# 目录

# 第4章　线性模型

基于贝叶斯决策论的模式分类方法虽然在理论上十分完美，但在实际应用中却难以处理高维大数据的概率估计问题。本章在贝叶斯决策论基础上，介绍基于线性函数的判别模型和用于解决分类任务的感知器算法；其次将从分类问题出发，探索回归问题的求解思路，并介绍最优化方法在线性模型上的实际应用；最后在线性模型的基础上，进一步探索逻辑回归算法及其在分类任务上的应用。

## 4.1　判别模型

### 4.1.1　生成模型与判别模型

根据贝叶斯决策论，分类问题应采用最大后验概率分类准则，即当前样本$\boldsymbol{x}$应该被分给对应的后验概率$P(\omega|\boldsymbol{x})$最大的类别。考虑到直接估计后验概率比较困难，所以通常会利用贝叶斯公式将后验概率$P(\omega|\boldsymbol{x})$转化为类条件概率密度函数$p(\boldsymbol{x}|\omega)$与先验概率$P(\omega)$的乘积。其中，$P(\omega)$可以基于训练样本中$\omega$类的样本比例进行近似估计，类条件概率$p(\boldsymbol{x}|\omega)$可以基于观测对象概率分布的某些先验知识（通常是类条件概率密度函数的数学形式）和训练样本采用最大似然法进行估计。基于上述思路建立的模型称为“生成模型（Generative Model）”。

有没有可能不通过类条件概率而直接估计后验概率$P(\omega|\boldsymbol{x})$呢？假设后验概率$P(\omega|\boldsymbol{x})$符合某个参数化函数形式，并可以通过对训练数据的学习进行参数估计，就有可能绕过类条件概率密度函数$p(\boldsymbol{x}|\omega)$，直接实现后验概率$P(\omega|\boldsymbol{x})$的估计。基于这种思路建立的模型称为“判别模型（Discriminant Model）”

直观上看，判别模型似乎比生成模型更直接，但判别模型对于使用哪种函数来拟合$P(\omega|\boldsymbol{x})$这一问题需要一些先验知识。本节将在类条件概率$p(\boldsymbol{x}|\omega)$服从正态分布的假设条件下，从生成模型的角度逆向去推导判别模型的数学形式。在某些特定的假设下，可以推导出判别模型表现为线性函数，而线性函数具有非常丰富的内涵和可扩展性，它为判别模型的实际应用奠定了数学基础。大多数经典的模式识别算法，如逻辑回归、支持向量机、决策树、Adaboost、人工神经网络等，其数学形式中都包含有线性函数的成分，因此深入理解线性模型对于学习和研究模式识别方法具有十分重要的意义。千里之行、始于足下，让我们先从为什么使用线性函数来拟合后验概率这一问题出发。

### 4.1.2　决策面与判别函数

假设样本空间$X$是连续的，对于一个二分类问题，必然存在分界面$S$,使得$\forall \boldsymbol{x} \in S$均有：

$$g(\boldsymbol{x}) = P(\omega_1|\boldsymbol{x}) - P(\omega_2|\boldsymbol{x}) = 0 \tag{4.1}$$

该分界面将整个样本空间分为两个部分$R_1, R_2$，且有$g(\boldsymbol{x}) > 0, \forall \boldsymbol{x} \in R_1$；$g(\boldsymbol{x}) < 0, \forall \boldsymbol{x} \in R_2$。

根据最大后验概率分类准则，处于$R_1$区域的样本将会被分为$\omega_1$类，处于$R_2$区域的样本将会被分为$\omega_2$类。此时，该分界面$S$被称为**决策面**（Decision Surface），公式(4.1)称为**决策面方程**。显然，决策面方程的数学形式取决于后验概率。考虑到后验概率$P(\omega|\boldsymbol{x})$的函数形式通常比较复杂，因此可以构造一个单调递增函数$f(\,)$，尝试改变决策面方程的原始数学形式。定义新的函数：

$$g_i(\boldsymbol{x}) = f\big(P(\omega_i|\boldsymbol{x})\big), i = 1,2 \tag{4.2}$$

则决策面方程可以重新写为：

$$g(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) = f\big(P(\omega_1|\boldsymbol{x})\big) - f\big(P(\omega_2|\boldsymbol{x})\big) = 0 \tag{4.3}$$

由于函数$f(\,)$是单调递增函数，因此公式(4.1)和(4.3)定义的决策面是完全相同的。对于多分类问题，从最大后验概率分类准则出发，可以通过比较函数$g_i(\boldsymbol{x})$来实现分类：

$$h(\boldsymbol{x}) = \begin{cases} \omega_1, & g_1(\boldsymbol{x}) > g_2(\boldsymbol{x}) \\ \omega_2, & g_1(\boldsymbol{x}) < g_2(\boldsymbol{x}) \end{cases} \tag{4.4}$$

此时，我们称$g_i(\boldsymbol{x})$为$\omega_i$类的**判别函数**（Discriminant Function）。原理上每个类别都可以有无穷多个判别函数，但在实际应用中选择一个合适的单调函数$f(\,)$可以在很大程度上简化判别函数的数学形式。下面，我们将以多元正态分布为例，解释这一操作。

假设$\omega_i$类服从多元正态分布，其类条件概率密度函数如下所示：

$$p(\boldsymbol{x}|\omega_i) = \frac{1}{2\pi^{\frac{l}{2}}|\Sigma_i|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right] \tag{4.5}$$

其中，$\boldsymbol{\mu}_i \in \Re^d$为均值矢量，$\Sigma_i \in \Re^{d\times d}$为协方差矩阵，则根据贝叶斯定理，后验概率$P(\omega_i|\boldsymbol{x}) = p(\boldsymbol{x}|\omega_i)P(\omega_i)/p(\boldsymbol{x})$。令单调函数$f(x) = \ln(x)$，则$\omega_i$类的判别函数可以写为：

$$\begin{aligned} g_i(\boldsymbol{x}) &= \ln\left[\frac{p(\boldsymbol{x}|\omega_i)P(\omega_i)}{p(\boldsymbol{x})}\right] \\ &= \ln p(\boldsymbol{x}|\omega_i) + \ln P(\omega_i) - \ln p(\boldsymbol{x}) \\ &= -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i) - \frac{l}{2}\ln 2\pi - \ln p(\boldsymbol{x}) \end{aligned} \tag{4.6}$$

将公式(4.6)带入决策面方程(4.3)，并假设两个类别的协方差矩阵$\Sigma_1 = \Sigma_2 = \Sigma$，消去相同的项，则决策面方程可以简化为：

$$g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0 = 0 \tag{4.7}$$

其中：

$$\boldsymbol{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \qquad w_0 = \ln P(\omega_1) - \ln P(\omega_2) - \frac{1}{2}(\boldsymbol{\mu}_1^T\Sigma^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^T\Sigma^{-1}\boldsymbol{\mu}_2) \tag{4.8}$$

可以看到此时的决策面方程已经简化为线性函数。因此“线性分类器可以视为贝叶斯分类器在具有相同先验概率$P(\omega)$和协方差矩阵$\Sigma$的正态分布假设下的特例。”

## 4.2 线性分类器基础

### 4.2.1 决策超平面的几何解释

对于一个正类(Positive class)和负类(Negative class)的二分类问题，在决策面方程$g(\boldsymbol{x})=0$中，等式左边的表达式$g(\boldsymbol{x})>0$时，样本$\boldsymbol{x}$被分为正类；否则被分为负类。如公式(4.7)所示，在某些假设条件下，决策面方程可以是一个线性方程。当样本$\boldsymbol{x}$的维度$d=2$时，决策面是一条直线；当$l>2$时，决策面将变成$d$维空间中的一个$d-1$维的超平面，称为**决策超平面**（Decision Hyperplane）。

为了更加准确地理解决策超平面，我们对$d=2$情况下的决策面方程与决策超平面之间的对应关系进行简单的分析：

$$g(\boldsymbol{x})=\boldsymbol{w}^T\boldsymbol{x}+w_0=[w_1,w_2]\begin{bmatrix}x_1\\x_2\end{bmatrix}+w_0=w_1x_1+w_2x_2+w_0=0 \tag{4.9}$$

公式(4.9)是定义在$x_1$-$x_2$平面直角坐标系上的直线方程，如图 4-1 所示。可以很容易推出该直线在$x_1,x_2$两个坐标轴的截距分别为$-w_0/w_1$和$-w_0/w_2$，该直线的法向量方向为$\boldsymbol{w}=[w_1,w_2]^T$，直线到坐标系原点的距离为：

$$d=\frac{|w_0|}{\sqrt{w_1^2+w_2^2}}=\frac{|w_0|}{\|\boldsymbol{w}\|} \tag{4.10}$$

需要特别注意的是，任意给定的样本点$\boldsymbol{x}$到决策超平面$g(\boldsymbol{x})=0$的距离为：

$$z=\frac{|w_1x_1+w_2x_2+w_0|}{\sqrt{w_1^2+w_2^2}}=\frac{|g(\boldsymbol{x})|}{\|\boldsymbol{w}\|} \tag{4.11}$$

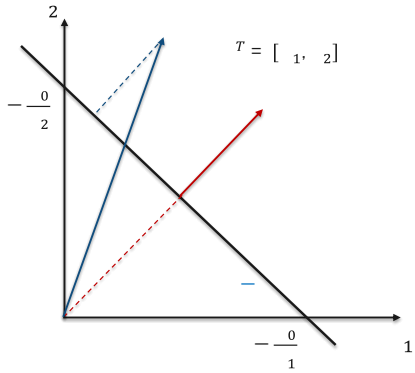



图 4-1 二维情况下决策面方程的几何示意图

公式(4.11)说明$g(\boldsymbol{x})$的绝对值与样本$\boldsymbol{x}$到决策超平面的距离成正比，$g(\boldsymbol{x})$的正负符号则取决于样本点$\boldsymbol{x}$落到决策超平面的哪一边。假设正样本的类别标签设为1，负样本的类别标签设

为$-1$，则模式识别系统的输入输出关系可以写为：

$$h(\boldsymbol{x}) = \text{sign}[g(\boldsymbol{x})] = \text{sign}[\boldsymbol{w}^T\boldsymbol{x} + w_0] \tag{4.12}$$

只要能够找到一组合适的参数$\boldsymbol{w} \in \Re^d$和常数项$w_0$，使得所有训练样本$(\boldsymbol{x}, y)$均满足公式(4.12)，就能构造出一个合格的判别模型。因为$g(\boldsymbol{x})$是一个线性函数，所以$h(\boldsymbol{x})$被称为线性分类器。

