第 1 章 绪论

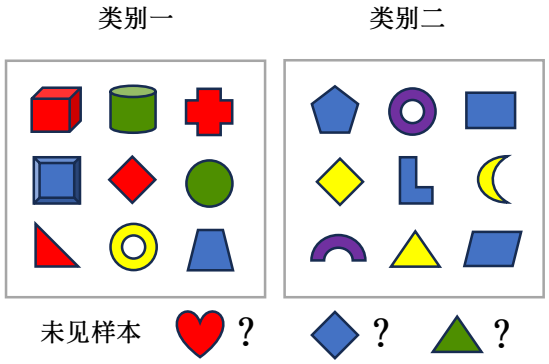
1.1 引言

2012 年之后的十年，人工智能取得了令人瞩目进展和突破。比如，2016 年 3 月美国 Google 公司 DeepMind 团队研发的人工智能程序 AlphaGo 以总比分 4 比 1 战胜了人类世界围棋冠军、职业九段的韩国选手李世石。2017 年 5 月 AlphaGo 又以 3 比 0 战胜了当时世界排名第一的中国选手柯洁。2020 年 11 月，DeepMind 团队所研发的名叫 AlphaFold 2 的人工智能系统在那一年国际蛋白质结构预测竞赛中取得桂冠，其预测准确性可以与使用冷冻电镜、核磁共振或 X 射线晶体衍射等实验技术解析出的三维蛋白质结构相媲美，有史以来首次将蛋白质结构预测做到了基本接近实用的水平。2022 年 11 月，美国人工智能研究实验室 OpenAI 发布了聊天机器人 ChatGPT，它能使用英语、汉语、法语等多种人类语言写诗做对、编写程序、回答问题、翻译撰文、闲聊对话等。发布后仅三个月，用户量就突破了一个亿，被认为是人工智能应用方面取得突破性进展的代表性成果。这些人工智能程序和系统的开发过程中都采用了机器学习技术，而机器学习则是人工智能领域的一个重要分支。机器学习的应用非常广泛，它能辅助医生根据所属人群、饮食习惯、检查指标来预测因突发心脏病而住院的病人近期再次发作的风险，协助投资者依据上市公司的经营状况、交易行情、宏观经济等信息来预测未来公司股价的走势，还能够进行垃圾邮件过滤、医学影像分析、语音和人脸识别等，新的应用还在日益增加。

简而言之，**人工智能**（Artificial Intelligence）的目标是制造出具有类似人类智能的机器，可以像人一样感知、学习、认知、交流、推理、决策和行动等。**机器学习**（Machine Learning）则是实现人工智能的一类方法，主要指从数据中获得规律，并利用规律对未知数据进行预测的算法。这里的“学习”是指从给定数据集中捕捉或发现潜在规律或模式的过程，而“机器”则是强调并非人类进行学习，而是让机器（一般指计算机）自动完成学习。机器学习所得到的结果就是模型，所以学习过程也被称为训练，训练模型所使用的数据集则称为训练集。评价模型的优劣不能只看模型在训练集上的性能，而要看模型在未知数据（训练时模型未见过，但一般与训练数据服从同一分布）上的预测能力。因为很容易让计算机“记住”每一个训练数据及其对应的预测结果，所以使用训练数据集对模型进行性能测试是毫无意义的。机器学习关注的是所构建模型的**泛化能力**，即对“未知数据”进行有效预测的能力。

训练或测试模型所使用的数据往往也称为样本。如图1.1 (a) 所示，我们有 18 个已被划分到两个类别的图形（训练样本），现要对图下方 3 个在训练样本中没有出现过的图形进行分类。机器学习模型需要从这些有限的图形样本中发现和归纳出分类所依据的主要特征（比如：形状和颜色等）和规律，然后能够根据这些特征并运用规律将未见过的新图形准确地划分到相应的类别。对于第一个红色的心形，合理的猜测是它应该属于类别一，因为所有红色的图形只在类别一中出现。对于第二个蓝色的菱形则难以判断，因为两个类别中都包含蓝色的图形，并且菱形也都出现在两个类别中。同样，对于第三个绿色的三角形，也不能确定它应该归属于哪一类。因

为从颜色上判断，它应该属于类别一，而从形状上判断，则它应该属于类别二。对于这些模棱两可的图形，可取的方法是让模型输出类别的概率分布。**概率论**提供了一种理论来量化这种不确定性（我们将在第二章回顾概率论的基础内容），不确定性可能来自于样本度量过程中所产生的噪声，也可能是由于可观察的样本数量有限造成的。但是，我们可以结合概率论与决策论（考虑决策行为可能带来的后果），在信息不完整、不充分或者存在歧义情况下，利用目前已知信息做出最优的预测。机器学习中绝大多数预测都是在不确定性情况下做出的，所以概率论是机器学习最重要的理论基石之一。



(a)

特征（或属性）

样本

形状	颜色	类型	类别
立方体	红	立体	一
圆柱体	绿	立体	一
十字形	红	平面	一
五边形	蓝	平面	二
同心圆	紫	平面	二
长方形	蓝	平面	二
⋮	⋮	⋮	⋮

(b)

图 1.1: 图形分类的例子。(a) 带有类别标签的 18 个训练样本和 3 个测试样本。(b) 表格形式表示的训练样本。前三列为描述图形的三种特征（或属性），最后一列则是类别标签。每一行表示一个训练样本。

现在已经是大数据时代了，我们每天能够获得的信息比古代帝王还要多，那么为什么不用足够多的样本来构建模型，从而降低预测的不确定性呢？首先，样本准备和人工标注一般非常的耗时和昂贵。比如我们想要构建一个花卉识别系统，人们可以通过拍摄花卉的照片并上传到系统来获得所拍摄花卉的种类、属性和简介等信息。大多数人一般只能识别少数常见的花卉，因而我们很可能需要聘请多位花卉方面的植物学家来对各种各样的花卉图片进行分类和标注。我们希望系统应能够对不同角度和光照环境所拍摄的花卉照片进行准确识别，所以每一种花卉往往还需要标注多张图片。此外，为了防止疏忽大意或意见不一致等原因所造成的人为分类错误，同一张花卉图片一般需要多人进行标注以保证样本的标注质量。简单想象一下目前世界上花卉的种类（虽然我们不需要也不可能收集到所有花卉的图片）和聘请植物学家的费用，就应该能够明白花卉识别系统构建过程中样本准备成本将是一笔不菲的开支。因而我们往往要在样本数量和系统性能之间进行权衡与选择，并且在样本数量有限的条件下，尽可能地提高系统的识别准确率。其次，有时我们看起来拥有大量的样本，但是实际对于某些类别的有效样本数量却还是很少。许多应用领域都观察到了类似被称为**长尾**（The Long Tail）的现象，即少数事件是很常见的，但大多数事件都相当罕见。例如，一般认为常用汉字约 3500 个，而古往今来使用过的汉字则超过了 10 万个。像“的”、“了”、“是”这些汉字使用和出现的频率很高，但它们的数量不多，而如“躞”（xiè，小步走路的样子）、“訇”（hōng，形容很大的声响）、“趔”（liè，脚步不稳

的模样）等字虽然相对少见，但它们的数量却占汉字总量的绝大多数。又如，仅少数电影被大众所熟知和喜爱，而多数电影只有少数人看过或喜欢。电影推荐系统不能总是仅推送少数几部广受欢迎的热门电影，还需要能够根据特定用户的个性化偏好推荐相对冷门的电影作品。长尾现象所造成的结果是，即使样本总体数量很多，但预测某些类别或行为所需的规律仍然只能从少量样本中学习。在这本书中，我们将系统讨论如何分析和处理类似数据的方法与技术。

1.2 机器学习的类型

机器学习主要可以分成三类：监督学习（Supervised Learning）、无监督学习（Unsupervised Learning）和强化学习（Reinforcement Learning）。机器学习主要类型及其应用例子如图1.2所示。

监督学习的目标是从训练数据集 $\mathcal{D}$ 中学习到将输入 $\boldsymbol{x}$ 对应到输出 y 的一种映射 $f(\boldsymbol{x})$。其中，我们用粗体 $\boldsymbol{x}$ 表示输入是一个向量，因为样本通常需要用多个特征（或属性）来进行描述，特征可以是如图1.1 (b) 所示的形状、颜色和类型等离散特征，也可以是时间、体积、长度等连续特征。向量 $\boldsymbol{x}$ 所包含特征的数量称为**维度**，以图1.1 (b) 为例，每一样本特征表示向量的维度等于 3（即形状、颜色和类型）。包含 N 个样本的训练数据集可表示为 $\mathcal{D}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，其中第 i 个训练样本 $\boldsymbol{x}_i$ 对应的输出为 y_i。根据 y_i 取值的类型，我们可以区分不同的问题。若 y_i 取离散值（如：男性或女性），则称为**分类**（Classification）或**模式识别**（Pattern Recognition）；若 y_i 取连续值（如：价格），则称为**回归**（Regression）。分类典型的应用包括人脸识别、文本情感分析（积极、消极或中性等）、手写体识别等。回归典型的应用包括人口增长预测、气象预测（温度、湿度、气压等）、房价和股价预测等。假设每一个训练样本用 D 维的向量表示，我们经常使用 $N\times D$ 的矩阵 $\mathbf{X}$ 来表示整个训练数据集的输入（如图1.1 (b) 除最后一列“类别”外的表格），其中矩阵的每一行对应一个训练样本的向量表示，而每一列则对应某个特征。类似地，我们也使用 $\mathbf{Y}$ 表示所有训练样本的输出（如图1.1 (b) 最后一列）。取离散值的特征和输出往往需要通过某种编码方法进行数值化。如图1.1的例子，可以将“类别一”记为 $y=0$，并且将“类别二”记为 $y=1$（此时 $\mathbf{Y}$ 是元素为 0 或 1 的列向量），形状、颜色和类型这些特征的取值也可以进行类似的数值化处理。在没有特殊说明的情况下，$\mathbf{X}$ 和 $\mathbf{Y}$ 的元素默认都是数值。

