

第2章 机器学习概述

机器学习是对能通过经验自动改进的计算机算法的研究。

— Mitchell [1997]

通俗地讲，机器学习（machine learning, ML）就是让计算机从数据中进行自动学习，得到某种知识（或规律）。作为一门学科，机器学习通常指一类问题以及解决这类问题的方法，即如何从观测数据（样本）中寻找规律，并利用学习到的规律（模型）对未知或无法观测的数据进行预测。

机器学习问题在早期的工程领域也经常称为模式识别（pattern recognition, PR），但模式识别更偏向于具体的应用任务，比如光学字符识别、语音识别，人脸识别等。这些任务的特点对于我们人类而言，这些任务很容易完成，但我们不知道自己是如何做到的，因此也很难人工设计一个计算机程序来解决这些任务。一个可行的方法是设计一个算法可以让计算机自己从有标注的样本上学习其中的规律，并用来完成各种识别任务。随着机器学习技术的应用越来越广，现在机器学习的概念逐渐替代模式识别，成为这一类问题及其解决方法的统称。

以手写体数字识别为例，我们需要让计算机能自动识别手写的数字。比如图2.1中的例子，将5识别为数字5，将6识别为数字6。手写数字识别是一个经典的机器学习任务，对人来说很简单，但对计算机来说却十分困难。我们很难总结每个数字的手写体特征，或者区分不同数字的规则，因此设计一套识别算法几乎是一项几乎不可能的任务。在现实生活中，很多问题都类似于手写体数字识别这类问题，比如物体识别、语音识别等。对于这类问题，我们不知道如何设计一个计算机程序来解决，即使可以通过一些启发式规则来实现，其过程

也是极其复杂的。因此，人们开始尝试采用另一种思路，即让计算机“看”大量的样本，并从中学习到一些经验，然后用这些经验来识别新的样本。要识别手写体数字，首先通过人工标注大量的手写体数字图像（即每张图像都通过人工标记了它是什么数字），这些图像作为训练数据，然后通过学习算法自动生成一套模型，并依靠它来识别新的手写体数字。这和人类学习过程也比较类似，我们教小孩子识别数字也是这样的过程。这种通过数据来学习的方法就称为机器学习的方法。



图 2.1 手写体数字识别示例。图片来源：MNIST 数据集 [LeCun et al., 1998]

本章先介绍机器学习的基本概念和要素，并较详细地描述一个简单的机器学习例子，线性回归。

2.1 基本概念

首先介绍下机器学习的一些基本概念：包括样本、特征、标签、模型、学习算法等。

以一个生活中的经验学习为例，假设我们要到市场上购买芒果，但是之前毫无挑选芒果的经验，那么我们如何通过学习来获取这些知识？

首先，我们从市场上随机选取一些芒果，列出每个芒果的特征（feature），包括颜色，大小，形状，产地，品牌，以及我们需要预测的标签（label）。标签可以连续值（比如关于芒果的甜度、水分以及成熟度的综合打分），也可以是离散值（比如“好”“坏”两类标签）。

特征也可以称为属性（attribute）。

样本 (sample)，也叫示例 (instance)。

在很多领域，数据集也经常称为语料库 (corpus)。并不是所有的样本特征都是数值型，需要通过转换表示为特征向量。

一个标记好特征以及标签的芒果可以看作是一个样本 (sample)。一组样本构成的集合称为数据集 (data set)。一般将数据集分为两部分：训练集和测试集。训练集 (training set) 中的样本是用来训练模型的，也叫训练样本 (training sample)，而测试集 (test set) 中的样本是用来检验模型好坏的，也叫测试样本 (test sample)。

我们用一个 d 维向量 $\mathbf{x} = [x_1, x_2, \dots, x_d]^T$ 表示一个芒果的所有特征构成的向量，称为特征向量 (feature vector)，其中每一维表示一个特征。

假设训练集由 N 个样本组成，其中每个样本都是独立同分布 (independent identically distributed, IID) 的，即独立地从相同的数据分布中抽取的，记为

$$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}. \quad (2.1)$$

给定训练集 \mathcal{D} ，我们希望让计算机自动寻找一个函数 $f(\mathbf{x}, \theta)$ 来建立每个样本特性向量 \mathbf{x} 和标签 y 之间的映射。对于一个样本 \mathbf{x} ，我们可以通过决策函数来预测其标签的值

$$\hat{y} = f(\mathbf{x}, \theta), \quad (2.2)$$

或标签的条件概率

$$p(y|\mathbf{x}) = f_y(\mathbf{x}, \theta), \quad (2.3)$$

其中 θ 为可学习的参数。

通过一个学习算法 (learning algorithm) \mathcal{A} ，在训练集上找到一组参数 θ^* ，使得函数 $f(\mathbf{x}, \theta^*)$ 可以近似真实的映射关系。这个过程称为学习 (learning) 或训练 (training) 过程，函数 $f(\mathbf{x}, \theta)$ 称为模型 (model)。

下次从市场上买芒果 (测试样本) 时，可以根据芒果的特征，使用学习到的模型 $f(\mathbf{x}, \theta^*)$ 来预测芒果的好坏。为了评价的公正性，我们还是独立同分布地抽取一组样本作为测试集 \mathcal{D}' ，并在测试集中所有样本上进行测试，计算预测结果的准确率。

$$Acc(f(\mathbf{x}, \theta^*)) = \frac{1}{|\mathcal{D}'|} \sum_{(\mathbf{x}, y) \in \mathcal{D}'} I(f(\mathbf{x}, \theta^*) = y), \quad (2.4)$$

其中 $I(\cdot)$ 为指示函数， $|\mathcal{D}'|$ 为测试集大小。

图 2.2 给出了机器学习的基本概念。对一个预测任务，输入特征向量为 \mathbf{x} ，输出标签为 y ，我们选择一个函数 $f(\mathbf{x}, \theta)$ ，通过学习算法 \mathcal{A} 和一组训练样本 \mathcal{D} ，找到一组最优的参数 θ^* ，得到最终的模型 $f(\mathbf{x}, \theta^*)$ 。这样就可以对新的输入 \mathbf{x} 进行预测。

学习算法在有些文献中也叫作学习器 (learner)。

第 2.7 节中会介绍更多的评价方法。

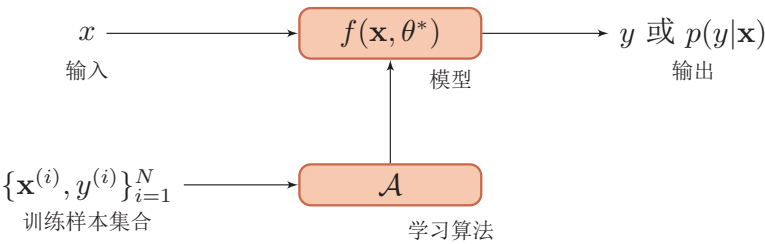


图 2.2 机器学习系统示例

2.2 机器学习的三个基本要素

机器学习是从有限的观测数据中学习（或“猜测”）出具有一般性的规律，并可以将总结出来的规律推广应用到未观测样本上。机器学习方法可以粗略地分为三个基本要素：模型、学习准则、优化算法。

2.2.1 模型

一个机器学习任务要先需要确定其输入空间 \mathcal{X} 和输出空间 \mathcal{Y} 。不同机器学习任务的主要区别在于输出空间不同。如果是两类分类问题， $\mathcal{Y} = \{+1, -1\}$ 。如果是 K 类分类问题， $\mathcal{Y} = \{1, 2, \dots, K\}$ 。如果是回归问题， $\mathcal{Y} = \mathbb{R}$ 。

这里，“输入空间”默认为样本的特征空间。

输入空间 \mathcal{X} 和输出空间 \mathcal{Y} 构成了一个样本空间。对于样本空间中的样本 $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ ，假定存在一个未知的真实映射函数 $g: \mathcal{X} \rightarrow \mathcal{Y}$ 使得

$$y = g(\mathbf{x}), \tag{2.5}$$

或者真实条件概率分布

$$p_r(y|\mathbf{x}), \tag{2.6}$$

机器学习的目标是找到一个模型来近似真实映射函数 $g(\mathbf{x})$ 或真实条件概率分布 $p_r(y|\mathbf{x})$ 。

由于我们不知道真实映射函数 $g(\mathbf{x})$ 或真实条件概率分布 $p_r(y|\mathbf{x})$ 的具体形式，只能根据经验来确定一个假设函数集合 \mathcal{F} ，称为假设空间（hypothesis space），然后通过观测其在训练集 \mathcal{D} 上的特性，从中选择一个理想的假设（hypothesis） $f^* \in \mathcal{F}$ 。

假设空间 \mathcal{F} 通常为一个参数化的函数族

$$\mathcal{F} = \{f(\mathbf{x}, \theta) | \theta \in \mathbb{R}^m\}, \quad (2.7)$$

其中 $f(\mathbf{x}, \theta)$ 为假设空间中的模型, θ 为一组可学习参数, m 为参数的数量。

常见的假设空间可以分为线性与非线性两种, 对应的模型 f 也分别称为线性模型和非线性模型。

2.2.1.1 线性模型

线性模型的假设空间为一个参数化的线性函数族,

$$f(\mathbf{x}, \theta) = \mathbf{w}^T \mathbf{x} + b, \quad (2.8)$$

对于分类问题, 一般为广义线性函数。参见公式 (3.3)。

其中参数 θ 包含了权重向量 \mathbf{w} 和偏置 b 。

2.2.1.2 非线性模型

广义的非线性模型可以写为多个非线性基函数 $\phi(\mathbf{x})$ 的线性组合

$$f(\mathbf{x}, \theta) = \mathbf{w}^T \phi(\mathbf{x}) + b, \quad (2.9)$$

其中 $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_K(\mathbf{x})]^T$ 为 K 个非线性基函数组成的向量, 参数 θ 包含了权重向量 \mathbf{w} 和偏置 b 。

如果 $\phi(\mathbf{x})$ 本身为可学习的基函数, 比如

$$\phi_k(\mathbf{x}) = h(\mathbf{w}_k^T \phi'(\mathbf{x}) + b_k), \forall 1 \leq k \leq K, \quad (2.10)$$