### 4.2.2 感知器算法

**感知器**（Perceptron）是最早出现的具备学习能力的线性分类器，是人工神经网络的里程碑式工作。虽然以现在的观点来看，感知器算法比较低效，且无法处理线性不可分问题，但它能帮助我们理解线性分类器的本质、代价函数的概念以及最优化方法在机器学习中的应用，是非常好的入门级算法。

为了使线性分类器的数学形式更加简洁，记$\hat{\boldsymbol{w}} = [w_1, \ldots, w_d, w_0]^T$，$\hat{\boldsymbol{x}} = [x_1, \ldots, x_d, 1]^T$，则有$g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0 = \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$，这里的$\hat{\boldsymbol{x}}$一般称为**增广样本**。对于二分类问题，感知器算法的目标是找到一个合适的参数向量$\hat{\boldsymbol{w}}$，使得:

$$\text{sign}(\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}) = \begin{cases} +1 & \boldsymbol{x} \in \omega_1 \\ -1 & \boldsymbol{x} \in \omega_2 \end{cases} \tag{4.13}$$

“合适”的参数这一说法已经暗示了这是一个优化问题。从最大似然估计方法开始，本书已经多次涉及了最优化方法，该方法将在本书介绍的大多数算法中反复出现。所以我们必须建立起“优化”的思想，并将其作为解决模式识别问题的核心思路。几乎所有的最优化方法都需要明确三个要素：

1）自变量，即待优化变量，在模式识别领域通常为模型参数；

2）代价函数（Cost function），也称为目标函数（Objective function）或损失函数（Loss function）,通常是对模型在训练数据上的识别结果的某种评价；

3）优化策略，通常是各种基于梯度的优化方法。

对照公式(4.13)所描述的线性分类问题，自变量为参数向量$\hat{\boldsymbol{w}}$；代价函数应为某个包含$\hat{\boldsymbol{w}}$的标量表达式，能够反映当前分类器$\text{sign}[\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}]$在训练数据集上的分类性能；优化策略使用梯度下降法即可。显然，感知器算法的关键在于代价函数的设计。同学读到此处，不妨自己先试试设计一个代价函数，看看能否与感知器算法不谋而合。

感知器的代价函数设计如下：

$$J(\hat{\boldsymbol{w}}) = \sum_{\boldsymbol{x} \in \Omega(\hat{\boldsymbol{w}})} \delta_x \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \tag{4.14}$$

其中，$\Omega(\hat{\boldsymbol{w}})$是$\hat{\boldsymbol{w}}$对应的分类器在当前训练集上产生的错分样本集合，$\delta_x$为符号系数：

$$\delta_x = \begin{cases} -1 & \boldsymbol{x} \in \omega_1 \\ 1 & \boldsymbol{x} \in \omega_2 \end{cases} \tag{4.15}$$

结合公式(4.13)，对于实际属于$\omega_1$但被当前模型错分的样本$\hat{\boldsymbol{x}}$，必然有$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \le 0$（等号成立意味着该样本落在了决策面上，也视为错误分类），此时$\delta_x = -1$，因此$\delta_x\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \ge 0$。对于

实际属于$\omega_2$却被错分为$\omega_1$的样本$\hat{\boldsymbol{x}}$，则有$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \geq 0$，此时$\delta_x = 1$，所以仍有$\delta_x\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \geq 0$。这意味着公式(4.14)的连加式中的每一项均为非负数，因此代价函数$J(\hat{\boldsymbol{w}})$的值变小有两种可能：一是错分样本减少；二是错分样本到决策面的距离变小。只有当所有样本都被正确分类时，$J(\hat{\boldsymbol{w}})$才可能得到理论上的最小值 0。至此，感知器模型的求解可以看做代价函数$J(\hat{\boldsymbol{w}})$的最小化问题，如公式(4.16)所示。

$$\hat{\boldsymbol{w}}^* = \underset{\hat{\boldsymbol{w}}}{\operatorname{argmin}} J(\hat{\boldsymbol{w}}) \tag{4.16}$$

根据梯度下降法，首先求出代价函数$J(\hat{\boldsymbol{w}})$对待优化变量$\hat{\boldsymbol{w}}$的导数。

$$\frac{\partial J(\hat{\boldsymbol{w}})}{\partial \hat{\boldsymbol{w}}} = \frac{\partial}{\partial \hat{\boldsymbol{w}}}\left(\sum_{\boldsymbol{x}\in\Omega(\hat{\boldsymbol{w}})} \delta_x \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}\right) = \sum_{\boldsymbol{x}\in\Omega(\hat{\boldsymbol{w}})} \delta_x\hat{\boldsymbol{x}} \tag{4.17}$$

则感知器算法中参数向量$\hat{\boldsymbol{w}}$的更新公式可以写为：

$$\hat{\boldsymbol{w}}(t+1) = \hat{\boldsymbol{w}}(t) - \rho\frac{\partial J(\hat{\boldsymbol{w}})}{\partial \hat{\boldsymbol{w}}} = \hat{\boldsymbol{w}}(t) - \rho_t \sum_{\boldsymbol{x}\in\Omega(\hat{\boldsymbol{w}})} \delta_x\hat{\boldsymbol{x}} \tag{4.18}$$

其中$\rho_t$为$t$时刻的学习步长。基于以上分析，则感知器算法步骤总结如图 4-2 所示。

| | |
|---|---|
| 输入: | 观测样本数据$X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}, \boldsymbol{x}_i \in \Re^d$ |
| 步骤: | |
| 1 | 随机初始化参数向量$\hat{\boldsymbol{w}}(0)$，开始迭代： |
| 2 | 设置一个合适的学习步长$\rho_0$ |
| 3 | Repeat： |
| 4 | 令$\Omega(\hat{\boldsymbol{w}}) = \emptyset$ |
| 5 | For $i = 1, \ldots, N$ |
| 6 | 如果 $\delta_x\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i \geq 0$，$\Omega(\hat{\boldsymbol{w}}) = \boldsymbol{x}_i \cup \Omega(\hat{\boldsymbol{w}})$ |
| 7 | 终止 For 循环 |
| 8 | 更新参数向量：$\hat{\boldsymbol{w}}(t+1) = \hat{\boldsymbol{w}}(t) - \rho\sum_{\boldsymbol{x}_i\in\Omega(\hat{\boldsymbol{w}})}\delta_x\hat{\boldsymbol{x}}_i$ |
| 9 | 调整学习步长$\rho_t$ |
| 10 | 更新计数器：$t = t+1$ |
| 11 | Until $\Omega(\hat{\boldsymbol{w}}) = \emptyset$ |
| 输出: | $\hat{\boldsymbol{w}}(t)$ |

图 4-2 感知器算法步骤

从梯度下降法可以搜索目标函数$J(\hat{\boldsymbol{w}})$的最小值的角度看，$J(\hat{\boldsymbol{w}})$应该至少具有一个局部极小值点。但从公式(4.14)的数学形式分析，$J(\hat{\boldsymbol{w}})$似乎是一个线性函数，不应该具有局部极小值。之所以产生这种困惑是因为读者忽略了错分样本集$\Omega(\hat{\boldsymbol{w}})$随$\hat{\boldsymbol{w}}$变化这一因素，对于某个给定的$\hat{\boldsymbol{w}}$，$J(\hat{\boldsymbol{w}})$确实服从公式(4.14)定义的线性函数，但对于两个不同的$\hat{\boldsymbol{w}}$的取值$\hat{\boldsymbol{w}}_1$和$\hat{\boldsymbol{w}}_2$，它们对应的错分样本集合$\Omega(\hat{\boldsymbol{w}}_1)$和$\Omega(\hat{\boldsymbol{w}}_2)$可能是不一致的，从而导致目标函数$J(\hat{\boldsymbol{w}}_1)$和$J(\hat{\boldsymbol{w}}_2)$的连加项不同。这意味着$J(\hat{\boldsymbol{w}})$在$\hat{\boldsymbol{w}}$的参数空间的不同区域具有不同的线性形式，因此$J(\hat{\boldsymbol{w}})$本质上是一个分段线性线性函数。对于线性可分问题而言，目标函数$J(\hat{\boldsymbol{w}})$必然在参数空间的某个区域上的

值均为零，而感知器算法的目的就是通过梯度下降法寻找这个区域中的解。但需要特别注意的是，感知器算法的分段线性目标函数不一定是凸函数，有可能存在多个局部最小值，因此梯度下降法无法确保一定能找到模型的全局最优解。从函数形式的角度看，感知器是一个典型的线性分类器。但如果将感知器算法与 MP 神经元模型相结合，就会构成一个简单的单层感知器网络，因此感知器算法同样也是人工神经网络领域的奠基性工作，其学习算法也呼应了神经网络发育的生物学机理，关于这方面的讨论将在神经网络一章给出。

### ♦ 例题-基于感知器算法的二维数据分类

如下图所示，已知正类$\omega_1$有 2 个训练样本$\boldsymbol{x}_1=[0,-1]^T,\boldsymbol{x}_2=[1,-1]^T$，对应标签$y=1$；负类$\omega_2$有 2 个训练样本$\boldsymbol{x}_3=[0,1]^T,\boldsymbol{x}_4=[1,1]^T$，对应标签$y=-1$。利用感知器算法进行线性分类器设计，初始化决策面方程为$x_1=0.5$，请对决策面方程进行一次更新，并给出计算过程。

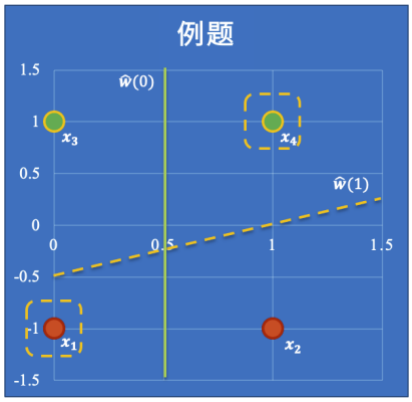



图 4-3　基于感知器算法的分类器更新结果示意图

**解答：**

第 1 步：将初始化决策面方程写为标准的感知器形式，并确定参数向量初始值$\hat{\boldsymbol{w}}(0)$

$$x_1=0.5\Longrightarrow 1x_1+0x_2+(-0.5)=0\Longrightarrow\hat{\boldsymbol{w}}(0)=\begin{bmatrix}1\\0\\-0.5\end{bmatrix}$$

