# 机器学习方法

李 航 著

清華大學出版社
北京

## 内 容 简 介

机器学习是以概率论、统计学、信息论、最优化理论、计算理论等为基础的计算机应用理论学科，也是人工智能、数据挖掘等领域的基础学科。本书全面系统地介绍了机器学习的主要方法，共分 3 篇。第 1 篇介绍监督学习的主要方法，包括感知机、$k$ 近邻法、朴素贝叶斯法、决策树、逻辑斯谛回归与最大熵模型、支持向量机、Boosting、EM 算法、隐马尔可夫模型、条件随机场等；第 2 篇介绍无监督学习的主要方法，包括聚类、奇异值分解、主成分分析、潜在语义分析、概率潜在语义分析、马尔可夫链蒙特卡罗法、潜在狄利克雷分配、PageRank 算法等；第 3 篇介绍深度学习的主要方法，包括前馈神经网络、卷积神经网络、循环神经网络、序列到序列模型、预训练语言模型、生成对抗网络等。书中每章介绍一两种机器学习方法。详细叙述各个方法的模型、策略和算法。从具体例子入手，由浅入深，帮助读者直观地理解基本思路，同时从理论角度出发，给出严格的数学推导，严谨详实，让读者更好地掌握基本原理和概念。目的是使读者能学会和使用这些机器学习的基本技术。为满足读者进一步学习的需要，书中还对各个方法的要点进行了总结，给出了一些习题，并列出了主要参考文献。

本书是机器学习及相关课程的教学参考书，适合人工智能、数据挖掘等专业的本科生、研究生使用，也可供计算机各个领域的专业研发人员参考。



**图书在版编目(CIP)数据**

机器学习方法/李航著.—北京：清华大学出版社，2022.1
ISBN 978-7-302-59730-8

Ⅰ. ①机… Ⅱ. ①李… Ⅲ. ①机器学习 Ⅳ. ①TP181

中国版本图书馆 CIP 数据核字(2021)第 277513 号

**责任编辑**：王 倩
**封面设计**：李祥榕
**责任校对**：王淑云
**责任印制**：丛怀宇

**出版发行**：清华大学出版社
**网 址**：http://www.tup.com.cn，http://www.wqbook.com
**地 址**：北京清华大学学研大厦 A 座 **邮 编**：100084
**社 总 机**：010-83470000 **邮 购**：010-62786544
**投稿与读者服务**：010-62776969，c-service@tup.tsinghua.edu.cn
**质量反馈**：010-62772015，zhiliang@tup.tsinghua.edu.cn
**印 装 者**：三河市东方印刷有限公司
**经 销**：全国新华书店
**开 本**：185mm×260mm **印 张**：35.75 **插 页**：4 **字 数**：880 千字
**版 次**：2022 年 3 月第 1 版 **印 次**：2022 年 3 月第 1 次印刷
**定 价**：138.00 元

产品编号：093532-01

献给我的母亲

# 序　　言

2012 年《统计学习方法 (第 1 版)》出版，内容涵盖监督学习的主要方法，2019 年第 2 版出版，增加了无监督学习的主要方法，都属于传统机器学习。在这段时间里，机器学习领域发生了巨大变化，深度学习在人工智能各个应用方向取得了巨大突破，成为机器学习的主流技术，彻底改变了机器学习的面貌。有些读者希望能看到与之前风格相同的讲解深度学习的书籍，这也触发了作者在原来《统计学习方法》的基础上增加深度学习内容的想法（计划今后再增加强化学习）。从 2018 年开始，历时 3 年左右，完成了深度学习的写作。

考虑到内容的变化，现将书名更改为《机器学习方法》。第 1 篇监督学习和第 2 篇无监督学习基本为原来的内容，增加第 3 篇深度学习，希望对读者有所裨益。传统机器学习是深度学习的基础，所以将这些内容放在一本书里讲述也有其合理之处。虽然深度学习目前是大家关注的重点，但传统机器学习仍然有其不容忽视的地位。事实上，传统机器学习和深度学习各自有更适合的应用场景，比如，深度学习长于大数据、复杂问题的预测，特别是人工智能的应用；传统机器学习善于小数据、相对简单问题的预测。

本书的定位是讲解机器学习的基本内容，并不完全是入门书。介绍的内容都是最基本的，在这种意义上适合初学者。但主旨是把最重要的原理和方法做系统的总结，方便大家经常阅读和复习。在写第 3 篇的时候也接受大家对第 1 篇和第 2 篇的反馈意见，在力求文字简练清晰的同时，也确保叙述的详尽明了，以方便读者理解。在各章方法的导入部分适当增加了背景和动机的介绍。

第 3 篇中使用的数学符号与第 1 篇和第 2 篇有一定的对应关系，但由于深度学习的特点也有一些改变，也都能自成体系。将符号完全统一于一个框架内还需要做大量的工作，希望在增加第 4 篇强化学习之后再做处理。

对第 3 篇的原稿，郑诗源、张新松等帮助做了校阅，对一些章节的内容提出了宝贵的意见。责任编辑王倩也为本书的出版做了大量工作。在此对他们表示衷心的感谢。

李　航

2021 年 5 月 27 日

# 《统计学习方法 (第 2 版)》序言

《统计学习方法 (第 1 版)》于 2012 年出版，讲述了统计机器学习方法，主要是一些常用的监督学习方法。第 2 版增加了一些常用的无监督学习方法，由此本书涵盖了传统统计机器学习方法的主要内容。

在撰写《统计学习方法》伊始，对全书内容做了初步规划。第 1 版出版之后，即着手无监督学习方法的写作。由于写作是在业余时间进行，常常被主要工作打断，历经 6 年多时间才使这部分工作得以完成。但犹未能加入深度学习和强化学习等重要内容，希望今后能够增补，完成整本书的写作计划。

《统计学习方法 (第 1 版)》的出版正值大数据和人工智能的热潮，生逢其时，截至 2019 年 4 月本书共印刷 25 次，152 000 册，得到了广大读者的欢迎和支持。有许多读者指出本书对学习和掌握机器学习技术有极大的帮助，也有许多读者通过电子邮件、微博等指出书中的错误，提出改进的建议和意见。一些高校将本书作为机器学习课程的教材或参考书。有的同学在网上发表了读书笔记，有的同学将本书介绍的方法在计算机上实现。清华大学深圳研究生院袁春老师精心制作了第 1 版 12 章的课件，在网上公布，为大家提供教学之便。众多老师、同学、读者的支持和鼓励，让作者深受感动和鼓舞。在这里向所有的老师、同学、读者致以诚挚的谢意!

能为中国的计算机科学、人工智能领域做出一点微薄的贡献，感到由衷的欣慰，同时也感受到作为知识传播者的重大责任，让作者决意把本书写好。也希望大家今后不吝指教，多提宝贵意见，以帮助继续提高本书的质量。在写作中作者也深切体会到教学相长的道理，经常发现自己对基础知识的掌握不够扎实，通过写作得以对相关知识进行深入的学习，受益匪浅。

本书是一部机器学习的基本读物，要求读者拥有高等数学、线性代数和概率统计的基础知识。书中主要讲述统计机器学习的方法，力求系统全面又简明扼要地阐述这些方法的理论、算法和应用，使读者能对这些机器学习的基本技术有很好的掌握。针对每个方法，详细介绍其基本原理、基础理论、实际算法，给出细致的数学推导和具体实例，既帮助读者理解，也便于日后复习。

第 2 版增加的无监督学习方法，王泉、陈嘉怡、柴琛林、赵程绮等帮助做了认真细致的校阅，提出了许多宝贵意见，在此谨对他们表示衷心的感谢。清华大学出版社的薛慧编辑一直对本书的写作给予非常专业的指导和帮助，在此对她表示衷心的感谢!

由于本人水平有限，本书一定存在不少错误，恳请各位专家、老师和同学批评指正。

李　航

2019 年 4 月

# 《统计学习方法 (第 1 版)》序言

计算机与网络已经融入人们的日常学习、工作和生活之中，成为人们不可或缺的助手和伙伴。计算机与网络的飞速发展完全改变了人们的学习、工作和生活方式。智能化是计算机研究与开发的一个主要目标。近几十年来的实践表明，统计机器学习方法是实现这一目标的最有效手段，尽管它还存在着一定的局限性。

本人一直从事利用统计学习方法对文本数据进行各种智能性处理的研究，包括自然语言处理、信息检索、文本数据挖掘。近 20 年来，这些领域发展之快，应用之广，实在令人惊叹！可以说，统计机器学习是这些领域的核心技术，在这些领域的发展及应用中起着决定性的作用。

本人在日常的研究工作中经常指导学生，并在国内外一些大学及讲习班上多次做过关于统计学习的报告和演讲。在这一过程中，同学们学习热情很高，希望得到指导，这使作者产生了撰写本书的想法。

国内外已出版了多本关于统计机器学习的书籍，比如，Hastie 等人的《统计学习基础》，该书对统计学习的诸多问题有非常精辟的论述，但对初学者来说显得有些深奥。统计学习范围甚广，一两本书很难覆盖所有问题。本书主要是面向将统计学习方法作为工具的科研人员与学生，特别是从事信息检索、自然语言处理、文本数据挖掘及相关领域的研究与开发的科研人员与学生。

本书力求系统而详细地介绍统计学习的方法。在内容选取上，侧重介绍那些最重要、最常用的方法，特别是关于分类与标注问题的方法。对其他问题及方法，如聚类等，计划在今后的写作中再加以介绍。在叙述方式上，每一章讲述一种方法，各章内容相对独立、完整；同时力图用统一框架来论述所有方法，使全书整体不失系统性，读者可以从头到尾通读，也可以选择单个章节细读。对每一种方法的讲述力求深入浅出，给出必要的推导证明，提供简单的实例，使初学者易于掌握该方法的基本内容，领会方法的本质，并准确地使用方法。对相关的深层理论，则予以简述。在每章后面，给出一些习题，介绍一些相关的研究动向和阅读材料，列出参考文献，以满足读者进一步学习的需求。本书第 1 章简要叙述统计学习方法的基本概念，最后一章对统计学习方法进行比较与总结。此外，在附录中简要介绍一些共用的最优化理论与方法。

本书可以作为统计机器学习及相关课程的教学参考书，适用于信息检索及自然语言处理等专业的大学生、研究生。

本书初稿完成后，田飞、王佳磊、武威、陈凯、伍浩铖、曹正、陶宇等人分别审阅了全部

或部分章节，提出了许多宝贵意见，对本书质量的提高有很大帮助，在此向他们表示衷心的感谢。在本书的写作和出版过程中，清华大学出版社的责任编辑薛慧给予了很多帮助，在此特向她致谢。

