}\Delta\boldsymbol{\alpha} \tag{4.14}$$

由于假设目标函数是二次函数，因而其二阶泰勒展开是精确的，不存在更高阶的余项。如果核矩阵 $\mathbf{K}$ 是半正定的，则对于任意 $\Delta\boldsymbol{\alpha}$，均有 $\Delta\boldsymbol{\alpha}^\top\mathbf{K}\Delta\boldsymbol{\alpha} \geq 0$。因此：：

$$\mathcal{L}(\boldsymbol{\alpha}+\Delta\boldsymbol{\alpha}) \geq \mathcal{L}(\boldsymbol{\alpha}) + \nabla\mathcal{L}(\boldsymbol{\alpha})^\top\Delta\boldsymbol{\alpha} \tag{4.15}$$

上式与公式3.142给出的凸函数一阶条件完全一致，因此该目标函数是凸函数。这也说明，核矩阵的半正定性能够保证上述二次型的凸性。凸优化问题具有良好的性质，其任意局部最优解都是全局最优解，并且已经发展出多种高效的求解算法。

如前所述，核函数可以对应于经过基函数变换后的向量内积。当基函数 $\boldsymbol{\phi}$ 确定之后，核函数的形式便随之确定，即为 $\kappa(\boldsymbol{x},\boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{\phi}(\boldsymbol{x}')$。但若先给定核函数的形式，再尝试找到类似公式4.12所示的一组基函数，则通常较为困难。尤其当变换后的特征空间为无限维时，通常无法显式计算和存储所有基函数的取值。然而，只要核函数所构造的格拉姆矩阵是正半定的，虽仍无法保证一定能够找到一组显式的基函数，但一定能够将核函数表示为特征向量的内积形式，其原因如下。

我们可以将公式4.8所示的格拉姆矩阵改写成如下等价形式：

$$\mathbf{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_1,\boldsymbol{x}_1) & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2,\boldsymbol{x}_1) & \kappa(\boldsymbol{x}_2,\boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x}_2,\boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x}_N,\boldsymbol{x}_1) & \kappa(\boldsymbol{x}_N,\boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x}_N,\boldsymbol{x}_N) \end{bmatrix} \tag{4.16}$$

由于格拉姆矩阵一般为实对称矩阵（复数情形类似），它可以被分解成：

$$\mathbf{K} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top \tag{4.17}$$

其中 $\mathbf{Q}$ 是由 $\mathbf{K}$ 的特征向量所组成的正交矩阵，而 $\boldsymbol{\Lambda}$ 是一个对角矩阵，其对角线元素 λ_i 为矩阵 $\mathbf{K}$ 的特征值。我们设 $\mathbf{Q} = \mathbf{U}^\top$，则公式4.17可以改写为：

$$\mathbf{K} = \mathbf{U}^\top\boldsymbol{\Lambda}\mathbf{U} \tag{4.18}$$

如果 $\mathbf{K}$ 是半正定的，则有 $\lambda_i \geq 0$，$i = \{1,2,\ldots,N\}$。根据公式4.18，格拉姆矩阵 $\mathbf{K}$ 的第 i 行、第 j 列的元素 k_{ij} 可表示为：

$$k_{ij} = \left(\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}_i\right)^\top\left(\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}_j\right) \tag{4.19}$$

其中 $\mathbf{U}_i$ 表示矩阵 $\mathbf{U}$ 的第 i 列向量。注意：若矩阵 $\mathbf{K}$ 不满足半正定性条件，则可能存在特征值 $\lambda_i < 0$，上式中的 $\mathbf{\Lambda}^{\frac{1}{2}}$ 不存在。我们定义 $\boldsymbol{\phi}(\boldsymbol{x}_i) = \mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U}_i$，则有：

$$k_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{\phi}(\boldsymbol{x}_j) \tag{4.20}$$

因此，当格拉姆矩阵为半正定时，存在函数 $\boldsymbol{\phi}$（由特征向量 $\mathbf{U}$ 隐式定义）将样本 $\boldsymbol{x}$ 映射到相应的特征空间（维度可以是无限维），使得 $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{x}')$。

综上所述，如果一个函数能够表示为经过基函数变换后的向量内积，那么它可以直接作为核函数使用。这是因为该函数本身具有对称性（由向量内积形式保证），并且由其构造的格拉姆矩阵可以证明是半正定的，即：

$$\boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} = \sum_{ij} \alpha_i \alpha_j \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{\phi}(\boldsymbol{x}_j) = \left\| \sum_i \alpha_i \boldsymbol{\phi}(\boldsymbol{x}_i) \right\|^2 \geq 0$$

在实际应用中，我们通常无法显式构造出对应的基函数。此时，要判断一个函数是否可以作为核函数使用。我们先要证明该函数是对称的，然后再证明由它构造的格拉姆矩阵（公式4.16）是半正定的。只有在同时满足这两个条件时，该函数才能被视为有效的核函数。

接下来，我们介绍几类常用的核函数及其特点。**径向基函数**（Radial Basis Function）是一类仅依赖于输入点到某个中心点之间距离（通常使用欧氏距离）的函数，其一般形式为：

$$\phi(\|\boldsymbol{x} - \boldsymbol{u}\|) \tag{4.21}$$

其中 $\boldsymbol{u}$ 表示中心点，而 $\phi(\cdot)$ 是作用在距离上的一维函数。径向基函数最初是为了解决函数插值问题而提出来的。具体来说，给定包含 N 个样本的数据集 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$，目标是找到一个函数，在所有样本点上能够精确匹配目标值，即满足 $f(\boldsymbol{x}_i) = y_i$，其中 $i = \{1, 2, \ldots, N\}$。为实现这一目标，可以将 $f(\boldsymbol{x})$ 表示为 N 个径向基函数的线性组合，其中每个径向基函数都以某个样本为中心：

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \phi(\|\boldsymbol{x} - \boldsymbol{x}_i\|) \tag{4.22}$$

权重参数 $\{w_1, w_2, \ldots, w_N\}$ 的取值通常通过最小二乘法来确定。由于权重参数的数量与训练样本提供的拟合条件数量相等，如果假设插值矩阵（其第 i 行、第 j 列的元素为 $\phi(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$）可逆，则可以得到一个在所有训练样本上精确拟合目标值的函数。然而，在实际应用中，目标值往往包含噪声，精确插值可能导致模型对噪声的过度拟合，从而影响其泛化能力。为缓解过拟合问题，之后发展出了相应的正则化的方法和理论。若将 $\phi(\|\boldsymbol{x} - \boldsymbol{u}\|)$ 视作一种样本之间的相似度度量（其任意有限样本的格拉姆矩阵均半正定），就得到了**径向基核**（RBF Kernel）：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \phi(\|\boldsymbol{x} - \boldsymbol{x}'\|) \tag{4.23}$$

高斯核（Gaussian Kernel）则是径向基核的一种特例，因为它也仅依赖于样本与中心点（即均值）之间的距离，其具体形式为：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}')^\top \mathbf{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{x}')\right) \tag{4.24}$$

如果上式中的 $\mathbf{\Sigma}$ 为对角矩阵，且应为正定矩阵，则该式可以简化为：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2}\sum_{j=1}^{D}\frac{1}{\sigma_j^2}(x_j - x_j')^2\right) \tag{4.25}$$

其中 σ_j 表示第 j 维的尺度。如果某一维的 $\sigma_j \to \infty$，该维度对核函数值的影响趋于零，相当于忽略该维度。当所有维度的 σ_j 取相同的值 σ 时，则被称为是各向同性（Isotropic Kernel）的，其中 σ 被称为带宽（Bandwidth）：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}\right) \tag{4.26}$$

如前所述，某些核函数等价于将数据映射到无限维特征空间，这是核方法的主要优势之一。高斯核就是一个典型例子，其对应的特征向量是无限维的。为了解释这一点，下面以各向同性的核函数为例展开讨论。为简化分析，我们假设输入样本是一维的，但类似的分析可以容易地推广到多维的情形。当输入样本为一维时，公式4.26变为：

$$\kappa(x, x') = \exp\left(-\frac{(x - x')^2}{2\sigma^2}\right) \tag{4.27}$$

将上式改写为以下乘积的形式：

$$\kappa(x, x') = \exp\left(\frac{-x^2}{2\sigma^2}\right)\exp\left(-\frac{x'^2}{2\sigma^2}\right)\exp\left(\frac{xx'}{\sigma^2}\right) \tag{4.28}$$

将以上等式右侧的最后一项作泰勒级数展开：

$$\exp\left(\frac{xx'}{\sigma^2}\right) = \sum_{M=0}^{\infty}\frac{1}{M!}\left(\frac{xx'}{\sigma^2}\right)^M \tag{4.29}$$

将公式4.29代入公式4.28，可将其表示为如下的内积形式：

$$\kappa(x, x') = \sum_{M=0}^{\infty}\left(\exp\left(\frac{-x^2}{2\sigma^2}\right)\frac{x^M}{\sigma^M\sqrt{M!}}\right)\left(\exp\left(\frac{-x'^2}{2\sigma^2}\right)\frac{x'^M}{\sigma^M\sqrt{M!}}\right) \tag{4.30}$$

因此，我们可以定义如下特征映射，其对应的映射空间具有无限维度：

$$\phi(x) = \left(\exp\left(\frac{-x^2}{2\sigma^2}\right)\frac{x^0}{\sigma\sqrt{0!}}, \exp\left(\frac{-x^2}{2\sigma^2}\right)\frac{x^1}{\sigma\sqrt{1!}}, \exp\left(\frac{-x^2}{2\sigma^2}\right)\frac{x^2}{\sigma^2\sqrt{2!}}, \dots\right)^\top \tag{4.31}$$

需要注意的是，在实际的应用中，“无限维”并不会带来计算上的困难，因为我们借助“核技巧”可以直接计算 $\kappa(\boldsymbol{x}, \boldsymbol{x}')$，从而避免显式构造无限维特征向量所带来的计算和存储方面的困难。

多项式核（Polynomial Kernel）的一般形式为：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \left(a\boldsymbol{x}^\top\boldsymbol{x}' + b\right)^d \tag{4.32}$$

其中 $a \geq 0$ 为缩放系数，$b \geq 0$ 为偏置项，而 d 则表示多项式的阶数（或次数），应为非负整数。与最简单的**线性核**（即 $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^\top\boldsymbol{x}'$）相比，多项式核能够显式表示变量之间的交互作用和高阶多项式关系，例如二阶交互项 x_ix_j 或三次项 x_i^3，并且可以通过调整 d 来控制模型的表达能力。然而，当 d 取值较大或训练样本较少时，模型往往容易出现过拟合。此外，由于多项式核并非基于距离的度量，其结果对输入数据的偏移较为敏感，因此需要通过合理设置偏置项 b 进

行修正。值得注意的是，a、b 和 d 均为可调超参数，可以通过交叉验证等方法选择最优值。

采用核方法的另一优势在于，它能够从长度不一致的样本中抽取特征，进而计算它们之间的相似度。以文档分类或信息检索为例，一种朴素的想法是：如果两个文档中存在较多相同的词汇，它们就越相似，也越有可能归属于同一类别。因而，比较简单的实现是：先统计每个文档中各个词出现的频率，并且将这些词频（Term Frequency）依次排列，拼接成一个向量。然后，通过计算两个词频向量之间的**余弦相似度**（Cosine Similarity）来衡量文档间的相似程度。具体而言，我们用 $\boldsymbol{x}_i$ 表示第 i 个文档的词频向量，其中 x_{ik} 表示第 k 个词在该文档中出现的次数。在此基础上，我们利用余弦相似度的公式，文档向量 $\boldsymbol{x}_i$ 和 $\boldsymbol{x}_j$ 的核函数可表示为：

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{\|\boldsymbol{x}_i\| \|\boldsymbol{x}_j\|} = \frac{\sum_{k=1}^{D} x_{ik} \cdot x_{jk}}{\sqrt{\sum_{k=1}^{D} x_{ik}^2} \sqrt{\sum_{k=1}^{D} x_{jk}^2}} \tag{4.33}$$

其中 D 表示词表的大小。余弦相似度的取值范围通常在 $[-1, 1]$ 之间。在文档的表示中，由于向量 $\boldsymbol{x}$ 的各分量均为非负数，所以文档间的余弦相似度实际落在 $[0, 1]$ 区间。其数值越接近 1，表示两个文档越相似；当取值为 0 时，相似度则最低，即两个向量相互正交，这意味着两个文档中所出现的词汇没有交集。

实际应用中，上述简单实现存在以下三个主要问题。第一个问题是：语言中存在一些主要起语法衔接作用、但缺乏实际语义的停用词（Stop Words）。例如，汉语中的“的”、“了”和“以及”等，又如英语中的“the”、“of”和“and”等。这类词通常不会出现在文档的向量表示中，而会在构建词表时直接删除。第二个问题是，不同词汇的使用频率差异较大，且往往相差显著。为了避免低频词在计算余弦相似度时被高频词掩盖，通常会对词频进行平滑处理，一种常用方法是对词频取自然对数，从而削弱高频词的影响，同时相对提升低频词的作用。具体地，将公式4.33中的 x_{ik} 需替换为：

$$\mathrm{tf}(i, k) = \ln(1 + x_{ik}) \tag{4.34}$$

其中，上式右侧自然对数内加上 1，是为了避免当 x_{ik} 为 0 时出现 $\ln(0)$ 无定义的情况。第三个问题是，如果某些词汇在所有文档或大多数文档中都出现，则这些词对于计算相似度作用不大。然而，如果某些词只出现在一个或少数文档中，则这些词对于表示这些文档的语义（区别于其它文档）具有重要意义。因此，我们往往会计算以下逆文档频率（Inverse Document Frequency）来对词频进行缩放：

$$\mathrm{idf}(k) = \ln\left(\frac{N}{\sum_{i=1}^{N} \mathbb{I}(x_{ik} > 0)}\right) \tag{4.35}$$