第 2 步：对原始样本$\boldsymbol{x}_i$增广得到$\hat{\boldsymbol{x}}_i,i=1,2,3,4$；带入初始化决策面方程，确定错分样本集：

$$\text{sign}(\hat{\boldsymbol{w}}(0)^T\hat{\boldsymbol{x}}_1)=\text{sign}\left([1,0,-0.5]\begin{bmatrix}0\\-1\\1\end{bmatrix}\right)=-1\Rightarrow\boldsymbol{x}_1\text{为错分样本}$$

同理可以得出$\boldsymbol{x}_4$为错分样本，因此错分样本集为$\Omega(\hat{\boldsymbol{w}})=\{\boldsymbol{x}_1,\boldsymbol{x}_4\}$。

第 3 步：根据错分样本的真实标签，确定符号系数$\delta(\boldsymbol{x}_1)=-1$，$\delta(\boldsymbol{x}_4)=1$；

第 4 步：设学习率为$\rho_0=0.5$，根据感知器算法对参数向量$\hat{\boldsymbol{w}}$进行更新：

$$\hat{\boldsymbol{w}}(t+1)=\hat{\boldsymbol{w}}(t)-\rho_t\sum_{\boldsymbol{x}_i\in\Omega(\hat{\boldsymbol{w}})}\delta(\boldsymbol{x}_i)\hat{\boldsymbol{x}}_i$$

$$\hat{\boldsymbol{w}}(1)=\begin{bmatrix}1\\0\\-0.5\end{bmatrix}-0.5(-1)\begin{bmatrix}0\\-1\\1\end{bmatrix}-0.5(+1)\begin{bmatrix}1\\1\\1\end{bmatrix}$$

$$\hat{\boldsymbol{w}}(1)=\begin{bmatrix}0.5\\-1\\-0.5\end{bmatrix}$$

**结论**：更新后的决策面方程为$0.5x_1-x_2-0.5=0$，如图 4-3 所示，全部 4 个样本均能够被$\hat{\boldsymbol{w}}(1)$对应的线性分类器正确分类。

## 4.3 逻辑回归

### 4.3.1 感知器算法的局限性

尽管感知器作为最早的线性分类器算法具有划时代的意义，但在它诞生不久后就遭到了人工智能领域著名专家 Marvin Minsky 的严厉批判，其批判的主要观点是感知器无法处理以 XOR 问题为代表的线性不可分问题。XOR 问题即逻辑运算中的“异或”问题，可以简单描述为以下规则：对于定义在{0,1}二值论域上的一个双输入单输出系统，如果两个输入相同，则输出为 0；否则，输出为 1。在模式识别领域，XOR 问题可以看作一个二分类问题，其中输入包含两个特征，每个特征的取值为 0 或 1，输出为 0 或 1，表示类别标签，如图 4-4 所示，$g_1(\boldsymbol{x})=0$或$g_2(\boldsymbol{x})=0$两个线性决策面都无法正确将两类分开。显然，任何线性分类器都不可能完全正确处理以 XOR 为代表的这类问题，因此这类问题称为线性不可分问题。

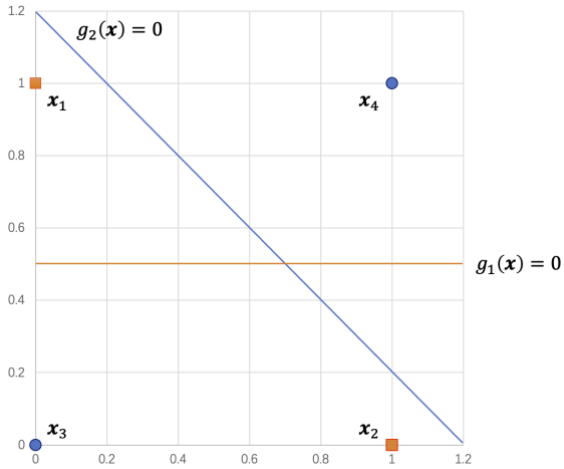



图 4-4 XOR 问题示意图

对于线性不可分问题，感知器算法找到的任意一个解都不可能使得错分样本集$\Omega(\hat{\boldsymbol{w}})$为空集，因此无法满足算法循环终止条件，算法将会在$\hat{\boldsymbol{w}}$参数空间的某些区域发生震荡，而无法收敛。另一方面，即便当前分类问题属于线性可分问题，则在能够正确分类所有训练样本的参数区域$R=\{\hat{\boldsymbol{w}}|\Omega(\hat{\boldsymbol{w}})=\emptyset\}$，目标函数$J(\hat{\boldsymbol{w}})=0,\forall\hat{\boldsymbol{w}}\in R$。这意味着感知器算法理论上存在无

穷多个最优解，不同$\hat{\boldsymbol{w}}$初始值会导致算法收敛于不同的最优解，而这些不同最优解对应的线性分类器在泛化性能上可能存在较大差异。考虑到上述原因，感知器算法并非完美的线性分类器算法，还存在较大的改进空间。本节的逻辑回归算法可以视为对感知器算法的一种改进。

### 4.3.2 逻辑回归函数

在给出逻辑回归算法的细节之前，本节将结合前文知识给出线性分类器算法设计要求：

1） 符合最大后验概率分类准则；

2） 能够处理线性不可分问题；

3） 便于模型参数的求解；

其中第一条要求模型应符合贝叶斯决策论的原理，第二条要求算法在面对线性不可分问题时仍然能够给出一些可以接受的解；第三条一般要求目标函数的导数数学形式容易计算和求解。基于上述分析，逻辑回归算法采用了 **S 型（Sigmoidal）函数**与线性函数相结合来构建逻辑回归函数，如公式(4.19)所示，其形状如图 4-5 所示。

$$h(\boldsymbol{x};\hat{\boldsymbol{w}}) = \frac{1}{1+e^{-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}}} \tag{4.19}$$

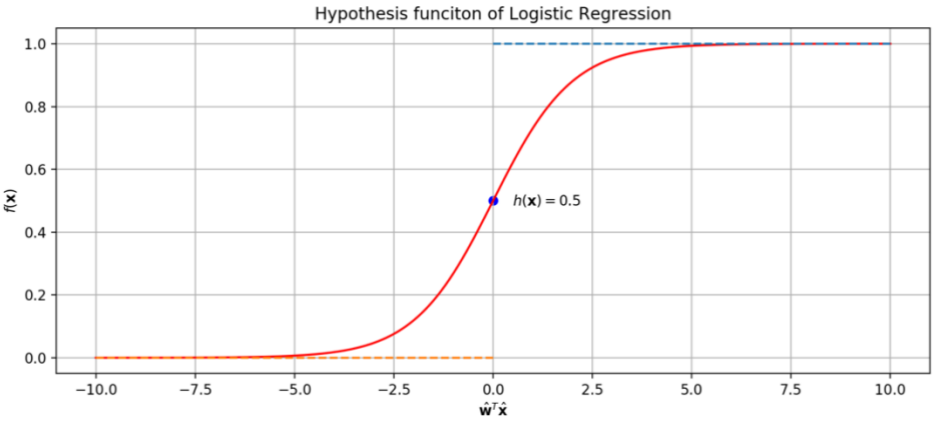



图 4-5 逻辑回归函数$h(\boldsymbol{x};\hat{\boldsymbol{w}})$相对于线性项$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$的曲线示意图

逻辑回归函数总体呈现出的 S 型曲线可以看作在保证函数连续可导的前提下对感知器函数sign$(\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}})$的一种近似。当$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$的值趋近于$+\infty$时，逻辑回归函数$h(\boldsymbol{x})$趋近于 1；当$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$的值趋近$-\infty$时，$h(\boldsymbol{x})$趋近于 0。对于二分类问题，可以将逻辑回归函数看作是对样本$\boldsymbol{x}$从属于$\omega_1$类的后验概率$P(\omega_1|\boldsymbol{x})$的估计。根据公式(4.19)，当$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} > 0$时，$h(\boldsymbol{x}) > 0.5$，按照最大后验概率分类准则，$\boldsymbol{x}$应被分给$\omega_1$类；当$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} < 0$时，$h(\boldsymbol{x}) < 0.5$，$\boldsymbol{x}$应被分给$\omega_2$类。显然，逻辑回归函数对应的决策面仍然是线性方程$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} = 0$描述的超平面，在效果上仍然等价于线性分类器。但从数学形式上看，逻辑回归算法的直接任务是对后验概率$P(\omega_1|\boldsymbol{x})$进行估计，其回归模型的映射函数在形式上是一个建立在线性项$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$基础上的非线性函数，因此既可以看作是一种**广义线性模型**，也可以看作是一种非线性回归模型。因此逻辑回归既是一个线性分类器，又是一个用于概率估计的非线性回归器。

- **逻辑回归函数的概率解释**

如果说逻辑回归函数是对后验概率$P(\omega_1|\boldsymbol{x})$的估计，为什么要设计成公式(4.19)描述的数学形式？这是一种主观上的选择还是一种客观的推论？在一个二分类问题中，设后验概率为$P_1 = P(\omega_1|\boldsymbol{x})$，则有$P_2 = P(\omega_2|\boldsymbol{x}) = 1 - P_1$。假设两类样本分别服从正态分布$\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1)$和$\mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2)$，根据贝叶斯公式有：

$$P_1 = P(\omega_1|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|\omega_1)P(\omega_1)}{P(\boldsymbol{x})}, \qquad P_2 = P(\omega_2|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|\omega_2)P(\omega_2)}{P(\boldsymbol{x})} \tag{4.20}$$

进一步假设两个类别的协方差矩阵$\Sigma_1 = \Sigma_2 = \Sigma$，将正态分布的概率密度函数公式(4.5)带入公式(4.20)，参考公式(4.7)的推导过程，得到：

$$\begin{aligned} \ln\frac{P_1}{1-P_1} &= \ln P(\boldsymbol{x}|\omega_1)P(\omega_1) - \ln P(\boldsymbol{x}|\omega_2)P(\omega_2) \\ &= \boldsymbol{w}^T\boldsymbol{x} + w_0 \\ &= \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \end{aligned} \tag{4.21}$$

其中$\boldsymbol{w}, w_0$的表达式详见公式(4.8)，$\hat{\boldsymbol{w}}$的定义详见 4.2.2 节的感知器算法。

公式(4.21)等式左侧的$P_1/1-P_1$，表示表示样本$\boldsymbol{x}$从属于$\omega_1$类和$\omega_2$类的后验概率比值，又称为**“几率”**（odds）。因此线性模型$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$可以看作是对几率的对数进行线性回归，因此逻辑回归算法又被称为**“对数几率回归”**。根据公式(4.21)可以进一步推导出$P_1, P_2$的表达式：

$$P_1 = P(\omega_1|\boldsymbol{x}) = \frac{1}{1+e^{-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}}}, \qquad P_2 = P(\omega_2|\boldsymbol{x}) = \frac{e^{-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}}}{1+e^{-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}}} \tag{4.22}$$