由于本人水平所限，书中难免有错误和不当之处，欢迎各位专家和读者给予批评指正。

李 航

2011 年 4 月 23 日

# 目　　录

## 第 1 篇　监 督 学 习

## 第 3 篇 深 度 学 习

# 第 1 篇　监 督 学 习

# 第 1 章　机器学习及监督学习概论

本书第 1 篇讲述监督学习方法。监督学习是从标注数据中学习模型的机器学习问题，是机器学习的重要组成部分。

本章简要叙述机器学习及监督学习的一些基本概念，使读者对机器学习及监督学习有初步了解。1.1 节叙述机器学习或统计机器学习的定义、研究对象与方法；1.2 节叙述机器学习的分类，基本分类是监督学习、无监督学习、强化学习；1.3 节叙述机器学习方法的三要素：模型、策略和算法；1.4 节至 1.7 节相继介绍监督学习的几个重要概念，包括模型评估与模型选择、正则化与交叉验证、学习的泛化能力、生成模型与判别模型；最后 1.8 节介绍监督学习的应用：分类问题、标注问题与回归问题。

## 1.1　机 器 学 习

### 1. 机器学习的特点

机器学习（machine learning）是关于计算机基于数据构建概率统计模型并运用模型对数据进行预测与分析的一门学科。机器学习也称为统计机器学习 (statistical machine learning）。

机器学习的主要特点是：①机器学习以计算机及网络为平台，是建立在计算机及网络上的；②机器学习以数据为研究对象，是数据驱动的学科；③机器学习的目的是对数据进行预测与分析；④机器学习以方法为中心，机器学习方法构建模型并应用模型进行预测与分析；⑤机器学习是概率论、统计学、信息论、计算理论、最优化理论及计算机科学等多个领域的交叉学科，并且在发展中逐步形成独自的理论体系与方法论。

赫尔伯特 • 西蒙（Herbert A. Simon）曾对“学习”给出以下定义：“如果一个系统能够通过执行某个过程改进它的性能，这就是学习。”按照这一观点，机器学习就是计算机系统通过运用数据及统计方法提高系统性能的学习。

### 2. 机器学习的对象

机器学习研究的对象是数据（data）。它从数据出发，提取数据的特征，抽象出数据的模型，发现数据中的知识，又回到对数据的分析与预测中去。作为机器学习的对象，数据是多样的，包括存在于计算机及网络上的各种数字、文字、图像、视频、音频数据以及它们的组合。

机器学习关于数据的基本假设是同类数据具有一定的统计规律性，这是机器学习的前

提。这里的同类数据是指具有某种共同性质的数据，例如英文文章、互联网网页、数据库中的数据等。由于它们具有统计规律性，所以可以用概率统计方法处理它们。比如，可以用随机变量描述数据中的特征，用概率分布描述数据的统计规律。在机器学习中，以变量或变量组表示数据。数据分为由连续变量和离散变量表示的类型。本书以讨论离散变量的方法为主。另外，本书只涉及利用数据构建模型及利用模型对数据进行分析与预测，对数据的观测和收集等问题不作讨论。

3. 机器学习的目的

机器学习用于对数据的预测与分析，特别是对未知新数据的预测与分析。对数据的预测可以使计算机更加智能化，或者说使计算机的某些性能得到提高；对数据的分析可以让人们获取新的知识，给人们带来新的发现。

对数据的预测与分析是通过构建概率统计模型实现的。机器学习总的目标就是考虑学习什么样的模型和如何学习模型，以使模型能对数据进行准确的预测与分析，同时也要考虑尽可能地提高学习效率。

4. 机器学习的方法

机器学习的方法是基于数据构建概率统计模型从而对数据进行预测与分析。机器学习由监督学习（supervised learning）、无监督学习（unsupervised learning）和强化学习（reinforcement learning）等组成。

本书第 1 篇讲述监督学习，第 2 篇讲述无监督学习。可以说监督学习、无监督学习方法是最主要的机器学习方法。第 3 篇讲述深度学习，既可以用于监督学习，也可以用于无监督学习。

机器学习方法可以概括如下：从给定的、有限的、用于学习的训练数据（training data）集合出发，假设数据是独立同分布产生的；并且假设要学习的模型属于某个函数的集合，称为假设空间（hypothesis space）；应用某个评价准则（evaluation criterion），从假设空间中选取一个最优模型，使它对已知的训练数据及未知的测试数据（test data）在给定的评价准则下有最优的预测；最优模型的选取由算法实现。这样，机器学习方法包括模型的假设空间、模型选择的准则以及模型学习的算法，称为机器学习方法的三要素，简称为模型（model）、策略（strategy）和算法（algorithm）。

实现机器学习方法的步骤如下：

（1）得到一个有限的训练数据集合；

（2）确定包含所有可能的模型的假设空间，即学习模型的集合；

（3）确定模型选择的准则，即学习的策略；

（4）实现求解最优模型的算法，即学习的算法；

（5）通过学习方法选择最优模型；

（6）利用学习的最优模型对新数据进行预测或分析。

本书第 1 篇介绍监督学习方法，主要包括用于分类、标注与回归问题的方法。这些方法在自然语言处理、信息检索、文本数据挖掘等领域中有着极其广泛的应用。

5. 机器学习的研究

机器学习研究一般包括机器学习方法、机器学习理论及机器学习应用三个方面。机器学

习方法的研究旨在开发新的学习方法；机器学习理论的研究在于探求机器学习方法的有效性与效率，以及机器学习的基本理论问题；机器学习应用的研究主要考虑将机器学习方法应用到实际问题中去，解决实际问题。

6. 机器学习的重要性

近几十年来，机器学习无论是在理论还是在应用方面都得到了巨大的发展，有许多重大突破，机器学习已被成功地应用到人工智能、模式识别、数据挖掘、自然语言处理、语音处理、计算视觉、信息检索、生物信息等许多计算机应用领域中，并且成为这些领域的核心技术。人们确信，机器学习将会在今后的科学发展和技术应用中发挥越来越大的作用。

机器学习学科在科学技术中的重要性主要体现在以下几个方面：

（1）机器学习是处理海量数据的有效方法。我们处于一个信息爆炸的时代，海量数据的处理与利用是人们必然的需求。现实中的数据不但规模大，而且常常具有不确定性，机器学习往往是处理这类数据最强有力的工具。

（2）机器学习是计算机智能化的有效手段。智能化是计算机发展的必然趋势，也是计算机技术研究与开发的主要目标。近几十年来，人工智能等领域的研究证明，利用机器学习模仿人类智能的方法虽有一定的局限性，但是还是实现这一目标的最有效手段。

（3）机器学习是计算机科学发展的一个重要组成部分。可以认为计算机科学由三维组成：系统、计算、信息。机器学习主要属于信息这一维，并在其中起着核心作用。

## 1.2 机器学习的分类

机器学习或统计机器学习是一个范围宽阔、内容繁多、应用广泛的领域，并不存在（至少现在不存在）一个统一的理论体系涵盖所有内容。下面从几个角度对机器学习方法进行分类。

### 1.2.1 基本分类

机器学习一般包括监督学习、无监督学习、强化学习。有时还包括半监督学习、主动学习。

1. 监督学习

监督学习（supervised learning）是指从标注数据中学习预测模型的机器学习问题。标注数据表示输入输出的对应关系，预测模型对给定的输入产生相应的输出。监督学习的本质是学习输入到输出的映射的统计规律。

（1）输入空间、特征空间和输出空间

在监督学习中，将输入与输出所有可能取值的集合分别称为输入空间（input space）与输出空间（output space）。输入空间与输出空间可以是有限元素的集合，也可以是整个欧氏空间。输入空间与输出空间可以是同一个空间，也可以是不同的空间，但通常输出空间远远小于输入空间。

每个具体的输入是一个实例（instance），通常由特征向量（feature vector）表示。这时，

所有特征向量存在的空间称为特征空间（feature space）。特征空间的每一维对应一个特征。有时假设输入空间与特征空间为相同的空间，对它们不予区分；有时假设输入空间与特征空间为不同的空间，将实例从输入空间映射到特征空间。模型实际上都是定义在特征空间上的。

在监督学习中，将输入与输出看作是定义在输入（特征）空间与输出空间上的随机变量的取值。输入输出变量用大写字母表示，习惯上输入变量写作 $X$，输出变量写作 $Y$。输入输出变量的取值用小写字母表示，输入变量的取值写作 $x$，输出变量的取值写作 $y$。变量可以是标量或向量，都用相同类型字母表示。除特别声明外，本书中向量均为列向量。输入实例 $x$ 的特征向量记作

$$x = \left(x^{(1)}, x^{(2)}, \cdots, x^{(i)}, \cdots, x^{(n)}\right)^{\mathrm{T}}$$

其中，$x^{(i)}$ 表示 $x$ 的第 $i$ 个特征。注意 $x^{(i)}$ 与 $x_i$ 不同，本书通常用 $x_i$ 表示多个输入变量中的第 $i$ 个变量，即

$$x_i = \left(x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(n)}\right)^{\mathrm{T}}$$

监督学习从训练数据（training data）集合中学习模型，对测试数据（test data）进行预测。训练数据由输入（或特征向量）与输出对组成，训练集通常表示为

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

测试数据也由输入与输出对组成。输入与输出对又称为样本（sample）或样本点。

输入变量 $X$ 和输出变量 $Y$ 有不同的类型，可以是连续的，也可以是离散的。人们根据输入输出变量的不同类型，对预测任务给予不同的名称：输入变量与输出变量均为连续变量的预测问题称为回归问题；输出变量为有限个离散变量的预测问题称为分类问题；输入变量与输出变量均为变量序列的预测问题称为标注问题。

（2）联合概率分布

监督学习假设输入与输出的随机变量 $X$ 和 $Y$ 遵循联合概率分布 $P(X,Y)$。$P(X,Y)$ 表示分布函数或分布密度函数。注意在学习过程中，假定这一联合概率分布存在，但对学习系统来说，联合概率分布的具体定义是未知的。训练数据与测试数据被看作是依联合概率分布 $P(X,Y)$ 独立同分布产生的。机器学习假设数据存在一定的统计规律，$X$ 和 $Y$ 具有联合概率分布就是监督学习关于数据的基本假设。

（3）假设空间

监督学习的目的在于学习一个由输入到输出的映射，这一映射由模型来表示。换句话说，学习的目的就在于找到最好的这样的模型。模型属于由输入空间到输出空间的映射的集合，这个集合就是假设空间（hypothesis space）。假设空间的确定意味着学习的范围的确定。