其中 $h(\cdot)$ 为非线性函数, $\phi'(\mathbf{x})$ 为另一组基函数, \mathbf{w}_k 和 b_k 为可学习的参数, 则 $f(\mathbf{x}, \theta)$ 就等价于神经网络模型。

2.2.2 学习准则

假设训练集 $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$ 是由 N 个独立同分布 (identically and independently distributed, IID) 的样本组成, 即每个样本 $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ 是从 \mathcal{X} 和 \mathcal{Y} 的联合空间中按照某个未知分布 $p_r(\mathbf{x}, y)$ 独立地随机产生的。这里要求样本分布 $p_r(\mathbf{x}, y)$ 必须是固定的 (虽然可以是未知的), 不会随时间而变化。如果 $p_r(\mathbf{x}, y)$ 本身可变的话, 我们就无法通过这些数据进行学习。

一个好的模型 $f(\mathbf{x}, \theta^*)$ 应该在所有 (\mathbf{x}, y) 的可能取值上都与真实映射函数 $y = g(\mathbf{x})$ 一致，即

$$f(\mathbf{x}, \theta^*) \sim y, \quad \forall (\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}, \quad (2.11)$$

或与真实条件概率分布 $p_r(y|\mathbf{x})$ 一致，即

$$f_y(\mathbf{x}, \theta^*) \sim p_r(y|\mathbf{x}), \quad \forall (\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}. \quad (2.12)$$

模型 $f(\mathbf{x}, \theta)$ 的好坏可以通过期望风险（Expected Risk） $\mathcal{R}(\theta)$ 来衡量。

期望风险也称为期望错误（expected error）。

$$\mathcal{R}(\theta) = \mathbb{E}_{(\mathbf{x}, y) \sim p_r(\mathbf{x}, y)} [\mathcal{L}(f(\mathbf{x}, \theta), y)], \quad (2.13)$$

其中 $p_r(\mathbf{x}, y)$ 为真实的数据分布， $\mathcal{L}(f(\mathbf{x}, \theta), y)$ 为损失函数，用来量化两个变量之间的差异。

2.2.2.1 损失函数

损失函数是一个非负实数函数，用来量化模型预测和真实标签之间的差异。下面介绍几种常用的损失函数。

0-1 损失函数 最直观的损失函数是模型预测的错误率，即 0-1 损失函数（0-1 Loss Function）。

$$\mathcal{L}(y, f(\mathbf{x}, \theta)) = \begin{cases} 0 & \text{if } y = f(\mathbf{x}, \theta) \\ 1 & \text{if } y \neq f(\mathbf{x}, \theta) \end{cases} \quad (2.14)$$

$$= I(y \neq f(\mathbf{x}, \theta)), \quad (2.15)$$

其中 $I(\cdot)$ 是指示函数。

虽然 0-1 损失能够客观的评价模型的好坏，但缺点是数学性质不是很好：不连续且不可导，比较难以优化。因此经常用连续可微的损失函数替代。

平方损失函数 平方损失函数（Quadratic Loss Function）经常用在预测标签 y 为实数值的任务中。

$$\mathcal{L}(y, f(\mathbf{x}, \theta)) = \frac{1}{2} (y - f(\mathbf{x}, \theta))^2. \quad (2.16)$$

平方损失函数一般不适用于分类问题。

参习题 2-1。

交叉熵损失函数 交叉熵损失函数（cross-entropy loss function）一般用于分类问题。假设样本的标签 $y \in \{1, \dots, C\}$ 为离散类别，模型 $f(\mathbf{x}, \theta) \in [0, 1]^C$ 的输出为类别标签的条件概率分布，即

$$P(y = c|\mathbf{x}) = f_c(\mathbf{x}, \theta), \quad (2.17)$$

并满足

$$f_c(\mathbf{x}, \theta) \in [0, 1], \quad \sum_{c=1}^C f_c(\mathbf{x}, \theta) = 1. \quad (2.18)$$

我们可以用一个 C 维的 one-hot 向量 \mathbf{y} 来表示样本标签。假设样本的标签为 k ，那么标签向量 \mathbf{y} 只有第 k 维的值为 1，其余元素的值都为 0。

标签向量 \mathbf{y} 可以看作是样本标签的真实概率分布，即第 c 维（记为 y_c ， $1 \leq c \leq C$ ）是类别为 c 的真实概率。假设样本的类别为 k ，那么它属于第 k 类的概率为 1，其它类的概率为 0。

交叉熵参见第 E.3.1 节。

对于两个概率分布，一般可以用交叉熵来衡量它们的差异。标签的真实分布 y_c 和模型预测分布 $f_c(\mathbf{x}, \theta)$ 之间的交叉熵为

$$\mathcal{L}(\mathbf{y}, f(\mathbf{x}, \theta)) = - \sum_{c=1}^C y_c \log f_c(\mathbf{x}, \theta). \quad (2.19)$$

假设对于三类分类问题，一个样本的标签向量为 $\mathbf{y} = [0, 0, 1]^T$ ，模型预测的标签分布为 $f(\mathbf{x}, \theta) = [0.3, 0.3, 0.4]^T$ ，则它们的交叉熵为

$$\begin{aligned} \mathcal{L}(\theta) &= -(0 \times \log(0.3) + 0 \times \log(0.3) + 1 \times \log(0.4)) \\ &= -\log(0.4). \end{aligned}$$

因为 \mathbf{y} 为 one-hot 向量，公式 (2.19) 也可以写为

$$\mathcal{L}(y, f(\mathbf{x}, \theta)) = -\log f_y(\mathbf{x}, \theta), \quad (2.20)$$

其中 $f_y(\mathbf{x}, \theta)$ 可以看作真实类别 y 的似然函数。因此，交叉熵损失函数也就是负对数似然损失函数（Negative Log-Likelihood Function）。

Hinge 损失函数 对于两类分类问题，假设 y 和 $f(x, \theta)$ 的取值为 $\{-1, +1\}$ 。Hinge 损失函数（Hinge Loss Function）为

$$\mathcal{L}(y, f(x, \theta)) = \max(0, 1 - yf(x, \theta)) \quad (2.21)$$

$$= |1 - yf(x, \theta)|_+. \quad (2.22)$$

2.2.2.2 风险最小化准则

一个好的模型 $f(\mathbf{x}, \theta)$ 应当有一个比较小的期望错误，但由于不知道真实的数据分布和映射函数，实际上无法计算期望风险 $\mathcal{R}(\theta; \mathbf{x}, y)$ 。给定一个训练集 $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$ ，我们可以计算的是经验风险（Empirical Risk），即在训练集上的平均损失。

经验风险也称为经验错误（Empirical Error）。

$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(y^{(i)}, f(x^{(i)}, \theta)). \quad (2.23)$$

因此，一个切实可行的学习准则是找到一组参数 θ^* 使得经验风险最小，

$$\theta^* = \arg \min_{\theta} \mathcal{R}_{\mathcal{D}}^{emp}(\theta), \quad (2.24)$$

这就是经验风险最小化（Empirical Risk Minimization, ERM）准则。

根据大数定理，当训练集大小 $|\mathcal{D}|$ 趋向于无穷大时，经验风险就趋向于期望风险。然而通常情况下，我们无法获取无限的训练样本，并且训练样本往往是真实数据的一个很小的子集或者包含一定的噪声数据，不能很好地反映全部数据的真实分布。经验风险最小化原则很容易导致模型在训练集上错误率很低，但是在未知数据上错误率很高。这就是所谓的过拟合（Overfitting）。

如何选择训练样本个数可以参考 PAC 理论，第 2.8.1 节。

定义 2.1—过拟合： 给定一个假设空间 \mathcal{F} ，一个假设 f 属于 \mathcal{F} ，如果存在其他的假设 f' 也属于 \mathcal{F} ，使得在训练集上 f 的损失比 f' 小，但在整个样本空间上 f' 比 f 的损失小，那么就假设 f 过度拟合训练数据 [Mitchell, 1997]。

过拟合问题往往是由于训练数据少和噪声以及模型能力强等原因造成的。为了解决过拟合问题，一般在经验风险最小化的基础上再引入参数的正则化（regularization），来限制模型能力，使其不要过度地最小化经验风险。这种准则就是结构风险最小化（Structure Risk Minimization, SRM）准则。

$$\theta^* = \arg \min_{\theta} \mathcal{R}_{\mathcal{D}}^{struct}(\theta) \quad (2.25)$$

$$= \arg \min_{\theta} \mathcal{R}_{\mathcal{D}}^{emp}(\theta) + \frac{1}{2} \lambda \|\theta\|_2^2 \quad (2.26)$$

$$= \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(y^{(i)}, f(x^{(i)}, \theta)) + \frac{1}{2} \lambda \|\theta\|^2, \quad (2.27)$$

更多的正则化方法参见第 7.7 节。

其中 $\|\theta\|_2$ 是 ℓ_2 范数的正则化项，用来减少参数空间，避免过拟合； λ 用来控制正则化的强度。

正则化项也可以使用其它函数，比如 ℓ_1 范数。 ℓ_1 范数的引入通常会使得参数有一定稀疏性，因此在很多算法中也经常使用。在贝叶斯学习的角度来讲，正则化是假设了参数的先验分布，不完全依赖训练数据。

总之，机器学习中的学习准则并不仅仅是拟合训练集上的数据，同时也要使得泛化错误最低。给定一个训练集，机器学习的目标是从假设空间中找到一个泛化错误较低的“理想”模型，以便更好地对未知的样本进行预测，特别是在训练集中出现的样本。因此，机器学习可以看作是一个从有限、高维、有噪声的数据上得到更一般性规律的泛化问题。

和过拟合相反的一个概念是欠拟合（underfitting），即模型不能很好地拟合训练数据，在训练集的错误率比较高。欠拟合一般是由于模型能力不足造成的。图 2.3 给出了欠拟合和过拟合的示例。