对比公式(4.22)中的$P_1$和公式(4.19)中的$h(\boldsymbol{x};\hat{\boldsymbol{w}})$，两者的函数形式完全相同。换言之，在二分类问题中，如果$\omega_1$和$\omega_2$均服从方差相同的正态分布，则可以推导出后验概率$P(\omega_1|\boldsymbol{x})$的数学形式就是公式(4.19)定义的逻辑回归函数。设样本$\boldsymbol{x}$的标签为$y$，且定义$y = 1$对应$\omega_1$类，$y = 0$对应$\omega_2$类，则后验概率$P(y|\boldsymbol{x})$可以统一写为：

$$P(y|\boldsymbol{x}) = P_1^y(1-P_1)^{1-y} \tag{4.23}$$

### 4.3.3 逻辑回归的目标函数

跟大多数机器学习算法相同，逻辑回归模型的学习是以模型参数向量$\hat{\boldsymbol{w}}$为自变量的优化问题，也需要设计目标函数。虽然逻辑回归函数是对后验概率$P(\omega|\boldsymbol{x})$的估计，但其根本任务是分类，因此应在贝叶斯决策论的框架下定义目标函数。设观测数据为$X = \{\boldsymbol{x}_i|i = 1, \dots, N\}$，对应的类别标签为$Y = \{y_i|i = 1, \dots, N\}$，假设观测相互独立，则有：

$$P(X,Y) = \prod_{i=1}^{N} P(\boldsymbol{x}_i, y_i) = \prod_{i=1}^{N} P(y_i|\boldsymbol{x}_i)p(\boldsymbol{x}_i) = \left[\prod_{i=1}^{N} P(y_i|\boldsymbol{x}_i)\right]\left[\prod_{i=1}^{N} p(\boldsymbol{x}_i)\right] \tag{4.24}$$

在逻辑回归算法中，后验概率$P(y_i|\boldsymbol{x}_i)$是一个参数化函数，可以写为$P(y_i|\boldsymbol{x}_i;\boldsymbol{\theta})$，其中$\boldsymbol{\theta}$为

参数向量。因此观测样本的概率$P(X,Y)$的似然函数可以写为$P(X,Y;\boldsymbol{\theta})$。观察公式(4.24)，概率$p(\boldsymbol{x}_i)$与参数向量$\boldsymbol{\theta}$无关，则可以根据公式(4.25)构造参数向量$\boldsymbol{\theta}$的**负对数似然函数**作为目标函数$J(\boldsymbol{\theta})$，则参数向量$\boldsymbol{\theta}$的最大似然估计等价于$J(\boldsymbol{\theta})$的最小化问题。

$$J(\boldsymbol{\theta}) = -\ln\left[\prod_{i=1}^{N} P(y_i|\boldsymbol{x}_i;\boldsymbol{\theta})\right] = -\sum_{i=1}^{N}\ln[P(y_i|\boldsymbol{x}_i;\boldsymbol{\theta})] \tag{4.25}$$

参考公式(4.19)给出的逻辑回归函数定义，可知$\boldsymbol{\theta} = \hat{\boldsymbol{w}} \in \Re^{d+1}$。将$\boldsymbol{\theta}$替换为$\hat{\boldsymbol{w}}$，再将公式(4.22)和(4.23)带入公式(4.25)，目标函数可以进一步写为：

$$J(\hat{\boldsymbol{w}}) = -\sum_{i=1}^{N}\left[y_i \ln\frac{1}{1+\exp(-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)} + (1-y_i)\ln\frac{\exp(-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)}{1+\exp(-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)}\right] \tag{4.26}$$

### 4.3.4 逻辑回归模型的优化

公式(4.26)给出的$J(\hat{\boldsymbol{w}})$的数学形式比较复杂，其偏导数方程不具备封闭解，因此一般采用梯度下降法进行求解。在给出$J(\hat{\boldsymbol{w}})$的导数形式前，首先介绍逻辑回归函数的导数性质。设$z = \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$，则逻辑回归函数$h(\boldsymbol{x};\hat{\boldsymbol{w}}) = S(z) = 1/(1+\exp(-z))$，这个函数 S( )，一般被称为 Sigmoid 函数，简称 S 函数。很容易推出 S 函数的导数为$S' = S(1-S)$，这意味着函数导数$S'$的数值计算只需要对原函数$S$的输出进行简单的运算即可。S 型函数良好的导数特性使其广泛应用于人工神经网络的激活函数，可以大幅度减少神经网络的梯度反传播计算量。基于 S 型函数的导数公式，很容易推出目标函数$J(\hat{\boldsymbol{w}})$的导数形式：

$$\begin{aligned}\frac{\partial J(\hat{\boldsymbol{w}})}{\partial \hat{\boldsymbol{w}}} &= \frac{-\sum_{i=1}^{N}[y_i \ln S + (1-y_i)\ln(1-S)]}{\partial \hat{\boldsymbol{w}}} \\ &= -\sum_{i=1}^{N}\left[y_i\frac{1}{S}S(1-S)\hat{\boldsymbol{x}}_i + (1-y_i)\frac{1}{1-S}\big(-S(1-S)\big)\hat{\boldsymbol{x}}_i\right] \\ &= -\sum_{i=1}^{N}[y_i\hat{\boldsymbol{x}}_i - S\hat{\boldsymbol{x}}_i] \\ &= \sum_{i=1}^{N}[h(\boldsymbol{x}_i;\hat{\boldsymbol{w}}) - y_i]\hat{\boldsymbol{x}}_i\end{aligned} \tag{4.27}$$

采用梯度下降法，则参数$\hat{\boldsymbol{w}}$的更新公式可以写为：

$$\hat{\boldsymbol{w}}(t+1) = \hat{\boldsymbol{w}}(t) - \rho_t\sum_{i=1}^{N}[h(\boldsymbol{x}_i;\hat{\boldsymbol{w}}(\boldsymbol{t})) - y_i]\hat{\boldsymbol{x}}_i \tag{4.28}$$

逻辑回归函数$h(\boldsymbol{x};\hat{\boldsymbol{w}})$与标签$y$之间差值可以被看作模型的预测误差，因此逻辑回归在本质上是一种误差学习（或称为残差学习），这一点与自动控制原理中的负反馈机制非常相似，都反映了维纳的控制论中的反馈控制思想。

总的来说，逻辑回归模型是一种非常有效的分类算法，其目标函数是定义在参数空间上的连续可导函数，因此即便对于线性不可分的数据集也能给出分类结果。但需要注意的是，

由于逻辑回归函数的目标函数不是凸函数，因此可能收敛于局部最优解，甚至在处理线性可分问题时仍有可能错误分类某些样本（这一点感知器也一样）。此外，逻辑回归对应的决策面仍然是线性超平面，因此逻辑回归的本质仍然是线性分类器，无法彻底解决线性不可分问题。逻辑回归函数需要对后验概率进行估计，因此是一个典型的判别模型，逻辑回归函数的后验概率解释有利于读者深刻理解生成模型与判别模型的关系。在目前最热门的深度学习领域，很多处理分类任务的深度网络所使用 softmax 层本质上是逻辑回归模型从二分类任务扩展到多分类任务的结果，因此学会并理解逻辑回归模型对于读者接受并理解神经网络与深度学习也具有重要的意义。

### ♦ 例题-基于逻辑回归算法求解 Iris 数据集的二分类问题

从 Iris 数据集中选择 Setosa 和 Versicolor 两类(每类各 50 个样本)作为训练样本集，记为$\boldsymbol{x}^{(i)}$，$i=1,2,\ldots,100$，$y^{(i)}$是$\boldsymbol{x}^{(i)}$类别标签。每个样本$\boldsymbol{x}^{(i)}$由花萼长度和花萼宽度两个特征表示，设样本标签$y^{(i)}=0,\forall i=1,\ldots 50$; $y^{(i)}=1,\forall i=51,\ldots,100$。请基于上述训练数据，使用逻辑回归算法训练一个线性分类器。

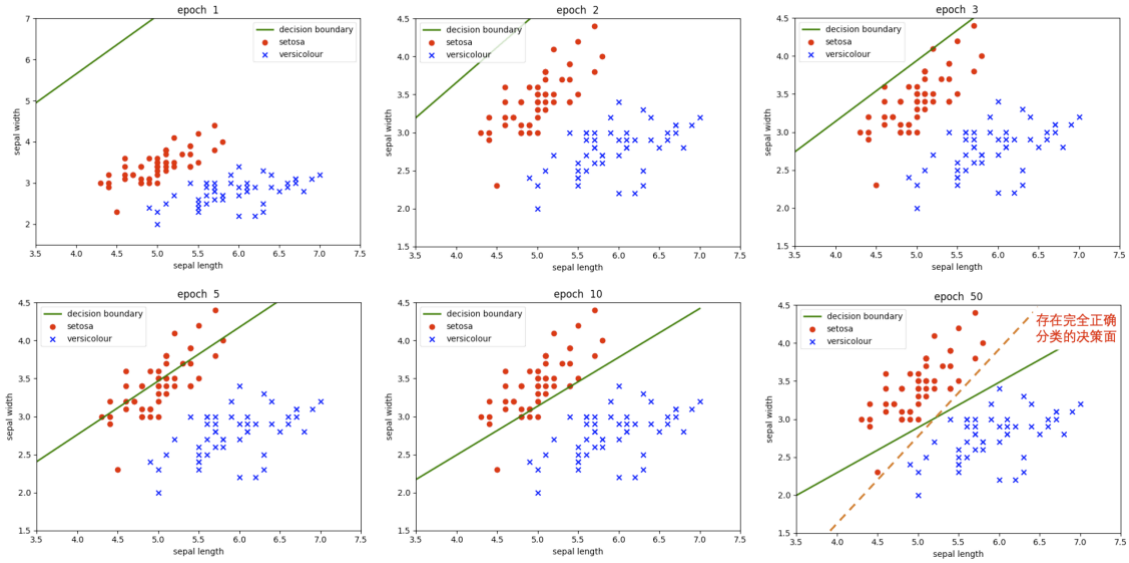



图 4-6　逻辑回归例题的决策面迭代收敛过程

**解答：**

第 1 步：

样本特征增广：$\hat{\boldsymbol{x}}^{(i)}=[x_1^{(i)},x_2^{(i)},1]^T$

第 2 步：

参数初始化：$\rho=0.1$，$\hat{\boldsymbol{w}}=[0,0,0]^T$

第 3 步：

基于训练样本计算目标函数梯度：

$$\frac{\partial J(\hat{\boldsymbol{w}})}{\partial \hat{\boldsymbol{w}}}=\frac{1}{M}\sum_{i=1}^{M}\left(h\left(\boldsymbol{x}^{(i)}\right)-y^{(i)}\right)\hat{\boldsymbol{x}}^{(i)}$$