监督学习的模型可以是概率模型或非概率模型，由条件概率分布 $P(Y|X)$ 或决策函数（decision function）$Y = f(X)$ 表示，随具体学习方法而定。对具体的输入进行相应的输出预测时，写作 $P(y|x)$ 或 $y = f(x)$。

（4）问题的形式化

监督学习利用训练数据集学习一个模型，再用模型对测试样本集进行预测。由于在这个

过程中需要标注训练数据集，而标注的训练数据集往往是人工给出的，所以称为监督学习。监督学习分为学习和预测两个过程，由学习系统与预测系统完成，可用图 1.1 来描述。

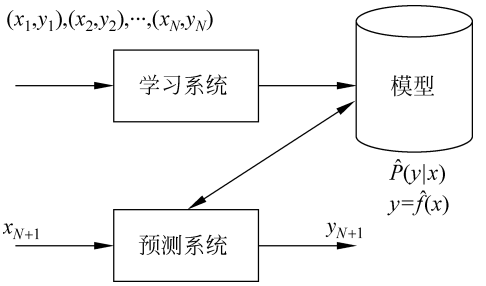


图 1.1 监督学习

首先给定一个训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中 $(x_i, y_i)$，$i = 1, 2, \cdots, N$，称为样本或样本点。$x_i \in \mathcal{X} \subseteq \boldsymbol{R}^n$ 是输入的观测值，也称为输入或实例，$y_i \in \mathcal{Y}$ 是输出的观测值，也称为输出。

监督学习分为学习和预测两个过程，由学习系统与预测系统完成。在学习过程中，学习系统利用给定的训练数据集，通过学习（或训练）得到一个模型，表示为条件概率分布 $\hat{P}(Y|X)$ 或决策函数 $Y = \hat{f}(X)$。条件概率分布 $\hat{P}(Y|X)$ 或决策函数 $Y = \hat{f}(X)$ 描述输入与输出随机变量之间的映射关系。在预测过程中，预测系统对于给定的测试样本集中的输入 $x_{N+1}$，由模型 $y_{N+1} = \arg\max\limits_{y} \hat{P}(y|x_{N+1})$ 或 $y_{N+1} = \hat{f}(x_{N+1})$ 给出相应的输出 $y_{N+1}$。

在监督学习中，假设训练数据与测试数据是依联合概率分布 $P(X, Y)$ 独立同分布产生的。

学习系统（也就是学习算法）试图通过训练数据集中的样本 $(x_i, y_i)$ 带来的信息学习模型。具体地说，对输入 $x_i$，一个具体的模型 $y = f(x)$ 可以产生一个输出 $f(x_i)$，而训练数据集中对应的输出是 $y_i$。如果这个模型有很好的预测能力，训练样本输出 $y_i$ 和模型输出 $f(x_i)$ 之间的差就应该足够小。学习系统通过不断地尝试，选取最好的模型，以便对训练数据集有足够好的预测，同时对未知的测试数据集的预测也有尽可能好的推广。

2. 无监督学习

无监督学习[①]（unsupervised learning）是指从无标注数据中学习预测模型的机器学习问题。无标注数据是自然得到的数据，预测模型表示数据的类别、转换或概率。无监督学习的本质是学习数据中的统计规律或潜在结构。

模型的输入与输出的所有可能取值的集合分别称为输入空间与输出空间。输入空间与输出空间可以是有限元素集合，也可以是欧氏空间。每个输入是一个实例，由特征向量表示。每一个输出是对输入的分析结果，由输入的类别、转换或概率表示。模型可以实现对数据的聚类、降维或概率估计。

假设 $\mathcal{X}$ 是输入空间，$\mathcal{Z}$ 是隐式结构空间。要学习的模型可以表示为函数 $z = g(x)$、条件

① 也译作非监督学习。

概率分布 $P(z|x)$ 或者条件概率分布 $P(x|z)$ 的形式，其中 $x \in \mathcal{X}$ 是输入，$z \in \mathcal{Z}$ 是输出。包含所有可能的模型的集合称为假设空间。无监督学习旨在从假设空间中选出在给定评价标准下的最优模型。

无监督学习通常使用大量的无标注数据学习或训练，每一个样本是一个实例。训练数据表示为 $U = \{x_1, x_2, \cdots, x_N\}$，其中 $x_i$，$i = 1, 2, \cdots, N$, 是样本。

无监督学习可以用于对已有数据的分析，也可以用于对未来数据的预测。分析时使用学习得到的模型，即函数 $z = \hat{g}(x)$、条件概率分布 $\hat{P}(z|x)$ 或者条件概率分布 $\hat{P}(x|z)$。预测时，和监督学习有类似的流程。由学习系统与预测系统完成，如图 1.2 所示。在学习过程中，学习系统从训练数据集学习，得到一个最优模型，表示为函数 $z = \hat{g}(x)$、条件概率分布 $\hat{P}(z|x)$ 或者条件概率分布 $\hat{P}(x|z)$。在预测过程中，预测系统对于给定的输入 $x_{N+1}$，由模型 $z_{N+1} = \hat{g}(x_{N+1})$ 或 $z_{N+1} = \arg\max\limits_{z} \hat{P}(z|x_{N+1})$ 给出相应的输出 $z_{N+1}$，进行聚类或降维，或者由模型 $\hat{P}(x|z)$ 给出输入的概率 $\hat{P}(x_{N+1}|z_{N+1})$，进行概率估计。

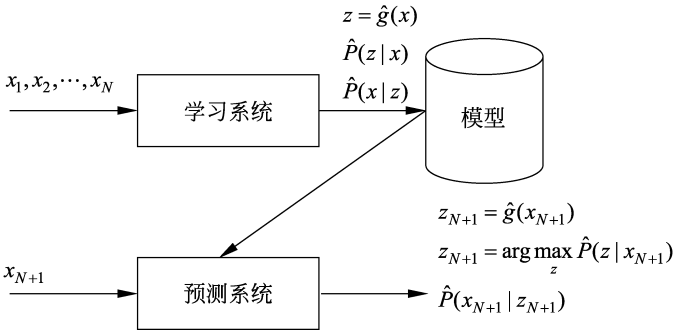


图 1.2 无监督学习

### 3. 强化学习

强化学习（reinforcement learning）是指智能系统在与环境的连续互动中学习最优行为策略的机器学习问题。假设智能系统与环境的互动基于马尔可夫决策过程（Markov decision process），智能系统能观测到的是与环境互动得到的数据序列。强化学习的本质是学习最优的序贯决策。

智能系统与环境的互动如图 1.3 所示。在每一步 $t$，智能系统从环境中观测到一个状态（state）$s_t$ 与一个奖励（reward）$r_t$，采取一个动作（action）$a_t$。环境根据智能系统选择的

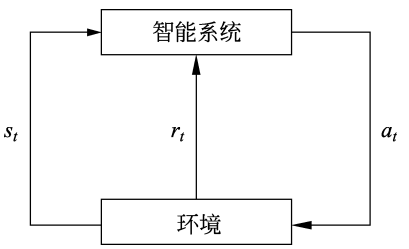


图 1.3 智能系统与环境的互动

动作，决定下一步 $t+1$ 的状态 $s_{t+1}$ 与奖励 $r_{t+1}$。要学习的策略表示为给定的状态下采取的动作。智能系统的目标不是短期奖励的最大化，而是长期累积奖励的最大化。强化学习过程中，系统不断地试错（trial and error），以达到学习最优策略的目的。

强化学习的马尔可夫决策过程是状态、奖励、动作序列上的随机过程，由四元组 $\langle S, A, P, r\rangle$ 组成。

- $S$ 是有限状态（state）的集合。
- $A$ 是有限动作（action）的集合。
- $P$ 是状态转移概率（transition probability）函数:

$$P(s'|s,a) = P(s_{t+1} = s'|s_t = s, a_t = a)$$

- $r$ 是奖励函数（reward function）: $r(s,a) = E(r_{t+1}|s_t = s, a_t = a)$。

马尔可夫决策过程具有马尔可夫性，下一个状态只依赖于前一个状态与动作，由状态转移概率函数 $P(s'|s,a)$ 表示。下一个奖励依赖于前一个状态与动作，由奖励函数 $r(s,a)$ 表示。

策略 $\pi$ 定义为给定状态下动作的函数 $a = f(s)$ 或者条件概率分布 $P(a|s)$。给定一个策略 $\pi$，智能系统与环境互动的行为就已确定（或者是确定性的或者是随机性的）。

价值函数（value function）或状态价值函数（state value function）定义为策略 $\pi$ 从某一个状态 $s$ 开始的长期累积奖励的数学期望：

$$v_\pi(s) = E_\pi[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots |s_t = s] \tag{1.1}$$

动作价值函数（action value function）定义为策略 $\pi$ 从某一个状态 $s$ 和动作 $a$ 开始的长期累积奖励的数学期望：

$$q_\pi(s,a) = E_\pi[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots |s_t = s, a_t = a] \tag{1.2}$$

强化学习的目标就是在所有可能的策略中选出价值函数最大的策略 $\pi^*$，而在实际学习中往往从具体的策略出发，不断优化已有策略。这里 $\gamma$ 是折扣率，表示未来的奖励会有衰减。

强化学习方法中有基于策略的（policy-based）、基于价值的（value-based），这两者属于无模型的（model-free）方法，还有有模型的（model-based）方法。

有模型的方法试图直接学习马尔可夫决策过程的模型，包括转移概率函数 $P(s'|s,a)$ 和奖励函数 $r(s,a)$。这样可以通过模型对环境的反馈进行预测，求出价值函数最大的策略 $\pi^*$。

无模型的、基于策略的方法不直接学习模型，而是试图求解最优策略 $\pi^*$，表示为函数 $a = f^*(s)$ 或者是条件概率分布 $P^*(a|s)$，这样也能达到在环境中做出最优决策的目的。学习通常从一个具体策略开始，通过搜索更优的策略进行。

无模型的、基于价值的方法也不直接学习模型，而是试图求解最优价值函数，特别是最优动作价值函数 $q^*(s,a)$。这样可以间接地学到最优策略，根据该策略在给定的状态下做出相应的动作。学习通常从一个具体价值函数开始，通过搜索更优的价值函数进行。

4. 半监督学习与主动学习

半监督学习（semi-supervised learning）是指利用标注数据和未标注数据学习预测模型的机器学习问题。通常有少量标注数据、大量未标注数据，因为标注数据的构建往往需要人