其中 N 表示文档集合中文档的总数，$\mathbb{I}(\cdot)$ 是一个指示函数，它根据输入的条件判断，只有 0 或 1 两种取值。公式4.35中的 $\mathbb{I}(x_{ik} > 0)$ 表示：当第 k 个词出现在文档 i 中时，其值为 1，否则为 0。取对数操作仍是为了对原始逆文档频率进行平滑处理。上式自然对数内分数的意义在于：当第 k 个词仅出现在少数文档中时，其对应的词频权重应按比值放大，反之则会被相对缩小。“逆文档频率”这一名称正是由此而来，因为它与某个词在文档集合中出现的文档数成反比关系。结合公式4.34和公式4.35，便得到了用于文档语义向量表示的 TF-IDF（Term Frequency–Inverse Document

Frequency）方法，在利用公式4.33计算余弦相似度时，该方法会将其中的 x_{ik} 项统一替换为：

$$\text{tf-idf}(i,k) = \text{tf}(i,k) \cdot \text{idf}(k) \tag{4.36}$$

上述方法还可以推广到短语或子串的情形。具体而言，可以从输入序列中提取若干个子串的出现次数，并将这些子串的出现次数依次排列后拼接成向量，再通过向量内积来定义相应的核函数。例如，考虑由字符表 $\mathcal{A} = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}\}$ 中字符所构成的两个字符串 $\boldsymbol{x}$ 和 $\boldsymbol{x}'$：

$$\boldsymbol{x} = \texttt{ATGCGTACCTGAACTACTGAGGAATCGTGACCTGACTGGAACCTTGACGTAACCG}$$

$$\boldsymbol{x}' = \texttt{CGTACTGATAACCGGATGCTTACGGTACCGATGCTTAGGACTCA}$$

这两个字符串都包含子串 ACTGA。我们可以将两个字符串的相似性定义为它们共有的子串数量。我们用 $\phi_s(\boldsymbol{x})$ 表示子串 s 在字符串 $\boldsymbol{x}$ 中出现的次数，则可以将两个字符串 $\boldsymbol{x}$ 和 $\boldsymbol{x}'$ 之间的核函数定义为：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sum_{s \in \mathcal{A}^*} w_s \phi_s(\boldsymbol{x}) \phi_s(\boldsymbol{x}') \tag{4.37}$$

其中 $w_s \geqslant 0$ 表示子串 s 的权重，而 $\mathcal{A}^*$ 表示由字符表 $\mathcal{A}$ 构成的所有可能字符串的集合，其长度可以是任意非负整数。

在大多数情况下，我们并不需要从零开始设计全新的核函数。通常的做法是，将一些较为简单的核函数作为基本模块，再通过组合的方式构造出新的核函数。只要所选的基础函数本身是有效的核函数，那么依据以下性质和方法进行组合之后，所得到的函数依然是有效的核函数。具体而言，若已知 $\kappa_1(\boldsymbol{x}, \boldsymbol{x}')$ 和 $\kappa_2(\boldsymbol{x}, \boldsymbol{x}')$ 为核函数，则按照以下方式构造的新函数同样可以作为核函数使用：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = c\kappa_1(\boldsymbol{x}, \boldsymbol{x}') \tag{4.38}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa_1(\boldsymbol{x}, \boldsymbol{x}') + \kappa_2(\boldsymbol{x}, \boldsymbol{x}') \tag{4.39}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa_1(\boldsymbol{x}, \boldsymbol{x}')\kappa_2(\boldsymbol{x}, \boldsymbol{x}') \tag{4.40}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp(\kappa_1(\boldsymbol{x}, \boldsymbol{x}')) \tag{4.41}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = g(\kappa_1(\boldsymbol{x}, \boldsymbol{x}')) \tag{4.42}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa_3(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}')) \tag{4.43}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = f(\boldsymbol{x})\kappa_1(\boldsymbol{x}, \boldsymbol{x}')f(\boldsymbol{x}') \tag{4.44}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^\top \mathbf{A} \boldsymbol{x}' \tag{4.45}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa_a(\boldsymbol{x}_a, \boldsymbol{x}'_a) + \kappa_b(\boldsymbol{x}_b, \boldsymbol{x}'_b) \tag{4.46}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa_a(\boldsymbol{x}_a, \boldsymbol{x}'_a)\kappa_b(\boldsymbol{x}_b, \boldsymbol{x}'_b) \tag{4.47}$$

其中 $c > 0$ 为常数，$f(\cdot)$ 可以是任意函数，$g(\cdot)$ 表示一个系数非负的多项式函数。函数 $\boldsymbol{\phi}(\boldsymbol{x})$ 是将输入 $\boldsymbol{x}$ 映射到 $\mathbb{R}^M$ 空间，$\kappa_3(\cdot,\cdot)$ 则是在 $\mathbb{R}^M$ 空间中定义的一个核函数。$\mathbf{A}$ 是一个对称半正定矩阵。$\boldsymbol{x}_a$ 和 $\boldsymbol{x}_b$ 为两个变量（它们不要求互不相交），并且 $\boldsymbol{x} = (\boldsymbol{x}_a, \boldsymbol{x}_b)$。同时，$\kappa_a(\cdot,\cdot)$ 和 $\kappa_b(\cdot,\cdot)$ 分别是在各自空间上定义的有效核函数。

我们还可以使用生成模型来构造核函数。给定一个生成模型 $p(\boldsymbol{x})$，可以定义如下核函数：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = p(\boldsymbol{x})p(\boldsymbol{x}') \tag{4.48}$$

以上函数显然是一个有效的核函数，因为我们可以将其理解为由映射 $p(x)$ 定义的一维特征空间中的内积。该核函数的直观含义是：如果两个样本 $\boldsymbol{x}$ 和 $\boldsymbol{x}'$ 在生成模型下的概率值都较高，则它们被认为是相似的。借助公式4.38和公式4.39，我们可以将这类核函数推广为多个不同概率分布乘积的加权和形式，其中每个分布对应一个权重系数 $p(i)$，具体表示为：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sum_i p(\boldsymbol{x}|i)p(\boldsymbol{x}'|i)p(i) \tag{4.49}$$

其中 i 可以看成取离散值的隐变量（Latent Variable）。上式可以理解为：如果两个样本 $\boldsymbol{x}$ 和 $\boldsymbol{x}'$ 在多种不同分布下都具有较高的概率值，那么相应核函数的值就会较大，从而被认为是相似的。当生成模型（或分布）的数量趋于无限时，公式4.49可以推广为如下积分形式：

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \int p(\boldsymbol{x}|z)p(\boldsymbol{x}'|z)p(z)\, dz \tag{4.50}$$

其中 z 是一个连续的隐变量。此外，基于生成模型所构造的核函数还可以利用第7章介绍的深度神经网络，将输入样本映射为分布式向量表示（Distributed Vector Representation），并以此替代公式4.48中的 $p(\boldsymbol{x})$ 来构造新的核函数。

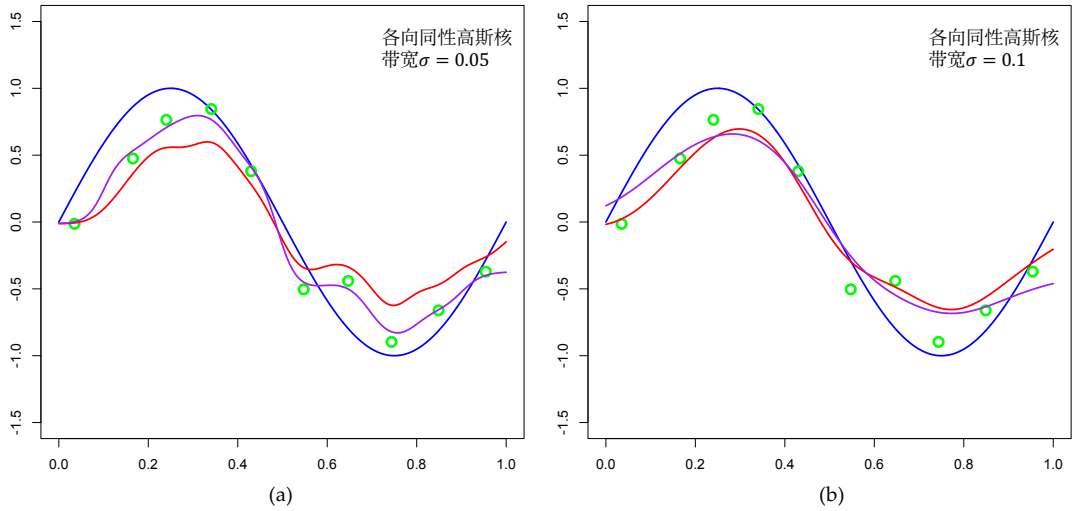


图 4.1: 核方法应用于线性回归的示例。绿色圆圈表示 10 个训练数据点，蓝色曲线表示原函数 $\sin(2\pi x)$，紫色曲线是将线性模型的权重参数直接设为训练集的实际观测值所得到的拟合函数，而红色曲线则表示使用带 ℓ_2 正则拟合得到的函数（正则化系数 $\lambda = 0.5$）。(a) 使用各向同性的高斯核，带宽 $\sigma = 0.05$ 的拟合结果。(b) 使用各向同性的高斯核，带宽 $\sigma = 0.1$ 的拟合结果。通过对比 (a) 和 (b) 两幅子图可以发现：当带宽较大时，核函数的作用范围更广，其拟合结果会受到更多邻近样本点的影响，因而拟合曲线较为平滑；反之，则拟合曲线显得更为曲折。

在介绍了一些常见的核函数之后，我们通过两个简单的例子说明如何在回归和分类任务中应用核函数。借助核技巧或核替换方法，我们可以为输入样本 $\boldsymbol{x}$ 构造如下特征向量：

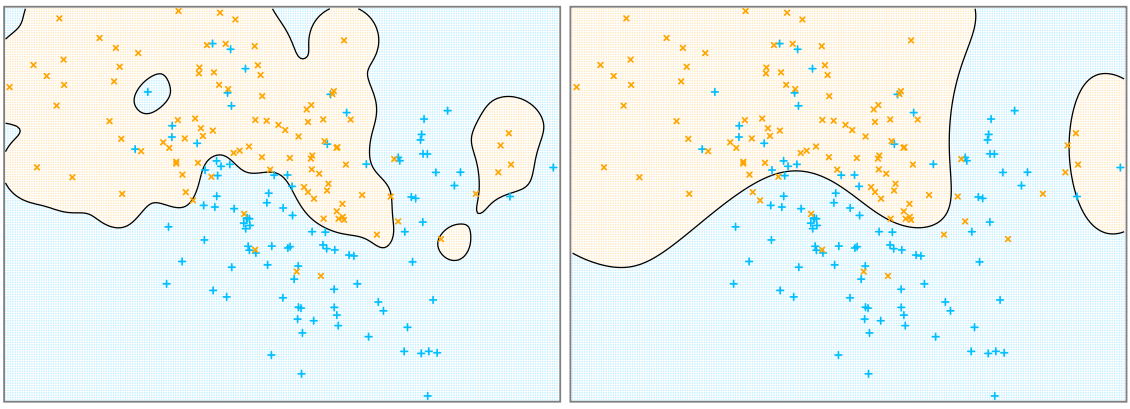
$$\boldsymbol{\phi}(\boldsymbol{x}) = (\kappa(\boldsymbol{x}, \boldsymbol{x}_1), \kappa(\boldsymbol{x}, \boldsymbol{x}_2), \ldots, \kappa(\boldsymbol{x}, \boldsymbol{x}_N))^\top \tag{4.51}$$

其中 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ 表示训练集中的样本。若采用第3.1节介绍的线性模型，则我们的目标是：在给定训练数据集 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ 的条件下，确定以下函数中的权重参数 $\boldsymbol{w}$：

$$f(\boldsymbol{x}, \boldsymbol{w}) = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{w} \tag{4.52}$$

在回归任务中，y_i 表示样本 $\boldsymbol{x}_i$ 的真实观测值。确定参数 $\boldsymbol{w}$ 的取值有两种常见方法。第一种较为简单的方法是，将第 i 个特征对应的权重 w_i 直接设为 y_i。在这种情况下，公式4.52可以直观地理解为：输入样本 $\boldsymbol{x}$ 的预测值等于以核函数计算得到的相似度作为权重来对训练集中的实际观测值进行加权求和。换言之，与输入样本 $\boldsymbol{x}$ 越相似（或距离越近）的训练样本，其观测值对 $\boldsymbol{x}$ 的预测结果影响越大。图4.1中的紫色曲线就是采用这种方法得到的拟合结果，其中权重参数 w_i 由 y_i 直接赋值，所使用的核函数为各向同性的高斯核（详见公式4.26）。第二种方法是将核函数看成一种特征变换或特征提取的手段，然后在变换之后的新特征空间中使用常规的线性回归模型。在这种情况下，还可以对权重参数 $\boldsymbol{w}$ 引入各种正则化方法以防止过拟合。图4.1中的红色曲线即是采用带 ℓ_2 正则的拟合得到的结果。

图4.1(a) 使用各向同性的高斯核，带宽设置为 $\sigma = 0.05$。图4.1(b) 也使用各向同性的高斯核，但带宽设为 $\sigma = 0.1$。通过对比两幅子图中的拟合曲线（紫色为上述方法一的结果，红色为方法二的结果），我们可以看出：当带宽较大时，核函数的作用范围更广，当前样本的拟合结果会受到更多邻近样本点的影响，因此曲线较为平滑；而带宽较小时，核函数仅关注局部的邻近样本，拟合曲线则明显更加曲折。核函数的带宽是一个重要的超参数，可以通过第1.6节介绍的交叉验证或使用独立验证集来选择合适的取值。



(a) (b)

图 4.2: 核方法应用于线性分类的示例。蓝色加号和橙色叉号分别表示两类样本，黑色曲线表示使用带 ℓ_2 正则的 Logistic 回归得到的分类决策边界（正则化系数 $\lambda = 0.5$）。(a) 使用各向同性的高斯核，带宽 $\sigma = 0.2$ 的结果。(b) 使用各向同性的高斯核，带宽 $\sigma = 0.5$ 的结果。通过对比 (a) 和 (b) 两幅子图可以看出：与回归任务类似，当带宽较大时，核函数的作用范围更广，决策边界会受到更多周围样本点的影响，因此曲线较为平滑；而带宽较小时，核函数只关注局部邻近样本，决策边界变得更加曲折和局部化。

对于分类任务，我们需要在线性函数 $f(\boldsymbol{x}, \boldsymbol{w})$ 的基础上引入一个非线性变换函数，将线性函数的输出进一步映射到 $[0, 1]$ 区间，以确保结果可以被解释为有效的概率值。若采用 Sigmoid