无监督学习主要特点是从无标签的数据 $\mathcal{D}=\{\boldsymbol{x}_i\}_{i=1}^N$ 中发现数据的内在结构、模式或表示。这里的“无标签”是强调：与监督学习不同，无监督学习所使用数据集 $\mathcal{D}$ 中的样本 $\boldsymbol{x}_i$，并没有给出对应的输出 y_i。无监督学习有时也被称为**知识发现**（Knowledge Discovery），它所面对的问题不像监督学习那样有较明确的定义，因为我们事先并不知道哪些“有趣”的模式可能会被发现，相应的也不像在监督学习时（一般只需比较预测值与真实观察值即可）有明确可靠的评估标准。**聚类**（Clustering）是一种常用的无监督学习技术，用于将一组数据样本划分成若干个类别或簇，使得同一簇内的样本彼此相似，而不同簇之间的样本则有较大差异。如图1.2所示（聚成红、黄、绿三个类别），虽然我们可以通过可视化的方法来观察和判断聚类结果的好坏，但某一个样本应该属于哪一个类别并没有标准答案，所以评估聚类结果一般没有显然而明确的标准（我们会在相关章节介绍和讨论一些间接的评估指标）。使用聚类技术，我们可以根据潜在顾客

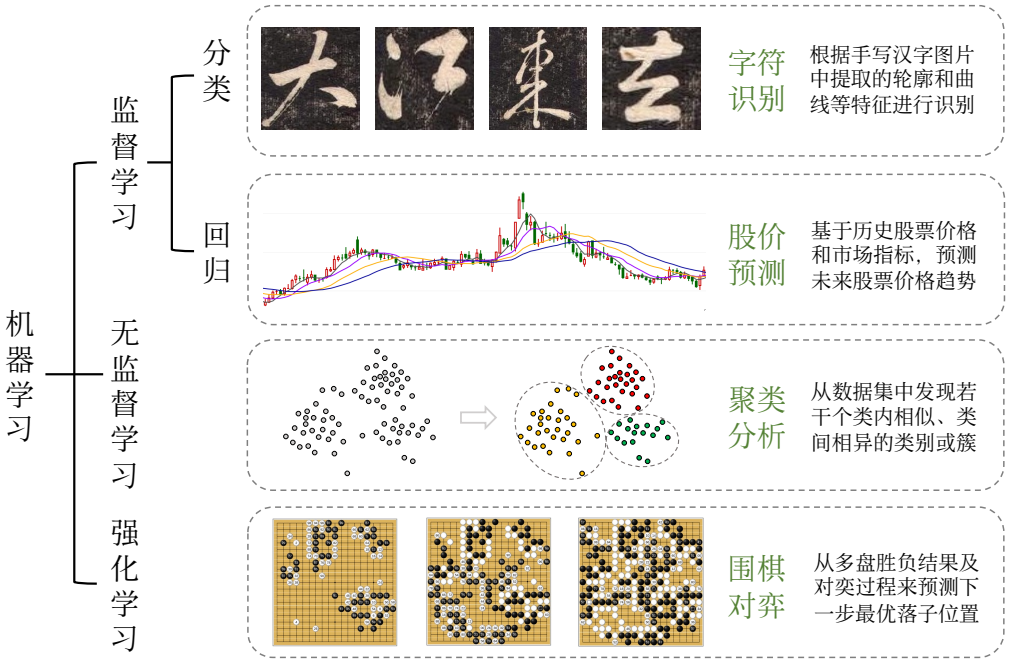


图 1.2: 机器学习的类型以及应用的例子。

的属性或特征将其划分成不同的群体，然后与目前市场上已有产品或服务定位的目标人群相对比，从而发现空白市场或竞争相对较弱的领域，通过填补市场空白或者进入竞争较小的市场来获得更好的发展机会。**频繁项集挖掘**（Frequent Itemset Mining）是另一个典型的无监督学习技术，它用于发现经常同时出现的项目（Item）集合，项目可以是商品、关键词、基因序列等。传说中“啤酒与尿布”的案例就是通过频繁项集挖掘技术发现啤酒与尿布（可能是年轻父亲的标配）经常一起被购买。基于此项发现，商家可以将啤酒与尿布摆放在一起，从而提高两者的销售量。如果啤酒与尿布确实总被一起购买，另一种策略是布置一条商品选购长廊，长廊一端摆放啤酒，另一端放置尿布（并用指示牌提醒另一商品的所在位置），其间摆放其他相关或日常销量较大的商品，迫使顾客在得到啤酒和尿布前要走过整个长廊，从而提高其他商品的销售量。

强化学习关注如何通过与环境的交互来最优化行动策略，以最大化累积的奖励。其原理是基于智能体与环境的交互学习。智能体在与环境的交互中尝试采取行动，并观察环境对其行动的反馈，通常是奖励或惩罚。智能体的目标是通过学习找到一种最优的行动策略，即在当前状态下选择未来期望奖励最大的行动。其特点是要求最优化整个过程，但过程中不是每一步都有明确的奖惩信号，往往需要通过试错和反馈来迭代优化决策。比如棋类游戏，我们只能在终局时才知道胜负（即奖惩信号），因而难以精确估计对奕过程中的每一步对于胜负的影响。所以强化学习关键问题之一是信用分配问题（Credit Assignment Problem），即如何将未来的奖励或惩罚归因和分配到过去做出的一系列动作，从而根据不同动作可能带来的期望奖惩来调整和优化动作选择的决策。强化学习在自动驾驶、机器人控制、对话系统（如聊天机器人 ChatGPT）等方面都有广泛的应用。

1.3 有参和非参模型

我们有很多方法可以用于构建机器学习模型，根据模型的参数数量是否固定可以将它们分为有参模型（Parametric Model）和非参模型（Non-parametric Model）。**有参模型**的参数数量是固定的，并且这些参数的数量不会随着训练数据的增加而增加。使用有参模型往往对目标数据的分布做了某种假设，并且在这种假设下定义了模型的结构（如：线性模型）。预先定义的模型结构是对建模过程的一种约束，即在所有可能的模型中仅考虑体现这种结构的模型。使用有参模型除了选择合适的模型结构外，主要是通过训练数据来确定参数的值。有参模型一般参数量相对较少，因而它们在处理大量数据时的计算效率较高，但也会由于所选模型结构与真实数据分布不相符导致较低的预测准确度。**非参模型**则较为灵活，事先不对模型结构和参数数量做出明确假设，其有效参数量会随着训练数据的增加而增加，因而可以更好地应对和处理复杂的数据分布。但是，非参模型在训练数据量较少时可能会表现出预测的不稳定，并且在大规模数据集上，通常需要更多的计算资源。我们下面介绍一种有参模型（线性模型）和一种非参模型（K 近邻方法），并且通过分类问题对它们进行讨论。

线性模型虽然形式相对简单，但它却是一种重要且实用的方法，其核心思想是将输出视为输入特征的加权求和，即如下的线性组合形式：

$$f(\boldsymbol{x}) = w_0 + \boldsymbol{x}^\top \boldsymbol{w} = w_0 + \sum_{j=1}^{D} x_j w_j \tag{1.1}$$

其中 w_0 称为偏置或截距，x_j 是输入 $\boldsymbol{x}$ 的第 j 个特征。使用线性模型，我们需要利用训练数据来确定 $\{w_0, w_1, \ldots, w_D\}$ 这 $D+1$ 个参数的值。参数数量一旦确定后将保持不变，不受训练数据规模的影响。我们往往会在 $\boldsymbol{x}$ 中增加一个值为 1 的特征（与公式1.1中参数 w_0 相对应），线性模型则可以方便地写成以下向量的内积形式：

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{w} \tag{1.2}$$

给定包含 N 个样本的训练数据集 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$，有许多方法可以用来确定线性模型中参数 $\boldsymbol{w}$ 的取值，其中最受欢迎的是最小二乘法（Least Squares），即最小化所有样本预测值与实际观测值之间误差的平方和，一般记为 RSS（Residual Sum of Squares）：

$$\text{RSS}(\boldsymbol{w}) \triangleq \sum_{i=1}^{N} (y_i - \boldsymbol{x}_i^\top \boldsymbol{w})^2 \tag{1.3}$$

我们需要求解出能使上式最小化的 $\boldsymbol{w}$ 值，求解过程用以下矩阵和向量表示比较简便：

$$\text{RSS}(\boldsymbol{w}) \triangleq (\mathbf{Y} - \mathbf{X}\boldsymbol{w})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{w}) \tag{1.4}$$

其中 $\mathbf{X}$ 是 $N \times D$ 的矩阵，用于表示整个训练数据集的输入，而 $\mathbf{Y}$ 表示所有训练样本实际的观测值。我们将上式对 $\boldsymbol{w}$ 求导数，并让导数等于 0 来求其极值：

$$\mathbf{X}^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{w}) = 0 \tag{1.5}$$

以上结果从公式1.3更容易推导出来。略去前面的加和操作 $\sum$，我们可以将公式1.3看成是 $(y_i -$

$\boldsymbol{x}_i^\top \boldsymbol{w}$) 与平方的复合函数，并采用链式法则依次对 $\boldsymbol{w}$ 求导。先对外层的平方求导后得到 $2(\mathbf{Y} - \mathbf{X}\boldsymbol{w})$，然后再对内层求导后得到 $-\mathbf{X}^\top$。将这两项相乘，并且略去负号和常数 2（求导后要置零，所以符号与常数并不影响求解结果），即得到如公式1.5的形式。将公式1.5简单改写后我们可以得到（详细推导过程见第三章）：

$$\mathbf{X}^\top \mathbf{Y} = \mathbf{X}^\top \mathbf{X}\boldsymbol{w} \tag{1.6}$$

如果 $\mathbf{X}^\top \mathbf{X}$ 是非奇异矩阵 (Non-singular)，则其可逆（如果 $\mathbf{X}^\top \mathbf{X}$ 不可逆，可以用其伪逆来替代），我们可以得到参数 $\boldsymbol{w}$ 的唯一解（公式1.6两边同时左乘 $(\mathbf{X}^\top \mathbf{X})^{-1}$）：

$$\boldsymbol{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y} \tag{1.7}$$