工，成本较高，未标注数据的收集不需太多成本。半监督学习旨在利用未标注数据中的信息，辅助标注数据，进行监督学习，以较低的成本达到较好的学习效果。

主动学习（active learning）是指机器不断主动给出实例让教师进行标注，然后利用标注数据学习预测模型的机器学习问题。通常的监督学习使用给定的标注数据，往往是随机得到的，可以看作是“被动学习”，主动学习的目标是找出对学习最有帮助的实例让教师标注，以较小的标注代价达到较好的学习效果。

半监督学习和主动学习更接近监督学习。

### 1.2.2 按模型分类

机器学习或统计机器学习方法可以根据其模型的种类进行分类。

1. 概率模型与非概率模型

机器学习的模型可以分为概率模型（probabilistic model）和非概率模型（non-probabilistic model）或者确定性模型（deterministic model）。在监督学习中，概率模型取条件概率分布形式 $P(y|x)$，非概率模型取函数形式 $y = f(x)$，其中 $x$ 是输入，$y$ 是输出。在无监督学习中，概率模型取条件概率分布形式 $P(z|x)$ 或 $P(x|z)$，非概率模型取函数形式 $z = g(x)$，其中 $x$ 是输入，$z$ 是输出。

本书介绍的决策树、朴素贝叶斯、隐马尔可夫模型、条件随机场、概率潜在语义分析、潜在狄利克雷分配、高斯混合模型是概率模型。感知机、支持向量机、$k$ 近邻、AdaBoost、$k$ 均值、潜在语义分析，以及神经网络是非概率模型。逻辑斯谛回归既可看作是概率模型，又可看作是非概率模型。

条件概率分布 $P(y|x)$ 和函数 $y = f(x)$ 可以相互转化（条件概率分布 $P(z|x)$ 和函数 $z = g(x)$ 同样可以）。具体地，条件概率分布最大化后得到函数，函数归一化后得到条件概率分布。所以，概率模型和非概率模型的区别不在于输入与输出之间的映射关系，而在于模型的内在结构。概率模型通常可以表示为联合概率分布的形式，其中的变量表示输入、输出、隐变量甚至参数。而非概率模型不一定存在这样的联合概率分布。

概率模型的代表是概率图模型（probabilistic graphical model），概率图模型是联合概率分布由有向图或者无向图表示的概率模型，而联合概率分布可以根据图的结构分解为因子乘积的形式。贝叶斯网络、马尔可夫随机场、条件随机场是概率图模型。无论模型如何复杂，均可以用最基本的加法规则和乘法规则（参照图 1.4）进行概率推理。

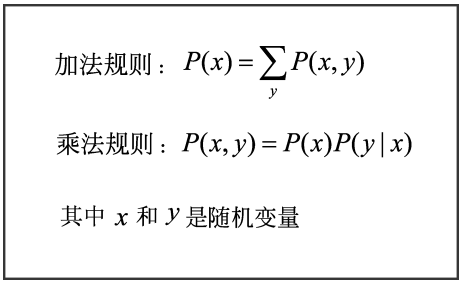


图 1.4 基本概率公式

2. 线性模型与非线性模型

机器学习模型，特别是非概率模型，可以分为线性模型（linear model）和非线性模型（non-linear model）。如果函数 $y = f(x)$ 或 $z = g(x)$ 是线性函数，则称模型是线性模型，否则称模型是非线性模型。

本书介绍的感知机、线性支持向量机、$k$ 近邻、$k$ 均值、潜在语义分析是线性模型，核函数支持向量机、AdaBoost、神经网络是非线性模型。

深度学习（deep learning）实际是复杂神经网络的学习，也就是复杂的非线性模型的学习。

3. 参数化模型与非参数化模型

机器学习模型又可以分为参数化模型（parametric model）和非参数化模型（non-parametric model）。参数化模型假设模型参数的维度固定，模型可以由有限维参数完全刻画；非参数化模型假设模型参数的维度不固定或者说无穷大，随着训练数据量的增加而不断增大。

本书介绍的感知机、朴素贝叶斯、逻辑斯谛回归、$k$ 均值、高斯混合模型、潜在语义分析、概率潜在语义分析、潜在狄利克雷分配是参数化模型，决策树、支持向量机、AdaBoost、$k$ 近邻是非参数化模型。

参数化模型适合问题简单的情况，现实中问题往往比较复杂，非参数化模型更加有效。

### 1.2.3 按算法分类

机器学习根据算法，可以分为在线学习（online learning）与批量学习（batch learning）。在线学习是指每次接受一个样本，进行预测，之后学习模型，并不断重复该操作的机器学习。与之对应，批量学习一次接受所有数据，学习模型，之后进行预测。有些实际应用的场景要求学习必须是在线的。比如，数据依次达到无法存储，系统需要及时做出处理；数据规模很大，不可能一次处理所有数据；数据的模式随时间动态变化，需要算法快速适应新的模式（不满足独立同分布假设）。

在线学习可以是监督学习，也可以是无监督学习，强化学习本身就拥有在线学习的特点。以下只考虑在线的监督学习。

学习和预测在一个系统，每次接受一个输入 $x_t$，用已有模型给出预测 $\hat{f}(x_t)$，之后得到相应的反馈，即该输入对应的输出 $y_t$；系统用损失函数计算两者的差异，更新模型，并不断重复以上操作，如图 1.5 所示。

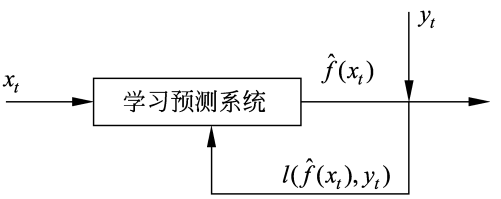


图 1.5 在线学习

利用随机梯度下降的感知机学习算法就是在线学习算法。

在线学习通常比批量学习更难，很难学到预测精确率更高的模型，因为每次模型更新中，可利用的数据有限。

### 1.2.4 按技巧分类

机器学习方法可以根据其使用的技巧进行分类。

1. 贝叶斯学习

贝叶斯学习（Bayesian learning）又称为贝叶斯推理（Bayesian inference），是统计学、机器学习中重要的方法。其主要想法是：在概率模型的学习和推理中，利用贝叶斯定理，计算在给定数据条件下模型的条件概率，即后验概率，并应用这个原理进行模型的估计，以及对数据的预测。将模型、未观测要素及其参数用变量表示，使用模型的先验分布是贝叶斯学习的特点。贝叶斯学习中也使用基本概率公式（图 1.4）。

本书介绍的朴素贝叶斯、潜在狄利克雷分配的学习属于贝叶斯学习。

假设随机变量 $D$ 表示数据，随机变量 $\theta$ 表示模型参数。根据贝叶斯定理，可以用以下公式计算后验概率 $P(\theta|D)$：

$$P(\theta|D) = \frac{P(\theta)P(D|\theta)}{P(D)} \tag{1.3}$$

其中，$P(\theta)$ 是先验概率，$P(D|\theta)$ 是似然函数。

模型估计时，估计整个后验概率分布 $P(\theta|D)$。如果需要给出一个模型，通常取后验概率最大的模型。

预测时，计算数据对后验概率分布的期望值：

$$P(x|D) = \int P(x|\theta, D)P(\theta|D)\mathrm{d}\theta \tag{1.4}$$

这里 $x$ 是新样本。

贝叶斯估计与极大似然估计在思想上有很大的不同，代表着统计学中贝叶斯学派和频率学派对统计的不同认识。其实，可以简单地把两者联系起来，假设先验分布是均匀分布，取后验概率最大，就能从贝叶斯估计得到极大似然估计。图 1.6 对贝叶斯估计和极大似然估计进行比较。

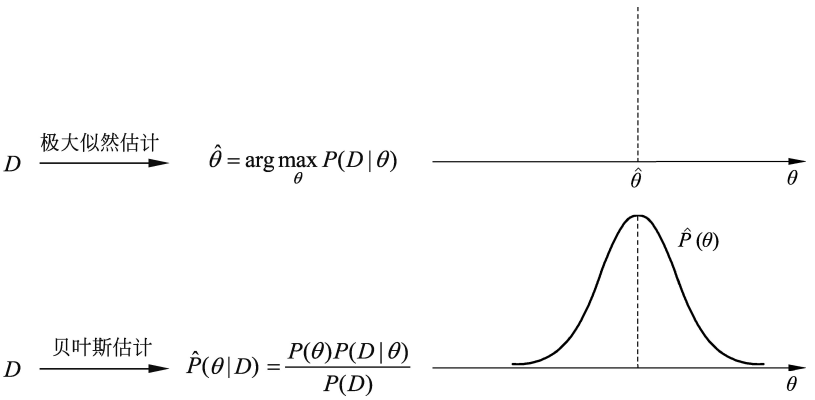


图 1.6 贝叶斯估计与极大似然估计

2. 核方法

核方法（kernel method）是使用核函数表示和学习非线性模型的一种机器学习方法，可以用于监督学习和无监督学习。有一些线性模型的学习方法基于相似度计算，更具体地，向量内积计算。核方法可以把它们扩展到非线性模型的学习，使其应用范围更广泛。

本书介绍的核函数支持向量机，以及核 PCA、核 $k$ 均值属于核方法。

把线性模型扩展到非线性模型，直接的做法是显式地定义从输入空间（低维空间）到特征空间（高维空间）的映射，在特征空间中进行内积计算。比如支持向量机，把输入空间的线性不可分问题转化为特征空间的线性可分问题，如图 1.7 所示。核方法的技巧在于不显式地定义这个映射，而是直接定义核函数，即映射之后在特征空间的内积。这样可以简化计算，达到同样的效果。

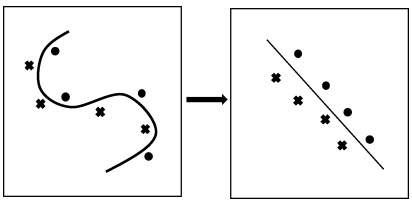

图 1.7 输入空间到特征空间的映射

假设 $x_1$ 和 $x_2$ 是输入空间的任意两个实例（向量），其内积是 $\langle x_1, x_2 \rangle$。假设从输入空间到特征空间的映射是 $\varphi$，于是 $x_1$ 和 $x_2$ 在特征空间的映像是 $\varphi(x_1)$ 和 $\varphi(x_2)$，其内积是 $\langle \varphi(x_1), \varphi(x_2) \rangle$。核方法直接在输入空间中定义核函数 $K(x_1, x_2)$，使其满足 $K(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle$。表示定理给出核函数技巧成立的充要条件。