函数作为非线性映射，即得到第3.6节所介绍的 Logistic 回归模型。与回归模型类似，我们同样可以在分类模型中对参数 $\boldsymbol{w}$ 施加正则化约束，以控制模型复杂度并提升泛化能力。图4.2为核方法应用于线性分类的例子，其中核函数采用各向同性的高斯核，其带宽大小对分类决策边界的影响与回归任务中的情况相似。值得注意的是：当采用 ℓ_1 正则化时，模型参数会呈现出稀疏化特性，即部分样本点（对应的 w_i 取值为零）在预测过程中不再被使用。这不仅减少了预测阶段需要存储或“记忆”的样本数量，同时也在一定程度上提升了预测时的计算效率。

4.3 高斯过程

在监督学习中，我们通过观测 N 个输入 $\boldsymbol{x}$ 与对应输出 y 之间的关系，来学习一个映射函数 $f(\boldsymbol{x})$。在此前的讨论中，我们通常假设映射函数 $f(\cdot)$ 具有某种预先设定的形式（例如线性或二次型），并完全由一组参数确定，因此学习问题的核心就转化为对这些参数的估计。另一种思路是不再预先假设某个具体的参数化模型，而是把函数看作一个“随机对象”，假设它在函数空间中服从某个先验分布。然后，我们通过观测数据来更新这个分布，得到函数的后验分布，并用它对新的输入进行预测。乍看之下，在无限维的函数空间中直接定义分布几乎是不可能的。然而，实际情况却比这简单得多。对于有限规模的训练数据，我们只关心函数在这些离散输入点上的取值，而不必描述函数的具体形式。因此，这一问题可以有效地转化为在有限维空间中进行处理，从而避免了在整个无限函数空间上直接建模的困难。

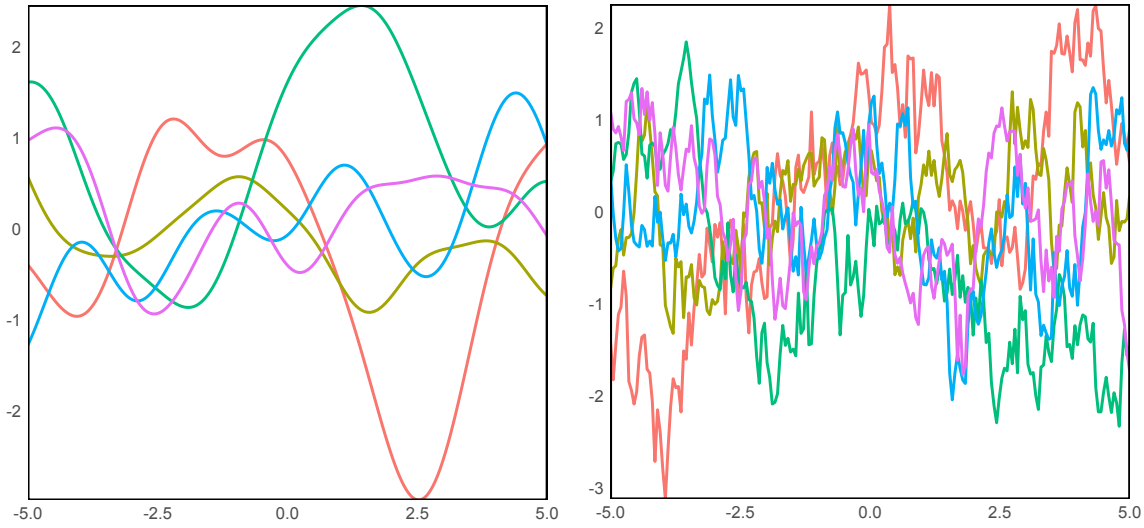
高斯过程（Gaussian Process）假设任意有限个输入点上的函数值的联合分布服从多维高斯分布（又称多元高斯分布）。满足这一假设的函数并非唯一，因此高斯过程实际上是对整个函数的分布进行约束，而具体的某一函数可以通过从该分布中采样得到。需要特别注意的是，高斯过程生成的函数并不是像单峰高斯钟形那样的曲线。实际上，高斯过程可以生成各种形状的函数：可以是平滑的直线、起伏的正弦波，甚至非常崎岖的曲线（如图4.3所示）。所生成的函数的形状与单个高斯分布的钟形并无直接关系。高斯过程的核心在于：虽然每个特定输入点的输出都是随机的，但任意多个输入点对应的函数值作为整体服从多维高斯分布。举个例子：假设我们要预测明天每个整点的温度。如果只看下午 2 点的温度，它可能服从一个一维高斯分布。但如果我们希望预测全天每个整点的温度，就必须考虑不同时间点之间的相关性。例如，若下午 1 点很热，那么下午 2 点大概率也很热，它们不能被看作是相互独立的随机变量。高斯过程正是用于刻画这种结构化依赖关系的工具。它不仅要求每个整点温度服从边际高斯分布，更重要的是，它定义了所有整点温度的联合分布是一个多维高斯分布，其中每个维度对应一个整点温度的随机变量。这个联合分布通过一个协方差矩阵描述了任意两个整点温度之间的相关性。

高斯过程与核函数的关系在于：高斯过程中的多维高斯分布的协方差矩阵即为核函数所构造的格拉姆矩阵。核函数通过衡量任意两个输入点之间的相似性，进而确定它们对应输出值之间的相关性。通过选择不同的核函数，我们可以灵活地定义协方差矩阵，从而调节函数在不同输入点之间的平滑度和相关性。图4.3展示了在两种不同核函数下，从高斯过程中采样得到的函数示例。第一种核函数是式4.26所示的各向同性的高斯核（左子图），第二种核函数为指数核（右

子图），其形式如下：

$$\kappa(x, x') = \exp\left(-\frac{|x - x'|}{\ell}\right) \tag{4.53}$$

其中 $\ell > 0$ 被称为长度尺度（Length Scale），其数值决定函数的平滑程度。ℓ 的值越大，相关性衰减越慢，函数变化越平滑。指数核在两个输入重合处（$x = x'$）存在不可微的尖点，因此其对应的高斯过程不具有均方可微性，生成的样本函数通常较为粗糙（存在尖锐的转折）。均方可微性是指随机过程在输入变化趋于零时，其变化率能够收敛到一个随机变量，并且两者之间的均方误差趋于零。相比之下，高斯核采用距离的平方构造，具有更高的光滑性，其对应的高斯过程具有任意阶的均方导数，因此生成的样本函数通常更加平滑。



(a) (b)

图 4.3: 从高斯过程采样得到的函数示例。(a) 使用高斯核。(b) 使用指数核。通过对比子图 (a) 和 (b) 可看出：高斯核生成的函数曲线明显比指数核更平滑。这是因为高斯核在输入空间中对相似性的度量随距离平方增加呈更为平滑的负指数衰减，而指数核的相似性随距离变化更为剧烈，导致其采样结果呈现出较强的局部波动。指数核在两个输入重合处（$x = x'$）存在不可微的尖点，因此其对应的高斯过程不具有均方可微性，生成的样本函数通常存在较多的尖锐转折。这些随机生成的、且函数值均满足高斯先验的函数构成了候选空间，之后通过引入观测数据来从这些函数中筛选出符合观测约束的函数来对新样本进行预测。

我们先讨论如何将高斯过程用于回归任务。高斯过程假设 $p(f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N))$ 服从以 $\boldsymbol{\mu}(\boldsymbol{x})$ 为均值、$\boldsymbol{\Sigma}(\boldsymbol{x})$ 为协方差的多维高斯分布。协方差矩阵的每一个元素由 $\Sigma_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ 给出，即该协方差矩阵正是由式4.16定义的格拉姆矩阵 $\mathbf{K}$。为了便于后续分析，我们假设 $f(\boldsymbol{x})$ 经过了中心化处理，即其均值为 0，因此有 $\boldsymbol{\mu}(\boldsymbol{x}) = \mathbf{0}$。

之所以上述假设是合理的，是因为它相当于在线性模型 $f(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{w}$ 中，对权重参数 $\boldsymbol{w}$ 引入如下的先验分布：

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) \tag{4.54}$$

其中 α 是高斯分布的精度。对于包含 N 个样本的数据集 $\mathcal{D}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，观察值 y_i 往往包含噪声。若假设观测值中叠加了均值为零、精度为 β 的高斯噪声，则真实目标值 t_i 与观测值 y_i 的关系可表示为：

$$p(\boldsymbol{y}|\boldsymbol{t})=\mathcal{N}(\boldsymbol{y}|\boldsymbol{t},\beta^{-1}\mathbf{I}) \tag{4.55}$$

其中 $\boldsymbol{t}=(t_1,\ldots,t_N)^\top$ 表示由所有真实目标值构成的向量，而 $\boldsymbol{y}=(y_1,\ldots,y_N)^\top$ 则表示对应的观测值向量。我们来分析所有真实目标值（即理想的映射函数的输出向量）$\boldsymbol{t}$ 的联合概率分布。根据线性模型的定义，输出向量 $\boldsymbol{t}$ 可以写作：

$$\boldsymbol{t}=\boldsymbol{\Phi}\boldsymbol{w} \tag{4.56}$$

其中 $\boldsymbol{\Phi}$ 为公式4.5所示的设计矩阵。由于假设权重参数 $\boldsymbol{w}$ 服从高斯先验，并且由公式4.56可知，向量 $\boldsymbol{t}$ 是高斯随机变量的线性组合，因此 $\boldsymbol{t}$ 也服从高斯分布，其均值和协方差矩阵可计算如下：

$$\mathbb{E}[\boldsymbol{t}]=\mathbb{E}[\boldsymbol{\Phi}\boldsymbol{w}]=\boldsymbol{\Phi}\mathbb{E}[\boldsymbol{w}]=\mathbf{0} \tag{4.57}$$

$$\mathrm{Cov}[\boldsymbol{t}]=\mathbb{E}[\boldsymbol{t}\boldsymbol{t}^\top]=\boldsymbol{\Phi}\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top]\boldsymbol{\Phi}^\top=\alpha^{-1}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top=\alpha^{-1}\mathbf{K} \tag{4.58}$$

其中式4.58中的常数 α^{-1} 可以合并到核函数中。由此，输出向量 $\boldsymbol{t}$ 的分布可表示为：

$$p(\boldsymbol{t})=\mathcal{N}(\boldsymbol{t}|\mathbf{0},\mathbf{K}) \tag{4.59}$$

从以上公式可以看出，若假设样本的输出值（即真实目标值）$\boldsymbol{t}$ 是由高斯过程生成的，这本质上相当于假设线性模型的参数 $\boldsymbol{w}$ 服从均值为 0 的高斯先验。

我们通常无法直接获得真实的目标值，而只能通过观测值对其进行推断。因而，需要了解观测值的概率分布 $p(\boldsymbol{y})$。我们通过将联合分布 $p(\boldsymbol{y},\boldsymbol{t})$ 对 $\boldsymbol{t}$ 积分来得到观测值的边缘分布 $p(\boldsymbol{y})$：

$$p(\boldsymbol{y})=\int p(\boldsymbol{y},\boldsymbol{t})\,d\boldsymbol{t}=\int p(\boldsymbol{y}|\boldsymbol{t})p(\boldsymbol{t})\,d\boldsymbol{t}=\mathcal{N}(\boldsymbol{y}|\mathbf{0},\mathbf{C}) \tag{4.60}$$

其中协方差矩阵 $\mathbf{C}$ 的每一个元素为：

$$c_{ij}=k_{ij}+\beta^{-1}\delta_{ij}=\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)+\beta^{-1}\delta_{ij} \tag{4.61}$$

其中 k_{ij} 是格拉姆矩阵 $\mathbf{K}$ 的相应元素，而 $\delta_{ij}=\mathbb{I}(i=j)$ 对应于每个样本观测值中所包含的独立随机噪声（即 δ_{ij} 在协方差矩阵的对角线上为 1，其他位置为 0）。上述结果不难理解。由于两个高斯随机向量相互独立，它们之和的协方差矩阵等于各自协方差矩阵之和。

至此，我们已经得到了观测值的边缘分布 $p(\boldsymbol{y})$。但是，我们的目标是在给定一个新的输入样本 $\boldsymbol{x}_{N+1}$ 的情况下，对其观测值 y_{N+1} 的分布进行预测，并且这一预测需要基于此前观测到的 N 个样本。为此，需要进一步推导条件概率分布 $p(y_{N+1}|\boldsymbol{y})$。需要注意的是，该分布同时以 $(x_1,\ldots,x_N)$ 和 x_{N+1} 作为条件。然而，为了保持符号简洁，这些条件变量在此不作显式表示。根据公式4.60，当增加一个新样本 $\boldsymbol{x}_{N+1}$ 后，观测值的联合分布（先验分布）可表示为：

$$p(\boldsymbol{y}_{N+1})=\mathcal{N}(\boldsymbol{y}_{N+1}|\mathbf{0},\mathbf{C}_{N+1}) \tag{4.62}$$

其中 $\boldsymbol{y}_{N+1}=(y_1,\ldots,y_N,y_{N+1})^\top$，而 $\mathbf{C}_{N+1}$ 是 $(N+1)\times(N+1)$ 的协方差矩阵，其每一个元素按公式4.61计算。根据第2.5节的讨论可知，如果两组变量服从联合高斯分布，那么在已知

其中一组变量的条件下，另一组变量的条件概率分布仍然是高斯分布。为了推导条件概率分布 $p(y_{N+1}|\boldsymbol{y})$ 的具体形式，我们将协方差矩阵 $\mathbf{C}_{N+1}$ 用如下分解形式表示：

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \boldsymbol{k} \\ \boldsymbol{k}^\top & c \end{bmatrix} \tag{4.63}$$