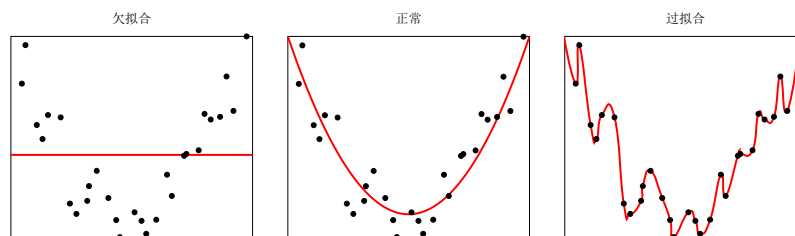


图 2.3 欠拟合和过拟合示例

2.2.3 优化算法

在确定了训练集 \mathcal{D} 、假设空间 \mathcal{F} 以及学习准则后，如何找到最优的模型 $f(\mathbf{x}, \theta^*)$ 就成了一个最优化（optimization）问题。机器学习的训练过程其实就是最优化问题的求解过程。

参数与超参数 在机器学习中，优化又可以分为参数优化和超参数优化。模型 $f(\mathbf{x}, \theta)$ 中的 θ 称为模型的参数，可以通过优化算法进行学习。除了可学习的参数 θ 之外，还有一类参数是用来定义模型结构或优化策略的，这类参数叫做超参数（hyper-parameter）。

在贝叶斯方法中，超参数可以理解为先验分布的参数，即控制模型参数分布的参数。

常见的超参数包括：聚类算法中的类别个数、梯度下降法的步长、正则项

的系数、神经网络的层数、支持向量机中的核函数等。超参数的选取一般都是组合优化问题，很难通过优化算法来自动学习。因此，超参数优化是机器学习的一个经验性很强的技术，通常是按照人的经验设定，或者通过搜索的方法对一组超参数组合进行不断试错调整。

超参数的优化参见第7.6节。

2.2.3.1 梯度下降法

为了充分利用凸优化中一些高效、成熟的优化方法，比如共轭梯度、拟牛顿法等，很多机器学习方法都倾向于选择合适的模型和损失函数以构造一个凸函数作为优化目标。但也有很多模型，比如神经网络，的优化目标是非凸的，只能退而求其次找到局部最优解。

不同机器学习算法的区别在于模型、学习准则（损失函数）和优化算法的差异。相同的模型也可以有不同的学习算法。比如，线性分类模型有感知器、logistic 回归和支持向量机，它们之间的差异在于使用了不同的学习准则和优化算法。

在机器学习中，最简单、常用的优化算法就是梯度下降法，即通过迭代的方法来计算所有样本上风险函数的最小值。

梯度下降法参见第C.1.0.2节。

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial \mathcal{R}(\theta)}{\partial \theta_t} \quad (2.28)$$

$$= \theta_t - \alpha \cdot \frac{1}{N} \sum_{n=1}^N \frac{\partial \mathcal{L}(y^{(n)}, f(\mathbf{x}^{(n)}, \theta_t))}{\partial \theta}. \quad (2.29)$$

其中 θ_t 为第 t 次迭代时的参数值， α 为搜索步长。在机器学习中 α 一般称为学习率（learning rate）。

2.2.3.2 提前停止

针对梯度下降的优化算法，除了加正则化项之外，还可以通过提前停止来防止过拟合。

在梯度下降训练的过程中，由于过拟合的原因，在训练样本上收敛的参数，并不一定在测试集上最优。因此，除了训练集和测试集之外，有时也会使用一个验证集（validation set）来进行模型选择，测试模型在验证集上是否最优。在每次迭代时，把新得到的模型 $f(\mathbf{x}, \theta)$ 在验证集上进行测试，并计算错误率。如果在验证集上的错误率不再下降，就停止迭代。这种策略叫提前停止（early stop）。如

验证集也叫开发集（development set）。

果没有验证集，可以在训练集上划分出一个小比例的子集作为验证集。图2.4给出了提前停止的示例。

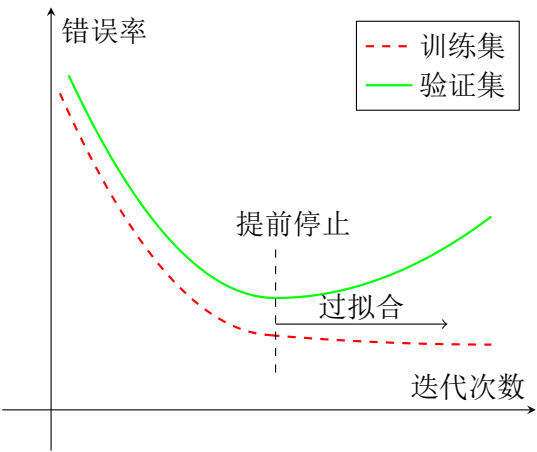


图 2.4 前提停止

2.2.3.3 随机梯度下降法

机器学习中的风险函数通过为每个样本损失函数的平均，并且每个样本都是独立的。如公式 (2.29) 所示，批量梯度下降每次迭代时需要计算每个样本上损失函数的梯度并求和。当训练集中的样本数量 N 很大时，空间复杂度比较高，每次迭代的计算开销也很大。为了减少每次迭代的计算复杂度，可以在每次迭代时随机选取一个样本，计算这个样本损失函数的梯度并更新参数，即随机梯度下降法（Stochastic Gradient Descent，SGD）。

随机梯度下降法也叫增量梯度下降。

批量梯度下降和随机梯度下降之间的区别在于每次迭代的优化目标是对所有样本的平均损失函数还是单个样本的损失函数。随机梯度下降因为实现简单，收敛速度也非常快，因此使用非常广泛。还有一种折中的方法就是小批量（Mini-Batch）随机梯度下降法。每次迭代时，只采用一小部分的训练样本，兼顾了批量梯度下降和随机梯度下降的优点。每次迭代时，选取一个包含 K 个样本的子集 \mathcal{I}_t ，计算这个子集上每个样本损失函数的梯度并进行平均，然后进行参数更新。

K 通常不会设置很大，一般在 $1 \sim 100$ 之间。在实际应用中为了提高计算效率，通常设置为 2 的 n 次方。

$$\theta \leftarrow \theta - \alpha \cdot \frac{1}{K} \sum_{(\mathbf{x}, y) \in \mathcal{I}_t} \frac{\partial \mathcal{L}(y, f(\mathbf{x}, \theta_t))}{\partial \theta}.$$

(2.30)

随机梯度下降法的训练过程如算法2.1所示。

算法 2.1: 随机梯度下降法

输入: 训练集 $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}, n = 1, \dots, N$, 验证集 \mathcal{V} , 学习率 α

```

1 随机初始化  $\theta$ ;
2 repeat
3   对训练集  $\mathcal{D}$  中的样本随机重排序;
4   for  $n = 1 \dots N$  do
5     从训练集  $\mathcal{D}$  中选取样本  $(\mathbf{x}^{(n)}, y^{(n)})$ ;
        // 更新参数
6      $\theta \leftarrow \theta - \alpha \frac{\partial \mathcal{L}(\theta; x^{(n)}, y^{(n)})}{\partial \theta}$ ;
7   end
8 until 模型  $f(\mathbf{x}, \theta)$  在验证集  $\mathcal{V}$  上的错误率不再下降;
输出:  $\theta$ 

```

在实际应用中, (小批量) 随机梯度下降方法有收敛快, 计算开销小的优点, 因此逐渐成为大规模的机器学习中的主要优化算法 [Bottou, 2010]。

2.3 机器学习的简单示例：线性回归

在本节中, 我们通过一个简单的模型 (线性回归) 来具体了解机器学习的一般过程, 以及不同学习准则 (经验风险最小化、结构风险最小化、最大似然估计、最大后验估计) 之间的关系。

线性回归 (Linear Regression) 是机器学习和统计学中最基础和广泛应用的模型, 是一种对自变量和因变量之间关系进行建模的回归分析。自变量数量为1时称为简单回归, 自变量数量大于1时称为多元回归。

从机器学习的角度来看, 自变量就是样本的特性向量 $\mathbf{x} \in \mathbb{R}^d$ (每一维对应一个自变量), 因变量是标签 y , 这里 $y \in \mathbb{R}$ 是连续值 (实数或连续整数)。假设空间是一组参数化的线性函数

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^T \mathbf{x} + b, \quad (2.31)$$

其中权重向量 \mathbf{w} 和偏置 b 都是可学习的参数, 函数 $f(\mathbf{x}; \mathbf{w}, b) \in \mathbb{R}$ 也称为线性模型。

为了简单起见，我们将公式 (2.31) 写为

$$f(\mathbf{x}; \hat{\mathbf{w}}) = \hat{\mathbf{w}}^T \hat{\mathbf{x}}, \quad (2.32)$$

其中 $\hat{\mathbf{w}}$ 和 $\hat{\mathbf{x}}$ 分别称为增广权重向量和增广特征向量。

$$\hat{\mathbf{x}} = \mathbf{x} \oplus 1 \triangleq \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ 1 \end{bmatrix}, \quad (2.33)$$

$$\hat{\mathbf{w}} = \mathbf{w} \oplus b \triangleq \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} = \begin{bmatrix} w_1 \\ \vdots \\ w_k \\ b \end{bmatrix}, \quad (2.34)$$

其中 \oplus 定义为两个向量的拼接操作。

不失一般性，在本章后面的描述中我们采用简化的表示方法，直接用 \mathbf{w} 和 \mathbf{x} 来表示增广权重向量和增广特征向量。即线性回归的模型简写为 $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x}$ 。

2.3.1 参数学习

给定一组包含 N 个训练样本的训练集 $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}, 1 \leq n \leq N$ ，我们希望能够学习一个最优的线性回归的模型参数 \mathbf{w} 。

我们介绍四种不同的参数估计方法：经验风险最小化、结构风险最小化、最大似然估计、最大后验估计。

2.3.1.1 经验风险最小化

由于线性回归的标签 y 和模型输出都为连续的实数值，因此平方损失函数平方损失函数参见第 2.2.2.1 节。非常合适来衡量真实标签和预测标签之间的差异。

根据经验风险最小化准则，训练集 \mathcal{D} 上的的经验风险定义为

$$\mathcal{R}(\mathbf{w}) = \sum_{n=1}^N \mathcal{L}(y^{(n)}, f(\mathbf{x}^{(n)}, \mathbf{w})) \quad (2.35)$$