## 1.3 机器学习方法三要素

机器学习方法都是由模型、策略和算法构成的，即机器学习方法由三要素构成，可以简单地表示为

$$\text{方法} = \text{模型} + \text{策略} + \text{算法}$$

下面论述监督学习中的机器学习三要素。非监督学习也同样拥有这三要素。可以说构建一种机器学习方法就是确定具体的机器学习三要素。

### 1.3.1 模型

机器学习首要考虑的问题是学习什么样的模型。在监督学习过程中，模型就是所要学习的条件概率分布或决策函数。模型的假设空间（hypothesis space）包含所有可能的条件概率分布或决策函数。例如，假设决策函数是输入变量的线性函数，那么模型的假设空间就是所有这些线性函数构成的函数集合。假设空间中的模型一般有无穷多个。

假设空间用 $\mathcal{F}$ 表示，可以定义为决策函数的集合：

$$\mathcal{F} = \{f|Y = f(X)\} \tag{1.5}$$

其中，$X$ 和 $Y$ 是定义在输入空间 $\mathcal{X}$ 和输出空间 $\mathcal{Y}$ 上的变量。这时 $\mathcal{F}$ 通常是由一个参数向量决定的函数族：

$$\mathcal{F} = \{f|Y = f_\theta(X), \theta \in \boldsymbol{R}^n\} \tag{1.6}$$

参数向量 $\theta$ 取值于 $n$ 维欧氏空间 $\boldsymbol{R}^n$，称为参数空间（parameter space）。

假设空间也可以定义为条件概率的集合：

$$\mathcal{F} = \{P|P(Y|X)\} \tag{1.7}$$

其中，$X$ 和 $Y$ 是定义在输入空间 $\mathcal{X}$ 和输出空间 $\mathcal{Y}$ 上的随机变量。这时 $\mathcal{F}$ 通常是由一个参数向量决定的条件概率分布族：

$$\mathcal{F} = \{P|P_\theta(Y|X), \theta \in \boldsymbol{R}^n\} \tag{1.8}$$

参数向量 $\theta$ 取值于 $n$ 维欧氏空间 $\boldsymbol{R}^n$，也称为参数空间。

本书中称由决策函数表示的模型为非概率模型，由条件概率表示的模型为概率模型。为了简便起见，当论及模型时，有时只用其中一种模型。

### 1.3.2 策略

有了模型的假设空间，机器学习接着需要考虑的是按照什么样的准则学习或选择最优的模型。机器学习的目标在于从假设空间中选取最优模型。

首先引入损失函数与风险函数的概念。损失函数度量模型一次预测的好坏，风险函数度量平均意义下模型预测的好坏。

1. 损失函数和风险函数

监督学习问题是在假设空间 $\mathcal{F}$ 中选取模型 $f$ 作为决策函数，对于给定的输入 $X$，由 $f(X)$ 给出相应的输出 $Y$，这个输出的预测值 $f(X)$ 与真实值 $Y$ 可能一致也可能不一致，用一个损失函数（loss function）或代价函数（cost function）来度量预测错误的程度。损失函数是 $f(X)$ 和 $Y$ 的非负实值函数，记作 $L(Y, f(X))$。

机器学习常用的损失函数有以下几种：

（1）0-1 损失函数（0-1 loss function）

$$L(Y, f(X)) = \begin{cases} 1, & Y \neq f(X) \\ 0, & Y = f(X) \end{cases} \tag{1.9}$$

（2）平方损失函数（quadratic loss function）

$$L(Y, f(X)) = (Y - f(X))^2 \tag{1.10}$$

（3）绝对损失函数（absolute loss function）

$$L(Y, f(X)) = |Y - f(X)| \tag{1.11}$$

（4）对数损失函数（logarithmic loss function）或对数似然损失函数（log-likelihood loss function）

$$L(Y, P(Y|X)) = -\log P(Y|X) \tag{1.12}$$

损失函数值越小，模型就越好。由于模型的输入、输出 $(X, Y)$ 是随机变量，遵循联合分布 $P(X, Y)$，所以损失函数的期望是

$$\begin{aligned} R_{\exp}(f) &= E_P[L(Y, f(X))] \\ &= \int_{\mathcal{X}\times\mathcal{Y}} L(y, f(x))P(x, y)\mathrm{d}x\mathrm{d}y \end{aligned} \tag{1.13}$$

这是理论上模型 $f(X)$ 关于联合分布 $P(X, Y)$ 的平均意义下的损失，称为风险函数（risk function）或期望损失（expected loss）。

学习的目标就是选择期望风险最小的模型。由于联合分布 $P(X, Y)$ 是未知的，$R_{\exp}(f)$ 不能直接计算。实际上，如果知道联合分布 $P(X, Y)$，可以从联合分布直接求出条件概率分布 $P(Y|X)$，也就不需要学习了。正因为不知道联合概率分布，所以才需要进行学习。这样一来，一方面根据期望风险最小学习模型要用到联合分布，另一方面联合分布又是未知的，所以监督学习就成为一个病态问题（ill-formed problem）。

给定一个训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

模型 $f(X)$ 关于训练数据集的平均损失称为经验风险（empirical risk）或经验损失（empirical loss），记作 $R_{\mathrm{emp}}$：

$$R_{\mathrm{emp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1.14}$$

期望风险 $R_{\exp}(f)$ 是模型关于联合分布的期望损失，经验风险 $R_{\mathrm{emp}}(f)$ 是模型关于训练样本集的平均损失。根据大数定律，当样本容量 $N$ 趋于无穷时，经验风险 $R_{\mathrm{emp}}(f)$ 趋于期望风险 $R_{\exp}(f)$。所以一个很自然的想法是用经验风险估计期望风险。但是，由于现实中训练样本数目有限，甚至很小，所以用经验风险估计期望风险常常并不理想，要对经验风险进行一定的矫正。这就关系到监督学习的两个基本策略：经验风险最小化和结构风险最小化。

2. 经验风险最小化与结构风险最小化

在假设空间、损失函数以及训练数据集确定的情况下，经验风险函数式 (1.14) 就可以确定。经验风险最小化（empirical risk minimization，ERM）的策略认为，经验风险最小的模型是最优的模型。根据这一策略，按照经验风险最小化求最优模型就是求解最优化问题：

$$\min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1.15}$$

其中，$\mathcal{F}$ 是假设空间。

当样本容量足够大时，经验风险最小化能保证有很好的学习效果，在现实中被广泛采用。比如，极大似然估计（maximum likelihood estimation） 就是经验风险最小化的一个例子。当模型是条件概率分布、损失函数是对数损失函数时，经验风险最小化就等价于极大似然估计。但是，当样本容量很小时，经验风险最小化学习的效果就未必很好，会产生“过拟合”（over-fitting）现象。

结构风险最小化（structural risk minimization，SRM）是为了防止过拟合而提出来的策略。结构风险最小化等价于正则化（regularization）。结构风险在经验风险上加上表示模型复杂度的正则化项（regularizer）或罚项（penalty term）。在假设空间、损失函数以及训练数据集确定的情况下，结构风险的定义是

$$R_{\text{srm}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.16}$$

其中，$J(f)$ 为模型的复杂度，是定义在假设空间 $\mathcal{F}$ 上的泛函。模型 $f$ 越复杂，复杂度 $J(f)$ 就越大；反之，模型 $f$ 越简单，复杂度 $J(f)$ 就越小。也就是说，复杂度表示了对复杂模型的惩罚。$\lambda \geqslant 0$ 是系数，用以权衡经验风险和模型复杂度。结构风险小需要经验风险与模型复杂度同时小。结构风险小的模型往往对训练数据以及未知的测试数据都有较好的预测。

比如，贝叶斯估计中的最大后验概率估计（maximum posterior probability estimation，MAP）就是结构风险最小化的一个例子。当模型是条件概率分布、损失函数是对数损失函数、模型复杂度由模型的先验概率表示时，结构风险最小化就等价于最大后验概率估计。

结构风险最小化的策略认为结构风险最小的模型是最优的模型，所以求最优模型就是求解最优化问题：

$$\min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.17}$$

这样，监督学习问题就变成了经验风险或结构风险函数的最优化问题 (1.15) 和 (1.17)。这时经验或结构风险函数是最优化的目标函数。

### 1.3.3 算法

算法是指学习模型的具体计算方法。机器学习基于训练数据集，根据学习策略，从假设空间中选择最优模型，最后需要考虑用什么样的计算方法求解最优模型。这时，机器学习问题归结为最优化问题，机器学习的算法成为求解最优化问题的算法。如果最优化问题有显式的解析解，这个最优化问题就比较简单。但通常解析解不存在，这就需要用数值计算的方法求解。如何保证找到全局最优解，并使求解的过程非常高效，就成为一个重要问题。机器学习可以利用已有的最优化算法，有时也需要开发独自的最优化算法。

机器学习方法之间的不同主要来自其模型、策略、算法的不同。确定了模型、策略、算法，机器学习的方法也就确定了。这就是将其称为机器学习方法三要素的原因。以下介绍监督学习的几个重要概念。

## 1.4 模型评估与模型选择

### 1.4.1 训练误差与测试误差

机器学习的目的是使学到的模型不仅对已知数据而且对未知数据都能有很好的预测能力。不同的学习方法会给出不同的模型。当损失函数给定时，基于损失函数的模型的训练误差（training error）和模型的测试误差（test error）就自然成为学习方法评估的标准。注意，机器学习方法具体采用的损失函数未必是评估时使用的损失函数。当然，让两者一致是比较理想的。

假设学习到的模型是 $Y = \hat{f}(X)$，训练误差是模型 $Y = \hat{f}(X)$ 关于训练数据集的平均损失：

$$R_{\text{emp}}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i)) \tag{1.18}$$

其中，$N$ 是训练样本容量。

测试误差是模型 $Y = \hat{f}(X)$ 关于测试数据集的平均损失：

$$e_{\text{test}} = \frac{1}{N'}\sum_{i=1}^{N'} L(y_i, \hat{f}(x_i)) \tag{1.19}$$

其中，$N'$ 是测试样本容量。

例如，当损失函数是 0-1 损失时，测试误差就变成了常见的测试数据集上的误差率（error rate）：

$$e_{\text{test}} = \frac{1}{N'}\sum_{i=1}^{N'} I(y_i \neq \hat{f}(x_i)) \tag{1.20}$$

这里 $I$ 是指示函数（indicator function），即 $y \neq \hat{f}(x)$ 时为 1，否则为 0。

相应地，常见的测试数据集上的精确率（accuracy）为

$$r_{\text{test}} = \frac{1}{N'}\sum_{i=1}^{N'} I(y_i = \hat{f}(x_i)) \tag{1.21}$$