按上述方法求得的 $\boldsymbol{w}$，一定是目标函数（公式1.3）的最小值。因为该目标函数没有上限，所以不可能是最大值。

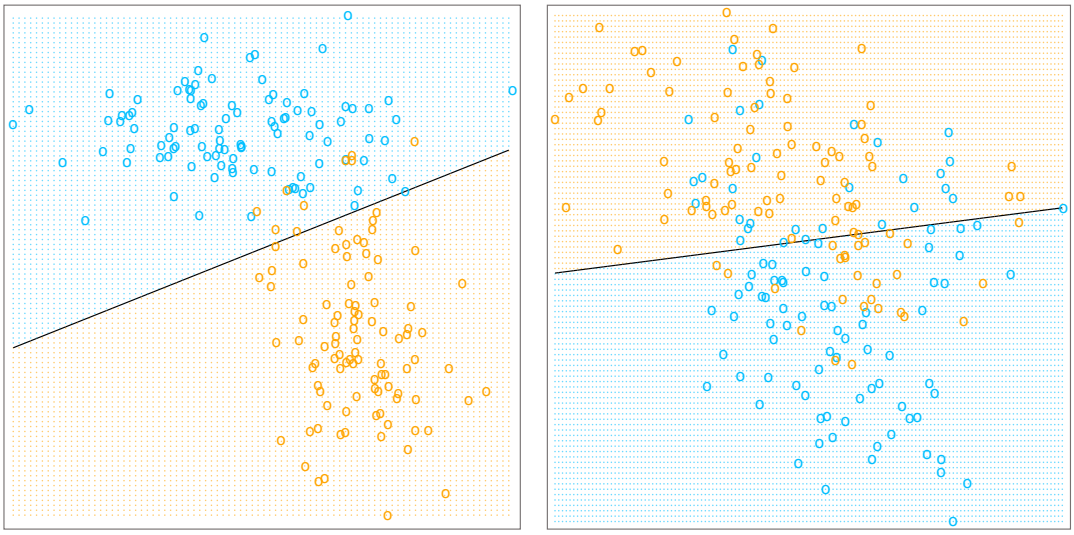
作为对比的非参模型，K 近邻方法使用训练集中 K 个在输入空间中与 $\boldsymbol{x}$ 最相近的样本来估算 $f(\boldsymbol{x})$：

$$f(\boldsymbol{x}) = \frac{1}{K} \sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} y_i \tag{1.8}$$

其中 $\mathcal{S}_K(\boldsymbol{x})$ 就是与输入 $\boldsymbol{x}$ 最近（或相似）的 K 个样本的集合。这种方法的想法也非常朴素，就是先找出 K 个与 $\boldsymbol{x}$ 最相似（或距离最近）的样本，然后将它们的输出取平均。稍微复杂一点，会按离 $\boldsymbol{x}$ 的距离进行加权，距离越近权重越高。如果是分类问题，相当于用这 K 个样本对 $\boldsymbol{x}$ 所属类别进行投票，少数服从多数。K 近邻方法虽然没有显式引入需要学习的参数，其实际发挥作用的有效参数量大体相当于 N/K（假设相邻集合之间没有重叠的情况）。也就是 K 个训练集的样本确定一种模式，并用它们的均值来拟合一个参数。显然，K 近邻方法的有效参数量会随着训练样本数量 N 增加而增加，同时会随着 K 值增大而减少。K 近邻方法没有像线性模型一样有可学习的参数 $\boldsymbol{w}$，只有超参数 K，即使用多少个近邻的信息来进行预测。

在机器学习中，**超参数**（Hyperparameters）是指那些在训练模型之前需要设置的参数，不能通过模型的优化算法直接学习得到。它们通常用来控制模型的学习过程和性能，影响着模型的结构、复杂度和训练过程中的各种配置。超参数的选择对于模型的性能和泛化能力至关重要。与超参数相对应的是模型参数（Model Parameters），**模型参数**是在模型训练过程中通过优化算法自动学习得到的。例如，在线性回归中，模型参数就是截距和回归系数，通过最小化目标函数来找到模型参数的最优值，使得预测结果与实际结果之间的误差最小。需要提醒的是：非参模型并非无参模型，非参主要描述或强调起作用的有效参数数量不是预先固定的，可以理解为“非参数化”，它不像有参模型将训练数据中隐含的模式捕捉和“压缩”到固定数量的参数中去。无论有参模型，还是非参模型，大多数模型既有超参数，也有模型参数。

让我们来看一个简单的例子。如图1.3(a) 所示的分布图，共计 100 个不同的点，分成了蓝色和橙色两类。我们希望所构建的模型能够根据点的横坐标（记为 x_1）和纵坐标（记为 x_2）来预测其所属类别。建模时我们规定：当属于蓝色类别时，$y = 0$；当属于橙色类别时，$y = 1$。使用



(a) (b)

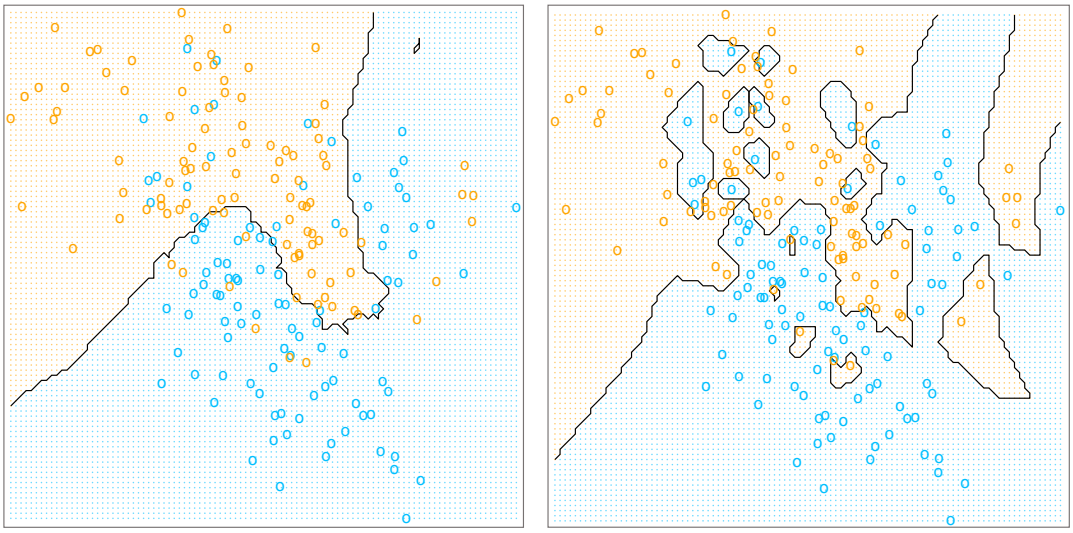
图 1.3: 二维空间中采用线性回归模型进行二分类的例子。(a) 线性模型能够较好地将蓝色和橙色的点区分开来，图中直线上方的蓝色区域为一类，而下方的橙色区域则被分为另一类。这条直线就是划定蓝色和橙色区域的决策边界（Decision Boundary），它由 $\boldsymbol{x}^\top \boldsymbol{w} = 0.5$ 所定义。(b) 简单形式的线性模型已经不能较好地将蓝色和橙色的点区分开来，因为合适的决策边界显然是非线性的。

线性模型拟合如图1.3(a) 所示的 100 样本点之后，我们得到以下线性函数：

$$f(\boldsymbol{x}) = 0.8888236 + 0.173223x_1 - 0.3118708x_2 \tag{1.9}$$

进行预测时，将点的横坐标和纵坐标代入以上函数。如果计算得到的 $f(\boldsymbol{x})$ 值小于 0.5，就预测为蓝色类别；如果大于等于 0.5，则预测为橙色类别。蓝色和橙色两类被图1.3(a) 中的直线（由公式1.9中 $y = 0.5$ 时定义）所划分，这条直线被称为**决策边界**（Decision Boundary），因为它确定了样本点所在区域的类别。仅包括三个参数（含截距）的线性模型就能够较好地将图1.3(a) 中蓝色和橙色的点区分开来，但将类似的线性模型应用于如图1.3(b) 所示的分布时就不再适合了，因为其决策边界显然是非线性的。

我们使用图1.3(b) 相同的数据，但采用十五近邻（15-nearest-neighbor）模型来进行建模，其结果如图1.4(a) 所示。预测结果是由 15 个最近邻居的多数票（多数邻居所属类别）来确定，等同于将 15 个近邻的 y 值取平均，然后根据平均值是否小于 0.5 来决定其类别。在这个例子中，样本点之间的距离远近由欧氏距离（Euclidean Distance）进行度量（采用什么距离定义样本之间的远近与 K 近邻模型本身无关，可以根据实际情况选择合适的距离公式）。为了便于比较，图1.4(b) 也显示了一近邻（1-nearest-neighbor）的结果。一近邻的决策边界显然较十五近邻更为复杂和曲折（将二维空间划分成了许多个不同的区域），也更容易受到噪声数据的影响。只要训练集中有一个数据点的类别标注错误，直接会导致离其最近所有点的分类错误。而在十五近邻的情况下，这样的标注错误会受相邻其它正确标注样本点的影响而以一定概率被修复。值得注意的是：我们不能再像使用线性模型那样以最小化训练样本误差平方和（公式1.3）为目标来选



(a) (b)

图 1.4: 与图1.3(b) 相同的二分类例子，但采用 K 近邻方法。(a) 使用最近 15 个邻居的类别来进行预测，预测的结果是由 15 个最近邻居的多数票进行确定。(b) 使用最近 1 个邻居的所属类别进行预测。这个例子中，十五近邻（15-nearest-neighbor）显然较一近邻（1-nearest-neighbor）更符合数据的分布，后者容易受到噪声的影响。此例子改编自文献 [27]。

择 K 的值，因为我们总是会选择 $K = 1$，即使得所有样本的误差为 0 的选择。选择合适的 K 需要使用训练样本之外的其它测试样本，这再一次说明了机器学习模型不能以训练集上的预测性能来进行评估。

从上述例子中可以看出，当有参模型（以线性模型为例）所做的假设与目标数据分布比较吻合时，有参模型有计算效率方面的优势（仅需将输入代入模型计算结果后进行预测）。但假设与目标数据分布不相符时会导致预测性能的严重下降（如图1.3(b) 所示）。非参模型（以 K 近邻为例）因其有效参数量会随着训练数据的增加而增加的特点，使其能够较灵活地自适应复杂的实际数据分布，但一般在预测和推理时计算代价较高（需要在所有训练样本中找到指定数量的近邻）。

1.4 高维灾难或诅咒

之前介绍的 K 近邻方法原理非常简单，并且符合直觉。当有一定规模的训练样本时，能够为任何一个 $\boldsymbol{x}$ 找到足够数量的近邻，然后用近邻的平均值作为 $\boldsymbol{x}$ 的预测值。但将这样的想法和直觉扩展到高维时就会失效，这种现象往往会被称为**高维灾难**或**高维诅咒**（Curse of Dimensionality）。