$$= \frac{1}{100}\left\{\left(\frac{1}{1+e^{-[0,0,0]\begin{bmatrix}5.1\\3.5\\1\end{bmatrix}}}-0\right)\begin{bmatrix}5.1\\3.5\\1\end{bmatrix}+\cdots+\left(\frac{1}{1+e^{-[0,0,0]\begin{bmatrix}7.0\\3.2\\1\end{bmatrix}}}-1\right)\begin{bmatrix}7.0\\3.2\\1\end{bmatrix}+\cdots\right\}$$

$$= [-0.232,\ 0.164,0]^T$$

第 4 步：

更新参数向量$\hat{\boldsymbol{w}}$:

$$\hat{\boldsymbol{w}}(\mathrm{t}+1) = \hat{\boldsymbol{w}}(\mathrm{t}) - \rho\frac{\partial J(\hat{\boldsymbol{w}})}{\partial \hat{\boldsymbol{w}}} = \begin{bmatrix}0\\0\\0\end{bmatrix} - 0.1\times\begin{bmatrix}0\\-0.232\\0.164\end{bmatrix} = \begin{bmatrix}0\\0.0232\\-0.0164\end{bmatrix}$$

第 5 步：

重复第 3 步和第 4 步直至$\hat{\boldsymbol{w}}(\mathrm{t})$收敛（如图 4-6 所示）。

**结论：**基于图 4-6 展示的线性分类器决策面的迭代收敛过程，可以得到以下结论：

1）逻辑回归算法能够较快地通过模型迭代更新实现数据的线性分类。

2）在个别样本没有或无法正确分类时，逻辑回归算法仍能正常运行并给出学习结果。

3）逻辑回归可能收敛于局部最优解，不能确保给出最优分类结果。

## 4.4 线性回归

线性模型不仅可以用来解决分类任务，还可以广泛应用于回归任务。假设回归模型的映射函数为线性函数：

$$f(\boldsymbol{x};\hat{\boldsymbol{w}}) = \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}} \tag{4.29}$$

则可以通过构建一个优化问题来寻找参数向量$\hat{\boldsymbol{w}}^T$的最优解，该模型称为线性回归模型。

### 4.4.1 均方误差与最小二乘法

从任务需求出发，线性回归算法应使得模型$f(\boldsymbol{x})$的输出尽可能接近真实的$y$。需要首先定义一个包含参数向量$\hat{\boldsymbol{w}}$的目标函数$J(\hat{\boldsymbol{w}})$。典型的线性回归模型通常使用**均方误差**（Mean Square Error，MSE）作为目标函数，如公式(4.30)所示。

$$J(\hat{\boldsymbol{w}}) = \mathbb{E}\left[\left(y-f(\boldsymbol{x})\right)^2\right] = \mathbb{E}[(y-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}})^2] \approx \frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)^2 \tag{4.30}$$

均方误差作为线性回归模型的目标函数具有以下优点：

1）均方误差在定义上等价于预测值与真实值之间欧式距离的平方，易于理解；

2）均方误差为凸函数，使用基于梯度的优化方法不会陷入局部最小；

3）均方误差函数的导数形式相对简单，有封闭解。

利用公式(4.30)定义的均方误差代价函数对线性模型的参数向量$\hat{\boldsymbol{w}}$进行优化求解的方法称为**最小二乘法**（Least Square Method，LSM），其最优解称为**最小二乘解**。

### 4.4.2 最小二乘法的梯度优化解

为了更加形象地理解均方误差$J(\hat{\boldsymbol{w}})$与自变量$\hat{\boldsymbol{w}}$的关系，我们假设原始样本$\boldsymbol{x}$是一维数据，则有$\hat{\boldsymbol{w}}=[w_1,w_0]^T,\hat{\boldsymbol{x}}_i=[x_i,1]^T$，则均方误差函数可以写为：

$$\begin{aligned}J(\hat{\boldsymbol{w}})&=\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)^2\\&=\frac{1}{N}\sum_{i=1}^{N}[y_i-(w_1x_i+w_0)]^2\end{aligned}\tag{4.31}$$

显然，均方$J(\hat{\boldsymbol{w}})$是变量$w_1,w_0$的二元二次函数，对应于一个凸曲面，如图 4-7 所示。

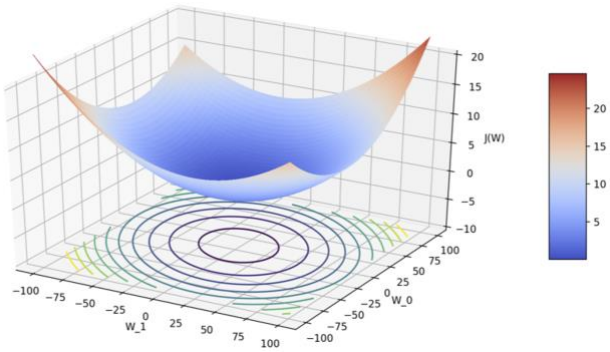



图 4-7 线性回归模型均方误差函数$J(\hat{\boldsymbol{w}})$的曲面示意图

凸函数的性质决定了该曲面只有一个极小值，即全局最小值。这意味着使用梯度下降法可以找到代价函数$J(\hat{\boldsymbol{w}})$的全局最优解。根据公式(4.30)，$J(\hat{\boldsymbol{w}})$的导数为：

$$\frac{\partial J(\hat{\boldsymbol{w}})}{\partial\hat{\boldsymbol{w}}}=\frac{2}{N}\sum_{i=1}^{N}-\hat{\boldsymbol{x}}_i(y_i-\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)=\frac{2}{N}\sum_{i=1}^{N}[(\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)-y_i]\hat{\boldsymbol{x}}_i=\frac{2}{N}\sum_{i=1}^{N}[f(\boldsymbol{x}_i)-y_i]\hat{\boldsymbol{x}}_i\tag{4.32}$$

根据梯度下降法，$\hat{\boldsymbol{w}}$迭代更新公式为：

$$\hat{\boldsymbol{w}}(t+1)=\hat{\boldsymbol{w}}(t)-\rho_t\sum_{i=1}^{N}\left[f\left(\boldsymbol{x}_i;\hat{\boldsymbol{w}}(t)\right)-y_i\right]\hat{\boldsymbol{x}}_i\tag{4.33}$$

首先对线性回归模型的增广参数向量$\hat{\boldsymbol{w}}$初始化，之后根据公式(4.33)反复迭代更新$\hat{\boldsymbol{w}}$直至收敛，可得到线性回归模型参数向量$\hat{\boldsymbol{w}}$的最小二乘解估计值。对比线性回归的梯度下降法更新公式(4.33)和逻辑回归算法的梯度更新公式(4.28)不难发现，两者几乎具有完全相同的数学形式，只是模型的假设函数不同，因此线性回归算法本质上也是一种误差学习。

### 4.4.3 最小二乘法的封闭解

虽然利用梯度下降法可以求取均方误差的最小二乘解，但需要经过多次迭代更新，速度

相对较慢。从公式(4.32)可以发现$J(\hat{\boldsymbol{w}})$的导数形式并不复杂，因此通过求解其偏导数方程可以直接得到该问题的**封闭解**（closed-form solution，又称**闭式解**或**解析解**），即用一个显式解析表达式来描述的解。根据公式(4.32)，$J(\hat{\boldsymbol{w}})$对参数向量$\hat{\boldsymbol{w}}$的偏导方程为：

$$\begin{aligned}\sum_{i=1}^{N}\left[\left(\hat{\boldsymbol{x}}_i\hat{\boldsymbol{x}}_i^T\right)\hat{\boldsymbol{w}}-\hat{\boldsymbol{x}}_i y_i\right]&=0\\ \left[\sum_{i=1}^{N}\hat{\boldsymbol{x}}_i\hat{\boldsymbol{x}}_i^T\right]\hat{\boldsymbol{w}}&=\sum_{i=1}^{N}\hat{\boldsymbol{x}}_i y_i\end{aligned}\tag{4.34}$$

为了让公式(4.34)的表述更加简洁，同时也为了让读者习惯方程的矩阵表达（这一点对于编程十分重要），我们用矩阵$X$和矢量$\boldsymbol{y}$分别表示全部训练样本的特征和标签：

$$X=\begin{bmatrix}\hat{\boldsymbol{x}}_1^T\\ \hat{\boldsymbol{x}}_2^T\\ \vdots\\ \hat{\boldsymbol{x}}_N^T\end{bmatrix}\in\Re^{N\times(d+1)};\qquad \boldsymbol{y}=\begin{bmatrix}y_1\\ y_2\\ \vdots\\ y_N\end{bmatrix}\in\Re^{N}\tag{4.35}$$

可以推导出：

$$X^TX=[\hat{\boldsymbol{x}}_1,\hat{\boldsymbol{x}}_1,\dots,\hat{\boldsymbol{x}}_N]\begin{bmatrix}\hat{\boldsymbol{x}}_1^T\\ \hat{\boldsymbol{x}}_2^T\\ \vdots\\ \hat{\boldsymbol{x}}_N^T\end{bmatrix}=\sum_{i=1}^{N}\hat{\boldsymbol{x}}_n\hat{\boldsymbol{x}}_n^T;\qquad X^T\boldsymbol{y}=[\hat{\boldsymbol{x}}_1,\hat{\boldsymbol{x}}_1,\dots,\hat{\boldsymbol{x}}_N]\begin{bmatrix}y_1\\ y_2\\ \vdots\\ y_N\end{bmatrix}=\sum_{n=1}^{N}\hat{\boldsymbol{x}}_i y_i\tag{4.36}$$

将公式(4.36)带入公式(4.34)，然后在公式左右两侧左乘$(X^TX)^{-1}$，可以得到：

$$\begin{aligned}(X^TX)\hat{\boldsymbol{w}}&=X^T\boldsymbol{y}\\ \hat{\boldsymbol{w}}&=(X^TX)^{-1}X^T\boldsymbol{y}\end{aligned}\tag{4.37}$$