显然，

$$r_{\text{test}} + e_{\text{test}} = 1$$

训练误差的大小对判断给定的问题是不是一个容易学习的问题是有意义的，但本质上不重要。测试误差反映了学习方法对未知的测试数据集的预测能力，是学习中的重要概念。显然，给定两种学习方法，测试误差小的方法具有更好的预测能力，是更有效的方法。通常将学习方法对未知数据的预测能力称为泛化能力（generalization ability），这个问题将在 1.6 节继续论述。

### 1.4.2 过拟合与模型选择

当假设空间含有不同复杂度（例如，不同的参数个数）的模型时，就要面临模型选择（model selection）的问题。我们希望选择或学习一个合适的模型。如果在假设空间中存在“真”模型，那么所选择的模型应该逼近真模型。具体地，所选择的模型要与真模型的参数个数相同，所选择的模型的参数向量与真模型的参数向量相近。

如果一味追求提高对训练数据的预测能力，所选模型的复杂度则往往会比真模型更高。这种现象称为过拟合（over-fitting）。过拟合是指学习时选择的模型所包含的参数过多，以至出现这一模型对已知数据预测得很好，但对未知数据预测得很差的现象。可以说模型选择旨在避免过拟合并提高模型的预测能力。

下面，以多项式函数拟合问题为例，说明过拟合与模型选择。这是一个回归问题。

**例 1.1** 假设给定一个训练数据集[①]：

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \boldsymbol{R}$ 是输入 $x$ 的观测值，$y_i \in \boldsymbol{R}$ 是相应的输出 $y$ 的观测值，$i = 1, 2, \cdots, N$。多项式函数拟合的任务是假设给定数据由 $M$ 次多项式函数生成，选择最有可能产生这些数据的 $M$ 次多项式函数，即在 $M$ 次多项式函数中选择一个对已知数据以及未知数据都有很好预测能力的函数。

假设给定如图 1.8 所示的 10 个数据点，用 0~9 次多项式函数对数据进行拟合。图中画出了需要用多项式函数曲线拟合的数据。

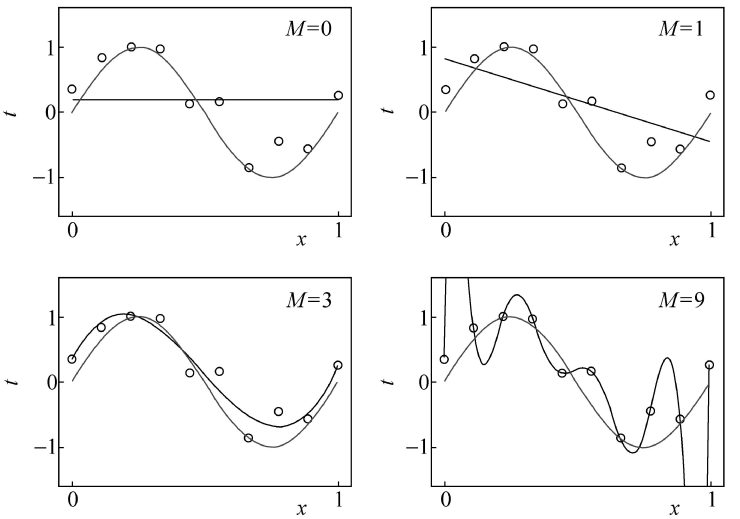


图 1.8 $M$ 次多项式函数拟合问题的例子

设 $M$ 次多项式为

$$f_M(x, w) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{1.22}$$

① 本例来自参考文献 [2]。

其中 $x$ 是单变量输入，$w_0, w_1, \cdots, w_M$ 是 $M+1$ 个参数。

解决这一问题的方法可以是这样的：首先确定模型的复杂度，即确定多项式的次数；然后在给定的模型复杂度下，按照经验风险最小化的策略求解参数，即多项式的系数。具体地，求以下经验风险最小化：

$$L(w) = \frac{1}{2}\sum_{i=1}^{N}(f(x_i, w) - y_i)^2 \tag{1.23}$$

这时，损失函数为平方损失，系数 $\frac{1}{2}$ 是为了计算方便。

这是一个简单的最优化问题。将模型与训练数据代入式 (1.23) 中，有

$$L(w) = \frac{1}{2}\sum_{i=1}^{N}\left(\sum_{j=0}^{M} w_j x_i^j - y_i\right)^2$$

这一问题可用最小二乘法求得拟合多项式系数的唯一解，记作 $w_0^*$, $w_1^*$, $\cdots$, $w_M^*$。求解过程这里不予叙述，读者可参阅有关材料。

图 1.8 给出了 $M=0$，$M=1$，$M=3$ 及 $M=9$ 时多项式函数拟合的情况。如果 $M=0$，多项式曲线是一个常数，数据拟合效果很差。如果 $M=1$，多项式曲线是一条直线，数据拟合效果也很差。相反，如果 $M=9$，多项式曲线通过每个数据点，训练误差为 0。从对给定训练数据拟合的角度来说，效果是最好的。但是，因为训练数据本身存在噪声，这种拟合曲线对未知数据的预测能力往往并不是最好的，在实际学习中并不可取。这时过拟合现象就会发生。这就是说，模型选择时，不仅要考虑对已知数据的预测能力，而且还要考虑对未知数据的预测能力。当 $M=3$ 时，多项式曲线对训练数据拟合效果足够好，模型也比较简单，是一个较好的选择。

在多项式函数拟合中可以看到，随着多项式次数（模型复杂度）的增加，训练误差会减小，直至趋向于 0，但是测试误差却不如此，它会随着多项式次数（模型复杂度）的增加先减小而后增大。而最终的目的是使测试误差达到最小。这样，在多项式函数拟合中，就要选择合适的多项式次数，以达到这一目的。这一结论对一般的模型选择也是成立的。

图 1.9 描述了训练误差和测试误差与模型复杂度之间的关系。当模型复杂度增大时，训练误差会逐渐减小并趋向于 0；而测试误差会先减小，达到最小值后又增大。当选择的模型复

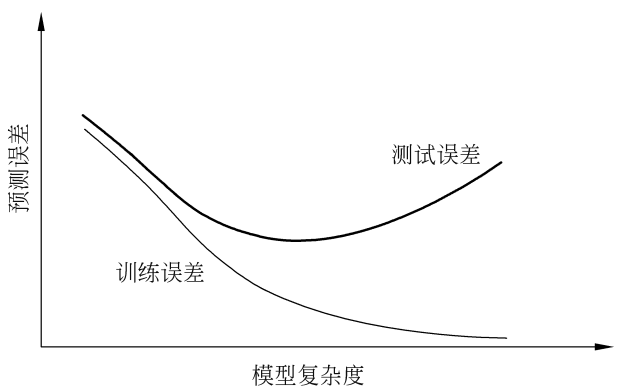


图 1.9　训练误差和测试误差与模型复杂度的关系

杂度过大时，过拟合现象就会发生。这样，在学习时就要防止过拟合，进行最优的模型选择，即选择复杂度适当的模型，以达到使测试误差最小的学习目的。下面介绍两种常用的模型选择方法：正则化与交叉验证。

## 1.5 正则化与交叉验证

### 1.5.1 正则化

模型选择的典型方法是正则化（regularization）。正则化是结构风险最小化策略的实现，是在经验风险上加一个正则化项（regularizer）或罚项（penalty term）。正则化项一般是模型复杂度的单调递增函数，模型越复杂，正则化值就越大。比如，正则化项可以是模型参数向量的范数。

正则化一般具有如下形式：

$$\min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.24}$$

其中，第 1 项是经验风险，第 2 项是正则化项，$\lambda \geqslant 0$ 为调整两者之间关系的系数。

正则化项可以取不同的形式。例如，在回归问题中，损失函数是平方损失，正则化项可以是参数向量的 $L_2$ 范数：

$$L(w) = \frac{1}{N}\sum_{i=1}^{N} (f(x_i; w) - y_i)^2 + \frac{\lambda}{2}\|w\|^2 \tag{1.25}$$

这里，$\|w\|$ 表示参数向量 $w$ 的 $L_2$ 范数。

正则化项也可以是参数向量的 $L_1$ 范数：

$$L(w) = \frac{1}{N}\sum_{i=1}^{N} (f(x_i; w) - y_i)^2 + \lambda\|w\|_1 \tag{1.26}$$

这里，$\|w\|_1$ 表示参数向量 $w$ 的 $L_1$ 范数。

第 1 项的经验风险较小的模型可能较复杂（有多个非零参数），这时第 2 项的模型复杂度会较大。正则化的作用是选择经验风险与模型复杂度同时较小的模型。

正则化符合奥卡姆剃刀（Occam's razor）原理。奥卡姆剃刀原理应用于模型选择时变为以下想法：在所有可能选择的模型中，能够很好地解释已知数据并且十分简单才是最好的模型，也就是应该选择的模型。从贝叶斯估计的角度来看，正则化项对应于模型的先验概率。可以假设复杂的模型有较小的先验概率，简单的模型有较大的先验概率。

### 1.5.2 交叉验证

另一种常用的模型选择方法是交叉验证（cross validation）。

如果给定的样本数据充足，进行模型选择的一种简单方法是随机地将数据集切分成三部分，分别为训练集（training set）、验证集（validation set）和测试集（test set）。训练集用来训练模型，验证集用于模型的选择，而测试集用于最终对学习方法的评估。在学习到的不同复杂度的模型中，选择对验证集有最小预测误差的模型。由于验证集有足够多的数据，用它对模型进行选择也是有效的。

但是，在许多实际应用中数据是不充足的。为了选择好的模型，可以采用交叉验证方法。交叉验证的基本想法是重复地使用数据，把给定的数据进行切分，将切分的数据集组合为训练集与测试集，在此基础上反复地进行训练、测试以及模型选择。

1. 简单交叉验证

简单交叉验证方法是：首先随机地将已给数据分为两部分，一部分作为训练集，另一部分作为测试集（例如，70%的数据为训练集，30%的数据为测试集）；然后用训练集在各种条件下（例如，不同的参数个数）训练模型，从而得到不同的模型；在测试集上评价各个模型的测试误差，选出测试误差最小的模型。

2. $S$ 折交叉验证

应用最多的是 $S$ 折交叉验证（$S$-fold cross validation），方法如下：首先随机地将已给数据切分为 $S$ 个互不相交、大小相同的子集；然后利用 $S-1$ 个子集的数据训练模型，利用余下的子集测试模型；将这一过程对可能的 $S$ 种选择重复进行；最后选出 $S$ 次评测中平均测试误差最小的模型。

3. 留一交叉验证

$S$ 折交叉验证的特殊情形是 $S=N$，称为留一交叉验证（leave-one-out cross validation），往往在数据缺乏的情况下使用。这里，$N$ 是给定数据集的容量。