其中 $\mathbf{C}_N$ 为原 N 个样本对应的协方差矩阵（即公式4.60中的 $\mathbf{C}$），向量 $\boldsymbol{k}$ 的元素为 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_{N+1})$，其中 $i \in \{1, \ldots, N\}$，而标量 $c = \kappa(\boldsymbol{x}_{N+1}, \boldsymbol{x}_{N+1}) + \beta^{-1}$。从第2.5节的讨论已知，给定多维高斯分布 $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$，将随机变量 $\boldsymbol{x}$ 任意划分成 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 两组。给定 $\boldsymbol{x}_1$ 时，$\boldsymbol{x}_2$ 的条件分布为：

$$p(\boldsymbol{x}_2|\boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{x}_2|\boldsymbol{\mu}_{2|1}, \boldsymbol{\Sigma}_{2|1}) \tag{4.64}$$

$$\boldsymbol{\mu}_{2|1} = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1) \tag{4.65}$$

$$\boldsymbol{\Sigma}_{2|1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12} \tag{4.66}$$

以上公式其实是公式2.114和2.115的另一种表达形式，只是将原先公式中出现的协方差矩阵的逆 $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ 改为使用协方差矩阵进行表示。将上述公式中的 $\boldsymbol{\Sigma}$ 用公式4.63中的 $\mathbf{C}_{N+1}$ 代入（其中 $\mathbf{C}_N$ 对应 $\boldsymbol{\Sigma}_{11}$，$\boldsymbol{k}$ 对应 $\boldsymbol{\Sigma}_{12}$，$\boldsymbol{k}^\top$ 对应 $\boldsymbol{\Sigma}_{21}$，c 对应 $\boldsymbol{\Sigma}_{22}$），我们可以得到条件概率分布 $p(y_{N+1}|\boldsymbol{y})$（即通过条件化引入观测信息 $\boldsymbol{y}$，并将其与先验协方差矩阵 $\mathbf{C}_{N+1}$ 相结合，从而得到预测目标的后验分布）：

$$p(y_{N+1}|\boldsymbol{y}) = \mathcal{N}(y_{N+1}|\mu(\boldsymbol{x}_{N+1}), \sigma^2(\boldsymbol{x}_{N+1})) \tag{4.67}$$

$$\mu(\boldsymbol{x}_{N+1}) = \mathbf{0} + \boldsymbol{k}^\top\mathbf{C}_N^{-1}(\boldsymbol{y} - \mathbf{0}) = \boldsymbol{k}^\top\mathbf{C}_N^{-1}\boldsymbol{y} \tag{4.68}$$

$$\sigma^2(\boldsymbol{x}_{N+1}) = c - \boldsymbol{k}^\top\mathbf{C}_N^{-1}\boldsymbol{k} \tag{4.69}$$

以上公式中需要对 $\mathbf{C}_N$ 求逆，因此要求 $\mathbf{C}_N$ 为正定矩阵，即其所有特征值均大于零。若 λ_i 是原格拉姆矩阵 $\mathbf{K}$ 的特征值，则对应的矩阵 $\mathbf{C}_N$ 的特征值为 $\lambda_i + \beta^{-1}$。因此，只要原格拉姆矩阵 $\mathbf{K}$ 是半正定的（即 $\lambda_i \geq 0$），并且噪声精度 $\beta > 0$，就可以保证 $\mathbf{C}_N$ 为正定矩阵（即 $\lambda_i + \beta^{-1} > 0$）。即使在不考虑观测噪声的情况下，为了确保 $\mathbf{C}_N$ 可逆以及计算逆矩阵的数值稳定性，通常也会在原矩阵中加入一个微小的、精度为正的随机噪声项。从公式4.68可以看出，新样本的预测均值本质上是对已知样本观测值 $\boldsymbol{y}$ 的加权线性组合。各样本的权重由新样本与已知样本间的核函数值（即相似度）确定。相似度越高，对应观测值对预测结果的贡献越大；相似度越低，则贡献则越小。此外，已知样本之间的协方差矩阵的逆 $\mathbf{C}_N^{-1}$ 起到“去冗余”的作用，即当两个已知样本彼此较为相似时，其贡献不会被简单叠加，而是会根据其相关性自动降低在预测中的贡献。

我们同样可以使用核技巧，选择合适的核函数来构建高斯过程所需的格拉姆矩阵。例如，在将高斯过程用于回归任务中，一种常见的核函数形式是对二次型取指数，并在此基础上再加上线性项和常数项：

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0 \exp\left(-\frac{\theta_1}{2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right) + \theta_2\boldsymbol{x}_i^\top\boldsymbol{x}_j + \theta_3 \tag{4.70}$$

其中 $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2, \theta_3)^\top$ 均为超参，其最优取值可借助数据集来确定，通常要求 $\theta_0, \theta_1, \theta_2, \theta_3 \geq 0$。

根据公式4.60，我们有：

$$\ln p(\boldsymbol{y}|\boldsymbol{\theta}) = \ln \mathcal{N}(\boldsymbol{y}|\mathbf{0}, \mathbf{C}_N) = -\frac{1}{2}\boldsymbol{y}^\top \mathbf{C}_N^{-1}\boldsymbol{y} - \frac{1}{2}\ln|\mathbf{C}_N| - \frac{N}{2}\ln(2\pi) \tag{4.71}$$

将上式对任一超参 θ_i 求偏导，可得：

$$\nabla_{\theta_i} \ln p(\boldsymbol{y}|\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{y}^\top \mathbf{C}_N^{-1}\nabla_{\theta_i}\mathbf{C}_N\mathbf{C}_N^{-1}\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\left(\mathbf{C}_N^{-1}\nabla_{\theta_i}\mathbf{C}_N\right) \tag{4.72}$$

$$= \frac{1}{2}\mathrm{tr}\left((\boldsymbol{z}\boldsymbol{z}^\top - \mathbf{C}_N^{-1})\nabla_{\theta_i}\mathbf{C}_N\right) \tag{4.73}$$

其中 $\boldsymbol{z} = \mathbf{C}_N^{-1}\boldsymbol{y}$。利用以上偏导公式，我们可以采用任意标准的基于梯度的优化算法来估计超参数的取值。但需要注意的是，由于优化目标并非凸函数，因此可能存在多个局部最优解。为此，我们可以借助验证集，对不同的超参数组合（每一超参可能有多种取值）进行评估，并选择在验证集上表现最佳的组合。

接下来，我们将简要讨论如何将高斯过程应用于分类任务。对于分类任务，为了赋予模型的预测值相应的概率解释，其值需要落入 $[0,1]$ 之间。实现这一点的常见方法是：先令函数输出为 $a = f(\boldsymbol{x})$，再通过 Sigmoid 函数将其映射到 $(0,1)$ 区间，从而得到 $y = \sigma(a)$，其中 $\sigma(z) = 1/(1+\exp(-z))$。

以二分类问题为例，设有一个包含 N 个样本的数据集 $\mathcal{D} = \{(\boldsymbol{x}_i, t_i)\}_{i=1}^N$，其中类别标签 $t_i \in \{0,1\}$，即只能取 0 或 1 两种可能值。与回归问题类似，我们的目标是在给定一个新的输入样本 $\boldsymbol{x}_{N+1}$ 的情况下，对其类别标签 t_{N+1} 进行预测。具体而言，就是构建预测分布 $p(t_{N+1}|\boldsymbol{t})$，其中 $\boldsymbol{t} = (t_1, \ldots, t_N)^\top$ 表示训练集中已观测到的类别标签。同样地，该分布同时以 $(x_1, \ldots, x_N)$ 和 x_{N+1} 作为条件。为了保持符号简洁，这些条件变量不在预测分布中显式表示。

我们假设函数输出值向量 $\boldsymbol{a}_{N+1} = (a_1, \ldots, a_N, a_{N+1})^\top$ 服从如下多维高斯分布：

$$p(\boldsymbol{a}_{N+1}) = \mathcal{N}(\boldsymbol{a}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1}) \tag{4.74}$$

其中协方差矩阵 $\mathbf{C}_{N+1}$ 的元素定义为：

$$c_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + \epsilon\delta_{ij} \tag{4.75}$$

其中 ϵ 是一个预先设定的超参，而 $\delta_{ij} = \mathbb{I}(i=j)$。尽管在分类任务中通常假设所有训练样本的标签都是正确的，但在协方差矩阵中引入类似噪声的修正项 $\epsilon\delta_{ij}$，可以保证该矩阵是正定的。在引入上述高斯过程先验后，经过 Sigmoid 函数映射，可以在 $\boldsymbol{t}_{N+1} = (t_1, \ldots, t_N, t_{N+1})^\top$ 上诱导出一个非高斯过程（Non-Gaussian Process）。虽然 $\boldsymbol{t}_{N+1}$ 的取值分布不再服从高斯分布，但取值之间的相关性仍受到原多维高斯分布的约束。

对于二分类问题，只需预测 $p(t_{N+1}=1|\boldsymbol{t})$，因为 $p(t_{N+1}=0|\boldsymbol{t})$ 的值等于 $1-p(t_{N+1}=1|\boldsymbol{t})$。相应的预测分布可表示为：

$$p(t_{N+1}=1|\boldsymbol{t}) = \int p(t_{N+1}=1|a_{N+1})p(a_{N+1}|\boldsymbol{t})\, da_{N+1} \tag{4.76}$$

其中 $p(t_{N+1}=1|a_{N+1})=\sigma(a_{N+1})$，而条件分布 $p(a_{N+1}|\boldsymbol{t})$ 可推导如下：

$$\begin{aligned}
p(a_{N+1}|\boldsymbol{t}) &= \int p(a_{N+1},\boldsymbol{a}|\boldsymbol{t})\,d\boldsymbol{a} \quad (\boldsymbol{a}=(a_1,\ldots,a_N)^\top) \\
&= \int \frac{p(a_{N+1},\boldsymbol{a},\boldsymbol{t})}{p(\boldsymbol{t})}\,d\boldsymbol{a} \\
&= \frac{1}{p(\boldsymbol{t})}\int p(a_{N+1},\boldsymbol{a})p(\boldsymbol{t}|a_{N+1},\boldsymbol{a})\,d\boldsymbol{a} \\
&= \frac{1}{p(\boldsymbol{t})}\int p(a_{N+1}|\boldsymbol{a})p(\boldsymbol{a})p(\boldsymbol{t}|\boldsymbol{a})\,d\boldsymbol{a} \quad (\text{条件概率 } p(\boldsymbol{t}|a_{N+1},\boldsymbol{a}) \text{ 与 } a_{N+1} \text{ 无关}) \\
&= \int p(a_{N+1}|\boldsymbol{a})p(\boldsymbol{a}|\boldsymbol{t})\,d\boldsymbol{a} \quad (p(\boldsymbol{a})p(\boldsymbol{t}|\boldsymbol{a})=p(\boldsymbol{a},\boldsymbol{t})=p(\boldsymbol{a}|\boldsymbol{t})p(\boldsymbol{t}))
\end{aligned} \tag{4.77}$$

根据公式4.68和4.69，上式中的 $p(a_{N+1}|\boldsymbol{a})=\mathcal{N}(a_{N+1}|\boldsymbol{k}^\top\mathbf{C}_N^{-1}\boldsymbol{a}, c-\boldsymbol{k}^\top\mathbf{C}_N^{-1}\boldsymbol{k})$。然而，上式中的另一项 $p(\boldsymbol{a}|\boldsymbol{t})$ 需要通过拉普拉斯近似（Laplace Approximation）或变分推断（Variational Inference）进行近似求解。这一近似过程涉及到本书后续章节将要介绍的内容，此处我们只需要理解整个推导思路即可。希望深入了解细节的读者，可以参考文献 [53] 第 15 章的相关内容。

4.4 支持向量机原理

核方法是属于基于记忆的学习方法，其主要不足之处在于：在预测阶段，每当输入一个新的样本 $\boldsymbol{x}$，都需要计算它与训练集中所有样本 $\boldsymbol{x}'$ 之间的核函数值 $\kappa(\boldsymbol{x},\boldsymbol{x}')$，这往往会带来较高的计算开销。**支持向量机**（Support Vector Machine）是核方法的一种稀疏化形式，它并不会保留训练集中的所有样本，而是仅保留对于预测有帮助的部分样本，即支持向量对应的样本。对于分类问题而言，支持向量就是那些位于决策边界附近[1]、用于确立分类超平面的样本。

我们从比较简单的二分类问题入手，并以线性模型为起点，然后引出并展开对支持向量机基本原理的讨论。给定包含 N 个样本的数据集 $\mathcal{D}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，我们构建如下形式的二分类线性模型：

$$f(\boldsymbol{x})=\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{w}+w_0 \tag{4.78}$$

其中 $\boldsymbol{\phi}(\boldsymbol{x})$ 表示将输入样本 $\boldsymbol{x}$ 通过一组基函数变换后得到的特征向量，而 $\boldsymbol{w}$ 和 w_0 分别表示权重参数与偏置项。在该模型中，分类标签的取值为 $y_i\in\{-1,1\}$，即仅可能为 -1 或 1 这两种值。采用这种取值方式的原因在于，此处无需对线性模型的输出赋予概率解释，同时可以通过 $f(\boldsymbol{x})$ 的符号（大于或小于零）来简单地判断输入样本 $\boldsymbol{x}$ 的类别。

我们暂时假设训练数据在特征空间中是线性可分的。也就是说，至少存在一组参数 $\boldsymbol{w}$ 和 w_0 使得形如4.78的线性模型满足以下条件：对于类别标签为 $y_i=1$ 的样本，有 $f(\boldsymbol{x}_i)>0$；对于类别标签为 $y_i=-1$ 的样本，有 $f(\boldsymbol{x}_i)<0$。因此，对于所有训练样本，都满足 $y_if(\boldsymbol{x}_i)>0$。为了确定一组合适的参数 $\boldsymbol{w}$ 和 w_0，使得模型 $f(\boldsymbol{x})$ 对于所有样本都满足 $y_if(\boldsymbol{x}_i)>0$ 的条件，支持向量机引入了间隔（Margin）的概念。间隔被定义为决策边界（如图4.4(a) 中的红色直线所示）

[1]硬间隔支持向量位于间隔边界上，而软间隔支持向量还可能位于间隔内部或被误分类（详见后文介绍）。

到所有样本点的最小距离。决策边界的选择原则是最大化间隔，从而使不同类别的样本能够更清晰地被区分开来。

由图3.7已知，权重向量 $\boldsymbol{w}$ 垂直于决策边界（即 $\boldsymbol{w}$ 为决策边界的法向量）。对于任意样本 $\boldsymbol{x}$，其到决策边界的带符号垂直距离为 $\frac{f(\boldsymbol{x})}{\|\boldsymbol{w}\|}$。由于假设训练样本是线性可分的，并且对所有训练样本均有 $y_i f(\boldsymbol{x}_i) > 0$，因此任意样本到决策边界的垂直距离可以表示为：

$$\frac{y_i f(\boldsymbol{x}_i)}{\|\boldsymbol{w}\|} = \frac{y_i(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0)}{\|\boldsymbol{w}\|} \tag{4.79}$$