我们在使用 K 近邻方法时，先要定义样本之间的距离，然后使用距离公式来找到近邻。在之前的例子中，我们采用了欧氏距离。在 n 维的欧几里得空间中，$\boldsymbol{x} = (x_1, \ldots, x_n)$ 和 $\boldsymbol{y} = (y_1, \ldots, y_n)$ 两点之间的欧氏距离为：

$$\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{1.10}$$

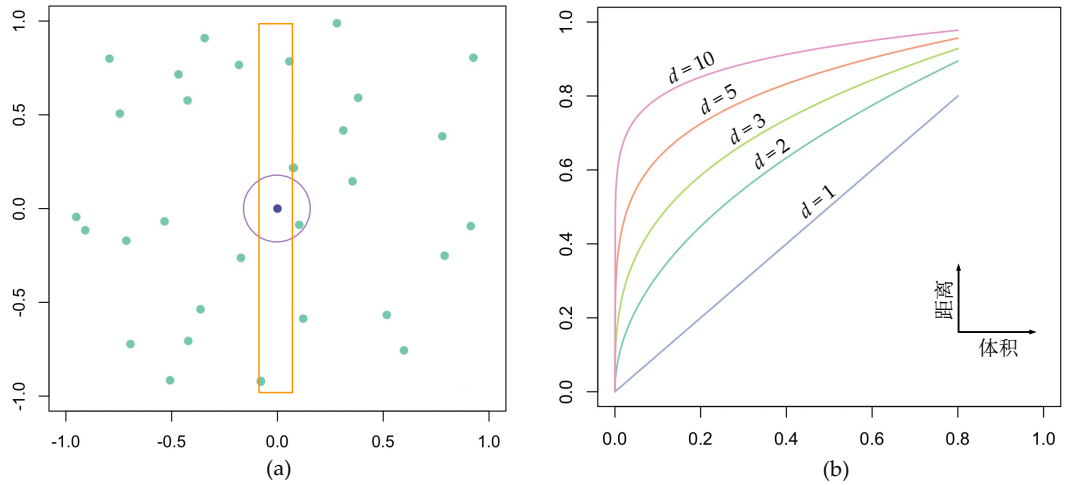


图 1.5: (a) 一维和二维空间中采用欧氏距离寻找近邻的例子。使用相同的半径，一维情况下坐标为 $(0,0)$ 的点有 3 个近邻（橙色矩形框内），而二维时则只有 1 个近邻（紫色圆圈内）。为了获得与低维时相同数量的近邻，在高维时需要大幅增加欧氏距离的半径。(b) 在各种维度下，覆盖不同比例数据（体积）所需超立方体的边长（距离）变化。在十维空间中，为了覆盖 10% 的数据作为近邻，每一维的边长几乎需要达到整个空间的 80%，此时已经并不能算是近邻了。

图1.5(a) 显示了二维平面中随机生成的 30 个绿色的点。我们采用相同的半径分别使用一维和二维欧氏距离为位于平面中心的蓝色点寻找其近邻。一维时我们只考虑两点之间 x 轴上的距离，蓝色点的近邻在图中橙色矩形框内，二维时其近邻则在紫色圆圈内（橙色矩形框的宽等于紫色圆圈的半径）。一维情况下，中心蓝色点有 3 个近邻，而二维时只有 1 个近邻落入相同的半径内。如图1.4(b) 所示，当近邻数量不足时，预测容易受到噪声或异常点的影响。因此，随着维度的上升，为了维持数据在空间中的密度，让任一点能够找到足够多的近邻用于预测，我们需要更多的训练数据（通常呈指数增长）。当高维且数据不足时，将会导致预测性能下降和不稳定。

为了更好地理解高维情况下会遇到的问题，我们以 d 维单位超立方体（Unit Hypercube）空间为例进行分析。单位超立方体的每个维度的边长都是 1，三维情况下就是长宽高相等的立方体。不论多少维度，单位超立方体的体积始终为 1。假设所有数据都均匀地分布在单位超立方的空间里，我们同样采用边长为 e 的更小的超立方体 $(e < 1)$ 所围成空间来定义近邻。因为数据分布是均匀的，边长为 e 的超立方体的体积 (e^d) 就近似等于近邻占所有数据的比例。当要求近邻占比为 r 时，覆盖这些近邻的超立方体边长则为 $e_d(r) = r^{1/d}$。在十维空间中，使得近邻包含 1% 数据的超立方体的边长为 $e_{10}(0.01) = 0.63$，需要包含 10% 数据时，边长则需为 $e_{10}(0.1) = 0.79$。整个超立方体空间的边长仅为 1，覆盖 1% 和 10% 数据的超立方体的边长却分别要达到 0.63 和 0.79，其实已经并不能算作是近邻了（已经超过整个空间一半的长度）。图1.5(b) 展示了维度分别为 $d = \{1, 2, 3, 5, 10\}$ 时，覆盖不同比例数据（体积）与所需超立方体的边长（距离）的变化关系。上述分析表明，基于距离的方法在高维空间中变得不再可靠。大幅地降低近邻所要求的边长也于事无补，因为随着边长 e 的缩短，近邻的数量会变少，而使用数量过少的近邻进行局部均值估计会导致拟合方差变大。

高维灾难或诅咒的另一个表现是大多数样本点会分布在样本空间的边界上，而非样本空间的内部。假设 N 个数据点均匀分布在以原点为中心的 d 维单位球中，我们可用以下公式估算离原点最近数据与原点之间距离的中位数：

$$(1-0.5^{1/N})^{1/d} \tag{1.11}$$

理解这个公式的关键在于半径为 r 的 d 维球体的体积与 r^d 成正比，而一个随机数据点落入半径为 r 的球体的概率为 $(r/R)^d$，其中 R 为整个样本空间的半径（由于是 d 维的单位球，所以 $R=1$）。最近点距离原点大于给定距离 r 的概率等于所有 N 个点距离原点都大于 r 的概率。由于这些点相互独立且在单位球中均匀分布，因而所有 N 个点距离原点都大于 r 的概率为 $(1-r^d)^N$。给定任一 r 的值，都可计算相应的概率。当这个概率为 0.5 时，对应的 r 即为从原点到最近数据点距离的中位数。设 $(1-r^d)^N=0.5$，我们可以得到公式1.11。如果有 500 个数据点且维度为 10，则距离中位数 r 约等于 0.5178（超过原点到球表面的一半）。这说明相对于原点而言，数据点更接近于球的表面（样本空间的边界）。理论上，原点已经是到所有数据点平均距离最短的。因此，在高维情况下，大多数数据点更接近样本空间的边界，而非空间的内部。这种情况带来的问题是，我们必须从相邻的样本点进行外推（推测超出已知数据点范围之外的未知值）来进行预测（因为需要预测的点大概率分布在边界上，而边界之外没有可观察的数据点），而不是在相邻样本之间进行内插（已知数据点之间进行推测）。外推则显然不如内插可靠。

1.5 欠拟合与过拟合

在机器学习中，欠拟合（Underfitting）和过拟合（Overfitting）是两种常见的问题，它们描述了模型对训练数据的拟合程度以及模型泛化到新数据的能力。**欠拟合**是指模型在训练数据上没有获得足够的学习，通常是由于模型过于简单导致无法捕捉到数据的基本结构。因此，它通常在训练集和测试集上都表现不佳。**过拟合**则是相对的概念，它指的是模型在训练数据上学习过好了，以至于它甚至开始学习数据中的噪声和异常值。这样的模型虽然在训练集上表现出色，但在未见的数据上往往表现不佳，从而失去了泛化能力。过拟合通常是由于模型太过复杂，超过了数据本身或问题的复杂度。在实际应用中，寻找一个既不过拟合也不欠拟合的模型，也就是具备较强泛化能力的模型，是机器学习中的一个重要任务。这通常涉及到模型选择和调参过程，需要使用交叉验证和学习曲线等方法来评估模型的泛化能力（详见1.6节）。

我们之前介绍的线性模型和 K 近邻方法用于如图1.3(b) 所示的数据时，线性模型就因为过于简单而发生欠拟合现象，而一近邻模型（图1.4(b)）则因过于复杂而产生了过拟合问题。我们将再通过一个简单的多项式拟合例子来分析因模型容量（Model Capacity）与问题复杂性（Problem Complexity）不匹配时导致的欠拟合和过拟合现象，并简要讨论对策。模型容量是指模型所能表示或学习的函数的复杂度，即模型能够拟合不同程度复杂度的数据模式的能力。高容量的模型可以更好地适应复杂的数据模式，但也更易过拟合。相反，低容量的模型可能无法很好地拟合复杂的数据模式，但不太容易过拟合。与模型容量相关但又不完全相同的概念是模型的复杂度