为了简化起见，这里的
风险函数省略了 $\frac{1}{N}$ 。

$$= \frac{1}{2} \sum_{n=1}^N \left(y^{(n)} - \mathbf{w}^T \mathbf{x}^{(n)} \right)^2 \quad (2.36)$$

$$= \frac{1}{2} \|\mathbf{y} - X^T \mathbf{w}\|^2, \quad (2.37)$$

其中 $\mathbf{y} \in \mathbb{R}^N$ 是由每个样本的真实标签 $y^{(1)}, \dots, y^{(N)}$ 组成的列向量, $X \in \mathbb{R}^{(d+1) \times N}$ 是所有输入 $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}$ 组成的矩阵

$$X = \begin{pmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ x_d^{(1)} & x_d^{(2)} & \cdots & x_d^{(N)} \\ 1 & 1 & \cdots & 1 \end{pmatrix}. \quad (2.38)$$

参见习题2-2。

风险函数 $\mathcal{R}(\mathbf{w})$ 是关于 \mathbf{w} 的凸函数，其对 \mathbf{w} 的偏导数为

$$\frac{\partial \mathcal{R}(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{2} \frac{\partial \|\mathbf{y} - X^T \mathbf{w}\|^2}{\partial \mathbf{w}} \quad (2.39)$$

$$= -X(\mathbf{y} - X^T \mathbf{w}), \quad (2.40)$$

令 $\frac{\partial}{\partial \mathbf{w}} \mathcal{R}(\mathbf{w}) = 0$ ，得到最优的参数 \mathbf{w}^* 为

$$\mathbf{w}^* = (X X^T)^{-1} X \mathbf{y} \quad (2.41)$$

$$= \left(\sum_{n=1}^N \mathbf{x}^{(n)} (\mathbf{x}^{(n)})^T \right)^{-1} \left(\sum_{n=1}^N \mathbf{x}^{(n)} y^{(n)} \right). \quad (2.42)$$

这种求解线性回归参数的方法也叫最小二乘法估计（Least Square Estimation, LSE）。图2.5给出了用最小二乘法估计方法来进行参数学习的示例。

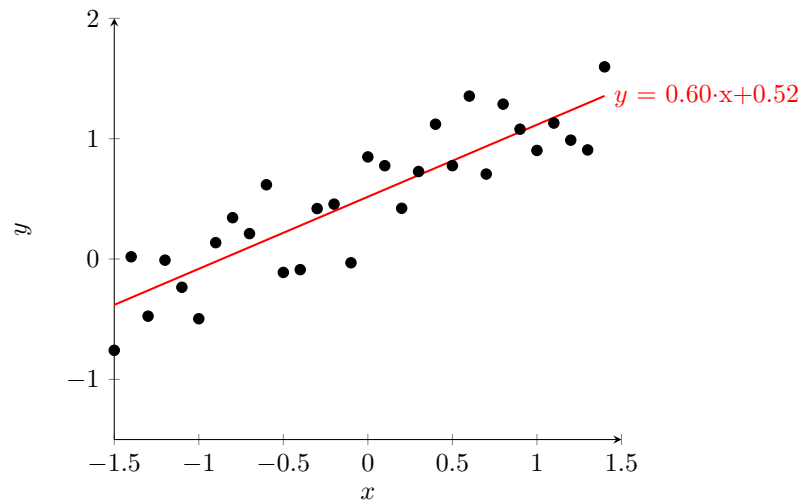


图 2.5 线性回归示例

一种常见的 XX^T 不可逆情况为 $N < (d + 1)$, 即样本的数量小于特征维数。

最小二乘法估计要求 XX^T 是满秩的，存在逆矩阵，也就是要求 \mathbf{x} 的每一维之间是线性不相关的。如果 XX^T 不可求逆矩阵，说明在训练数据上，输入的不同特征之间是线性相关的。

因此，当 XX^T 不可逆时，可以使用主成分分析等方法来预处理数据，消除不同特征之间的相关性，然后再使用最小二乘估计方法来求解。或者是通过用梯度下降法来求解。初始化 $\mathbf{w}_0 = 0$ ，通过下面公式进行迭代，

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha X(\mathbf{y} - X^T \mathbf{w}),$$

(2.43)

其中 α 是学习率。这种方法也称为最小均方误差（Least Mean Squares, LMS）算法。

2.3.1.2 结构风险最小化

最小二乘法估计的基本要求是各个特征之间要相互独立，保证 XX^T 可逆。但实际应用中，特征之间可能会有较大的共线性（multicollinearity），即 XX^T 的秩会接近于 0。数据集 X 上一些小的扰动就会导致 $(XX^T)^{-1}$ 发生大的改变，进而使得最小二乘法估计的计算变得很不稳定。为了解决这个问题，Hoerl and Kennard [1970] 提出了岭回归（Ridge Regression），给 XX^T 的对角线元素都

加上一个常数 λI 使得 $(XX^T + \lambda I)$ 的秩不为 0。最优的参数 \mathbf{w}^* 为

$$\mathbf{w}^* = (XX^T + \lambda I)^{-1} X \mathbf{y}, \quad (2.44)$$

其中 $\lambda > 0$ 为预先设置的超参数， I 为单位矩阵。

岭回归的解 \mathbf{w}^* 可以看做是结构风险最小化准则下的最小二乘法估计。

参习题 2-3。

$$\mathcal{R}(\mathbf{w}) = \frac{1}{2} \|\mathbf{y} - X^T \mathbf{w}\|^2 + \frac{1}{2} \lambda \|\mathbf{w}\|^2, \quad (2.45)$$

其中 $\lambda > 0$ 为正则化系数。

2.3.1.3 最大似然估计

机器学习任务可以分为两类，一类是样本的特征向量 \mathbf{x} 和标签 y 之间如果存在未知的函数关系 $y = h(\mathbf{x})$ ，另一类是条件概率 $p(y|\mathbf{x})$ 服从某个未知分布。第??中介绍的最小二乘估计是属于第一类，直接建模 \mathbf{x} 和标签 y 之间的函数关系。此外，线性回归还可以通过建模条件概率 $p(y|\mathbf{x})$ 的角度来进行参数估计。

假设标签 y 为一个随机变量，其服从以均值为 $f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \mathbf{x}$ 为中心，方差为 β^2 的高斯分布。

这里 \mathbf{x} 看作是确定值的参数。

$$p(y|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(y|\mathbf{w}^T \mathbf{x}, \beta^2) \quad (2.46)$$

$$= \frac{1}{\sqrt{2\pi}\beta} \exp\left(-\frac{(y - \mathbf{w}^T \mathbf{x})^2}{2\beta^2}\right). \quad (2.47)$$

参数 \mathbf{w} 在训练集 \mathcal{D} 上的似然函数 (likelihood) 为

似然函数是关于统计模型的参数的函数。

$$p(\mathbf{y}|X, \mathbf{w}, \beta) = \prod_{i=1}^N p(y^{(n)}|\mathbf{x}^{(n)}, \mathbf{w}, \beta) \quad (2.48)$$

似然 $p(x|w)$ 和 概率 $p(x|w)$ 之间的区别在于：概率 $p(x|w)$ 是描述固定参数 w 时，随机变量 x 的分布情况，而似然 $p(x|w)$ 则是描述已知随机变量 x 时，不同的参数 w 对其分布的影响。

$$= \prod_{i=1}^N \mathcal{N}(y^{(n)}|\mathbf{w}^T \mathbf{x}^{(n)}, \beta^2), \quad (2.49)$$

其中 $\mathbf{y} = [y^{(1)}, \dots, y^{(N)}]^T$ 为所有样本标签组成的向量， $X = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}]$ 为所有样本特征向量组成的矩阵。

为了方便计算，对似然函数取对数得到对数似然函数 (log likelihood)，

$$\log p(\mathbf{y}|X, \mathbf{w}, \beta) = \sum_{i=1}^N \log \mathcal{N}(y^{(n)}|\mathbf{w}^T \mathbf{x}^{(n)}, \beta^2). \quad (2.50)$$

最大似然估计 (Maximum Likelihood Estimate, MLE) 是指找到一组参数 \mathbf{w} 使得似然函数 $p(\mathbf{y}|X, \mathbf{w}, \beta)$ 最大, 等价于对数似然函数 $\log p(\mathbf{y}|X, \mathbf{w}, \beta)$ 最大。

参见习题??。

$$\text{令 } \frac{\partial \log p(\mathbf{y}|X, \mathbf{w}, \beta)}{\partial \mathbf{w}} = 0, \text{ 得到}$$

$$\mathbf{w}^{ML} = (X^T X)^{-1} X^T \mathbf{y}. \quad (2.51)$$

最小二乘估计解参见公式 (2.41)。

可以看出, 最大似然估计的解和最小二乘估计的解相同。

2.3.1.4 最大后验估计

贝叶斯公式参见公式 (D.31)。

假设参数 \mathbf{w} 为一个随机向量, 并服从一个先验分布 $p(\mathbf{w}|\alpha)$ 。根据贝叶斯公式, 那么参数 \mathbf{w} 的后验概率分布 (posterior distribution) 为

分母为和 \mathbf{w} 无关的常量。

$$p(\mathbf{w}|X, \mathbf{y}, \alpha, \beta) = \frac{p(\mathbf{w}, \mathbf{y}|X, \alpha, \beta)}{\sum_{\mathbf{w}} p(\mathbf{w}, \mathbf{y}|X, \alpha, \beta)} \quad (2.52)$$

$$\approx p(\mathbf{y}|X, \mathbf{w}, \beta) p(\mathbf{w}|\alpha), \quad (2.53)$$

其中 $p(\mathbf{y}|X, \mathbf{w}, \beta)$ 为 \mathbf{w} 的似然函数, 定义见公式 (2.48), $p(\mathbf{w}|\alpha)$ 为 \mathbf{w} 的先验。