间隔被定义为训练样本中距离决策边界最近样本 $\boldsymbol{x}_i$ 的垂直距离。距离决策边界最近的样本可以有多个（如图4.4(a) 中由红圈标出的样本点）。对于二分类问题，至少每个类别各有一个这样的样本。我们的目标是通过调整参数 $\boldsymbol{w}$ 和 w_0 来最大化该最小垂直距离。因此，求解最大间隔的决策边界问题可以表述为以下优化问题：

$$\arg\max_{\boldsymbol{w},w_0} \left(\frac{1}{\|\boldsymbol{w}\|} \min_i \left[y_i(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0) \right] \right) \tag{4.80}$$

在以上优化问题中，内层的最小化用于选出距离决策边界最近的样本，而外层的最大化则旨在使这一最小距离（即间隔）尽可能大。需要注意的是，我们将因子 $\frac{1}{\|\boldsymbol{w}\|}$ 从内层的最小化中提取出来，这一处理并不会改变最终的最优解。其原因在于，权重向量 $\boldsymbol{w}$ 的模长与具体的样本无关。若同时对权重向量和偏置项（也与具体样本无关）进行缩放，即令 $\boldsymbol{w} \to c\boldsymbol{w}$ 和 $w_0 \to cw_0$，那么任意样本到决策边界的距离保持不变：

$$\frac{y_i(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top c\boldsymbol{w} + cw_0)}{\|c\boldsymbol{w}\|} = \frac{cy_i(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0)}{c\|\boldsymbol{w}\|} = \frac{y_i(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0)}{\|\boldsymbol{w}\|} \tag{4.81}$$

其中 c 是任何大于零的常数项。利用这一性质，我们可以通过适当的缩放，使得距离决策边界最近的样本满足如下条件：

$$y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) = 1 \tag{4.82}$$

这相当于要求所有训练样本都满足以下不等式约束：

$$y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) \geq 1 \tag{4.83}$$

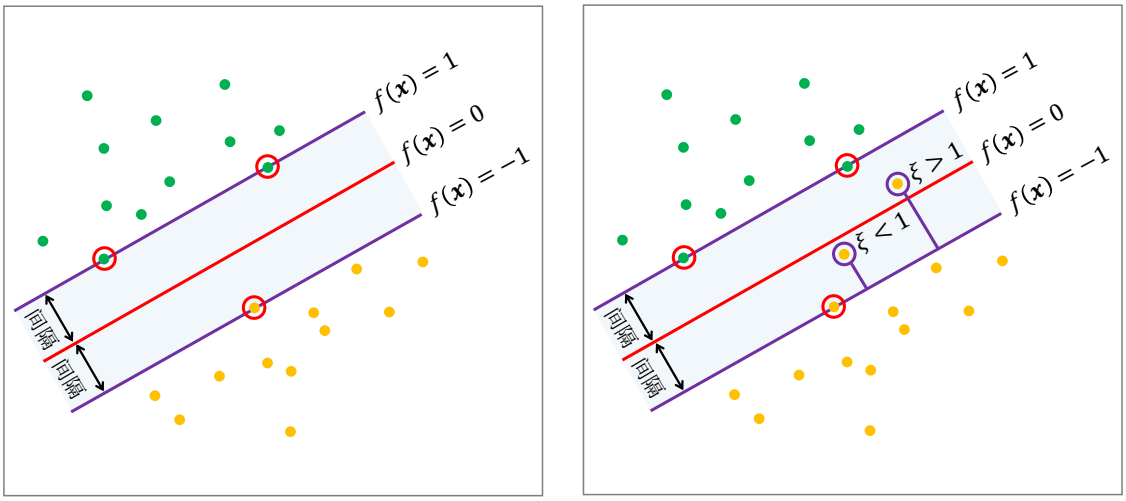
即对于那些距离决策边界较远的样本，其公式4.83左侧的值会更大。对于满足等号的那些样本（即支持向量），其对应的约束称为活跃约束（Active Constraint）；而对于其余的样本，其约束则称为非活跃约束（Inactive Constraint）。

直接求解公式4.80中的优化问题往往较为复杂。借助上述推导得到的不等式约束，我们可以将其转化为一个带约束的二次规划问题（Quadratic Programming Problem）。由于最大化 $\|\boldsymbol{w}\|^{-1}$ 等价于最小化 $\|\boldsymbol{w}\|^2$，因此公式4.80所示的优化问题可以改写为：

$$\arg\min_{\boldsymbol{w},w_0} \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{4.84}$$

$$\text{s.t. } y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) \geq 1,\ i = 1, \ldots, N \tag{4.85}$$

其中 “s.t.” 是 “受限于” 的意思（即英文 Subject to 的缩写），表示后续的条件为优化问题的约束。



(a) (b)

图 4.4: 支持向量机确定决策边界的示意图。图中红线表示决策边界，带红圈的样本点表示用于确定该边界的支持向量。(a) 两类样本完全可分的情况。间隔定义为距离决策边界最近样本点到决策边界的垂直距离。通过最大化间隔，我们可以得到一个唯一的最优决策边界，这一边界只由部分样本所确定，这些样本即为支持向量。(b) 两类样本不完全可分的情况：部分样本可能被错分或落入间隔内，因此引入的松弛变量 $\xi \geq 0$ 来衡量违反约束的程度。

在目标函数中引入系数 $\frac{1}{2}$ 是为了简化后续的推导过程，本身并不会改变最优解。需要注意的是，虽然偏置项 w_0 并未显式出现在目标函数中，但它通过约束条件间接影响权重参数 $\boldsymbol{w}$ 的最优解。公式4.85所示的不等式约束可以改写为以下标准形式：

$$1 - y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) \leq 0 \tag{4.86}$$

为了求解公式4.84所示的带约束的优化问题，我们为每个样本对应的约束条件（见上式）分别引入一个拉格朗日乘子 $\mu_i \geq 0$，从而可以将原问题转化为如下的拉格朗日形式：

$$\begin{aligned} \mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\mu}) &= \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{N} \mu_i \left(1 - y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) \right) \\ &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \mu_i \left(y_i (\boldsymbol{\phi}\left(\boldsymbol{x}_i \right)^\top \boldsymbol{w} + w_0 \right) - 1 \right) \end{aligned} \tag{4.87}$$

上述优化问题还必须满足 Karush-Kuhn-Tucker（KKT）条件，这要求同时满足以下三个条件：

$$\mu_i \geq 0 \tag{4.88}$$

$$y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) - 1 \geq 0 \tag{4.89}$$

$$\mu_i \left(y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) - 1 \right) = 0 \tag{4.90}$$

其中 $i = 1, \ldots, N$。公式4.88表示拉格朗日乘子的定义域约束，公式4.89对应于公式4.85所示的原始约束条件，而公式4.90则体现了互补松弛原则：对于每一个样本，要么 $y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0 \right) = 1$（即该样本为支持向量）；要么对应的拉格朗日乘子 $\mu_i = 0$（即该约束不产生实质性影响）。满足

这些约束条件对于支持向量机的实际应用至关重要。它们确保在模型训练完成之后，部分训练样本（即乘子为零 $\mu_i = 0$ 的样本）可以被舍弃，只需保留支持向量即可进行预测，从而显著提高预测时的计算效率。

我们将公式4.87的目标函数 $\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\mu})$ 分别对 $\boldsymbol{w}$ 和 w_0 求偏导，并令偏导数为零，可以得到以下等式（之前引入的系数 $\frac{1}{2}$ 就是为了在此处求导时简化计算）：

$$\boldsymbol{w} = \sum_{i=1}^{N} \mu_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i) \tag{4.91}$$

$$0 = \sum_{i=1}^{N} \mu_i y_i \tag{4.92}$$

以上等式给出了公式4.87所示的目标函数 $\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\mu})$ 取得极值时所需满足的驻点条件。为了将等式4.91代回到公式4.87中，我们先将公式4.87展开为如下形式：

$$\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} - \sum_{i=1}^{N} \mu_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} - w_0 \sum_{i=1}^{N} \mu_i y_i + \sum_{i=1}^{N} \mu_i \tag{4.93}$$

由于等式4.92，因而上式中包含 w_0 的项为零。同时，根据等式4.91，上式中的 $\boldsymbol{w}^\top \boldsymbol{w}$ 可以表示为 $\sum_{i=1}^{N} \mu_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w}$（正好等于上式右侧的第二项）。我们将等式4.91代入公式4.87之后，可以得到原优化问题的对偶形式。该对偶问题以最大化为目标，其表达式如下：

$$\mathcal{J}(\boldsymbol{\mu}) = \sum_{i=1}^{N} \mu_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \mu_i \mu_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4.94}$$

$$\text{s.t.} \;\; \mu_i \geq 0, \; i = 1, \ldots, N \tag{4.95}$$

$$\sum_{i=1}^{N} \mu_i y_i = 0 \tag{4.96}$$

其中 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{\phi}(\boldsymbol{x}_j)$。采用这种基于核函数的对偶形式的优势在于，我们可以通过核替换来避免在高维特征空间中进行显式的特征映射，甚至可以使用无限维的特征表示。在约束条件中，公式4.95给出了拉格朗日乘子的定义域约束，而公式4.96则对应原始问题最优解应满足的条件（即公式4.92）。对偶形式本身仍然是一个带约束的二次规划问题。无论从原始形式还是对偶形式来看，标准支持向量机的优化问题都属于凸优化问题。其目标函数为凸二次函数（对偶形式中为凹二次函数的最大化），约束条件均为线性约束，因此任意局部最优解都是全局最优解。从理论上讲，该优化问题可以转化为标准形式的二次规划问题，并利用一系列经典优化方法加以求解，例如内点法、有效集法、椭球法、增广拉格朗日法、梯度投影法等 [13]。然而，当训练样本规模较大时，受制于计算和存储开销，直接采用这些方法往往不太可行。我们将在第4.5节介绍更为高效的求解算法。

当完成二次规划问题的求解，并获得参数 μ_i 的最优值后，我们可以用以下公式来对新样本 $\boldsymbol{x}$ 进行预测：

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \mu_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + w_0 \tag{4.97}$$

上式是将公式4.91中关于权重参数 $\boldsymbol{w}$ 的表达式代入原线性模型（公式4.78）的结果。值得注意的是，对于 $\mu_i = 0$ 的训练样本，在预测过程中不再参与计算（对应的样本也不需要保留），因而支持向量机是一种核方法的稀疏化实现。对于偏置项 w_0 的取值，由于任意一个支持向量 $\boldsymbol{x}_i$ 都满足公式4.82，即 $y_i\left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0\right) = y_i f(\boldsymbol{x}_i) = 1$。我们将公式4.97代入该约束，可得：

$$y_i\left(\sum_{\boldsymbol{x}_j \in \mathcal{S}} \mu_j y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + w_0\right) = 1 \tag{4.98}$$

其中 $\mathcal{S}$ 表示支持向量的集合。虽然理论上可以选取任一支持向量 $\boldsymbol{x}_i$ 来求解偏置项 w_0，但为了提高计算过程的数值稳定性，我们通常会使用所有支持向量来计算 w_0 的值。对上式两边同时乘以 y_i，并且利用 $y_i^2 = 1$（因为 $y_i \in \{-1, 1\}$）的性质，可以得到：

$$w_0 = \frac{1}{N_{\mathcal{S}}} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \left(y_i - \sum_{\boldsymbol{x}_j \in \mathcal{S}} \mu_j y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) \tag{4.99}$$

其中 $N_{\mathcal{S}}$ 表示支持向量的数量。上式实际上是先用每个支持向量分别计算 w_0 的值，然后再取其平均。在上述计算过程中，计算 $f(\boldsymbol{x}_i)$ 需要依赖 μ_i 和 w_0，而求解 w_0 又依赖于 μ_i，这似乎形成了一个循环依赖关系。在实际求解最优问题时（详见第4.5节），在给定合适初值（通常都可以设为零）的情况下，算法通常采用迭代更新的方式，交替调整 μ_i 和 w_0（更新某一参数时保持其他参数的值不变），直至收敛。图4.5(a) 展示了采用高斯核函数的支持向量机在二分类问题中的应用示例。需要注意的是，支持向量机在原特征空间中所形成的决策边界可以是非线性的。

4.5 软间隔支持向量机

到目前为止，我们假设训练数据在特征空间 $\boldsymbol{\phi}(\boldsymbol{x})$ 中是线性可分的。然而，在实际应用中，不同类别的样本在特征空间 $\boldsymbol{\phi}(\boldsymbol{x})$ 中的分布往往存在重叠区域，并不能做到完全线性可分。如果仍然强行要求在训练数据上实现完全分离，反而可能导致过拟合，从而削弱模型在未见数据上的泛化能力。为了解决这一问题，我们需要对此前介绍的支持向量机进行改进，使其能够允许部分训练样本出现分类错误。改进的核心思想是在约束条件中引入松弛变量，从而将公式4.83所示的不等式约束修改为：

$$y_i\left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0\right) \geq 1 - \xi_i \tag{4.100}$$

其中 $\xi_i \geq 0$ 为松弛变量，用于衡量第 i 个样本违反约束条件的程度。如图4.4(b) 所示，当 $\xi_i = 0$ 时，与原约束条件相同，表示第 i 个样本被正确分类，并且位于间隔之外或正好落在间隔边界上（即为典型的支持向量）；当 $0 < \xi_i \leq 1$ 时，样本仍然被正确分类，但其位置落入了间隔区域之内；当 $\xi > 1$ 时，该样本已经越过决策边界，从而被误分类。之前的约束条件称为硬间隔，而允许部分训练样本落入间隔区域甚至被误分类的约束形式，则被称为软间隔（Soft Margin）。软