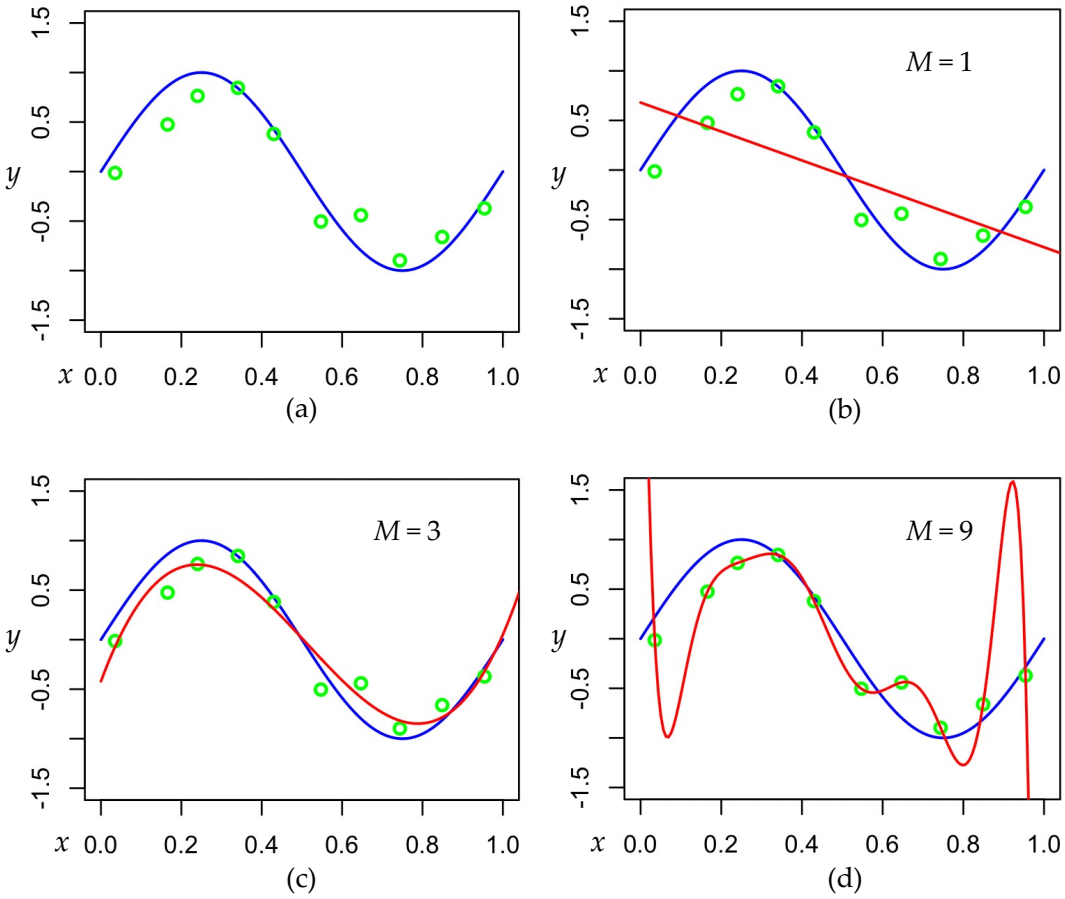


图 1.6: 多项式拟合结果，其中蓝色曲线表示原 $\sin(2\pi x)$ 函数，红色曲线表示拟合所得到的函数。(a) 绿色圆圈代表 10 个训练数据点，每一个点表示输入变量 x 和输出目标 y 的对应关系。为了模拟实际情形，这些点是由蓝色曲线所表示的 $\sin(2\pi x)$ 函数上采样后加随机噪声生成的。建模的目标是在不了解真实函数 $\sin(2\pi x)$ 的情况下，通过学习这 10 个训练数据点，预测任何新 x 所对应的 y 值。(b) 当多项式阶数为 $M = 1$ 时，拟合结果呈现出欠拟合现象。(c) 当多项式阶数为 $M = 3$ 时，拟合结果表明模型容量与数据复杂度相匹配。(d) 当多项式阶数为 $M = 9$ 时，拟合结果出现了过拟合。此例子改编自文献 [8]。

表 1.1: 各阶多项式拟合的参数值

	$M=1$	$M=3$	$M=5$	$M=7$	$M=9$	$M=9$ ($\lambda=0.001$)
w_0	0.68	−0.42	−0.06	0.60	6.01	−0.0038
w_1	−1.46	10.96	−0.62	−26.72	−300.22	4.5121
w_2		−29.87	51.96	309.29	4795.98	−7.1532
w_3		19.38	−200.56	−1284.30	−37326.96	−4.6056
w_4			249.82	2549.70	166656.30	0.1140
w_5			−101.01	−2664.05	−453352.90	3.4788
w_6				1417.44	760751.60	4.3785
w_7				−302.04	−766628.60	2.9054
w_8					424460.90	−0.0768
w_9					−99080.74	−3.5092

(Model Complexity)。模型复杂度通常指的是模型本身的结构和参数数量。同样，一个复杂度较高的模型通常拥有更多的参数和更灵活的结构，可以更好地适应复杂的数据模式，但也更容易过拟合。反之，一个复杂度较低的模型一般拥有较少的参数和更简单的结构，可能无法很好地拟合复杂的数据模式，但也不容易过拟合。模型复杂度和模型容量都涉及到模型的拟合能力和泛化能力，但模型复杂度更侧重于模型本身的结构和参数数量，而模型容量更侧重于模型所能表示或学习的函数的复杂度。

假设我们有如图1.6(a) 中绿色圆圈表示的 10 个训练数据，这些数据点是由蓝色曲线表示的 $y = \sin(2\pi x)$ 函数在定义域 $[0, 1]$ 区间等距采样，并且在输出 y 值上叠加高斯随机噪声生成的。在预测目标 y 值上添加高斯随机噪声是为了模拟实际的情形，即我们实际获得的训练数据往往带有各种噪声，这些噪声可能来源于数据产生过程中固有的随机性（如生物生长速度或放射性衰变），但更多的是人为疏忽或计量不准等因素所引入的。我们的目标是在不知道生成数据所使用真实函数 $\sin(2\pi x)$ 的情况下，从这 10 个被随机噪声所影响的有限训练数据点学习到数据背后的基本规律，从而较准确地预测任一新 x 所对应的 y 值。

我们仍然使用线性模型来拟合这 10 个数据点，但是将输入 x 用不同次项的幂函数进行扩展，变成以下多项式函数形式：

$$f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{k=0}^{M} w_k x^k \tag{1.12}$$

其中 M 表示多项式的阶数，x^k 表示 x 的 k 次幂。我们需要利用训练数据来确定 $\{w_0, w_1, \ldots, w_M\}$ 这 $M+1$ 个参数或权重的取值。我们仍然使用最小二乘法，通过最小化所有训练数据预测值与实际值之间的误差平方和（公式1.3）来进行多项式拟合。参数集合 $\boldsymbol{w} = \{w_0, w_1, \ldots, w_M\}$ 的取值则可以使用公式1.7计算得出。

图1.6展示了阶数 $M = \{1, 3, 9\}$ 时多项式拟合结果。阶数 $M = 1$ 时的多项式（一条带斜率的直线）显然不足以建模 sin 函数的 S 型曲线变化，从而产生了欠拟合现象。当阶数 $M = 3$ 时，看起来能较好地拟合 $\sin(2\pi x)$ 函数。当阶数 M 增加到 9 时，所得到的多项式函数完美地拟合了所有带噪声的训练数据，图1.6(d) 所示的红色曲线准确地贯穿所有训练数据点，即所有训练数据预测值与实际值之间的误差为零。然而，拟合曲线在两个训练数据点之间剧烈震荡，产生了严重的过拟合现象。相对于真实函数 $\sin(2\pi x)$，阶数 $M = 9$ 的多项式拟合曲线变化过于陡峭，不够平滑（Smooth）。事实上，阶数 $M = 9$ 的多项式的各阶导数都远大于阶数 $M = 3$ 的情况，而各阶导数的绝对值大小反映了函数的平滑程度。较平滑的模型更倾向于捕捉数据中的总体趋势，而不是每一个特定数据点的具体值，这通常也意味着能够更加稳健地泛化到未见过的数据，而不是过分关注训练数据中的细节和噪声。

我们来分析一下为何采用阶数 $M = 9$ 的多项式会导致严重的过拟合现象。表1.1的前五列列出了五种不同阶数多项式拟合的参数值。明显地，随着多项式阶数的增加，其参数的绝对值显著上升。特别是对于阶数 $M = 9$ 的多项式，通过让参数取较大的绝对值来 “精细” 地匹配训练数据，使相应的多项式函数完美拟合每一个数据点。随着 M 值的增大，多项式模型的拟合能

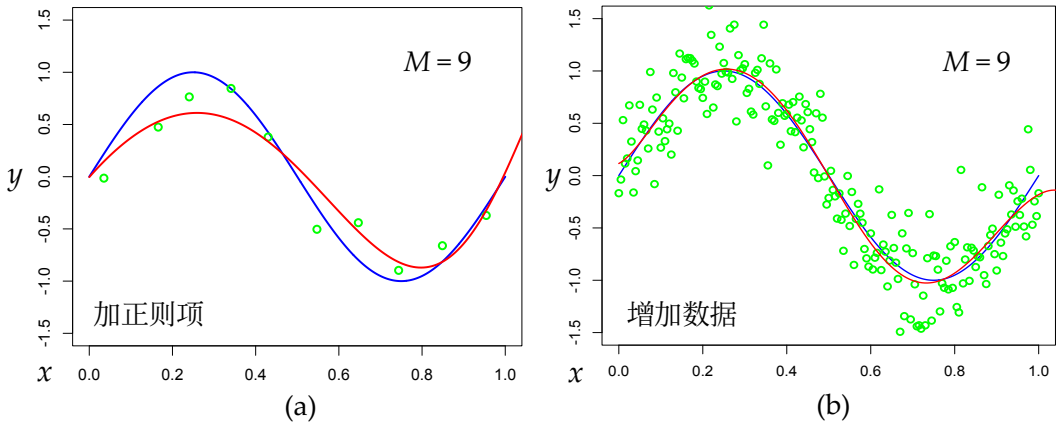


图 1.7: 多项式拟合结果，其中蓝色曲线表示原 $\sin(2\pi x)$ 函数，红色曲线表示拟合所得到的函数。(a) 优化目标函数中加入正则项（$\lambda = 0.001$），使用图1.6(a) 中相同的 10 个训练数据点进行阶数 $M = 9$ 的多项式拟合结果。(b) 当训练数据点增加到 200 个时，阶数 $M = 9$ 的多项式拟合结果。

力不断增强。在数据有限的情况下，相对较少的约束使得更多的过剩参数被调校到目标值的随机噪声上。直觉上，我们可以通过控制多项式参数的绝对值大小来缓解过拟合问题。实现这种控制的常用方法就是在误差平方和的优化目标函数中加入关于参数的**正则项**（Regularization）：

$$\sum_{i=1}^{N}(y_i - \boldsymbol{x}_i^{\top}\boldsymbol{w})^2 + \lambda\|\boldsymbol{w}\|^2 \tag{1.13}$$