统计推断参见第 11.2 节。

这种估计参数 \mathbf{w} 的后验概率分布的方法称为贝叶斯估计 (Bayesian Estimation), 是一种统计推断问题。采用贝叶斯估计的线性回归也称为贝叶斯线性回归 (Bayesian Linear Regression)。

贝叶斯估计是一种参数的区间估计, 即参数在一个区间上的分布。如果我们希望得到一个最优的参数值 (即点估计), 可以使用最大后验估计。最大后验估计 (Maximum A Posteriori Estimation, MAP) 是指最优参数为后验分布 $p(\mathbf{w}|X, \mathbf{y}, \alpha, \beta)$ 中概率密度最高的参数 \mathbf{w} 。

$$\mathbf{w}^{MAP} = \arg \max_{\mathbf{w}} p(\mathbf{y}|X, \mathbf{w}, \beta) p(\mathbf{w}|\alpha), \quad (2.54)$$

令似然函数 $p(\mathbf{y}|X, \mathbf{w}, \beta)$ 为公式 (2.49) 中定义的高斯密度函数, 先验分布 $p(\mathbf{w}|\alpha)$ 各向同性的高斯分布

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha I), \quad (2.55)$$

后验分布 $p(\mathbf{w}|X, \mathbf{y}, \alpha, \beta)$ 的对数为

$$\log p(\mathbf{w}|X, \mathbf{y}, \alpha, \beta) \approx \log p(\mathbf{y}|X, \mathbf{w}, \beta) + \log p(\mathbf{w}|\alpha) \quad (2.56)$$

$$\approx -\frac{1}{2\beta^2} \sum_{i=1}^N \left(y^{(n)} - \mathbf{w}^T \mathbf{x}^{(n)} \right)^2 - \frac{1}{2\alpha^2} \mathbf{w}^T \mathbf{w}, \quad (2.57)$$

$$= -\frac{1}{2\beta^2} \sum_{i=1}^N \|\mathbf{y} - \mathbf{w}^T X\|^2 - \frac{1}{2\alpha^2} \mathbf{w}^T \mathbf{w}. \quad (2.58)$$

可以看出，最大后验概率等价于平方损失的结构方法最小化，其中正则化系数 $\lambda = \beta^2/2\alpha^2$ 。

最大似然估计和贝叶斯估计可以看作是频率学派和贝叶斯学派对需要估计的参数 \mathbf{w} 的不同解释。当 $\alpha \rightarrow \infty$ 时，先验分布 $p(\mathbf{w}|\alpha)$ 退化为均匀分布，称为无信息先验（non-informative prior），最大后验估计退化为最大似然估计。

2.4 偏差-方差分解

为了避免过拟合，我们经常会在模型的拟合能力和复杂度之间进行权衡。拟合能力强的模型一般复杂度会比较高，容易导致过拟合。相反，如果限制模型的复杂度，降低其拟合能力，又可能会导致欠拟合。因此，如何在模型能力和复杂度之间取得一个较好的平衡对一个机器学习算法来讲十分重要。偏差-方差分解（Bias-Variance Decomposition）为我们提供一个很好的分析和指导工具。

以回归问题为例，假设样本的真实分布为 $p_r(\mathbf{x}, y)$ ，并采用平方损失函数，模型 $f(\mathbf{x})$ 的期望错误为

$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x}, y) \sim p_r(\mathbf{x}, y)} \left[(y - f(\mathbf{x}))^2 \right]. \quad (2.59)$$

那么最优的模型为

$$f^*(\mathbf{x}) = \mathbb{E}_{y \sim p_r(y|\mathbf{x})} [y]. \quad (2.60)$$

其中 $p_r(y|\mathbf{x})$ 为样本的真实条件分布， $f^*(\mathbf{x})$ 为使用平方损失作为优化目标的最优模型，其损失为

$$\varepsilon = \mathbb{E}_{(\mathbf{x}, y) \sim p_r(\mathbf{x}, y)} \left[(y - f^*(\mathbf{x}))^2 \right]. \quad (2.61)$$

损失 ε 通常是由于样本分布以及噪声引起的，无法通过优化模型来减少。

期望错误可以分解为

$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x}, y) \sim p_r(\mathbf{x}, y)} \left[\left(y - f^*(\mathbf{x}) + f^*(\mathbf{x}) - f(\mathbf{x}) \right)^2 \right] \quad (2.62)$$

$$= \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})} \left[\left(f(\mathbf{x}) - f^*(\mathbf{x}) \right)^2 \right] + \varepsilon, \quad (2.63)$$

本节介绍的偏差-方差分解以回归问题为例，但其结论同样适用于分类问题。

这里省略了模型参数 θ 。

参见习题2-8。

根据公式2.60, $\mathbb{E}_{\mathbf{x}} \mathbb{E}_y [y - f^*(\mathbf{x})] = 0$.

其中第一项是机器学习算法可以优化的真实目标，是当前模型和最优模型之间的差距。

在实际训练一个模型 $f(\mathbf{x})$ 时，训练集 \mathcal{D} 是从真实分布 $p_r(\mathbf{x}, y)$ 上独立同分布地采样出来的有限样本集合。不同的训练集会得到不同的模型。令 $f_{\mathcal{D}}(\mathbf{x})$ 表示在训练集 \mathcal{D} 学习到的模型，一个机器学习算法（包括模型以及优化算法）的能力可以用不同训练集上的模型的平均性能来评价。

对于单个样本 \mathbf{x} ，不同训练集 \mathcal{D} 得到模型 $f_{\mathcal{D}}(\mathbf{x})$ 和最优模型 $f^*(\mathbf{x})$ 的上的期望差距为

$$\begin{aligned} \mathbb{E}_{\mathcal{D}} \left[\left(f_{\mathcal{D}}(\mathbf{x}) - f^*(\mathbf{x}) \right)^2 \right] \\ = \mathbb{E}_{\mathcal{D}} \left[\left(f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] + \mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] - f^*(\mathbf{x}) \right)^2 \right] \end{aligned} \quad (2.64)$$

$$= \underbrace{\left(\mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] - f^*(\mathbf{x}) \right)^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}} \left[\left(f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] \right)^2 \right]}_{\text{variance}}. \quad (2.65)$$

其中第一项为偏差（bias），是指一个模型的在不同训练集上的平均性能和最优模型的差异。偏差可以用来衡量一个模型的拟合能力；第二项是方差（variance），是指一个模型在不同训练集上的差异，可以用来衡量一个模型是否容易过拟合。

结合公式 (2.63) 和 (2.65)，期望错误可以分解为

$$\mathcal{R}(f) = (\text{bias})^2 + \text{variance} + \varepsilon. \quad (2.66)$$

其中

$$(\text{bias})^2 = \mathbb{E}_{\mathbf{x}} \left[\left(\mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] - f^*(\mathbf{x}) \right)^2 \right], \quad (2.67)$$

$$\text{variance}^2 = \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathcal{D}} \left[\left(f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x})] \right)^2 \right] \right]. \quad (2.68)$$

最小化期望错误等价于最小化偏差和方差之和。

图2.6给出了机器学习算法的偏差和方差的四种不同组合情况。每个图的中心点为最优模型 $f^*(\mathbf{x})$ ，蓝点为不同训练集 \mathcal{D} 上得到的模型 $f_{\mathcal{D}}(\mathbf{x})$ 。图2.6a给出了一种理想情况，方差和偏差都比较小。图2.6b为高偏差低方差的情况，表示模型的泛化能力很好，但拟合能力不足。图2.6c为低偏差高方差的情况，表示

模型的拟合能力很好，但泛化能力比较差。当训练数据比较少时会导致过拟合。图2.6d为高偏差高方差的情况，是一种最差的情况。

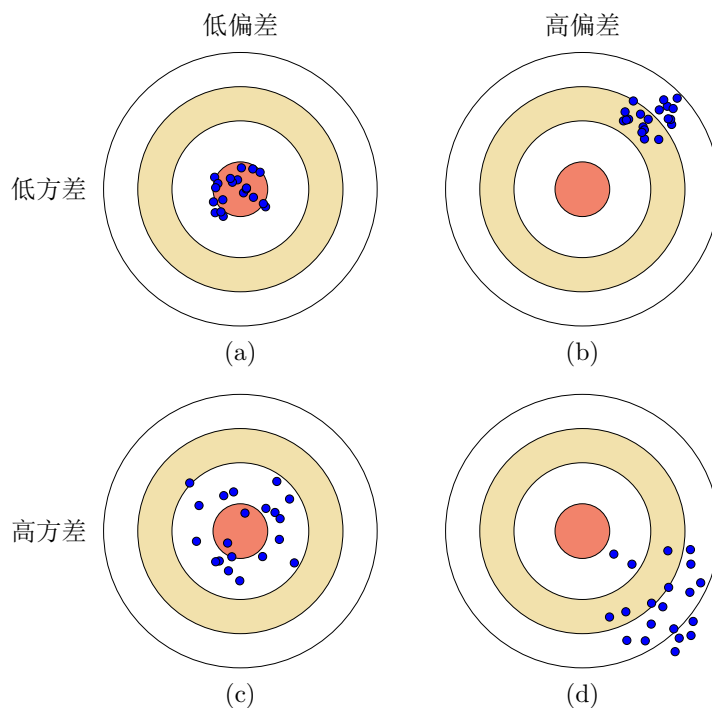


图 2.6 偏差和方差的四种组合

方差一般会随着训练样本的增加而减少。当样本比较多时，方差比较少，我们可以选择能力强的模型来减少偏差。然而在很多机器学习任务上，训练集上往往都比较有限，最优的偏差和最优的方差就无法兼顾。

随着模型复杂度的增加，模型的拟合能力变强，偏差减少而方差增大，从而导致过拟合。以结构错误最小化为例，我们可以调整正则化系数 λ 来控制模型的复杂度。当 λ 变大时，模型复杂度会降低，可以有效地减少方差，避免过拟合，但偏差会上升。当 λ 过大时，总的期望错误反而会上升。因此，一个好的正则化系数 λ 需要在偏差和方差之间取得比较好的平衡。图2.7给出了机器学习模型的期望错误、偏差和方差随复杂度的变化情况。最优的模型并不一定是偏差曲线和方差曲线的交点。