间隔支持向量机的优化问题可以表示为：

$$\arg\min_{\boldsymbol{w},w_0,\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{4.101}$$

$$\text{s.t. } y_i\left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{w} + w_0\right) \geq 1-\xi_i,\ i=1,\ldots,N \tag{4.102}$$

$$\xi_i \geq 0,\ i=1,\ldots,N \tag{4.103}$$

其中 $\boldsymbol{\xi}=(\xi_1,\ldots,\xi_N)^\top$，而超参数 $C>0$ 的大小反映了对违反约束样本的惩罚程度与增大间隔之间的权衡。由于任何被误分类的样本都满足 $\xi_i>1$，因此 $\sum_{i=1}^{N}\xi_i$ 可以视作误分类样本数量的一个上界。当数据线性可分时，足够大的 $C\to\infty$ 可以使软间隔解与硬间隔解一致，即模型将不允许出现误分类样本。由此可见，硬间隔支持向量机是软间隔支持向量机的一个特例。而当数据不可分时，增大 C 只能加强对违反间隔的惩罚。

我们的目标是在满足约束条件的前提下，最小化公式4.101所示的目标函数，其对应的拉格朗日形式如下：

$$\mathcal{L}(\boldsymbol{w},w_0,\boldsymbol{\xi},\boldsymbol{\mu},\boldsymbol{\eta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\mu_i\left(y_i(\boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^\top\boldsymbol{w}+w_0\right) - 1 + \xi_i\Big) - \sum_{i=1}^{N}\eta_i\xi_i \tag{4.104}$$

其中拉格朗日乘子 $\mu_i\geq 0$ 和 $\eta_i\geq 0$ 分别对应公式4.102和4.103所示的约束条件。需要注意的是，约束条件 $\xi_i\geq 0$ 的标准形式是 $-\xi_i\leq 0$。将其乘以对应的拉格朗日乘子 η_i 并加入目标函数后，使得 $\sum_{i=1}^{N}\eta_i\xi_i$ 这一项前为负号。相应的 KKT 条件包括：

$$\mu_i \geq 0 \tag{4.105}$$

$$y_i\left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{w}+w_0\right) - 1 + \xi_i \geq 0 \tag{4.106}$$

$$\mu_i\left(y_i\left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{w}+w_0\right) - 1 + \xi_i\right) = 0 \tag{4.107}$$

$$\eta_i \geq 0 \tag{4.108}$$

$$\xi_i \geq 0 \tag{4.109}$$

$$\eta_i\xi_i = 0 \tag{4.110}$$

其中 $i=1,\ldots,N$。以上公式4.105至4.107对应于硬间隔情况下的 KKT 条件（公式4.88至4.90），公式4.108表示拉格朗日乘子的定义域约束，公式4.109给出了松弛变量的非负性约束，而最后一项条件 $\eta_i\xi_i=0$ 体现了互补松弛原则：对于每个样本，要么松弛变量 $\xi_i=0$，即不等式约束 $\xi_i\geq 0$ 到达边界，此时样本未违反约束，其对应拉格朗日乘子 η_i 可以大于零；要么对应的拉格朗日乘子 $\eta_i=0$，此时样本允许被误分类或位于间隔之内。换言之，ξ_i 和 η_i 不能同时严格大于零。互补松弛的含义是：只有当不等式约束在最优解处收紧（生效成为活跃约束）时，其对应的拉格朗日乘子才可以非零，从而对优化产生影响，否则该乘子必须为零。在推导出对偶形式之后，我们将进一步讨论互补松弛条件的必要性和作用。

将公式4.104中的目标函数 $\mathcal{L}(\boldsymbol{w},w_0,\boldsymbol{\xi},\boldsymbol{\mu},\boldsymbol{\eta})$ 分别对 $\boldsymbol{w}$、w_0 和 ξ_i 求偏导，并令偏导数为零，

我们可以得到以下等式（即驻点条件）：

$$\boldsymbol{w} = \sum_{i=1}^{N} \mu_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i) \tag{4.111}$$

$$0 = \sum_{i=1}^{N} \mu_i y_i \tag{4.112}$$

$$\mu_i = C - \eta_i \tag{4.113}$$

将公式4.111中求得的权重 $\boldsymbol{w}$ 表达式代入拉格朗日目标函数（公式4.104），并且结合公式4.112和公式4.113，即可得到原最优化问题对应的对偶形式（以最大化为目标）：

$$\mathcal{J}(\boldsymbol{\mu}) = \sum_{i=1}^{N} \mu_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \mu_i \mu_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4.114}$$

$$\text{s.t. } 0 \leq \mu_i \leq C,\ i = 1, \ldots, N \tag{4.115}$$

$$\sum_{i=1}^{N} \mu_i y_i = 0 \tag{4.116}$$

我们来分析一下拉格朗日乘子 μ_i 取不同值时所对应的情形：当 $\mu_i = 0$ 时，由公式4.111 可知，该样本对权重向量 $\boldsymbol{w}$ 没有贡献，同时在对偶优化目标函数（公式4.114）中也不起作用。因此，该样本不会成为支持向量；当 $\mu_i > 0$ 且 $\mu_i < C$ 时，根据公式4.113可知，必然有 $\eta_i > 0$。结合互补松弛条件 $\eta_i \xi_i = 0$，可以推知 $\xi_i = 0$。再由约束条件4.107，推出 $y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0\right) - 1 = 0$，说明该样本正好位于间隔边界上，因而它是一个典型的支持向量；当 $\mu_i > 0$ 且 $\mu_i = C$ 时，由公式4.113可知，此时必然有 $\eta_i = 0$。根据互补松弛条件，ξ_i 的取值可以大于零，即允许该样本被误分类（即 $\xi_i > 1$）或落入间隔内（即 $\xi_i \leq 1$）。从以上分析可以看出，互补松弛条件 $\eta_i \xi_i = 0$ 在软间隔支持向量机中起到了区分不同类型支持向量的作用，一类正好位于间隔边界上，而另一类样本不受间隔的限制，允许其落入间隔内或甚至被误分类。对于后一类样本，在以拉格朗日形式表示的优化问题（公式4.104）中，其违反间隔的程度由松弛变量 ξ_i 衡量，并通过常数 C 对违反情况进行线性惩罚。

与硬间隔支持向量机的情况类似，我们同样利用公式4.82来求解偏置项 w_0。具体来说，w_0 由那些满足 $y_i \left(\boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{w} + w_0\right) = 1$ 条件的样本确定。这类样本的松弛变量 $\xi_i = 0$，且其拉格朗日乘子满足 $0 < \mu_i < C$，即它们正好位于间隔边界上。这类样本通常称为自由支持向量（Free Support Vectors）。

设 $\mathcal{M}$ 表示所有拉格朗日乘子满足 $0 < \mu_i < C$ 的自由支持向量所组成的集合，则对于其中任意一个样本，都有：

$$y_i \left(\sum_{i=1}^{N} \mu_j y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + w_0 \right) = 1 \tag{4.117}$$

其中 N 为训练样本的数量。需要注意的是，上式中的求和包含所有训练样本，但由于非支持向量对应的拉格朗日乘子为零，因此实际计算时只需要考虑所有满足 $\mu_j > 0$ 的支持向量。

由上式可以求得偏置项 w_0。为了减小数值计算误差，通常会对多个自由支持向量计算得到的偏置项取平均，从而获得更稳定的估计结果：

$$w_0 = \frac{1}{N_{\mathcal{M}}} \sum_{\boldsymbol{x}_i \in \mathcal{M}} \left(y_i - \sum_{i=1}^{N} \mu_j y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \right) \tag{4.118}$$

其中 $N_{\mathcal{M}}$ 表示典型支持向量的数量，即那些对应拉格朗日乘子满足不等式 $0 < \mu_i < C$ 的支持向量的个数。

将求得的权重向量 $\boldsymbol{w} = \sum_{i=1}^{N} \mu_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)$ 和偏置项 w_0 代入公式4.78之后，我们可以得到与硬间隔支持向量机形式类似的预测公式：

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \mu_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + w_0 \tag{4.119}$$

在对新的未见样本进行预测时，只需保留训练集中对应拉格朗日乘子 $\mu_i > 0$ 的支持向量。

虽然对新输入样本进行预测仅依赖支持向量，但在训练阶段确定模型参数的过程中，需要利用整个训练数据集。因此，高效求解支持向量机所涉及二次规划问题对于实际应用至关重要。然而，直接采用通用的二次规划求解器通常并不可行。因为这类方法在计算复杂度和内存开销方面过于庞大，难以处理大规模训练数据集。为此，已经提出了一系列改进算法，其核心思想是将原始的大规模优化任务分解为一系列规模更小的子问题，并通过在这些子问题上的迭代更新逐步逼近最优解，直至整体收敛。在已有的求解方法中，序列最小优化 SMO（Sequential Minimal Optimization）是一种具有代表性的算法 [58]。该方法在每次迭代时仅选择两个拉格朗日乘子进行更新，从而将子问题简化为一个具有闭式解析解的二次规划问题。这种设计显著降低了计算复杂性，使得支持向量机能够在较大规模的数据集上高效训练。

具体而言，SMO 算法首先对预测公式4.119中的参数 $\boldsymbol{\mu}$ 和 w_0 进行初始化。最常见的做法是将它们的初始值统一设为零。然后进入迭代更新阶段。在每次迭代中，算法采用启发式方法选择一对拉格朗日乘子 μ_i 和 μ_j。其中，第一个 μ_i 一般选择违背 KKT 条件（即公式4.105至4.107）程度最大的乘子，第二个则选择使目标函数改进最大的乘子。后一个乘子常见的选择策略是选取使预测误差差异 $|e_i - e_j|$ 最大的样本 j，其中 $e_i = f(\boldsymbol{x}_i) - y_i$ 表示第 i 个样本在当前参数下的预测误差。之所以采用这样的选择策略，是因为预测误差差异与参数的更新步长成正比（可由闭式解析解推出），从而能够带来更快的目标函数改进。如果无法找到符合条件的样本 j，则随机选择一个样本。当仅考虑 μ_i 和 μ_j，并固定其他所有参数时，约束条件公式4.116可以改写为：

$$\mu_i y_i + \mu_j y_j = r \tag{4.120}$$

其中 $r = -\sum_{k \neq i,j} \mu_k y_k$。由此可以将 μ_j 用 μ_i 来表示：

$$\mu_j = \frac{r - \mu_i y_i}{y_j} \tag{4.121}$$

将该表达式代入目标函数公式4.114后，并在其他参数固定的情况下，优化问题被简化为一个关于 μ_i 的单变量二次规划问题，相关的约束是 $0 \leq \mu_i \leq C$。这一优化问题具有闭式解析解，因此可以高效求解。得到新的 μ_i 和 μ_j 后，再利用公式4.118更新偏置项 w_0。按照上述步骤不断迭

代，当所有训练样本违反 KKT 条件的程度均在允许的容忍范围之内时，算法即可停止。最终，所有满足 $\mu_i > 0$ 的样本将作为支持向量。不过在实际应用中，通常会要求 μ_i 显著大于零。对于数值上接近零的乘子，对应的样本也会被舍弃，以避免引入冗余的支持向量。

支持向量机本质上是一种二分类模型。但在实际应用中，我们往往需要处理类别数 $K > 2$ 的多分类问题。正如第3.5节中介绍的那样，常见的思路是通过构建多个二分类器，并将它们以适当方式组合，从而将二分类模型推广到多分类场景。常用的构建策略主要有三类：“一对其余”、“一对一” 和 “最大值判决”。注意，前两种策略可能在某些区域出现类别无法判定的情况。使用 “一对其余” 策略时，每次训练将某个类别的样本作为正样本，并将其余所有类别的样本作为负样本。最终训练出 K 个二分类器。但这种策略在应用时会面临两个主要问题：一是不同二分类器可能会给出相互冲突的预测结果；二是在构建每个二分类器时，会面临类别样本数量严重不平衡的问题。针对第一个问题，有时会采用如以下公式所示的 “最大值判决” 策略：

$$f(\boldsymbol{x}) = \arg\max_k f_k(\boldsymbol{x}) \tag{4.122}$$

其中 $f_k(\boldsymbol{x})$ 是第 k 个二分类器的输出得分。该策略选择输出值最大的分类器所对应的类别作为最终的预测结果。不过，这又带来了另一个问题。由于不同分类器在数据集的不同划分上单独训练，不同分类器的输出值并不能保持相同的尺度，因此直接比较各二分类器的预测得分并不可靠。对于类别不平衡问题，一个常用的改进方法是调整目标值：将正样本的目标设为 1，而将负样本的目标设为 $-1/(K-1)$，其中 K 是类别总数。这样可以在一定程度上缓解样本比例失衡带来的不利影响。另一种常用的改进方法是对不同类别设置不同惩罚系数 C。对于 “一对一” 策略，则需要为每一对类别训练一个二分类器，共需训练 $K(K-1)/2$ 个分类器。在预测时，所有分类器分别给出预测结果，最终类别由 “少数服从多数” 的原则确定。尽管这一策略可以减轻一对其余所引入的数量失衡，但由于训练和预测过程中需要使用更多的二分类器，当类别数量较多时，其计算开销较大。在实际应用中，“一对其余” 策略由于实现简单、计算开销相对较小，往往是支持向量机扩展到多分类任务时的常用选择。

4.6 回归支持向量机

我们现在将支持向量机的方法扩展到回归问题，并且依然保留其稀疏性的特点，即模型的参数仅由一部分训练样本决定。在基本的线性回归模型中，最优目标是最小化一个带有正则化项的均方误差损失函数，其形式为：

$$\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2 + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \tag{4.123}$$

为了获得稀疏解，我们将均方误差损失函数替换为 ϵ-容忍误差函数 (ϵ-insensitive Error Function)，即当某个样本的预测值 $f(\boldsymbol{x}_i)$ 与目标值 y_i 之间差异的绝对值小于 ϵ 时，其损失为零；而当差异