## 1.6 泛化能力

### 1.6.1 泛化误差

学习方法的泛化能力（generalization ability）是指由该方法学习到的模型对未知数据的预测能力，是学习方法本质上重要的性质。现实中采用最多的办法是通过测试误差来评价学习方法的泛化能力。但这种评价是依赖于测试数据集的。因为测试数据集是有限的，很有可能由此得到的评价结果是不可靠的。机器学习理论试图从理论上对学习方法的泛化能力进行分析。

首先给出泛化误差的定义。如果学到的模型是 $\hat{f}$，那么用这个模型对未知数据预测的误差即为泛化误差（generalization error）：

$$\begin{aligned} R_{\exp}(\hat{f}) &= E_P[L(Y,\hat{f}(X))] \\ &= \int_{\mathcal{X}\times\mathcal{Y}} L(y,\hat{f}(x))P(x,y)\mathrm{d}x\mathrm{d}y \end{aligned} \tag{1.27}$$

泛化误差反映了学习方法的泛化能力，如果一种方法学习的模型比另一种方法学习的模型具有更小的泛化误差，那么这种方法就更有效。事实上，泛化误差就是所学习到的模型的期望风险。

### 1.6.2 泛化误差上界

学习方法的泛化能力分析往往是通过研究泛化误差的概率上界进行的，简称为泛化误差上界（generalization error bound）。具体来说，就是通过比较两种学习方法的泛化误差上界的大小来比较它们的优劣。泛化误差上界通常具有以下性质：它是样本容量的函数，当样本容量增加时，泛化误差上界趋于 0；它是假设空间容量（capacity）的函数，假设空间容量越大，模型就越难学，泛化误差上界就越大。

下面给出一个简单的泛化误差上界的例子：二类分类问题的泛化误差上界。

考虑二类分类问题。已知训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，$N$ 是样本容量，$T$ 是从联合概率分布 $P(X,Y)$ 独立同分布产生的，$X\in\boldsymbol{R}^n$，$Y\in\{-1,+1\}$。假设空间是函数的有限集合 $\mathcal{F}=\{f_1,f_2,\cdots,f_d\}$，$d$ 是函数个数。设 $f$ 是从 $\mathcal{F}$ 中选取的函数。损失函数是 0-1 损失。关于 $f$ 的期望风险和经验风险分别是

$$R(f)=E[L(Y,f(X))] \tag{1.28}$$

$$\hat{R}(f)=\frac{1}{N}\sum_{i=1}^{N}L(y_i,f(x_i)) \tag{1.29}$$

经验风险最小化函数是

$$f_N=\arg\min_{f\in\mathcal{F}}\hat{R}(f) \tag{1.30}$$

$f_N$ 依赖训练数据集的样本容量 $N$。人们更关心的是 $f_N$ 的泛化能力

$$R(f_N)=E[L(Y,f_N(X))] \tag{1.31}$$

下面讨论从有限集合 $\mathcal{F}=\{f_1,f_2,\cdots,f_d\}$ 中任意选出的函数 $f$ 的泛化误差上界。

**定理 1.1（泛化误差上界）** 对二类分类问题，当假设空间是有限个函数的集合 $\mathcal{F}=\{f_1,f_2,\cdots,f_d\}$ 时，对任意一个函数 $f\in\mathcal{F}$，至少以概率 $1-\delta$，$0<\delta<1$，以下不等式成立：

$$R(f)\leqslant\hat{R}(f)+\varepsilon(d,N,\delta) \tag{1.32}$$

其中，

$$\varepsilon(d,N,\delta)=\sqrt{\frac{1}{2N}\left(\log d+\log\frac{1}{\delta}\right)} \tag{1.33}$$

不等式 (1.32) 左端 $R(f)$ 是泛化误差，右端即为泛化误差上界。在泛化误差上界中，第 1 项是训练误差，训练误差越小，泛化误差也越小。第 2 项 $\varepsilon(d,N,\delta)$ 是 $N$ 的单调递减函数，当 $N$ 趋于无穷时趋于 0；同时它也是 $\sqrt{\log d}$ 阶的函数，假设空间 $\mathcal{F}$ 包含的函数越多，其值越大。

**证明** 在证明中要用到 Hoeffding 不等式，先叙述如下。

设 $X_1, X_2, \cdots, X_N$ 是独立随机变量，且 $X_i \in [a_i, b_i]$，$i = 1, 2, \cdots, N$；$\bar{X}$ 是 $X_1, X_2, \cdots, X_N$ 的经验均值，即 $\bar{X} = \dfrac{1}{N}\sum\limits_{i=1}^{N} X_i$，则对任意 $t > 0$，以下不等式成立：

$$P(\bar{X} - E(\bar{X}) \geqslant t) \leqslant \exp\left[-\frac{2N^2t^2}{\sum\limits_{i=1}^{N}(b_i - a_i)^2}\right] \tag{1.34}$$

$$P(E(\bar{X}) - \bar{X} \geqslant t) \leqslant \exp\left[-\frac{2N^2t^2}{\sum\limits_{i=1}^{N}(b_i - a_i)^2}\right] \tag{1.35}$$

Hoeffding 不等式的证明省略，这里用来推导泛化误差上界。

对任意函数 $f \in \mathcal{F}$，$\hat{R}(f)$ 是 $N$ 个独立的随机变量 $L(Y, f(X))$ 的样本均值，$R(f)$ 是随机变量 $L(Y, f(X))$ 的期望值。如果损失函数取值于区间 $[0, 1]$，即对所有 $i$，$[a_i, b_i] = [0, 1]$，那么由 Hoeffding 不等式 (1.35) 不难得知，对 $\varepsilon > 0$，以下不等式成立：

$$P(R(f) - \hat{R}(f) \geqslant \varepsilon) \leqslant \exp(-2N\varepsilon^2) \tag{1.36}$$

由于 $\mathcal{F} = \{f_1, f_2, \cdots, f_d\}$ 是一个有限集合，故

$$\begin{aligned} P(\exists f \in \mathcal{F} : R(f) - \hat{R}(f) \geqslant \varepsilon) &= P\Big(\bigcup_{f\in\mathcal{F}} \{R(f) - \hat{R}(f) \geqslant \varepsilon\}\Big) \\ &\leqslant \sum_{f\in\mathcal{F}} P(R(f) - \hat{R}(f) \geqslant \varepsilon) \\ &\leqslant d\exp(-2N\varepsilon^2) \end{aligned}$$

或者等价地，对任意 $f \in \mathcal{F}$，有

$$P(R(f) - \hat{R}(f) < \varepsilon) \geqslant 1 - d\exp(-2N\varepsilon^2) \tag{1.37}$$

令

$$\delta = d\exp(-2N\varepsilon^2) \tag{1.38}$$

则

$$P(R(f) < \hat{R}(f) + \varepsilon) \geqslant 1 - \delta$$

即至少以概率 $1 - \delta$ 有 $R(f) < \hat{R}(f) + \varepsilon$，其中 $\varepsilon$ 由式 (1.38) 得到，即为式 (1.33)。 ■

从泛化误差上界可知：

$$R(f_N) \leqslant \hat{R}(f_N) + \varepsilon(d, N, \delta) \tag{1.39}$$

其中，$\varepsilon(d, N, \delta)$ 由式 (1.33) 定义，$f_N$ 由式 (1.30) 定义。

以上讨论的只是假设空间包含有限个函数情况下的泛化误差上界，对一般的假设空间要找到泛化误差上界就没有这么简单，这里不作介绍。

## 1.7 生成模型与判别模型

监督学习的任务就是学习一个模型，应用这一模型，对给定的输入预测相应的输出。这个模型的一般形式为决策函数：

$$Y = f(X)$$

或者条件概率分布：

$$P(Y|X)$$

监督学习方法又可以分为生成方法（generative approach）和判别方法（discrimina-tive approach）。所学到的模型分别称为生成模型（generative model）和判别模型（discriminative model）。

生成方法原理上由数据学习联合概率分布 $P(X,Y)$，然后求出条件概率分布 $P(Y|X)$ 作为预测的模型，即生成模型：

$$P(Y|X) = \frac{P(X,Y)}{P(X)} \tag{1.40}$$

这样的方法之所以称为生成方法，是因为模型表示了给定输入 $X$ 产生输出 $Y$ 的生成关系。典型的生成模型有朴素贝叶斯法和隐马尔可夫模型，将在后面章节进行相关讲述。

判别方法由数据直接学习决策函数 $f(X)$ 或者条件概率分布 $P(Y|X)$ 作为预测的模型，即判别模型。判别方法关心的是对给定的输入 $X$，应该预测什么样的输出 $Y$。典型的判别模型包括：$k$ 近邻法、感知机、逻辑斯谛回归模型、最大熵模型、支持向量机、提升方法和条件随机场等，将在后面章节讲述。

在监督学习中，生成方法和判别方法各有优缺点，适合于不同条件下的学习问题。

生成方法的特点：生成方法可以还原出联合概率分布 $P(X,Y)$，而判别方法不能；生成方法的学习收敛速度更快，即当样本容量增加的时候，学到的模型可以更快地收敛于真实模型；当存在隐变量时，仍可以用生成方法学习，此时判别方法就不能用。

判别方法的特点：判别方法直接学习的是条件概率分布 $P(Y|X)$ 或决策函数 $f(X)$，直接面对预测，往往学习的精确率更高；由于直接学习 $P(Y|X)$ 或 $f(X)$，可以对数据进行各种程度上的抽象、定义特征并使用特征，因此可以简化学习问题。

## 1.8 监督学习应用

监督学习的应用主要在三个方面：分类问题、标注问题和回归问题。

### 1.8.1 分类问题

分类是监督学习的一个核心问题。在监督学习中，当输出变量 $Y$ 取有限个离散值时，预

测问题便成为分类问题。这时，输入变量 $X$ 可以是离散的，也可以是连续的。监督学习从数据中学习一个分类模型或分类决策函数，称为分类器（classifier）。分类器对新的输入进行输出的预测，称为分类（classification）。可能的输出称为类别（class）。分类的类别为多个时，称为多类分类问题。本书主要讨论二类分类问题。

分类问题包括学习和分类两个过程。在学习过程中，根据已知的训练数据集利用有效的学习方法学习一个分类器；在分类过程中，利用学习的分类器对新的输入实例进行分类。分类问题可用图 1.10 描述。图中 $(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)$ 是训练数据集，学习系统由训练数据学习一个分类器 $P(Y|X)$ 或 $Y = f(X)$；分类系统通过学到的分类器 $P(Y|X)$ 或 $Y = f(X)$ 对新的输入实例 $x_{N+1}$ 进行分类，即预测其输出的类标记 $y_{N+1}$。