结构错误最小化参见公式(2.27)。

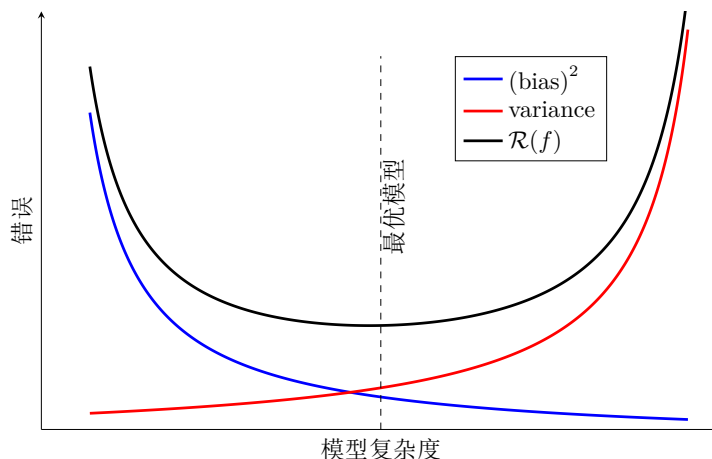


图 2.7 模型的期望错误、偏差和方差随复杂度的变化情况

偏差和方差分解给机器学习模型提供了一种分析途径，但在实际操作中难以直接衡量。一般来说，当一个模型在训练集上的错误率比较高时，说明模型的拟合能力不够，偏差比较高。这种情况可以增加数据特征、提高模型复杂度，减少正则化系数等操作来改进模型。当模型在训练集上的错误率比较低，但验证集上的错误率比较高时，说明模型过拟合，方差比较高。这种情况可以通过降低模型复杂度，加大正则化系数，引入先验等方法来缓解。此外，还有一种有效的降低方差的方法为集成模型，即通过多个高方差模型的平均来降低方差。

集成模型参见第16.1节。

2.5 机器学习算法的类型

机器学习算法可以按照不同的标准来进行分类。比如按函数 $f(\mathbf{x}, \theta)$ 的不同，机器学习算法可以分为线性模型和非线性模型；按照学习准则的不同，机器学习算法也可以分为统计方法和非统计方法。

但一般来说，我们会按照训练样本提供的信息以及反馈方式的不同，将机器学习算法分为以下几类：

监督学习 如果机器学习的目标是通过建模样本的特征 \mathbf{x} 和标签 y 之间的关系： $y = f(\mathbf{x}, \theta)$ 或 $p(y|\mathbf{x}, \theta)$ ，并且训练集中每个样本都有标签，那么这类机器学习称为监督学习（Supervised Learning）。根据标签类型的不同，监督学习又可以分为回归和分类两类。

1. 回归 (Regression) 问题中的标签 y 是连续值 (实数或连续整数), $f(\mathbf{x}, \theta)$ 的输出也是连续值。
2. 分类 (Classification) 问题中的标签 y 是离散的类别 (符号)。在分类问题中, 学习到模型也称为分类器 (Classifier)。分类问题根据其类别数量又可分为两类分类 (Binary Classification) 和多类分类 (Multi-class Classification) 问题。

无监督学习 无监督学习 (Unsupervised Learning) 是用来学习的数据不包含目标标签, 需要学习算法自动学习到一些有价值的信息。典型的无监督学习问题有聚类、密度估计、降维等。

无监督学习参见第9章。

强化学习 强化学习 (Reinforcement Learning), 也叫增强学习, 是一类通过交互来学习的机器学习算法。在强化学习中, 智能体根据环境的状态做出一个动作, 并得到即时或延时的奖励。智能体在和环境的交互中不断学习并调整策略, 以取得最大化的期望总回报。

强化学习参见第15章。

表2.1给出了三种机器学习类型的比较。

	监督学习	无监督学习	强化学习
训练样本	训练集 $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$	训练集 $\{\mathbf{x}^n\}_{n=1}^N$	智能体和环境交互的 轨迹 τ 和累积奖励 G_τ
优化目标	$y = f(\mathbf{x})$ 或 $p(y \mathbf{x})$	$p(\mathbf{x})$ 或带隐变量 \mathbf{z} 的 $p(\mathbf{x} \mathbf{z})$	期望总回报 $\mathbb{E}_\tau[G_\tau]$
学习准则	期望风险最小化 最大似然估计	最大似然估计 最小重构错误	策略评估 策略改进

表 2.1 三种机器学习类型的比较

监督学习需要每个样本都有标签, 而无监督学习则不需要标签。一般而言, 监督学习通常大量的有标签数据集, 这些数据集是一般都需要由人工进行标注, 成本很高。因此, 也出现了很多弱监督学习 (Weak Supervised Learning) 和半监督学习 (Semi-Supervised Learning) 的方法, 希望从大规模的无标注数据中充分挖掘有用的信息, 降低对标注样本数量的要求。强化学习和监督学习的不

同在于强化学习不需要显式地以“输入/输出对”的方式给出训练样本，是一种在线的学习机制。

2.6 表示学习

在实际应用中，我们要处理的数据类型也是多种多样，比较有代表性的类型为文本、音频、图像、视频等。不同类型的数据，其原始特征（Raw Features）的空间也不相同。比如一张灰度图像（像素数量为 n ）的特征空间为 $[0, 255]^n$ ，一个自然语言句子（长度为 L ）的特征空间为 $|\mathcal{V}|^L$ ，其中 \mathcal{V} 为词表集合。而很多机器学习算法要求是输入的样本特征是数学上可计算的，因此，在机器学习之前我们需要将这些不同类型的数据转换为向量表示。

也有一些机器学习算法（比如决策树）不需要向量形式的特征。

比如在手写体数字识别任务中，样本 \mathbf{x} 为待识别的图像，类别 $y \in \{0, 1, \dots, 9\}$ 分别对应10个不同数字。为了识别 \mathbf{x} 是什么数字，我们可以从图像中抽取一些特征。如果图像是一张大小为 $m \times n$ 的图像，其特征向量可以简单地表示为 $m \times n$ 维的向量，每一维的值为图像中对应像素的灰度值。为了提高模型准确率，也会经常加入一个额外的特征，比如直方图、宽高比、笔画数，纹理特征、边缘特征等。假设我们总共抽取了 d 个特征，这些特征可以表示为一个向量 $\mathbf{x} \in \mathbb{R}^d$ 。

而在文本情感分类任务中，样本 \mathbf{x} 为自然语言文本，类别 $y \in \{+1, -1\}$ 分别表示正面或负面的评价。为了将样本 \mathbf{x} 从文本形式转为为向量形式，我们可以使用词袋模型（Bag-Of-Words, BoW）模型。假设训练集合中的词都来自一个词典 \mathcal{V} ，则每个样本可以表示为一个 $|\mathcal{V}|$ 维的向量 $\mathbf{x} \in \mathbb{R}^{|\mathcal{V}|}$ ，向量中每一维 x_i 的值为词典中的第 i 个词是否在 \mathbf{x} 中出现。如果出现值为1，否则为0。

参见习题2-9。

表示学习 数据的原始特征往往不能直接用来预测，这些原始特征可能存在以下几种情况：（1）特征比较单一，需要进行（非线性的）组合才能发挥其作用；（2）特征之间冗余度比较高；（3）并不是所有的特征都对预测有用；（4）很多特征通常是易变的；（5）特征中往往存在一些噪声。

为了提高机器学习算法的能力，我们需要抽取有效、稳定的特征。传统的特征提取是通过人工方式进行的，需要大量的人工和专家知识。一个成功的机器学习系统通常需要尝试大量的特征，称为特征工程（Feature Engineering）。但即使这样，人工设计的特征在很多任务上也不能满足需要。因此，如何让机器自动地学习出有效的特征也成为机器学习中的一项重要研究内容，称为特征学习（Feature Learning），也叫表示学习（Representation Learning）。特征学习在一定程度上也可以减少预测模型复杂性、缩短训练时间、提高模型泛化

表示学习可以看作是一个特殊的机器学习任务，即有自己的模型、学习准则和优化方法。

能力、避免过拟合等。

2.6.1 特征选择和特征抽取

传统的特征学习一般是通过人为地设计一些准则，然后根据这些准则来选取有效的特征，具体又可以分为两种：特征选择和特征抽取。

2.6.1.1 特征选择

特征选择 (Feature Selection) 是选取原始特征集合的一个有效子集，使得基于这个特征子集训练出来的模型准确率最高。简单地说，特征选择就是保留有用特征，移除冗余或无关的特征。

子集搜索 一种直接的特征选择方法为子集搜索 (subset search)。假设原始特征数为 d ，则共有 2^d 个候选子集。特征选择的目标是选择一个最优的候选子集。最暴力的做法是测试每个特征子集，看机器学习模型哪个子集上的准确率最高。但是这种方式效率太低。常用的方法是采样贪心的策略：由空集合开始，每一轮添加该轮最优的特征，称为前向搜索 (forward search)；或者从原始特征集合开始，每次删除最无用的特征，称为反向搜索 (backward search)。

子集搜索方法又可以分为过滤式和包裹式的方法。

过滤式 (filter) 方法不依赖具体的机器学习模型。每次增加对最有信息量的特征，或删除最没有信息量的特征 [Hall, 1999]。信息量可以通过信息增益 (information gain) 来衡量。

包裹式 (wrapper) 方法是用后续机器学习模型的准确率来评价一个特征子集。每次增加对后续机器学习模型最有用的特征，或删除对后续机器学习任务最无用的特征。这种方法是将机器学习模型的包裹到特征选择过程的内部。