其中 $\|\boldsymbol{w}\|^2 \triangleq \boldsymbol{w}^{\top}\boldsymbol{w} = w_0^2 + w_1^2 + \cdots + w_M^2$，超参 λ 用于调控正则项相对误差平方和损失的重要程度。正则项 $\|\boldsymbol{w}\|^2$ 的作用是让参数不要离零值过远，从而防止其值过大。超参 λ 值越大，正则项的约束就越强，所得到的多项式函数就越平滑，但误差平方和会较大，即对训练数据的拟合度降低。反之亦然。选择合适的超参 λ 值以平衡公式1.13中两项不同优化目标对于构建具有较强泛化能力的模型至关重要。我们将在下一节 “模型选择与评估” 中进一步探讨这个问题。图1.7(a) 表明阶数 $M = 9$ 的多项式在目标函数中加入正则项之后有效缓解了过拟合问题。使用更多的训练数据往往能够达到相同或更好的效果。如图1.7(b) 所示，当我们将训练数据从 10 个增加到 200 个时，虽然输入仍然是加了噪声的采样点，阶数 $M = 9$ 的多项式基本能够较好地拟合 $\sin(2\pi x)$ 函数（除了 $x = 0$ 和 $x = 1$ 这两端外。其原因是缺少另一侧的数据，所以对边界附近变化趋势的估计容易产生较大偏差）。增加训练数据可以为模型参数的求解引入更多约束。由于随机噪声的均值通常为零，更多的数据点（从统计均值的意义上来看）可以有效地过滤原始目标值中的噪声。欠拟合通常是因为模型太简单或者特征未能充分表达数据的关键信息，而过拟合则通常是因为模型过于复杂或者训练数据量不足。解决欠拟合的方法包括增加模型复杂度、添加更多有效特征，以及减少数据预处理中的简化步骤（导致关键信息丢失）等。解决过拟合的方法则包括减少模型复杂度、使用正则化技术、增加训练样本的数量，以及进行特征选择等。

在公式1.13中引入正则化项是基于对表1.1中参数取值观察后所获得的直觉。接下来，我们将从贝叶斯分析的角度对通过这种方式引入正则化约束的理论依据进行简要讨论，并从中理解超参数 λ 选取的一般指导原则。

在之前用多项式拟合 $y = \sin(2\pi x)$ 函数的例子中，为了模拟实际的情况，我们在每个训练数据的输出 y 值上添加了高斯随机噪声。为了表达所观察到 y 值的不确定性，我们可以合理地假设，对于特定的输入 x，其 y 值服从以 $f(x)$ 为均值的高斯分布。换句话说，理想模型 $f(x)$ 的预测值应该是消除了随机噪声后的均值。基于这一假设，我们可以将其表示为：

$$p(y|x, \boldsymbol{w}, \beta) = \mathcal{N}(y|f(x, \boldsymbol{w}), \beta^{-1}) \tag{1.14}$$

其中 β 是方差的倒数，被称为精度（Precision）。这里选择使用精度而不是更常用的方差来描述高斯分布，是为了方便之后的计算过程。此外，我们将多项式的权重 $\boldsymbol{w}$ 放到函数 f 中来强调该函数的值受 $\boldsymbol{w}$ 的影响。公式1.14所表达的意思是对于给定的输入 x，其对应的输出 y 不是唯一确定的，而是服从一个以 $f(x, \boldsymbol{w})$ 为均值、β^{-1} 为方差的高斯分布。换言之，对于每个给定的 x，y 可以取多个值，这些值的概率分布由 $\mathcal{N}(y|f(x, \boldsymbol{w}), \beta^{-1})$ 描述。这是因为我们实际得到的输入和输出对 (x, y) 中的 y 受多种因素（如：测量误差、传输干扰和人为失误等）的影响而包含了噪声信息。我们希望通过优化 $\boldsymbol{w}$ 的取值来最大化公式1.14的概率，即使得所观察到的值的发生概率最大化。给定包含 N 个样本的训练数据集 $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$，假设这些数据是独立地从分布（公式1.14）中采样出来的，那么它们的似然（Likelihood）可以表示为：

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{w}, \beta) = \prod_{i=1}^{N} \mathcal{N}(y_i|f(x_i, \boldsymbol{w}), \beta^{-1}) \tag{1.15}$$

为了简化上述式子中复杂的连乘运算，我们对等式两边取对数。对数函数（ln）是单调递增的，因此不会改变最大化的极值点。因此，我们可以得到：

$$\ln p(\mathbf{Y}|\mathbf{X}, \boldsymbol{w}, \beta) = \ln \sum_{i=1}^{N} \mathcal{N}(y_i|f(x_i, \boldsymbol{w}), \beta^{-1}) \tag{1.16}$$

我们知道高斯分布的概率密度函数为：

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.17}$$

其中 $\sigma^2 = \beta^{-1}$，$\exp(\cdot)$ 为自然常数 e 的指数函数。将公式1.17代入到公式1.16，可得：

$$\ln p(\mathbf{Y}|\mathbf{X}, \boldsymbol{w}, \beta) = -\frac{\beta}{2} \sum_{i=1}^{N} (y_i - f(x_i, \boldsymbol{w}))^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) \tag{1.18}$$

上式右边的最后两项与 $\boldsymbol{w}$ 无关，所以可以忽略。这时我们可以发现，在假设高斯噪声的情况下，最大化似然等价于最小化平方误差损失函数（见公式1.3）。

在统计学中，正式将先验信息纳入并探索如何利用这些信息的方法被称为贝叶斯分析。在贝叶斯分析中，我们利用参数的先验分布来表示对参数不确定性的先验知识或信念。为了简单起见，我们在这里引入以下高斯分布作为多项式权重 $\boldsymbol{w}$ 的先验分布（Prior Distribution）：

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left(-\frac{\alpha}{2}\boldsymbol{w}^\top \boldsymbol{w}\right) \tag{1.19}$$

其中 α 是高斯分布的精度，$M+1$ 是 M 阶多项式权重向量 $\boldsymbol{w}$ 的维度（包括了截距 w_0）。$\alpha^{-1}\mathbf{I}$ 是这个多维高斯分布的协方差矩阵。为了减少计算复杂度，这个协方差矩阵只有对角线上有值且都所有值均等于超参 α，而其他位置都为零。这意味着各个维度之间是相互独立的，即各个

维度的变化不会影响其他维度的变化。使用贝叶斯公式 (Bayes' Theorem)，参数 $\boldsymbol{w}$ 的后验分布可以表示为正比于先验分布和似然函数的乘积（详见第2.2节的公式2.13）：

$$p(\boldsymbol{w}|\mathbf{X},\mathbf{Y},\alpha,\beta) \propto p(\mathbf{Y}|\mathbf{X},\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha) \tag{1.20}$$

以上公式体现了贝叶斯方法的核心思想，即将先验信息和样本信息（即似然函数）结合，得出给定样本信息后的后验分布。后验分布把对 $\boldsymbol{w}$ 的先验信息与含于样本内关于 $\boldsymbol{w}$ 的信息结合起来得出最后信念的合成图像。现在我们可以通过最大化后验分布来求解 $\boldsymbol{w}$ 的取值，这一技术被称为最大后验概率估计（Maximum a Posterior，简称 MAP）。结合公式1.15和1.19，并取其对数形式，然后去除与 $\boldsymbol{w}$ 无关的项，最大化后验分布等价于最小化以下形式（相差一个负号）：

$$\frac{\beta}{2}\sum_{i=1}^{N}(y_i - f(x_i,\boldsymbol{w}))^2 + \frac{\alpha}{2}\boldsymbol{w}^\top\boldsymbol{w} \tag{1.21}$$

将上式乘以 $2/\beta$，并设 $\lambda = \alpha/\beta$，就与加了正则项的误差平方和的损失函数公式（1.13）相同了。至此，我们可以观察到 λ 的取值与 α 与 β 的比值有关。当 β 较小时，样本数据所包含的噪声就越大（β 是样本噪声方差的倒数）。在这种情况下，我们更倾向于选择能够更好地对抗噪声的更为平滑的模型，此时 λ 的值应该选择得更大，以增加正则化的强度。相反，当 α 较小时，高斯先验分布的精度较小，表明对权重的先验信念不太强烈，即对权重的先验知识不太确定。因此，我们应该选择较小的 λ 值。换言之，精度越小，高斯先验分布在权重空间中（给定均值的情况下）的方差越大，表明我们对权重的取值范围没有太多的先验偏好，因此公式1.21中来自先验分布的最后一项所起的作用应越小。

1.6 模型选择与评估

模型选择和评估是机器学习中两个相关但又不同的概念。模型选择涉及从多个模型中挑选性能表现最佳的模型，或确定特定模型的超参数取值。而模型评估则是在选定模型及其超参数的情况下，对其在未见数据上的泛化能力进行估计。之前我们强调评价模型的优劣不能只看模型在训练集上的性能，而要看模型在未知数据上的预测或泛化能力。为了估计模型在未知数据的预测能力，一般会从数据集中随机地划分出一部分作为测试集（Test Set）。在模型构建过程中，测试集要保持“未见”状态，不应用于模型选择和超参确定，只能在最终评估阶段才被使用。如果我们在模型构建过程中反复使用测试集，并且选择具有最小测试集误差的模型，那么最终选择的模型的测试集误差将可能会严重低估真实的测试误差。

既然测试集不能用于模型选择阶段，那么如何进行模型挑选和超参选取呢？直接在训练集上进行模型选择仍然会导致所选择的模型过拟合于训练集。举例来说，在之前提到的 K 近邻模型中，如果我们根据训练集的预测准确度来确定超参 K 的取值，通常会选择 $K = 1$（此时训练集预测误差为零），而这显然会导致过拟合的情况。因此，我们需要进一步从数据集中随机划分出一部分数据作为验证集（Validation Set），用于模型挑选和超参选取。总之，在模型的构建过程中，只能在验证集上进行模型选择（包括超参数的选取），而测试集则应该在模型选择之后的

最终评估阶段才被使用。即在整个模型构建过程中，测试集应始终“锁在保险箱”里，直至最终评估阶段，以确保评估的公平性和可靠性。

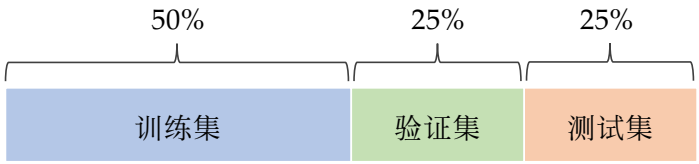


图 1.8: 当数据量比较充足时，可以随机选取 50% 的数据作为训练集，然后在剩余的数据中随机选取一半作为验证集，另一半作为测试集。

对数据集的划分比例没有统一的规则，这通常依赖于可用数据的数量、数据的分布以及模型的复杂性等因素。在数据量较充足的情况下，可以采取以下典型的划分方法（如图1.8所示）：先从整个数据集中随机选取 50% 的数据作为训练集。然后将剩余的数据再随机平分，一半用作验证集，另一半用作测试集。测试集和验证集的样本数量不宜过少，否则可能导致预测误差估计出现显著偏差，进而影响模型的实际泛化能力。