其中矩阵$X^TX$被称为**自相关矩阵**（Auto-correlation Matrix），记为$R_X\in\Re^{(d+1)\times(d+1)}$；$X^T\boldsymbol{y}\in\Re^{d+1}$被称为**互相关向量**（Cross-correlation Vector），则线性回归的封闭解如公式(4.37)所示。

### 4.4.4 理解线性回归的“三面镜子”

为了更加深刻和准确地理解线性回归的本质，本书设置了三面不同的“镜子”，尝试从不同的角度去理解线性回归。

- **第一面镜子——矩阵运算**

如果把线性回归看作凸函数优化问题，其目标函数是均方误差函数$J(\hat{\boldsymbol{w}})$，待优化变量是线性模型参数$\hat{\boldsymbol{w}}$，优化过程可以采用梯度下降法也可以采用封闭解。如采用梯度下降法，将公式(4.35)带入公式(4.33)，则参数更新公式可以写为：

$$\hat{\boldsymbol{w}}(t+1)=\hat{\boldsymbol{w}}(t)-\eta_t X^T[X\hat{\boldsymbol{w}}(t)-\boldsymbol{y}]\tag{4.38}$$

从中不难返现，梯度下降法的参数向量更新公式从计算复杂度上最多只涉及到矩阵和向

量的乘法，即使当训练样本数量 $N$ 或样本维度$d$很大时，上述更新公式的计算速度仍然可以得到一定的保障。

如果采用封闭解方法，其数学形式更加简洁，也不需要反复迭代。但必须注意到，公式(4.37)中不但包含了两个矩阵的乘法$X^TX$，同时也包含了一个对称阵的求逆：$(X^TX)^{-1}$。当样本维度$d$很大时，自相关矩阵$R_X = X^TX \in \Re^{(d+1)\times(d+1)}$的存储和求逆都面临巨大的挑战。同时还需要注意封闭解在不同数据条件下的差异：

1） 当$d+1>N$时，自相关矩阵$R_X$的秩为$N$，不满秩，为奇异矩阵，不存在逆矩阵，但存在无穷多个左逆矩阵，因此$\hat{\boldsymbol{w}}$有无穷多组解；

2） 当$d+1\le N$时，自相关矩阵$R_X$满秩，$\hat{\boldsymbol{w}}$有唯一解。

- **第二面镜子——线性方程组**

从线性回归任务需求出发，最理想的参数向量$\hat{\boldsymbol{w}}$应使得$\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i = y_i, \forall i = 1,\dots,N$。根据公式(4.35)可以用矩阵形式描述为：

$$X\hat{\boldsymbol{w}} = \boldsymbol{y} \tag{4.39}$$

这意味着由$\hat{\boldsymbol{w}}$描述的线性回归模型能够毫无误差地预测每一个训练样本$\boldsymbol{x}$的输出$y$。则线性回归模型的参数学习问题可以看作是求解公式(4.39)中的矢量方程对应的线性方程组问题。为了便于区分，这里使用上角标标记样本的序号，下角标标记每个样本的特征序号，既$x_j^i, i = 1,\dots,N;\ j = 1,\dots,d$表示第$i$个样本的第$j$个分量，则方程组可以写为：

$$\begin{cases} w_1x_1^1 + w_2x_2^1 + \cdots + w_dx_d^1 + w_0 = y^1 \\ w_1x_1^2 + w_2x_2^2 + \cdots + w_dx_d^2 + w_0 = y^2 \\ \cdots \\ w_1x_1^N + w_2x_2^N + \cdots + w_dx_d^N + w_0 = y^N \end{cases} \tag{4.40}$$

公式(4.40)中$w_i, i = 0,\dots,d$为未知数，因此该线性方程组包含$N$个方程和$d+1$个未知数。假设上述$N$个线性方程线性不相关，则有：

1） 当$d+1>N$时，方程组有无穷多个解；

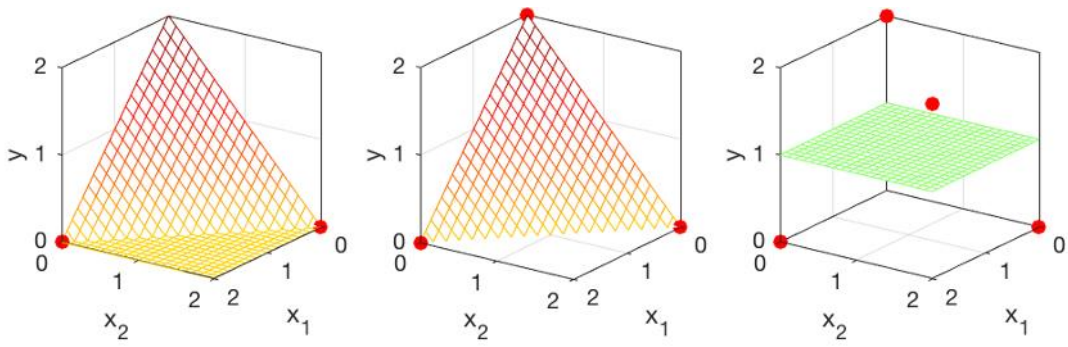
2） 当$d+1=N$时，有唯一解；

3） 当$d+1<N$时，无解；

需要特别提到的是，第二面镜子中方程组的解是指使得回归误差为零的解，而第一面镜子的矩阵运算则是使得回归误差最小化的解。这种定义上的差别导致在第一面镜子中$d+1<N$时存在唯一解，而在第二面镜子中$d+1<N$时无解。显然，最小二乘解可以保证均方误差的最小化，但不能保证均方误差为零。

- **第三面镜子——几何形态**

前两面镜子都是从代数角度去理解线性回归模型，而想要更加直观和形象的理解该算法还需要构建起线性模型求解的几何形态。对于线性回归模型$f(\boldsymbol{x}) = \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}} \in \Re^{d+1}$，假设$d=2$，则有$f(\boldsymbol{x}) = w_1x_1 + w_2x_2 + w_0$，该函数对应于$x_1$-$x_2$-$y$三维坐标系中的一个二维平面。此时，分别令样本数量$N = 2,3,4$。可以得到以下三种情况，如图 4-8 所示。




(a) $N=2$ (b) $N=3$ (c) $N=4$

图 4-8 三种情况下线性回归模型的几何形态。

图 4-8 中的红点对应于训练样本$\boldsymbol{x}$，其三维坐标为$(x_1, x_2, y)$。

1） $N=2$，$d+1>N$，能找到无穷多个平面穿过 2 个样本，有无穷多组解；

2） $N=3$，$d+1=N$，只能找到唯一平面穿过 3 个样本，有唯一解；

3） $N=4$，$d+1<N$，无法找到任何一个平面能令 4 个样本点都在该平面上，此时最小二乘解对应的平面使得所有样本点沿着$y$轴到该平面的截距平方和最小。

当$d>2$时，可以对上述情况进行扩展，则线性回归模型$f(\boldsymbol{x})=\hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}, \hat{\boldsymbol{w}}\in\Re^{d+1}$对应于一个镶嵌在$d+1$维空间中的$d$维超平面，上述三种解的条件和结论仍然成立。

### ♦ 知识扩展——过拟合与正则化（一）

线性模型的“三面镜子”对于我们理解模式识别模型的本质以及预测模型的性能具有深远的意义。首先回顾一下第一章提到的**“泛化”**、**“过拟合”**与**“欠拟合”**的概念。泛化能力是指模型对于未观察过的新数据的识别能力。如果模型在训练样本上的性能很好，但实际应用中的泛化能力很差，则称为过拟合现象；如果模型在训练样本集上的性能也不好，称为欠拟合现象。过拟合通常在训练数据规模有限但模型复杂度较高时出现，而欠拟合则通常在训练数据规模较大而模型复杂度较低时出现。训练数据规模一般用样本数量来衡量，模型复杂度则与模型参数数量密切相关。在线性回归模型中，样本数量为$N$，参数数量为$d+1$，因此可以对照理解线性回归的三面镜子来解释过拟合与欠拟合现象：

1） $d+1>N$，模型复杂度高于数据规模，存在无穷多组解；在无穷多个解中随机选择一个恰好具有良好泛化能力的解的可能性很小，容易出现过拟合现象；

2） $d+1=N$，模型复杂度与数据规模相当，存在唯一解；该解在训练样本集上达到理论上的最佳性能，且具有唯一性，因此模型的泛化性能相对较好；

3） $d+1<N$，无解；无法找到一组参数能够在当前训练样本集上得到令人满意的性能，属于欠拟合现象。

在机器学习方法的实际应用中，由于训练数据的标签已知，因此欠拟合现象比较容易被发现，也相对容易解决，只需适当增加模型复杂度即可。但由于实际应用数据极其标签无法事先获取，过拟合现象的发现与解决都是比较棘手的问题。在机器学习领域，专门用于解决过拟合问题，提高模型泛化能力的技术手段一般称为**正则化**（Regularization）。在诸多正则

化方法中，通过在目标函数中加入某些项来改变参数向量的优化结果是一种常见的策略，这些项一般被称为**正则化项**（Regularization Term）。仍以线性回归模型为例，可以考虑在目标函数中增加一个正则化项$\|\hat{\boldsymbol{w}}\|^2$，则改造后的目标函数可以写为：

$$J_R(\hat{\boldsymbol{w}}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}_i)^2 + \lambda\|\hat{\boldsymbol{w}}\|^2 \tag{4.41}$$