ℓ_1 正则化 此外，我们还可以通过 ℓ_1 正则化来实现特征选择。由于 ℓ_1 正则化会导致稀疏特征，间接实现了特征选择。

2.6.1.2 特征抽取

特征抽取 (Feature Extraction) 是构造一个新的特征空间，并将原始特征投影在新的空间中。以线性投影为例，原始特征向量 $\mathbf{x} \in \mathbb{R}^d$ ，经过线性投影后得到在新空间中的特征向量 \mathbf{x}' 。

$$\mathbf{x}' = P\mathbf{x}, \quad (2.69)$$

其中 $P \in \mathbb{R}^{k \times d}$ 为映射矩阵。

特征抽取又可以分为监督和无监督的方法。监督的特征学习的目标是抽取对一个特定的预测任务最有用的特征，比如线性判别分析（Linear Discriminant Analysis, LDA）。而无监督的特征学习和具体任务无关，其目标通常是减少冗余信息和噪声，比如主成分分析（Principle Components Analysis, PCA）。

表2.2列出了一些传统的特征选择和特征抽取方法。

	监督学习	无监督学习
特征选择	标签相关的子集搜索、 ℓ_1 正则化、决策树	标签无关的子集搜索
特征抽取	线性判别分析	主成分分析、独立成分分析、流形学习、自编码器

表 2.2 传统的特征选择和特征抽取方法

特征选择和特征抽取的优点是可以用较少的特征来表示原始特征中的大部分相关信息，去掉噪声信息，并进而提高计算效率和减小维度灾难（Curse Of Dimensionality）。对于很多没有正则化的模型，特征选择和特征抽取非常必要。经过特征选择或特征抽取后，特征的数量一般会减少，因此特征选择和特征抽取也经常称为维数约减或降维（Dimension Reduction）。

正则化参见第??节。

2.6.2 深度学习方法

传统的特征抽取一般是和预测模型的学习分离的。我们会先通过主成分分析或线性判别分析等方法抽取出有效的特征，然后再基于这些特征来训练一个具体的机器学习模型。

如果我们将特征的表示学习和机器学习的预测学习有机地统一到一个模型中，建立一个端到端的学习算法，可以有效地避免它们之间准则的不一致性。这种表示学习方法就称为深度学习（Deep Learning, DL）。深度学习方法的难点是如何评价表示学习对最终系统输出结果的贡献或影响，即贡献度分配问题。目前比较有效的模型是神经网络，即将最后的输出层作为预测学习，其它层作为表示学习。

参见第1.3.1节。

2.7 评价指标

为了衡量一个机器学习模型的好坏，需要给定一个测试集，用模型对测试集中的每一个样本进行预测，并根据预测结果计算评价分数。

对于分类问题，常见的评价标准有正确率、准确率、召回率和F值等。

给定测试集 $\mathcal{T} = (\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})$ ，假设标签 $y^{(n)} \in \{1, \dots, C\}$ ，用学习好的模型 $f(\mathbf{x}, \theta)$ 对测试集中的每一个样本进行预测，结果为 $Y = \hat{y}^{(1)}, \dots, \hat{y}^{(N)}$ 。

准确率 最常用的的平均指标为**准确率**（Accuracy）

$$ACC = \frac{1}{N} \sum_{n=1}^N I(y^{(n)} = \hat{y}^{(n)}), \quad (2.70)$$

其中 $I(\cdot)$ 为指示函数。

错误率 和准确率相对应的就是**错误率**（Error Rate）。

$$\mathcal{E} = 1 - ACC \quad (2.71)$$

$$= \frac{1}{N} \sum_{n=1}^N I(y^{(n)} \neq \hat{y}^{(n)}). \quad (2.72)$$

查准率和查全率 准确率是所有类别整体性能的平均，如果希望对每个类都进行性能估计，就需要计算查准率和查全率。查准率和查全率是广泛用于信息检索和统计学分类领域的两个度量值，在机器学习的评价中也被大量使用。

对于类别 c 来说，模型在测试集上的结果可以分为以下四种情况：

1. **真正例**（True Positive, TP）：一个样本的真实类别为 c 并且模型正确地预测为类别 c 。这类样本数量记为

TP 表示。

$$TP_c = \sum_{n=1}^N I(y^{(n)} = \hat{y}^{(n)} = c). \quad (2.73)$$

2. **假负例**（False Negative, FN）：一个样本的真实类别为 c ，模型错误地预测为其它类。这类样本数量记为

$$FN_c = \sum_{n=1}^N I(y^{(n)} = c \wedge \hat{y}^{(n)} \neq c). \quad (2.74)$$

3. 假正例 (False Positive, FP) 一个样本的真实类别为其它类, 模型错误地预测为类 c 。这类样本数量记为

$$FP_c = \sum_{n=1}^N I(y^{(n)} \neq c \wedge \hat{y}^{(n)} = c). \quad (2.75)$$

4. 真负例 (True Negative, TN): 一个样本的真实类别为其它类, 模型也预测为其它类。这类样本数量记为 TN_c 。对于类别 c 来说, 这种情况一般不需要关注。

这四种情况的关系如表2.3所示的混淆矩阵来表示。

		预测类别	
		$\hat{y} = c$	$\hat{y} \neq c$
真实类别	$y = c$	TP_c	FN_c
	$y \neq c$	FP_c	TN_c

表 2.3 类别 c 的预测结果的混淆矩阵

查准率 (Precision), 也叫精确率或精度, 类别 c 的查准率是所有预测为类别 c 的样本中, 预测正确的比例。

$$\mathcal{P}_c = \frac{TP_c}{TP_c + FP_c}, \quad (2.76)$$

查全率 (Recall), 也叫召回率, 类别 c 的查全率是所有真实标签为类别 c 的样本中, 预测正确的比例。

$$\mathcal{R}_c = \frac{TP_c}{TP_c + FN_c}, \quad (2.77)$$

F 值 (F Measure) 是一个综合指标, 为查准率和查全率的调和平均。

$$\mathcal{F}_c = \frac{(1 + \beta^2) \times \mathcal{P}_c \times \mathcal{R}_c}{\beta^2 \times \mathcal{P}_c + \mathcal{R}_c}, \quad (2.78)$$

其中 β 用于平衡查全率和查准率的重要性, 一般取值为 1。 $\beta = 1$ 时的 F 值称为 $F1$ 值, 是查准率和查全率的调和平均。

宏平均和微平均 为了计算分类算法在所有类别上的总体准确率、召回率和 F1 值, 经常使用两种平均方法, 分别称为宏平均 (macro average) 和微平均 (micro average) [Yang, 1999]。

宏平均是每一类的性能指标的算术平均值,

$$\mathcal{P}_{macro} = \frac{1}{C} \sum_{c=1}^C \mathcal{P}_c, \quad (2.79)$$

$$\mathcal{R}_{macro} = \frac{1}{C} \sum_{c=1}^C \mathcal{R}_c, \quad (2.80)$$

$$\mathcal{F1}_{macro} = \frac{2 \times \mathcal{P}_{macro} \times \mathcal{R}_{macro}}{\mathcal{P}_{macro} + \mathcal{R}_{macro}}. \quad (2.81)$$

而微平均是每一个样本的性能指标的算术平均。对于单个样本而言, 它的准确率和召回率是相同的 (要么都是 1, 要么都是 0)。因此准确率的微平均和召回率的微平均是相同的。同理, F1 值的微平均指标是相同的。当不同类别的样本数量不均衡时, 使用宏平均会比微平均更合理些。宏平均会更关注于小类别上的评价指标。

在实际应用中, 我们通过调整分类模型的阈值来进行更全面的评价, 比如 AUC (Area Under Curve)、ROC (Receiver Operating Characteristic) 曲线、PR (Precision-Recall) 曲线等。此外, 很多任务还有自己专门的评价方式, 比如 TopN 准确率。

交叉验证 交叉验证 (Cross Validation) 是一种比较好的可能衡量机器学习模型的统计分析方法, 可以有效避免划分训练集和测试集时的随机性对评价结果造成的影响。我们可以把原始数据集平均分为 K 组不重复的子集, 每次选 $K - 1$ 组子集作为训练集, 剩下的一组子集作为验证集。这样可以进行 K 次试验并得到 K 个模型。这 K 个模型在各自验证集上的错误率的平均作为分类器的评价。

有的文献上, F1 值的宏平均为 $\mathcal{F1}_{macro} = \frac{1}{C} \sum_{c=1}^C \mathcal{F1}_c$ 。

关于更详细的模型评价指标, 可以参考《机器学习》[周志华, 2016] 的第二章。

K 一般大于 3。

2.8 理论和定理

在机器学习中, 有一些非常有名的理论或定理, 对理解机器学习的内在特性非常有帮助。

2.8.1 PAC 学习理论

当使用机器学习方法来解决某个特定问题时，通常靠经验或者多次试验来选择合适的模型、训练样本数量以及学习算法收敛的速度等。但是经验判断或多次试验往往成本比较高，也不太可靠，因此希望有一套理论能够分析问题难度、计算模型能力，为学习算法提供理论保证，并指导机器学习模型和学习算法的设计。这就是计算学习理论。计算学习理论（computational learning theory）是关于机器学习的理论基础，其中最基础的理论就是可能近似正确（Probably Approximately Correct, PAC）学习理论。

机器学习中一个很关键的问题是期望错误和经验错误之间的差异，称为泛化错误（Generalization Error）。泛化错误可以衡量一个机器学习模型 f 是否可以很好地泛化到未知数据。

“泛化错误”在有些文献中也指“期望错误”，指在未知样本上的错误。

$$\mathcal{G}_{\mathcal{D}}(f) = \mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f). \quad (2.82)$$

根据大数定律，当训练集大小 $|\mathcal{D}|$ 趋向于无穷大时，泛化错误趋向于 0，即经验风险趋近于期望风险。

$$\lim_{|\mathcal{D}| \rightarrow \infty} \mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f) = 0. \quad (2.83)$$

由于我们不知道真实的数据分布 $p(\mathbf{x}, y)$ ，也不知道真实的目标函数 $g(\mathbf{x})$ ，因此期望从有限的训练样本上学习到一个期望错误为 0 的函数 $f(\mathbf{x})$ 是不切实际的。因此，需要降低对学习算法能力的期望，只要求学习算法可以以一定的概率学习到一个近似正确的假设，即 PAC 学习。