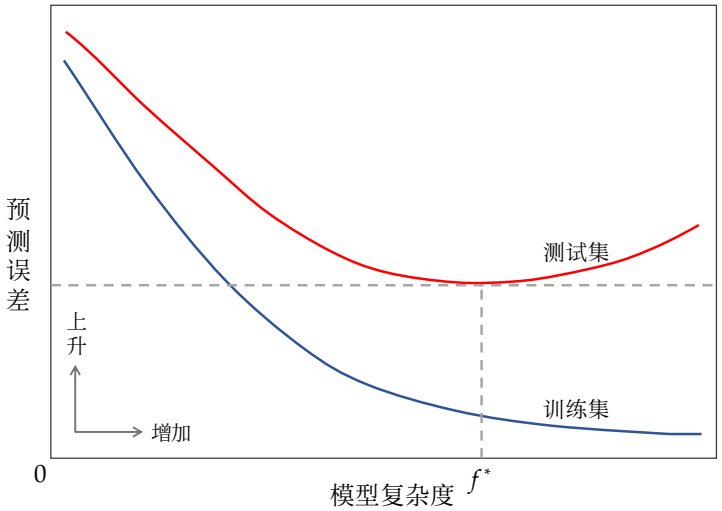


图 1.9: 当模型复杂度增加时，在训练集和测试集上的预测误差的变化趋势。随着模型复杂度的增加，训练集上的预测误差（蓝色曲线）通常会减少。而对于测试集，预测误差（红色曲线）的趋势往往会先降后升。

不同复杂度的模型在训练集和测试集的性能表现通常如图1.9所示。随着模型复杂度的增加，训练误差通常会持续减少。如果模型复杂度增加到足够高时，训练误差还可能会降至零。然而，训练误差为零的模型往往严重过拟合于训练数据，通常会导致泛化能力较差。对于测试集而言，预测误差则通常会经历一个先降低后上升的过程。初期，模型复杂度的增加有助于缓解欠拟合的问题，使模型能够更好地捕捉到数据中的复杂关系和模式，从而提升其对于未知数据的预测能力。然而，当模型复杂度继续增加超过某个临界点后，过拟合现象开始显现，模型过度拟合于训练数据的特定噪声和细节，而这些特性并不适用于新的未知数据。这就导致了模型在未见数据上的泛化能力下降。

理论上，我们可以合理假设随机选择的验证集和测试集服从于同一分布，模型在验证集上的表现可以预见其在测试集上的性能。因而，模型在验证集的上性能变化趋势也与图1.9中红色曲线相似。如果在模型选择阶段，并且将该图中测试集替换成验证集，则两条虚线的交点标识了欠拟合与过拟合之间的临界点，对应于我们应该选择的最优模型 f^*。一旦选定了最优模型，就可以在测试集上对其进行评估，以确定模型的最终性能。对模型的超参进行选择时，也可以采用类似的方法，利用验证集来进行择优，使模型达到性能与复杂度之间的平衡。

图 1.10: 五折交叉验证。先将整个数据集随机划分为五个互不相交的子集。在验证过程中，每一次实验选择其中四个子集作为训练集，而将剩余的一个子集用作测试集。这样可以在五组不同的数据组合上进行训练和测试，最后将这五次不同的测试结果的均值作为最终评估结果。

模型评估的关键在于能够准确地估计模型在未见数据上的性能，而上述方法虽然会随机地划分出测试集来模拟未见数据，但仍然不能排除因不同数据划分造成对模型评估结果的偏差。因而，一种简单而广泛采用的评估方法是 K 折交叉验证法（K-fold Cross-validation）。该方法先将数据集 $\mathcal{D}$ 随机划分成 K 个大小相同的互斥子集，满足 $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K, \mathcal{D}_i \cap \mathcal{D}_j = \varnothing$ $(i \neq j)$。然后，每次选择 $K-1$ 个子集合并组成训练集，剩余的一个子集则作为测试集。如此，可以让模型在 K 种不同的训练集和测试集的组合上进行多次训练和测试。最后，将 K 次测试结果的平均值作为模型的最终评估结果（五折交叉验证的过程如图1.10所示）。采用 K 折交叉验证不仅提高了评估的稳健性，还有助于减少因数据划分不同所可能引起的模型性能评估偏差。K 折交叉验证模拟了所有可能的数据划分情况，既可用于模型选择，也可用于模型评估。值得注意的是，在开始 K 折交叉验证之前，我们应避免在整个数据集上进行任何形式的特征选择或超参选取。为了有效隔离模型选择和评估过程，可以采用嵌套交叉验证，即在模型选择和评估阶段采用不同的数据划分，以确保评估的独立性和可靠性。

将数据集划分成不同的 K 个子集也会存在因多种不同数据划分结果所带来的偶然性影响。为了更稳健地评估模型的性能，K 折交叉验证法还可以使用不同的随机划分重复 P 次，最终的

评估结果取这 P 次 K 折交叉验证结果的均值。假设数据集的样本数量为 N，K 折交叉验证法最极端的形式是取 $K=N$，即留一法（Leave-one-out Cross-validation）。留一法不受数据随机划分的影响，因为它只有一种划分方式（每个子集仅包含一个样本）。在留一法中，每次只有一个样本被留作测试集，其余 $N-1$ 个样本则用作训练集。留一法能够比较精确地估计模型的性能，但计算代价也非常高，不太适合数据集规模较大的情况。使用留一法需要考虑计算资源和时间成本，通常只在对模型评估的准确性要求极高且样本数量有限的情况下采用。

对于监督学习而言，机器学习模型选择与评估主要标准是模型在未见数据上的泛化能力。偏差和方差分解（Bias–variance Decomposition）提供了一种分析模型及其学习算法在未见数据上期望预测误差（Expected Error）的产生原因与组成因素的方法。该方法能够帮助我们进一步理解模型复杂度与其拟合及泛化能力之间的关系。

假设我们有包含 N 个有限数量样本的训练数据集 $\mathcal{D}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，这些数据是从样本的真实分布 $p(\boldsymbol{x},y)$ 随机采样出来的。假设样本的输出 $y(\boldsymbol{x})$ 是在函数 $g(\boldsymbol{x})$ 的结果上叠加高斯随机噪声后生成的（类似第1.5节中拟合 $\sin(2\pi x)$ 函数的例子），则输出 $y(\boldsymbol{x})$ 可以表示为 $y(\boldsymbol{x})=g(\boldsymbol{x})+\epsilon$，其中随机噪声 ϵ 服从均值为零、方差为 σ_ϵ^2 的高斯分布 $\mathcal{N}(0,\sigma_\epsilon^2)$。我们的目标是使得在数据集 $\mathcal{D}$ 上训练得到的模型 $f_{\mathcal{D}}(\boldsymbol{x})$ 尽可能地近似目标函数 $g(\boldsymbol{x})$，这种近似程度理论上可以通过模型在未见样本 $\boldsymbol{x}$ 上的均方误差（Mean Squared Error）的期望值来衡量，即希望 $\mathbb{E}_{\mathcal{D}}[(y(\boldsymbol{x})-f_{\mathcal{D}}(\boldsymbol{x}))^2]$ 的值尽可能小。计算这个期望值需要用训练集 $\mathcal{D}$ 的分布对误差平方求期望。由于我们可以从样本的真实分布 $p(\boldsymbol{x},y)$ 中随机抽取多个不同的训练集，而不同的训练集往往又会得到不同的模型。为了公平且合理地评估模型及其优化算法的性能，可以采用不同训练集上模型性能的平均值作为衡量标准，因而需要对数据集 $\mathcal{D}$ 的分布求期望。模型在未见样本上的期望均方误差可以被分解为不可约误差（Irreducible Error）、偏差（Bias）的平方和方差（Variance）三个组成部分：

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}[(y(\boldsymbol{x})-f_{\mathcal{D}}(\boldsymbol{x}))^2] &= \sigma_\epsilon^2+[\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x})]-g(\boldsymbol{x})]^2+\mathbb{E}[f(\boldsymbol{x})-\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x})]]^2\\ &= \underbrace{\sigma_\epsilon^2}_{\text{不可约误差}}+\underbrace{\text{Bias}^2[f(\boldsymbol{x})]}_{\text{偏差的平方}}+\underbrace{\text{Var}[f(\boldsymbol{x})]}_{\text{方差}}\end{aligned}$$

上述期望均方误差的分解可推导如下（为了简洁，省略了下标 $\mathcal{D}$）：

$$\begin{aligned}&\mathbb{E}[(y(\boldsymbol{x})-f(\boldsymbol{x}))^2]\\ =&\mathbb{E}[y(\boldsymbol{x})^2-2y(\boldsymbol{x})f(\boldsymbol{x})+f(\boldsymbol{x})^2]\\ =&\mathbb{E}[y(\boldsymbol{x})^2]-2\mathbb{E}[y(\boldsymbol{x})f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})^2] && \text{（期望的线性性质）}\\ =&\mathbb{E}[(g(\boldsymbol{x})+\epsilon)^2]-2\mathbb{E}[(g(\boldsymbol{x})+\epsilon)f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})^2] && \text{（}y(\boldsymbol{x})=g(\boldsymbol{x})+\epsilon\text{）}\\ =&\mathbb{E}[g(\boldsymbol{x})^2+2g(\boldsymbol{x})\epsilon+\epsilon^2]-2\mathbb{E}[g(\boldsymbol{x})f(\boldsymbol{x})+\epsilon f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})^2]\\ =&g(\boldsymbol{x})^2+2g(\boldsymbol{x})\mathbb{E}[\epsilon]+\mathbb{E}[\epsilon^2]-2g(\boldsymbol{x})\mathbb{E}[f(\boldsymbol{x})]-2\mathbb{E}[\epsilon]\mathbb{E}[f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})^2] && \text{（}g(\boldsymbol{x})\text{ 不依赖于 }\mathcal{D}\text{）}\\ =&g(\boldsymbol{x})^2+\sigma_\epsilon^2-2g(\boldsymbol{x})\mathbb{E}[f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})^2] && \text{（}\mathbb{E}[\epsilon]=0,\mathbb{E}[\epsilon^2]=\sigma_\epsilon^2\text{）}\\ =&\sigma_\epsilon^2+g(\boldsymbol{x})^2-2g(\boldsymbol{x})\mathbb{E}[f(\boldsymbol{x})]+\mathbb{E}[f(\boldsymbol{x})]^2+\text{Var}[f(\boldsymbol{x})] && \text{（Var}[x]=\mathbb{E}[x^2]-\mathbb{E}[x]^2\text{）}\\ =&\sigma_\epsilon^2+[\mathbb{E}[f(\boldsymbol{x})]-g(\boldsymbol{x})]^2+\mathbb{E}[f(\boldsymbol{x})-\mathbb{E}[f(\boldsymbol{x})]]^2 && \text{（方差的定义）}\end{aligned}$$