其中$\lambda$是正则化项系数，用于平衡均方误差项与正则化项的比例。关于正则化项的理解可以从两个角度入手：1）从目标函数的角度，正则化项$\|\hat{\boldsymbol{w}}\|^2$的加入会使得优化结果倾向于选择使得$\|\hat{\boldsymbol{w}}\|^2$更小的参数向量$\hat{\boldsymbol{w}}$，也就是希望$\hat{\boldsymbol{w}}$的每个元素的幅度尽量趋近于零，这种趋势从某种意义上限制了假设空间的大小；2）正则化项$\|\hat{\boldsymbol{w}}\|^2$也可以视为约束条件$\|\hat{\boldsymbol{w}}\|^2 \leq \xi$对应的拉格朗日乘子项，既在假设空间中划定一个球形区域$r \leq \xi$作为$\hat{\boldsymbol{w}}$的搜索范围。无论从哪个角度理解，正则化项都有助于降低模型复杂度，从而减少过拟合风险。对目标函数$J_R(\hat{\boldsymbol{w}})$求导，可以推导出新的封闭解表达式：

$$\hat{\boldsymbol{w}} = (X^TX + \lambda I)^{-1}X^T\boldsymbol{y} \tag{4.42}$$

可以看出，需要求逆的矩阵从$X^TX$变为了$X^TX + \lambda I$，其中$I \in \Re^{(d+1)\times(d+1)}$是一个单位阵，当$X^TX$为奇异阵时，$X^TX + \lambda I$仍然是个满秩的正定对称阵，存在唯一的逆矩阵，因此$\hat{\boldsymbol{w}}$有唯一解，可以从矩阵运算的角度规避过拟合风险。进一步分析，随着正则化系数$\lambda$的不断增大，$X^TX + \lambda I$的正定性逐渐增强，$\hat{\boldsymbol{w}}$的解倾向于从多个不稳定的解收敛到一个相对稳定的解，从横向看类似于一道山岭从山脚逐渐收缩到山脊的过程，因此该算法又称为岭回归（Ridge regression），系数$\lambda$称为岭系数。岭回归的本质是对线性模型真实参数的一种有偏估计，这种偏置（Bias）与机器学习理论中的“偏好”在本质上是一致的，即在诸多满足观测数据的假设中更倾向于选择哪一个假设。关于该问题更加全面和详细的论述将在模型估计一章给出。

## 4.5 线性回归的扩展

线性回归的假设函数为线性函数$h(\boldsymbol{x}) = \hat{\boldsymbol{w}}^T\hat{\boldsymbol{x}}$，因此难以胜任非线性观测数据的回归任务。为了解决这一问题，学界提出了线性回归的扩展方法，通过将模型假设函数设计为一系列非线性函数的线性组合，仍以均方误差作为目标函数进行优化学习，可以得到具有非线性拟合能力的回归算法。上述思路的代表性算法有基函数线性回归与核函数线性回归。

### 4.5.1 基函数线性回归

以一维数据的回归问题$y = f(x)$为例，模型的参数化假设函数可以写为：

$$h(x;\boldsymbol{\theta}) = \sum_{j=1}^{b}\theta_j\phi_j(x) = \boldsymbol{\theta}^T\boldsymbol{\phi}(x) \tag{4.43}$$

其中，$\phi_j(x)$是输入标量$x$的**基函数**向量$\boldsymbol{\phi}(x) = [\phi_1(x), \dots, \phi_b(x)] \in \Re^b$的第$j$个元素，$b$是基函数的个数，$\theta_j$是参数向量$\theta = [\theta_1, \dots, \theta_b]$的第$j$个元素。显然公式(4.43)给出的假设函数$h(x;\boldsymbol{\theta})$

是基函数$\phi_j(x), j=1,\dots,b$的线性加权组合，因此可以看作是一种广义线性模型。

在具体应用中，如果将基函数向量$\boldsymbol{\phi}(x)$设置成一簇特殊的非线性函数，假设函数$h(x;\boldsymbol{\theta})$就会具备非线性回归能力，例如以幂函数作为基函数：

$$\boldsymbol{\phi}(x)=[x^0,x^1,x^2,\dots,x^{b-1}]^T \tag{4.44}$$

或以三角函数作为基函数：

$$\boldsymbol{\phi}(x)=[1,\sin x,\cos x,\sin 2x,\cos 2x,\dots,\sin mx,\cos mx]^T \tag{4.45}$$

回顾一下高等数学中讲述的泰勒展开和傅里叶展开，不难发现，只要基函数的个数$b$足够大，上述两种基函数线性回归方法都能够以任意精度去拟合观测数据，一般将参数$b$称为模型的阶数。因此理论上只要阶数足够高，基函数线性回归可以解决任意复杂函数的拟合问题。需要注意的是，公式(4.43)只描述了输入$x$是一维标量的情况，但基函数线性回归方法可以扩展到任意$d$维数据$\boldsymbol{x}=[x_1,\dots,x_d]^T$。首先构造出最基础的标量基函数，设输入变量$\boldsymbol{x}$的维数为$d$，原始基函数数量为$b$，则一共有$d\times b$个基函数分量$\phi_j(x_i), i=1,\dots,d,\ j=1,\dots,b$，可以用一个矩阵$\Phi(\boldsymbol{x})\in\Re^{d\times b}$表示：

$$\Phi(\boldsymbol{x})=\begin{bmatrix}\phi_1(x_1) & \cdots & \phi_b(x_1)\\ \vdots & \ddots & \vdots\\ \phi_1(x_d) & \cdots & \phi_b(x_d)\end{bmatrix} \tag{4.46}$$

利用公式(4.46)中列出的$d\times b$个基函数分量可构造基函数线性回归模型。比较常见的构造方法包括**加法模型**和**乘法模型**两种。加法模型相对简单，只需要为每一个基函数向量$\phi_j(x_i)$配置一个对应的权重$\theta_{j,i}$即可，加法模型的假设函数为：

$$h(\boldsymbol{x};\boldsymbol{\theta})=\sum_{j=1}^{b}\sum_{i=1}^{d}\theta_{j,i}\phi_j(x_i) \tag{4.47}$$

乘法模型相对复杂，需要首先从矩阵$\Phi(\boldsymbol{x})$的每一行取出 1 个元素，即共取出 d 个元素，分别记为$\phi_{j_i}(x_i), j_i$是在第$i$行取出的元素的序号。然后将这$d$个元素相乘得到一个基础乘法项：$\phi_{j_1}(x_1)\phi_{j_2}(x_2)\dots\phi_{j_d}(x_d)$。由于$d$个连乘项的每一项都有$b$种选择，基础乘法项一共有$b^d$种可能。为每一个可能的基础乘法项赋予一个权重，记为$\theta_{j_1,j_2,\dots,j_d}$，则乘法模型的假设函数为：

$$h(\boldsymbol{x};\boldsymbol{\theta})=\sum_{j_1=1}^{b}\sum_{j_2=1}^{b}\cdots\sum_{j_d=1}^{b}\theta_{j_1,j_2,\dots,j_d}\left[\phi_{j_1}(x_1)\phi_{j_2}(x_2)\dots\phi_{j_d}(x_d)\right] \tag{4.48}$$

乘法模型的基础项更多，拟合能力更强，但同时参数也更多，计算复杂度更高。例如，当输入变量的维度为 100 时，即便只是一个阶数$b=2$的低阶乘法模型，也需要计算$2^{100}$个参数，这显然是无法接受的。因此乘法模型在实际应用中只能处理模型阶数和变量维度都比较低的情况。而加法模型的参数数量仅为$b\times d$，学习难度和计算量均较小，实用性更强。

另一个需要注意的问题是，无论乘法模型还是加法模型，也无论使用多项式还是三角函数，基函数线性回归模型中真正的参数只有权重系数$\boldsymbol{\theta}$，而基函数内部并没有可以学习的参

数。因此在真正使用基函数线性回归方法时，一般首先将原始数据样本$\boldsymbol{x}$通过基函数$\boldsymbol{\phi}(\boldsymbol{x})$转变为新的特征向量（新数据的样本数量不变，但样本特征向量维度通常会增加），然后再基于新的特征表达去训练一个线性回归模型。因此其学习的本质仍然是线性回归，这也是为什么基函数线性回归方法可以视为线性回归模型的扩展算法的原因。

### 4.5.2 核函数线性回归

在基函数线性回归方法中，无论是多项式或是三角函数，基函数都与训练样本无关。这意味着基函数线性回归提供的是一种通用性的解决方案。当样本数量不多但维度较高时，基函数线性回归模型通常需要较多的参数以保证模型容量足以胜任当前的回归任务，这会增加模型学习的难度与时间。是否能够寻找到一种与训练数据相关性更强的方法，从而降低模型的复杂度和学习难度呢？核函数线性回归提供了一种非常有益的思路。核函数线性回归，又称核模型，使用一种称为**核函数**的二元函数$K(x_1, x_2)$取代基函数作为线性回归模型的基本单元进行模型学习。

$$h(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{j=1}^{m} \theta_j K(\boldsymbol{x}, \boldsymbol{c}_j) \tag{4.49}$$

其中的核函数$K(\boldsymbol{x}, \boldsymbol{c}_j)$通常会使用高斯核函数：

$$K(\boldsymbol{x}, \boldsymbol{c}_j) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{2\sigma^2}\right) \tag{4.50}$$

高斯核函数$K(\boldsymbol{x}, \boldsymbol{c}_j)$仅在中心$\boldsymbol{c}_j$附近有较大的数值，一旦采样位置$\boldsymbol{x}$与中心$\boldsymbol{c}_j$的欧式距离超过方差根$\sigma$，函数值就会快速下降并趋近于 0。这意味着每一个核函数$K(\boldsymbol{x}, \boldsymbol{c}_j)$对应的权重系数$\theta_j$只需要对训练数据在$\boldsymbol{c}_j$附近的局部分布进行拟合，而不像多项式或者三角函数等基函数模型那样在整个训练数据集分布的区域上都需要拟合，这大大降低了核回归模型的学习难度。如图 4-9 所示，蓝色实线为训练数据，绿色虚线表示每一个核函数加权后对应的曲线，红色虚线为核函数线性模型的回归结果。