PAC 学习可以分为两部分：

一是“近似正确”（Approximately Correct）。一个假设 $f \in \mathcal{F}$ 是“近似正确”的，是指其在泛化错误 $\mathcal{G}_{\mathcal{D}}(f)$ 小于一个界限 ϵ 。 ϵ 一般为 0 到 $\frac{1}{2}$ 之间的数， $0 < \epsilon < \frac{1}{2}$ 。如果 $\mathcal{G}_{\mathcal{D}}(f)$ 比较大，说明模型不能用来做正确的“预测”。

二是“可能”。一个学习算法 \mathcal{A} 有“可能”以 $1 - \delta$ 的概率学习到这样一个“近似正确”的假设。 δ 一般为 0 到 $\frac{1}{2}$ 之间的数， $0 < \delta < \frac{1}{2}$ 。

一个 PAC 可学习的算法是指该学习算法能够在多项式时间内从合理数量的训练数据中学习到一个近似正确的 $f(\mathbf{x})$ 。

$$P\left((\mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f)) \leq \epsilon\right) \geq 1 - \delta, \quad (2.84)$$

其中 ϵ, δ 是和样本数量 n 、假设空间 \mathcal{F} 相关的变量。如果固定 ϵ, δ ，可以反过来计算出样本复杂度为

$$n(\epsilon, \delta) \geq \frac{1}{2\epsilon^2} (\ln |\mathcal{F}| + \ln \frac{2}{\delta}), \quad (2.85)$$

其中 $|\mathcal{F}|$ 为假设空间的大小。

[Blum et al., 2016] 定理 5.3。

PAC 学习理论也可以帮助分析一个机器学习方法在什么条件下可以学习到一个近似正确的分类器。从公式 (2.85) 可以看出，如果希望模型的假设空间越大，泛化错误越小，其需要的样本数量越多。

2.8.2 没有免费午餐定理

没有免费午餐定理 (No Free Lunch Theorem, NFL) 是由 Wolpert 和 Macerday 在最优化理论中提出的。没有免费午餐定理证明：对于基于迭代的最优化算法，不存在某种算法对所有问题（有限的搜索空间内）都有效。如果一个算法对某些问题有效，那么它一定在另外一些问题上比纯随机搜索算法更差。也就是说，不能脱离具体问题来谈论算法的优劣，任何算法都有局限性。必须要“具体问题具体分析”。

没有免费午餐定理对于机器学习算法也同样适用。不存在一种机器学习算法适合于任何领域或任务。如果有人宣称自己的模型在所有问题上都好于其他模型，那么他肯定是在吹牛。

2.8.3 丑小鸭定理

丑小鸭定理 (Ugly Duckling Theorem) 是 1969 年由渡边慧提出的 [Watanabe, 1969]。“丑小鸭与白天鹅之间的区别和两只白天鹅之间的区别一样大”。这个定理初看好像不符合常识，但是仔细思考后是非常有道理的。因为世界上不存在相似性的客观标准，一切相似性的标准都是主观的。如果以体型大小的角度来看，丑小鸭和白天鹅的区别大于两只白天鹅的区别；但是如果以基因的角度来看，丑小鸭与它父母的差别要小于他父母和其他白天鹅之间的差别。

渡边慧 (Satosi Watanabe), 1910-1993, 美籍日本学者，理论物理学家，也是模式识别的最早研究者之一。

这里的“丑小鸭”是指白天鹅的幼雏，而不是“丑陋的小鸭子”。

2.8.4 奥卡姆剃刀

奥卡姆剃刀 (Occam's Razor) 是由 14 世纪逻辑学家 William of Occam 提出的一个解决问题的法则：“如无必要，勿增实体”。奥卡姆剃刀思想和机器学习上正则化思想十分类似：简单的模型泛化能力更好。如果有两个性能相近

的模型，我们应该选择更简单的模型。因此，在机器学习的学习准则上，我们经常会引入参数正则化来限制模型能力，避免过拟合。

奥卡姆剃刀的一种形式化是最小描述长度（Minimum Description Length, MDL）原则，即对一个数据集 \mathcal{D} ，最好的模型 $f \in \mathcal{F}$ 是会使得数据集的压缩效果最好，即编码长度最小。

最小描述长度也可以通过贝叶斯学习的观点来解释 [MacKay and Mac Kay, 2003]。模型 f 在数据集 \mathcal{D} 上的对数后验概率为

$$\max_f \log p(f|\mathcal{D}) = \max_f \log p(\mathcal{D}|f) + \log p(f) \quad (2.86)$$

$$= \min_f -\log p(\mathcal{D}|f) - \log p(f), \quad (2.87)$$

其中 $-\log p(f)$ 和 $-\log p(\mathcal{D}|f)$ 可以分别看作是模型 f 的编码长度和在该模型下数据集 \mathcal{D} 的编码长度。也就是说，我们不但要使得模型 f 可以编码数据集 \mathcal{D} ，也要使得模型 f 尽可能简单。

2.8.5 归纳偏置

在机器学习中，很多学习算法经常会对学习的问题做一些假设，这些假设就称为归纳偏置（Inductive Bias）Mitchell [1997]。比如在最近邻分类器中，我们会假设在特征空间中，一个小的局部区域中的大部分样本都同属一类。在朴素贝叶斯分类器中，我们会假设每个特征的条件概率是相互独立的。

归纳偏置在贝叶斯学习中也经常称为先验（priors）。

2.9 总结和深入阅读

本章简单地介绍了机器学习的基础知识，并为后面介绍的神经网络进行一些简单的铺垫。机器学习算法虽然种类繁多，但其中三个基本的要素为：模型、学习准则、优化算法。大部分的机器学习算法都可以看作是这三个基本要素的不同组合。如果需要快速全面地了解机器学习的基本概念和体系可以阅读《Pattern Classification》[Duda et al., 2001]、《Machine Learning: a Probabilistic Perspective》[Robert, 2014] 和《机器学习》[周志华, 2016]。

目前机器学习中最主流的一类方法是统计学习方法，将机器学习问题看作是统计推断问题，并且又可以进一步分为频率学派和贝叶斯学派。频率学派将模型参数 θ 看作是固定常数；而贝叶斯学派将参数 θ 看作是随机变量，并且存在

某种先验分布。想进一步深入了解统计学习的知识，可以阅读《Pattern Recognition and Machine Learning》[Bishop, 2007] 和《The Elements of Statistical Learning》[Hastie et al., 2009]。关于统计学习理论的知识可以参考 [Vapnik, 1998]。

此外，机器学习中一个重要内容是表示学习。Bengio et al. [2013] 系统地给出了关于表示学习的全面综述。传统的表示学习方法，即特征选择和特征抽取，可以参考《机器学习》[周志华, 2016] 中的第 10 章和第 11 章。

习题

习题 2-1 分析为什么平方损失函数不适用于分类问题。

习题 2-2 在线性回归中，如果我们给每个样本 $(\mathbf{x}^{(n)}, y^{(n)})$ 赋予一个权重 $r^{(n)}$ ，经验风险函数为

$$\mathcal{R}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N r^{(n)} \left(y^{(n)} - \mathbf{w}^T \mathbf{x}^{(n)} \right)^2, \quad (2.88)$$

计算其最优参数 \mathbf{w}^* ，并分析权重 $r^{(n)}$ 的作用。

习题 2-3 在线性回归中，验证岭回归的解为结构风险最小化准则下的最小二乘法估计，见公式 (2.45)。

习题 2-4 在线性回归中，若假设标签 $y \sim \mathcal{N}(\mathbf{w}^T \mathbf{x}, \beta)$ ，并用最大似然估计来优化参数时，验证最优参数为公式 (2.51) 的解。

习题 2-5 假设有 N 个样本 $x^{(1)}, x^{(2)}, \dots, x^{(N)}$ 服从正态分布 $\mathcal{N}(\mu, \sigma^2)$ ，其中 μ 未知。(1) 使用最大似然估计来求解最优参数 μ^{ML} 。(2) 若参数 μ 为随机变量，并服从正态分布 $\mathcal{N}(\mu_0, \sigma_0^2)$ ，使用最大后验估计来求解最优参数 μ^{MAP} 。

习题 2-6 在习题 2-5 中，证明当 $N \rightarrow \infty$ 时，最大后验估计趋向于最大似然估计。

习题 2-7 验证公式 (2.60)。

习题 2-8 验证公式 (2.65)。

习题 2-9 验证公式 (2.65)。

习题 2-10 对于一个三类分类问题，数据集的真实标签和模型的预测标签如下：

真实标签	1	1	2	2	2	3	3	3	3
预测标签	1	2	2	2	3	3	3	1	2

分别计算模型的查准率、查全率、F1 值以及它们的宏平均和微平均。

参考文献

周志华. 机器学习. 清华大学出版社, 北京, 2016. ISBN 978-7-302-206853-6.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Christopher M. Bishop. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007. ISBN 9780387310732.

Avrim Blum, John Hopcroft, and Ravindran Kannan. Foundations of data science. *Vorabversion eines Lehrbuchs*, 2016.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In

Proceedings of COMPSTAT, pages 177–186. Springer, 2010.

Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern classification, 2nd Edition*. Wiley, 2001. ISBN 9780471056690.

Mark Andrew Hall. Correlation-based feature selection for machine learning. 1999.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The elements of statistical learning: data mining, inference, and prediction, 2nd Edition*. Springer series in statistics. Springer, 2009. ISBN 9780387848570.

Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estima-

- tion for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.
- Yann LeCun, Corinna Cortes, and Christopher JC Burges. MNIST handwritten digit database. Online, 1998. URL <http://yann.lecun.com/exdb/mnist>.
- David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.
- T.M. Mitchell. *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill, 1997. ISBN 9780071154673.
- Christian Robert. Machine learning, a probabilistic perspective, 2014.
- Vladimir Vapnik. *Statistical learning theory*. Wiley, New York, 1998.
- S Watanable. Knowing and guessing: A quantitative study of inference and information, 1969.
- Yiming Yang. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90, 1999.