不可约误差是指即使采用最优预测模型也无法消除的误差。这种误差源于数据本身的随机性或内在的噪声。偏差衡量了在不同数据集上训练的模型预测值与最优模型预测值之间的平均差异，体现了模型的拟合能力；而方差则衡量了在不同数据集上训练的模型之间的差异，反映了模型是否容易过拟合于特定的训练数据。图1.11展示了模型预测偏差和方差的四种典型表现。在四幅子图中，中心点表示理想最优模型 $g(\boldsymbol{x})$ 的预测值，绿色圆圈代表在不同数据集上训练得到的模型。图1.11(a) 是低偏差、低方差的理解情况；图1.11(b) 为高偏差、低方差的情况，表示模型的拟合能力不足，但不易对特定的训练集产生过拟合；图1.11(c) 是低偏差、高方差的情形，表示模型拟合能力较强，但容易导致过拟合问题；图1.11(d) 则是偏差和方差都高的最差情况。

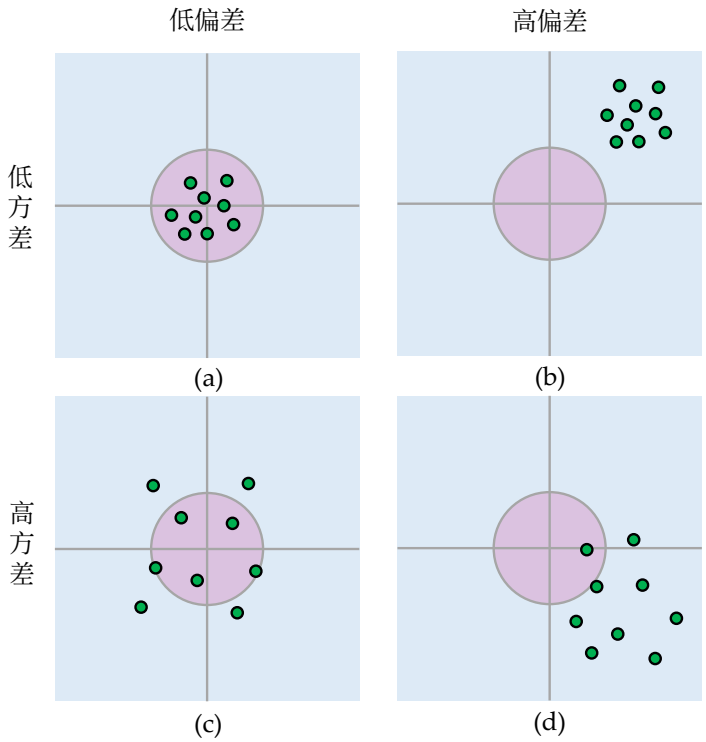


图 1.11: 机器学习模型预测偏差和方差的四种典型表现。

偏差与方差之间的权衡（Bias–variance Tradeoff）是监督学习中的一个核心问题。在理想的情况下，我们希望选择一个既能准确捕捉训练数据中的主要规律，又能很好地泛化到未见数据的模型。然而，这两者通常难以兼顾。高复杂度的模型尽管可以很好地拟合训练集，但容易过拟合于数据中的噪声，导致泛化能力下降；而低复杂度的模型虽然不易过拟合于特定数据集，但可能无法充分捕捉数据中的主要特征和规律，从而导致欠拟合。需要强调的是，认为复杂模型必然伴随高方差是一个常见的误解。虽然高方差的模型通常具有较高的复杂度，但反之则并不成立，即复杂度高并不一定意味着模型会产生高方差的预测（例如人工神经网络模型 [49]）。还需要注意的是，用模型参数的数量来衡量模型的复杂性并不总是可靠。举例来说，函数 $a\sin(bx)$ 只有两个参数 a 和 b。但如果频率足够的高，该函数能够通过高频振荡来拟合任意多的数据点，这会同时导致预测的高偏差和高方差。

对于 K 近邻方法，其偏差和方差分解有如下直观的形式：

$$\mathbb{E}[(y(\boldsymbol{x}) - f(\boldsymbol{x}))^2] = \sigma_\epsilon^2 + \left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} g(\boldsymbol{x}_i) - g(\boldsymbol{x})\right]^2 + \frac{\sigma_\epsilon^2}{K}$$

其中 $\mathcal{S}_K(\boldsymbol{x})$ 是与 $\boldsymbol{x}$ 最近（或相似）的 K 个样本的集合。从上述公式可以看出，当 K 值较小时，偏差的平方项会减小（当 $K = 1$ 时，偏差为零），但方差项会相应增大（与 K 呈反比）。反之，随着 K 值的增大，偏差的平方项会增大，而方差项会减小。上述偏差项的具体推导如下：

$$\begin{aligned}
&[\mathbb{E}[f(\boldsymbol{x})] - g(\boldsymbol{x})]^2 \\
=&\left[\mathbb{E}\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} y(\boldsymbol{x}_i)\right] - g(\boldsymbol{x})\right]^2 && (K \text{ 近邻方法的预测}) \\
=&\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} \mathbb{E}[y(\boldsymbol{x}_i)] - g(\boldsymbol{x})\right]^2 && (\text{期望的线性性质}) \\
=&\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} \mathbb{E}[g(\boldsymbol{x}_i) + \epsilon] - g(\boldsymbol{x})\right]^2 && (y(\boldsymbol{x}) = g(\boldsymbol{x}) + \epsilon) \\
=&\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} \mathbb{E}[g(\boldsymbol{x}_i)] - g(\boldsymbol{x})\right]^2 && (\mathbb{E}[\epsilon] = 0) \\
=&\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} g(\boldsymbol{x}_i) - g(\boldsymbol{x})\right]^2 && (\text{对于特定的 } \boldsymbol{x}_i,\ g(\boldsymbol{x}_i) \text{ 为常数})
\end{aligned}$$

方差项的推导过程如下：

$$\begin{aligned}
&\text{Var}[f(\boldsymbol{x})] \\
=&\text{Var}\left[\frac{1}{K}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} y(\boldsymbol{x}_i)\right] && (K \text{ 近邻方法的预测}) \\
=&\frac{1}{K^2}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} \text{Var}[y(\boldsymbol{x}_i)] && (\text{假设样本 } \boldsymbol{x}_i \text{ 之间相互独立}) \\
=&\frac{1}{K^2}\sum_{\boldsymbol{x}_i \in \mathcal{S}_K(\boldsymbol{x})} \text{Var}[g(\boldsymbol{x}_i) + \epsilon] && (y(\boldsymbol{x}) = g(\boldsymbol{x}) + \epsilon) \\
=&\frac{1}{K^2} K \cdot \text{Var}[\epsilon] && (\text{常数 } g(\boldsymbol{x}_i) \text{ 的方差为零}) \\
=&\frac{\sigma_\epsilon^2}{K} && (\text{Var}[\epsilon] = \sigma_\epsilon^2)
\end{aligned}$$

统计学家乔治 · 博克斯 (George Box) 有一句名言 [10]：

“所有模型都是错的，但其中有些是有用的!” ——没有免费午餐定理

这句话想表达的意思是：现实世界中的数据分布通常极其复杂（比如语言和图像），而我们所考察的模型都会基于一系列假设，而这些假设所获得的理论分布必然与数据的真实分布存在一定差异。尽管如此，通过精心的设计和选择，我们仍然能够找到一些实用的模型来解决实际问题。机器学习的许多研究都致力于设计不同的模型以及拟合这些模型的不同算法。我们可以应用交叉验证等方法筛选出针对特定问题有效的模型和算法。然而，并不存在一个普适的最佳模型。在某个应用场景中表现出色的模型（这通常基于对特定数据分布的假设），在另一个场景中可能就不那么有效了。这句话也被称为“没有免费午餐定理”（No Free Lunch Theorem）。因此，面对现实世界中数据的多样性，我们需要开发各种类型的模型。而对于每种模型，又可采用不同的算法来训练它们。这些算法在效率、准确性和复杂度方面各有千秋，需要我们在多个维度上进行权衡取舍。

与模型选择相关的另一个原理是奥卡姆剃刀原理（Occam’s Razor），又称为奥卡姆的剃刀。它原本是一种解决问题和理论选择的哲学原则，由 14 世纪的英国哲学家和逻辑学家奥卡姆的威廉（William of Occam）所提出。主张在所有其他条件相同的前提下，应优先选择假设最少、最简单的解释或理论。这并不意味着最简单的理论总是正确的，但是在缺乏额外证据的情况下，这种方法有助于避免不必要的复杂性和过度解释。这里的“剃刀”是一种比喻，用来形容简化或剔除多余的、复杂的、没有必要的假设，从而让理论更加简洁明了。在机器学习领域，奥卡姆剃刀原理也被用来指导模型或特征选择过程，意味着当多个模型都能够有效解释数据时，应选择复杂度最低的简单模型。因为简单模型通常更易于理解、训练成本更低，且更不容易过拟合。一言以蔽之，可选者众，取其最简。

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

[9] 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.

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

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

[12] 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.

[13] 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.

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

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

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

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

[18] 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.

[19] 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).

[20] 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).

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

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

[23] 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.

[24] 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.

[25] 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.

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

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

[28] 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.

[29] 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.

[30] 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.

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

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

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

[34] 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.

[35] 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.

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

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

[38] 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.

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

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

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

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

[43] 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.

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

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

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

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

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

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

[50] 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.

[51] 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.

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

[53] 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).

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

[55] 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.

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

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

[58] 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.

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

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

[61] 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).

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

[63] 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.

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

[65] 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).

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

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

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

[69] 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.

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

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

[72] 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.

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

[74] 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.