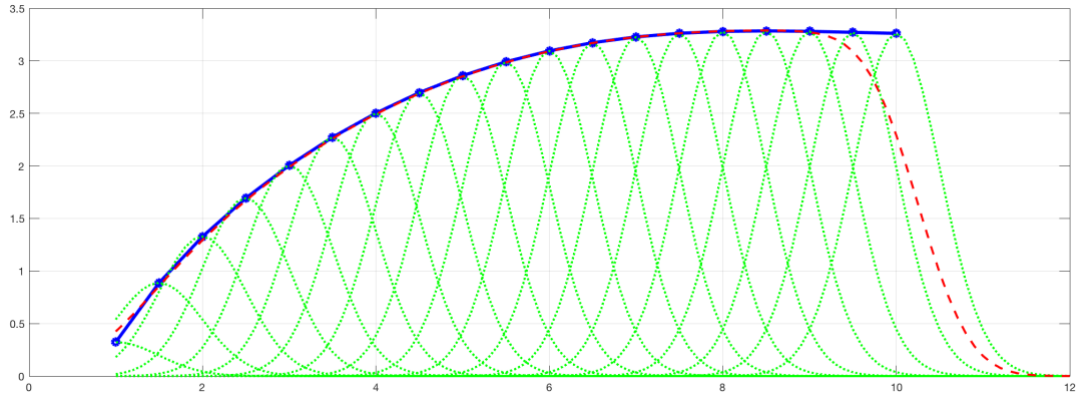



图 4-9 核函数线性回归拟合效果示意图。

显然，核函数线性回归的假设函数由一系列不同位置和高度的高斯函数加权组合而成，在每一个局部的函数形状主要由其邻域内的少数几个核函数决定。这说明核函数回归模型的关键在于核中心$\boldsymbol{c}_j$的选取。一种比较简单的方式是直接将全部$N$个训练样本均作为核中心，即$\boldsymbol{c}_j = \boldsymbol{x}_j, j = 1,2,\ldots.,m,$ 且$m = N$. 这种设置对于样本数量较少，但样本维度较高的问题有很好的学习效果。但对于样本数量非常大的问题显然是不划算的。此外，当训练数据分布密度并不均匀时，可以在样本聚集密度较高的区域，选择少量样本作为核中心或使用某些聚类算法求取类中心作为核中心。设定好核中心后将每一个训练样本向量$\boldsymbol{x}_i, i = 1,\ldots,N$分别代入$m$个核函数$K(\boldsymbol{x}, \boldsymbol{c}_j), j = 1,\ldots,m$，则数据由原来的$d$维变为$m$维，之后再使用标准线性回归模型就可以求出相应的参数$\theta_j, j = 1,\ldots,m.$

需要注意的是，核函数回归模型尽管在数学形式上与基函数回归类似，但其背后蕴藏的思想已经发生了非常大的变化。我们以预测一只猫的年龄为例，基函数回归通过定义牙齿的磨损程度、毛发的光泽度、瞳孔的浑浊度，奔跑的速度，平均睡觉时间等一系列特征来猜测猫的年龄；核回归方法则列出小猫、青年猫、成年猫、老猫等不同年龄段的猫作为原型，通过观测当前猫与猫原型之间的相似性来估计这只猫的年龄。因此，基函数回归是一种**基于规律**的描述，而核函数回归是一种**基于样例**的描述。这两种思想在很多经典的机器学习算法里面都有体现，例如：在概率估计上，高斯模型估计倾向于分布规律学习，而 Parzen 窗倾向于样例统计；在特征学习上，PCA（主成分分析）倾向于子空间规律，而稀疏编码倾向于样例学习；在分类算法上，决策树倾向于规律描述，而支持向量机倾向于样例选取。我们很难脱离具体问题去评价这两种思想的优劣，一般来说数据量大、维度低、分布一致性好的数据适合使用基于规律的描述；数据量小、维度高、分布不均匀的数据适合使用基于样例的描述。

# 本章思维导图


第四章 线性模型
线性回归扩展
基函数线性回归
基函数概念
泰勒展开
傅里叶展开
模型结构
加法模型
乘法模型
核函数线性回归
核函数概念
高斯核函数
核函数与基函数的差异
基于规律
基于样例
线性回归算法
目标函数
均方误差
距离解释
凸函数
导数/封闭解
最小二乘法
最小二乘解
优化策略
梯度下降法
梯度表达式
凸函数/全局最优解
参数更新公式
封闭解
数据的向量化与矩阵化表示
自相关矩阵与互相关向量
封闭解解析表达式
理解线性回归
矩阵运算
线性方程组
几何形态
无解/唯一解/无穷多解
过拟合与正则化(一)
泛化/过拟合/欠拟合
正则化项
岭回归
生成模型与判别模型
决策面
判别函数
线性模型参数
样本增广
参数向量表达
从贝叶斯决策到线性模型
线性分类器基础
基础概念
决策超平面
几何解释
感知器算法
优化问题
自变量
代价函数
错分样本集
符号系数
优化策略
梯度下降法
算法步骤
感知器局限性
线性不可分问题
不收敛
线性可分问题
无穷多最优解
逻辑回归算法
逻辑回归函数
Sigmoid函数
后验概率解释
对数几率回归
目标函数
最大似然估计
负对数似然函数
优化策略
逻辑回归函数的导数
梯度下降法
参数更新公式
误差学习

# 本章习题

## 一、填空题

1. 生成模型估计（　　）概率，判别模型估计（　　）概率。
2. 感知器算法中，错分样本 x 当前输出为−1，则对应的符号系数$\delta_x$取值为（　　）；感知器算法目标函数对参数向量$\widehat{\boldsymbol{w}}$的导数表达式为（　　）。
3. 逻辑回归函数的取值范围是（　　），其输出结果可以视为对（　　）概率的估计。
4. 利用线性回归算法解决标准的波士顿房价回归问题，自相关矩阵大小为（　　）。
5. 如果利用基函数线性回归算法中的加法模型解决波士顿房价回归问题，当$b=3$时（相当于最高阶次为平方项），模型共有（　　）个基函数项。

## 二、判断题

6. 对于两个服从正态分布的类别，最大后验概率准则的决策面必然是线性的。（　）
7. 逻辑回归算法不存在封闭解的显示表达式。（　）
8. 对于线性可分数据集，感知器算法一定能够学得完全分类正确的决策面。（　）
9. 4 个非共线性的三维样本，无法用线性回归模型得到唯一解。（　）
10. 基函数线性回归的目标函数不是凸函数，所以无法采用封闭解方法。（　）

## 三、选择题

11. 利用感知器算法求解一个二分类问题，样本$\boldsymbol{x}_i \in \Re^3, y_i \in \{+1,-1\}, i=1,\ldots,100$，算法共有多少个需要学习的参数：（　）

A. 3　　B. 4　　C. 100　　D. 不能确定

12. 逻辑回归算法主要用于解决哪种模式识别任务：（　）

A. 分类任务　　B. 回归任务　　C. 概率估计任务　　D. 异常检测任务

13. 已知 3 个训练样本及标签分别为：$x_1=1, y_1=1; x_2=2, y_2=1; x_3=3, y_3=2$，则线性回归算法的自相关矩阵为：（　）

A. $\begin{bmatrix}14 & 6\\ 6 & 3\end{bmatrix}$　　B. $\begin{bmatrix}14 & 9\\ 9 & 6\end{bmatrix}$　　C. $\begin{bmatrix}2 & 3 & 4\\ 3 & 5 & 7\\ 4 & 7 & 10\end{bmatrix}$　　D. $\begin{bmatrix}14 & 9 & 6\\ 9 & 6 & 4\\ 6 & 4 & 3\end{bmatrix}$

14. 上题中解的情况为：（　）

A. 无解　　B. 唯一解　　C. 无穷多组解　　D. 不能确定

15. 如上题中允许使用幂函数作为基函数进行线性回归，且$b=3$;则对应的自相关矩阵大小为：（　）

A. 1× 1　　B. 2× 2　　C. 3× 3　　D. 4× 4

## 四、简答题

16. 请分析总结感知器算法的不足之处，并将其与逻辑回归算法进行比较。
17. 如果逻辑回归算法使用均方误差作为目标函数，请推导其梯度下降法的更新公式。
18. 标签$y$是一个随机变量，由函数$\widehat{\boldsymbol{w}}^T\widehat{\boldsymbol{x}}$加上一个随机噪声生成，可以写为$y=\widehat{\boldsymbol{w}}^T\widehat{\boldsymbol{x}}+\epsilon$；其

中噪声$\epsilon$服从正态分布$\mathcal{N}(0,\sigma^2)$。设当前训练样本集为$X=\{\boldsymbol{x}_i|i=1,\dots,N\}, Y=\{y_i|i=1,\dots,N\}$；

**问题 1：** 试推导$\widehat{\boldsymbol{w}}$的最大似然解的数学形式；

**问题 2：** 假设$\widehat{\boldsymbol{w}}$的先验分布服从$d+1$元正态分布$\mathcal{N}\left(\mathbf{0}, I_{(d+1)\times(d+1)}\right)$，试采用最大后验概率估计法推导$\widehat{\boldsymbol{w}}$的最优解的数学形式。

19. 请设计一种正则化方法（岭回归算法除外）防止线性回归模型出现过拟合现象。
20. 请论证使用基函数线性回归方法是否能够给出参数向量的封闭解。

## 五、计算题

21. 已知两个类别$\omega_1$和$\omega_2$，其先验概率相等，两类的类条件概率密度服从正态分布，有$\mathrm{p}(\boldsymbol{x}|w_1)=\mathcal{N}(\mu_1,\Sigma)$和$\mathrm{p}(\boldsymbol{x}|w_2)=\mathcal{N}(\mu_2,\Sigma)$，且$\mu_1=\begin{bmatrix}0\\0\end{bmatrix}$，$\mu_2=\begin{bmatrix}3\\3\end{bmatrix}$，$\Sigma=\begin{bmatrix}1.1 & 0.3\\0.3 & 1.9\end{bmatrix}$，试写出上述问题每一类的判别函数$g_i(\boldsymbol{x}), i=1,2$的线性形式以及整个分类问题的决策面方程的线性形式。并判断样本$\boldsymbol{x}=[1.2,1.9]^T$属于哪一类，给出计算过程，并作图说明。
22. 已知决策面方程：$5x_1-x_2-1=0$，当前错分样本为：$(0.4,0.6), (0.1,-0.25)$。采用感知器算法进行模型参数学习：
    (1) 写出参数向量$\widehat{\boldsymbol{w}}$，并设定合适的学习步长$\eta$；
    (2) 列出迭代公式和计算结果；
    (3) 列出一次迭代后的决策面方程；
    (4) 画出一次迭代后的决策面；
    (5) 给出结论与分析。
23. 尝试设计一组线性可分的二分类数据，每个样本包含 1 个特征。请采用绘图软件画出该组数据对应的感知器算法目标函数曲面图，并勾画出使得目标函数值为零的区域。
24. 针对第 16 题的数据集，请画出逻辑回归模型目标函数的曲面图并用作图法找到最优解，将该解与梯度下降法求得的最优解进行比较，并分析和总结。
25. 已知包含 5 个一维样本的训练集为$X_{\mathrm{tr}}=\{0,1,2,3,4\}$，回归标签为$Y_{\mathrm{tr}}=\{0,0.6,2.2,4.8,8.4\}$；测试集包含 4 个样本：$X_{\mathrm{tr}}=\{0.5,1.5,2.5,3.5\}$。请采用线性回归、基函数线性回归和核函数线性回归分别给出模型训练结果及其在测试集上的预测结果，请自行选择基函数和核函数的形式与数量，并对比分析三种算法在该问题上的优劣。