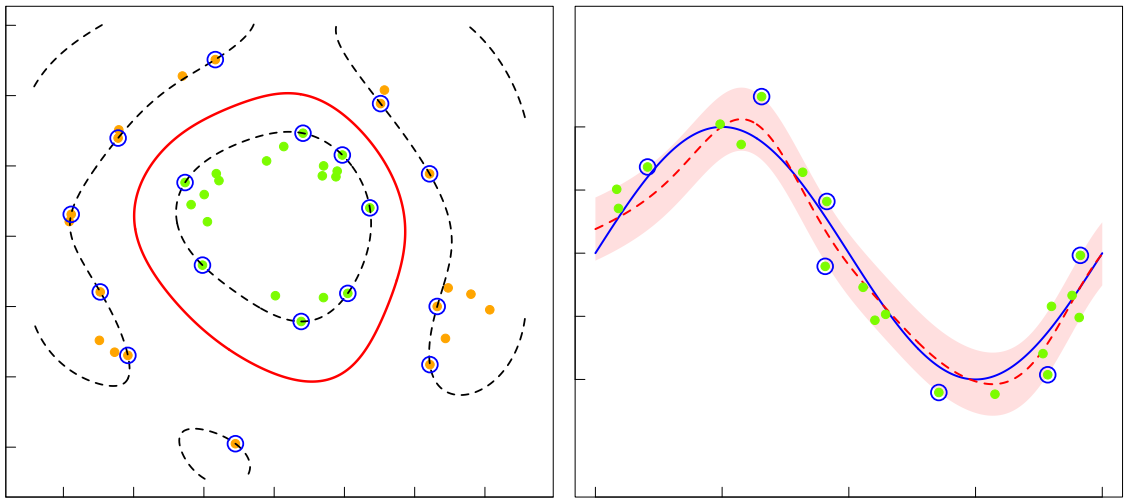


图 4.5: 支持向量机应用示例。支持向量用蓝色圆圈标出。(a) 采用高斯核函数的硬间隔支持向量机用于二分类问题。两类样本分别用绿色和橙色圆点表示。红色实线表示决策边界，黑色虚线表示分类间隔边界。注意，支持向量机在原特征空间的决策边界可以是非线性的。(b) 采用高斯核函数的支持向量机用于回归问题，对函数 $\sin(2\pi x)$ 进行拟合。红色虚线表示预测曲线。

绝对值超过 ϵ 时，损失则随着差异程度呈线性增长。一种常用的 ϵ-容忍误差函数可以表示为：

$$\mathcal{L}_\epsilon(y_i - f(\boldsymbol{x}_i)) = \begin{cases} 0, & \text{当 } |y_i - f(\boldsymbol{x}_i)| \leq \epsilon \\ |y_i - f(\boldsymbol{x}_i)| - \epsilon, & \text{当 } |y_i - f(\boldsymbol{x}_i)| > \epsilon \end{cases} \tag{4.124}$$

该损失函数如图4.6(a) 所示。为了便于比较，图中还同时显示了均方误差损失函数的曲线。

加上正则化项之后，ϵ-容忍误差的损失函数可以表示为：

$$C \sum_{i=1}^{N} \mathcal{L}_\epsilon(y_i - f(\boldsymbol{x}_i)) + \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{4.125}$$

其中 $f(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{w} + w_0$。按照惯例，在支持向量机的目标函数中，我们将原来的正则化系数 λ 替换为 C，并将其置于误差项前。C 的数值大小与 $1/\lambda$ 相对应，C 的值越大，模型越倾向于拟合训练数据；而 C 的值越小，则正则化作用越强。

与软间隔支持向量机类似，我们可以引入松弛变量来改写回归问题的目标优化函数。对于每一个样本 $\boldsymbol{x}_i$，定义两个非负的松弛变量 $\xi_i^+ \geq 0$ 和 $\xi_i^- \geq 0$，分别表示预测结果的误差超出容忍区间时，仍低于或高于目标值的额外偏差。当 $y_i > f(\boldsymbol{x}_i) + \epsilon$ 时，有 $\xi_i^+ > 0$ 且 $\xi_i^- = 0$，使得 $y_i \leq f(\boldsymbol{x}_i) + \epsilon + \xi_i^+$；当 $y_i < f(\boldsymbol{x}_i) - \epsilon$，有 $\xi_i^+ = 0$ 且 $\xi_i^- > 0$，使得 $y_i \geq f(\boldsymbol{x}_i) - \epsilon - \xi_i^-$。其中，$\epsilon > 0$ 是一个超参数，用于控制误差的容忍区间。若预测误差的绝对值小于 ϵ，则有 $\xi_i^+ = \xi_i^- = 0$。各种偏差情况如图4.6(b) 所示。

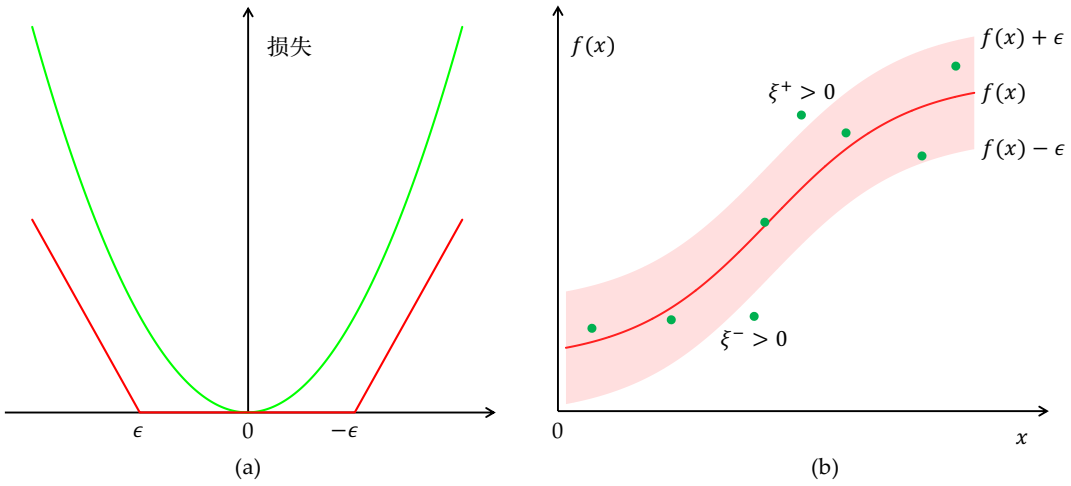


图 4.6: ϵ-容忍误差损失函数及其在回归支持向量机中的作用。(a) 红线表示 ϵ-容忍误差损失函数。在容忍区间 $[-\epsilon, \epsilon]$ 内，损失为零；超出容忍区间时，损失随差异绝对值线性增长。为便于对比，图中绿色曲线表示均方误差损失函数。(b) 回归支持向量机示意图。红线表示预测曲线，淡红色区域表示 ϵ-容忍区间。位于容忍区间上方的样本点对应 $\xi^+ > 0$ 且 $\xi^- = 0$；位于容忍区间下方的样本点对应 $\xi^+ = 0$ 且 $\xi^- > 0$；位于容忍区间内的样本点则有 $\xi^+ = \xi^- = 0$。

在引入松弛变量之后，回归支持向量机的损失函数可以表示为：

$$\arg\min_{\boldsymbol{w},w_0,\boldsymbol{\xi}} C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) + \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{4.126}$$

$$\text{s.t.}\quad y_i \le f(\boldsymbol{x}_i) + \epsilon + \xi_i^+,\ i = 1,\dots,N \tag{4.127}$$

$$y_i \ge f(\boldsymbol{x}_i) - \epsilon - \xi_i^-,\ i = 1,\dots,N \tag{4.128}$$

$$\xi_i^+ \ge 0,\ i = 1,\dots,N \tag{4.129}$$

$$\xi_i^- \ge 0,\ i = 1,\dots,N \tag{4.130}$$

其中 $\boldsymbol{\xi} = (\boldsymbol{\xi}^+, \boldsymbol{\xi}^-)$，而 $\boldsymbol{\xi}^+ = (\xi_1^+, \dots, \xi_N^+)$ 且 $\boldsymbol{\xi}^- = (\xi_1^-, \dots, \xi_N^-)$。对每个约束条件分别引入相应的拉格朗日乘子后，可以得到如下拉格朗日形式：

$$\begin{aligned}\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\eta}) = C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) + \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N}\mu_i^+(f(\boldsymbol{x}_i) - y_i + \epsilon + \xi_i^+) \\ - \sum_{i=1}^{N}\mu_i^-(y_i - f(\boldsymbol{x}_i) + \epsilon + \xi_i^-) - \sum_{i=1}^{N}(\eta_i^+\xi_i^+ + \eta_i^-\xi_i^-)\end{aligned} \tag{4.131}$$

其中 $\boldsymbol{\mu} = (\boldsymbol{\mu}^+, \boldsymbol{\mu}^-)$，$\boldsymbol{\eta} = (\boldsymbol{\eta}^+, \boldsymbol{\eta}^-)$。在公式4.131中，我们以 $\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{w} + w_0$ 代替 $f(\boldsymbol{x}_i)$，然后将目标函数 $\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\eta})$ 分别对 $\boldsymbol{w}$、w_0、ξ_i^+ 和 ξ_i^- 求偏导，并令偏导数为零，我们可以得到

以下等式（即驻点条件）：

$$\boldsymbol{w} = \sum_{i=1}^{N}(\mu_i^+ - \mu_i^-)\boldsymbol{\phi}(\boldsymbol{x}_i) \tag{4.132}$$

$$0 = \sum_{i=1}^{N}(\mu_i^+ - \mu_i^-) \tag{4.133}$$

$$C = \mu_i^+ + \eta_i^+ \tag{4.134}$$

$$C = \mu_i^- + \eta_i^- \tag{4.135}$$

我们将等式4.132代入公式4.131后，即可得到原优化问题的对偶形式。该对偶问题以最大化为目标，其表达式如下所示：

$$\begin{aligned}\mathcal{J}(\boldsymbol{\mu}) = &-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\mu_i^+ - \mu_i^-)(\mu_j^+ - \mu_j^-)\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &+ \sum_{i=1}^{N} y_i(\mu_i^+ - \mu_i^-) - \sum_{i=1}^{N}\epsilon(\mu_i^+ + \mu_i^-)\end{aligned} \tag{4.136}$$

$$\text{s.t. } 0 \le \mu_i^+ \le C,\ i = 1, \ldots, N \tag{4.137}$$

$$0 \le \mu_i^- \le C,\ i = 1, \ldots, N \tag{4.138}$$

$$\sum_{i=1}^{N}(\mu_i^+ - \mu_i^-) = 0 \tag{4.139}$$

其中 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^\top \boldsymbol{\phi}(\boldsymbol{x}_j)$ 表示核函数。不等式约束4.138来源于公式4.134以及拉格朗日乘子非负性条件 $\eta_i^+ \ge 0$，不等式约束4.139来源于公式4.135以及拉格朗日乘子非负性条件 $\eta_i^- \ge 0$，而等式约束4.139则继承自驻点条件4.133。相应的 KKT 条件包括：

$$\mu_i^+ \ge 0 \tag{4.140}$$

$$f(\boldsymbol{x}_i) - y_i + \epsilon + \xi_i^+ \ge 0 \tag{4.141}$$

$$\mu_i^+(f(\boldsymbol{x}_i) - y_i + \epsilon + \xi_i^+) = 0 \tag{4.142}$$

$$\mu_i^- \ge 0 \tag{4.143}$$

$$y_i - f(\boldsymbol{x}_i) + \epsilon + \xi_i^- \ge 0 \tag{4.144}$$

$$\mu_i^-(y_i - f(\boldsymbol{x}_i) + \epsilon + \xi_i^-) = 0 \tag{4.145}$$

$$\eta_i^+, \eta_i^- \ge 0 \tag{4.146}$$

$$\xi_i^+, \xi_i^- \ge 0 \tag{4.147}$$

$$(C - \mu_i^+)\xi_i^+ = 0 \quad (\text{即 } \eta_i^+\xi_i^+ = 0) \tag{4.148}$$

$$(C - \mu_i^-)\xi_i^- = 0 \quad (\text{即 } \eta_i^-\xi_i^- = 0) \tag{4.149}$$

公式4.140至公式4.142，以及公式4.143至公式4.145均以三个约束一组同时出现。公式4.148所示的 KKT 条件，是将驻点条件4.134（即 $\eta_i^+ = C - \mu_i^+$）代入原 KKT 条件 $\eta_i^+\xi_i^+ = 0$ 之后得到的。

同理，也可得 KKT 条件4.149。

从上述 KKT 条件出发，我们可以得到一些有用的结论。首先，由公式4.142可知，拉格朗日乘子 μ_i^+ 仅当 $f(\boldsymbol{x}_i)-y_i+\epsilon+\xi_i^+=0$ 成立时才可能取非零值。此时，对应的样本要么位于容忍区间的上边界（即 $\xi_i^+=0$），要么位于容忍区间之外的上方（即 $\xi_i^+>0$）。同理，当 μ_i^- 取非零值时，则必须满足 $y_i-f(\boldsymbol{x}_i)+\epsilon+\xi_i^-=0$，对应的样本要么位于容忍区间的下边界（即 $\xi_i^-=0$），要么位于容忍区间之外的下方（即 $\xi_i^->0$）。此外，对于每个样本 $\boldsymbol{x}_i$，对应的两个乘子 μ_i^+ 和 μ_i^- 不可能同时取非零值。因为如果两者同时大于零，则必须同时满足 $f(\boldsymbol{x}_i)-y_i+\epsilon+\xi_i^+=0$ 和 $y_i-f(\boldsymbol{x}_i)+\epsilon+\xi_i^-=0$。将两式相加得 $2\epsilon+\xi_i^++\xi_i^-=0$，然而由于 $\epsilon>0$，$\xi_i^+\geq 0$ 和 $\xi_i^-\geq 0$，$2\epsilon+\xi_i^++\xi_i^-$ 必然严格大于零，这与等式相矛盾，因而是不可能的。

将公式4.132代入到 $f(\boldsymbol{x})=\boldsymbol{\phi}(\boldsymbol{x})^\top\boldsymbol{w}+w_0$，我们即可得到对于新样本的预测公式：

$$f(\boldsymbol{x})=\sum_{i=1}^{N}(\mu_i^+-\mu_i^-)\kappa(\boldsymbol{x},\boldsymbol{x}_i)+w_0 \tag{4.150}$$

其中偏置 w_0 可以通过满足 $0<\mu_i^+<C$ 的样本来计算。根据 KKT 条件4.148，即 $(C-\mu_i^+)\xi_i^+=0$，若 $0<\mu_i^+<C$，则必然有 $\xi_i^+=0$。再结合公式4.142，即 $\mu_i^+(f(\boldsymbol{x}_i)-y_i+\epsilon+\xi_i^+)=0$，可推出 $f(\boldsymbol{x}_i)-y_i+\epsilon=0$。由此可得：

$$\begin{aligned} w_0 &= y_i-\epsilon-\boldsymbol{\phi}(\boldsymbol{x}_i)^\top\boldsymbol{w} \\ &= y_i-\epsilon-\sum_{j=1}^{N}(\mu_j^+-\mu_j^-)\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j) \end{aligned} \tag{4.151}$$

理论上，只需任意选择一个满足 $0<\mu_i^+<C$ 的样本即可计算 w_0。但在实际应用中，通常会选取所有满足该条件的样本，分别通过上式求得 w_0 的值，并取其平均作为最终结果，以提高估计的可靠性。由预测公式4.150可知，支持向量是那些对应乘子 $\mu_i^+\neq 0$ 或 $\mu_i^-\neq 0$ 的样本。而对于 $\mu_i^+=\mu_i^-=0$ 的样本，即落在容忍区间之内的样本，由于其不参与模型预测，因此可以被舍弃。由此可见，回归支持向量机同样具有稀疏性，其预测阶段仅需计算与支持向量相关的项，从而有效降低了计算复杂度和存储开销。回归支持向量机的训练同样可以采用第4.5节介绍的 SMO 算法来求解。使用该方法对函数 $\sin(2\pi x)$ 进行拟合的示例如图4.5(b) 所示。

主要符号表

数学符号

$\boldsymbol{x}$	向量
y	标量
D	向量维度
x_j	向量的第 j 个分量（向量分量为标量）
f	输入到输出的某种映射
$\mathbf{X}^\top$	矩阵转置
$\mathbf{X}^{-1}$	逆矩阵
$\|\mathbf{X}\|$	矩阵的行列式
$\mathrm{tr}(\mathbf{X})$	矩阵的迹（矩阵所有特征值之和，也等于主对角线元素的总和）
$\ln$	以自然常数 e 为底的对数
e^x 或 $\exp(x)$	自然常数 e 的指数函数
$f'(x)$	一阶导数
$f''(x)$	二阶导数
$\nabla f(\boldsymbol{x})$	一阶梯度（梯度向量）
$\nabla^2 f(\boldsymbol{x})$	二阶梯度（海森矩阵）
$\frac{\partial f(x)}{\partial x}$	一阶偏导
$\frac{\partial^2 f(x)}{\partial x^2}$	二阶偏导

机器学习

$\mathcal{D}$	数据集
N	样本数量
$\boldsymbol{x}_i$	第 i 个训练样本
y_i	第 i 个训练样本对应的输出
$\kappa(\boldsymbol{x}, \boldsymbol{x}')$	核函数
$\mathcal{L}(\cdot)$	损失函数
$\mathcal{J}(\cdot)$	目标函数或代价函数

概率统计

$\mathbb{E}[x]$	均值或期望
$\text{Var}[x]$	方差
$\text{Cov}[x, z]$	随机变量 x 和 z 之间的协方差
$\Gamma(z)$	伽玛函数
$\mathcal{N}(x\|\mu, \sigma^2)$	高斯分布，其中 μ 为均值，σ 为标准差
$\mathcal{N}(x\|\mu, \lambda^{-1})$	高斯分布，其中 μ 为均值，λ 为精度
$\text{Bern}(x\|\mu)$	伯努利分布
$\text{Bin}(k\|\mu, N)$	二项分布
$\text{Beta}(\mu\|\alpha, \beta)$	贝塔分布
$\text{Poisson}(k\|\lambda)$	泊松分布
$\text{Exp}(\lambda\|\beta)$	指数分布
$\text{Gamma}(\lambda\|\alpha, \beta)$	伽玛分布，其中 α 为形状参数，β 为逆尺度参数
$\text{Mult}(\boldsymbol{m}\|\boldsymbol{\mu}, N)$	多项分布
$\text{Dir}(\boldsymbol{\mu}\|\boldsymbol{\alpha})$	狄利克雷分布
$\mathcal{N}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Sigma})$	多维高斯分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Sigma}$ 为协方差矩阵
$\mathcal{N}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Lambda})$	多维高斯分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Lambda}$ 为精度矩阵
$\mathcal{W}(\boldsymbol{\Lambda}\|\mathbf{W}, \eta)$	威沙特分布，其中 $\mathbf{W}$ 为尺度矩阵，η 为自由度
$\text{NormalGamma}(\mu, \lambda\|\upsilon, \nu, \alpha, \beta)$	高斯-伽玛分布
$\text{NormalWishart}(\boldsymbol{\mu}, \boldsymbol{\Lambda}\|\boldsymbol{\upsilon}, \nu, \mathbf{W}, \eta)$	高斯-威沙特分布
$\text{St}(x\|\mu, \tau, \eta)$	学生氏分布，其中 μ 为均值，τ 为精度，η 为自由度
$\text{St}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Lambda}, \eta)$	多维学生氏分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Lambda}$ 为精度矩阵
$\text{Laplace}(x\|\mu, \tau)$	拉普拉斯分布，其中 μ 为位置参数，τ 为尺度参数
$\chi^2(x\|\eta)$	卡方分布，其中 η 为自由度
$\text{U}(x\|a, b)$	连续型均匀分布，其中 a 为下界、b 为上界 $(a < b)$
$\text{H}[x]$	信息熵，其中 x 为随机变量
$\text{H}[y\|x]$	条件熵，其中 x 和 y 为随机变量
$\text{H}[p, q]$	分布 p 和 q 之间的交叉熵
$D_{\text{KL}}(p\|\|q)$	分布 p 和 q 之间的 KL 散度

参考文献

[1] Daniel J. Amit, Hanoch Gutfreund, and Haim Sompolinsky. “Storing infinite numbers of patterns in a spin-glass model of neural networks”. In: *Physical Review Letters* 55.14 (1985), pp. 1530–1533.

[2] Mikel Artetxe et al. “On the role of bidirectionality in language model pre-training”. In: *Findings of the Conference on Empirical Methods in Natural Language Processing* (2022), pp. 3973–3985.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv: 1607.06450* (2016).

[4] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166.

[5] Jame O. Berger. *Statistical decision theory and Bayesian analysis (second edition)*. Springer, 1980.

[6] James Bergstra and Yoshua Bengio. “Random search for hyper-parameter optimization”. In: *Journal of Machine Learning Research* 13.2 (2012).

[7] Jeremy Bernstein and Laker Newhouse. “Old optimizer, new norm: An anthology”. In: *Proceedings of the 16th Annual Workshop on Optimization for Machine Learning* (2024), pp. 1–19.

[8] J Besag. “On spatio-temporal models and Markov fields”. In: *Transactions of the 7th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes* (1974), pp. 47–75.

[9] Julian Besag. “On the statistical analysis of dirty pictures”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 48.3 (1986), pp. 259–279.

[10] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[11] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. “A training algorithm for optimal margin classifiers”. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. 1992, pp. 144–152.

[12] George E.P. Box and Norman Richard Draper. *Empirical model-building and response surfaces*. John Wiley & Sons, 1987.

[13] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[14] Tom Brown et al. “Language models are few-shot learners”. In: *Proceedings of the Conference on Neural Information Processing Systems* 33 (2020), pp. 1877–1901.

[15] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder–decoder for statistical machine translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (2014), pp. 1724–1734.

[16] Peter Clifford. “Markov random fields in statistics”. In: *Disorder in Physical Systems: A Volume in Honour of John M. Hammersley* (1990), pp. 19–32.

[17] Thomas H. Cormen et al. *Introduction to algorithms (third edition)*. The MIT Press, 2022.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley, 1991.

[19] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[20] Morris H. DeGroot and Mark J. Schevish. *Probability and statistics (forth edition)*. China Machine Press, 2012.

[21] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2019), pp. 4171–4186.

[22] Alexey Dosovitskiy et al. “An image is worth 16×16 words: Transformers for image recognition at scale”. In: *Proceedings of the International Conference on Learning Representations* (2021).

[23] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization”. In: *Journal of Machine Learning Research* 12.7 (2011).

[24] Walter D. Fisher. “On grouping for maximum homogeneity”. In: *Journal of the American Statistical Association* 53.284 (1958), pp. 789–798.

[25] Roger Fletcher. *Practical methods of optimization (second edition)*. Wiley, 1987.

[26] Yoav Freund and Robert E. Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139.

[27] Stuart Geman and Donald Geman. “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), pp. 721–741.

[28] Felix A. Gers, Nicol N. Schraudolph, and Jürgen Schmidhuber. “Learning precise timing with LSTM recurrent networks”. In: *Journal of Machine Learning Research* 3 (2002), pp. 115–143.

[29] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256.

[30] Ian J Goodfellow et al. “Generative adversarial nets”. In: *Proceedings of the Conference on Neural Information Processing Systems* 27 (2014).

[31] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction (second edition)*. Springer, 2009.

[32] Kaiming He and Jian Sun. “Convolutional neural networks at constrained time cost”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 5353–5360.

[33] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[34] Kaiming He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1026–1034.

[35] Nicholas J. Higham. *Functions of matrices: Theory and computation*. SIAM, 2008.

[36] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv: 1503.02531* (2015).

[37] Geoffrey E. Hinton. “Training products of experts by minimizing contrastive divergence”. In: *Neural Computation* 14.8 (2002), pp. 1771–1800.

[38] John J. Hopfield. “Neural networks and physical systems with emergent collective computational abilities”. In: *Proceedings of the National Academy of Sciences* 79.8 (1982), pp. 2554–2558.

[39] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *Proceedings of the International Conference on Machine Learning* (2015), pp. 448–456.

[40] Arieh Iserles. *A first course in the numerical analysis of differential equations*. 44. Cambridge university press, 2009.

[41] Max Jaderberg et al. “Population based training of neural networks”. In: *arXiv: 1711.09846* (2017).

[42] Kevin Jamieson and Ameet Talwalkar. “Non-stochastic best arm identification and hyperparameter optimization”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics* (2016), pp. 240–248.

[43] Keller Jordan et al. *Muon: An optimizer for hidden layers in neural networks*. 2024. URL: https://kellerjordan.github.io/posts/muon/.

[44] Diederik P. Kingma and Jimmy L. Ba. “Adam: A method for stochastic optimization”. In: *Proceedings of the International Conference on Learning Representations* (2015).

[45] Josef Kittler and Janos Föglein. “Contextual classification of multispectral pixel data”. In: *Image and vision computing* 2.1 (1984), pp. 13–29.

[46] Solomon Kullback and Richard A. Leibler. “On information and sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[47] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (2002), pp. 2278–2324.

[48] Mike Lewis et al. “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 7871–7880.

[49] Yaron Lipman et al. “Flow matching guide and code”. In: *arXiv: 2412.06264* (2024).

[50] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv: 1711.05101* (2017).

[51] Calvin Luo. “Understanding diffusion models: A unified perspective”. In: *arXiv: 2208.11970* (2022).

[52] Marvin Minsky and Seymour Papert. *Perceptrons: An introduction to computational geometry*. The MIT Press, 1969.

[53] Kevin P. Murphy. *Machine learning: A probabilistic perspective*. The MIT Press, 2012.

[54] Brady Neal et al. “A modern take on the bias-variance tradeoff in neural networks”. In: *arXiv: 1810.08591* (2018).

[55] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Proceedings of the Conference on Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[56] Judea Pearl. *Probabilistic reasoning in intelligent systems*. Morgan kaufmann, 2014.

[57] Matthew E. Peters et al. “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2018), pp. 2227–2237.

[58] John C. Platt. “Fast training of support vector machines using sequential minimal optimization”. In: *Advances in Kernel Methods* (1999), pp. 185–208.

[59] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *Proceedings of the International Conference on Learning Representations* (2016).

[60] Alec Radford et al. “Improving language understanding by generative pre-training”. In: *OpenAI’s Technical Report* (2018).

[61] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67.

[62] Scott Reed et al. “Generative adversarial text to image synthesis”. In: *Proceedings of the International Conference on Machine Learning* (2016), pp. 1060–1069.

[63] Edmund T. Rolls. *Cerebral cortex: principles of operation*. Oxford University Press, 2016.

[64] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional networks for biomedical image segmentation”. In: *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention* (2015), pp. 234–241.

[65] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning representations by back-propagating errors”. In: *Nature* 323.6088 (1986), pp. 533–536.

[66] Leonard J. Savage. *The subjective basis of statistical practice*. University of Michigan, 1961.

[67] Andrew M Saxe, James L McClelland, and Surya Ganguli. “Exact solutions to the nonlinear dynamics of learning in deep linear neural networks”. In: *Proceedings of the International Conference on Learning Representations* (2014).

[68] Robert E. Schapire. “The strength of weak learnability”. In: *Machine learning* 5.2 (1990), pp. 197–227.

[69] Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural machine translation of rare words with subword units”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), pp. 1715–1725.

[70] Claude Elwood Shannon. “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[71] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. “Practical Bayesian optimization of machine learning algorithms”. In: *Proceedings of the Conference on Neural Information Processing Systems* 25 (2012).

[72] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *Proceedings of the International Conference on Learning Representations* (2021).

[73] Gilbert Strang. *Introduction to linear algebra*. Wellesley-Cambridge Press, 2016.

[74] Jianlin Su et al. “Roformer: Enhanced transformer with rotary position embedding”. In: *Neurocomputing* 568 (2024), p. 127063.

[75] Tijmen Tieleman and Geoffrey E. Hinton. “Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural Networks for Machine Learning* 4.2 (2012), pp. 26–26.

[76] Antonio Torralba, Phillip Isola, and William Freeman. *Foundations of computer vision*. The MIT Press, 2024.

[77] Ashish Vaswani et al. “Attention is all you need”. In: *Proceedings of the Conference on Neural Information Processing Systems* 30 (2017).

[78] Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. “Deep learning for Chinese word segmentation and POS tagging”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (2013), pp. 647–657.

[79] Ji Zhu et al. “Multi-class AdaBoost”. In: *Statistics and its Interface* 2.3 (2009), pp. 349–360.

[80] Hui Zou and Trevor Hastie. “Regularization and variable selection via the elastic net”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67.2 (2005), pp. 301–320.