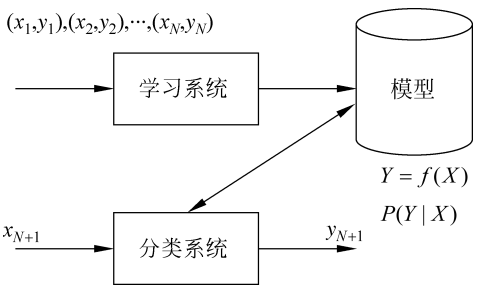


图 1.10 分类问题

评价分类器性能的指标一般是分类精确率（accuracy），其定义是：对于给定的测试数据集，分类器正确分类的样本数与总样本数之比。也就是损失函数是 0-1 损失时测试数据集上的精确率（见式 (1.21)）。

对于二类分类问题，常用的评价指标是准确率（precision）与召回率（recall）。通常以关注的类为正类，其他类为负类，分类器在测试数据集上的预测或正确或不正确，4 种情况出现的总数分别记作：

TP—— 将正类预测为正类数；

FN—— 将正类预测为负类数；

FP—— 将负类预测为正类数；

TN—— 将负类预测为负类数。

准确率定义为

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1.41}$$

召回率定义为

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{1.42}$$

此外，还有 $F_1$，是准确率和召回率的调和均值，即

$$\frac{2}{F_1} = \frac{1}{P} + \frac{1}{R} \tag{1.43}$$

$$F_1 = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{1.44}$$

准确率和召回率都高时，$F_1$ 值也会高。

许多机器学习方法可以用于分类，包括 $k$ 近邻法、感知机、朴素贝叶斯法、决策树、决策列表、逻辑斯谛回归模型、支持向量机、提升方法、贝叶斯网络、神经网络、Winnow 等。本书将讲述其中一些主要方法。

分类在于根据其特性将数据“分门别类”，所以在许多领域都有广泛的应用。例如，在银行业务中，可以构建一个客户分类模型，对客户按照贷款风险的大小进行分类；在网络安全领域，可以利用日志数据的分类对非法入侵进行检测；在图像处理中，分类可以用来检测图像中是否有人脸出现；在手写识别中，分类可以用于识别手写的数字；在互联网搜索中，网页的分类可以帮助网页的抓取、索引与排序。

举一个分类应用的例子 —— 文本分类 (text classification)。这里的文本可以是新闻报道、网页、电子邮件、学术论文等。类别往往是关于文本内容的，如政治、经济、体育等；也有关于文本特点的，如正面意见、反面意见；还可以根据应用确定，如垃圾邮件、非垃圾邮件等。文本分类是根据文本的特征将其划分到已有的类中。输入是文本的特征向量，输出是文本的类别。通常把文本中的单词定义为特征，每个单词对应一个特征。单词的特征可以是二值的，如果单词在文本中出现则取值是 1，否则是 0；也可以是多值的，表示单词在文本中出现的频率。直观地，如果“股票”“银行”“货币”这些词出现很多，这个文本可能属于经济类；如果“网球”“比赛”“运动员”这些词频繁出现，这个文本可能属于体育类。

### 1.8.2 标注问题

标注（tagging）也是一个监督学习问题。可以认为标注问题是分类问题的一个推广，标注问题又是更复杂的结构预测（structure prediction）问题的简单形式。标注问题的输入是一个观测序列，输出是一个标记序列或状态序列。标注问题的目标在于学习一个模型，使它能够对观测序列给出标记序列作为预测。注意，可能的标记个数是有限的，但其组合所成的标记序列的个数是依序列长度呈指数级增长的。

标注问题分为学习和标注两个过程（如图 1.11 所示）。首先给定一个训练数据集

$$T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$$

这里，$x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$，$i=1,2,\cdots,N$，是输入观测序列；$y_i=(y_i^{(1)},y_i^{(2)},\cdots,y_i^{(n)})^{\mathrm{T}}$ 是相应的输出标记序列；$n$ 是序列的长度，对不同样本可以有不同的值。学习系统基于训练数据集构建一个模型，表示为条件概率分布：

$$P(Y^{(1)},Y^{(2)},\cdots,Y^{(n)}|X^{(1)},X^{(2)},\cdots,X^{(n)})$$

这里，每一个 $X^{(i)}$（$i=1,2,\cdots,n$）取值为所有可能的观测，每一个 $Y^{(i)}$（$i=1,2,\cdots,n$）取值为所有可能的标记，一般 $n\ll N$。标注系统按照学习得到的条件概率分布模型，对新的输入观测序列找到相应的输出标记序列。具体地，对一个观测序列 $x_{N+1}=(x_{N+1}^{(1)},x_{N+1}^{(2)},\cdots,x_{N+1}^{(n)})^{\mathrm{T}}$，找到使条件概率 $P((y_{N+1}^{(1)},y_{N+1}^{(2)},\cdots,y_{N+1}^{(n)})^{\mathrm{T}}|(x_{N+1}^{(1)},x_{N+1}^{(2)},\cdots,x_{N+1}^{(n)})^{\mathrm{T}})$ 最大的标记序列 $y_{N+1}=(y_{N+1}^{(1)},y_{N+1}^{(2)},\cdots,y_{N+1}^{(n)})^{\mathrm{T}}$。

评价标注模型的指标与评价分类模型的指标一样，常用的有标注精确率、准确率和召回率。其定义与分类模型相同。

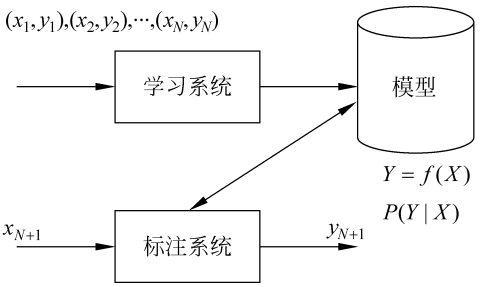


图 1.11 标注问题

标注常用的机器学习方法有隐马尔可夫模型、条件随机场。

标注问题在信息抽取、自然语言处理等领域被广泛应用，是这些领域的基本问题。例如，自然语言处理中的词性标注（part of speech tagging）就是一个典型的标注问题：给定一个由单词组成的句子，对这个句子中的每一个单词进行词性标注，即对一个单词序列预测其对应的词性标记序列。

举一个信息抽取的例子。从英文文章中抽取基本名词短语（base noun phrase）。为此，要对文章进行标注。英文单词是一个观测，英文句子是一个观测序列，标记表示名词短语的“开始”、“结束”或“其他”（分别以 B，E，O 表示），标记序列表示英文句子中基本名词短语的所在位置。信息抽取时，将标记“开始”到标记“结束”的单词作为名词短语。例如，给出以下的观测序列，即英文句子，标注系统产生相应的标记序列，即给出句子中的基本名词短语。

输入：At Microsoft Research, we have an insatiable curiosity and the desire to create new technology that will help define the computing experience.

输出：At/O Microsoft/B Research/E, we/O have/O an/O insatiable/B curiosity/E and/O the/O desire/BE to/O create/O new/B technology/E that/O will/O help/O define/O the/O computing/B experience/E.

### 1.8.3 回归问题

回归（regression）是监督学习的另一个重要问题。回归用于预测输入变量（自变量）和输出变量（因变量）之间的关系，特别是当输入变量的值发生变化时，输出变量的值随之发生的变化。回归模型正是表示从输入变量到输出变量之间映射的函数。回归问题的学习等价于函数拟合：选择一条函数曲线使其很好地拟合已知数据且很好地预测未知数据（参照 1.4.2 节）。

回归问题分为学习和预测两个过程（如图 1.12 所示）。首先给定一个训练数据集：

$$T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$$

这里，$x_i\in\boldsymbol{R}^n$ 是输入，$y\in\boldsymbol{R}$ 是对应的输出，$i=1,2,\cdots,N$。学习系统基于训练数据构建一个模型，即函数 $Y=f(X)$；对新的输入 $x_{N+1}$，预测系统根据学习的模型 $Y=f(X)$ 确定相应的输出 $y_{N+1}$。

回归问题按照输入变量的个数，分为一元回归和多元回归；按照输入变量和输出变量之间关系的类型即模型的类型，分为线性回归和非线性回归。

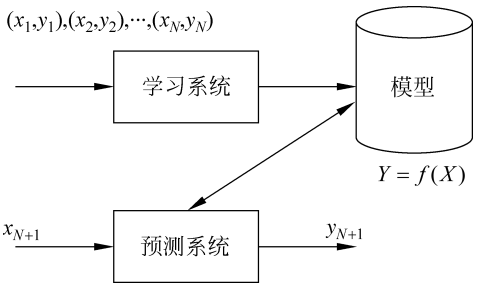


图 1.12 回归问题

回归学习最常用的损失函数是平方损失函数，在此情况下，回归问题可以由著名的最小二乘法（least squares）求解。

许多领域的任务都可以形式化为回归问题，比如，回归可以用于商务领域，作为市场趋势预测、产品质量管理、客户满意度调查、投资风险分析的工具。作为例子，简单介绍股价预测问题。假设知道某一公司在过去不同时间点（比如，每天）的市场上的股票价格（比如，股票平均价格），以及在各个时间点之前可能影响该公司股价的信息（比如，该公司前一周的营业额、利润）。目标是从过去的数据学习一个模型，使它可以基于当前的信息预测该公司下一个时间点的股票价格。可以将这个问题作为回归问题解决。具体地，将影响股价的信息视为自变量（输入的特征），而将股价视为因变量（输出的值）。将过去的数据作为训练数据就可以学习一个回归模型，并对未来的股价进行预测。可以看出这是一个困难的预测问题，因为影响股价的因素非常多，我们未必能判断出哪些信息（输入的特征）有用并能得到这些信息。

## 本章概要

1. 机器学习或统计机器学习是关于计算机基于数据构建概率统计模型并运用模型对数据进行分析与预测的一门学科。机器学习包括监督学习、无监督学习和强化学习。

2. 机器学习方法三要素 —— 模型、策略、算法，对理解机器学习方法起到提纲挈领的作用。

3. 本书第 1 篇主要讨论监督学习，监督学习可以概括如下：从给定的有限训练数据出发，假设数据是独立同分布的，而且假设模型属于某个假设空间，应用某一评价准则，从假设空间中选取一个最优的模型，使它对已给训练数据及未知测试数据在给定评价标准意义下有最准确的预测。

4. 机器学习中，进行模型选择或者说提高学习的泛化能力是一个重要问题。如果只考虑减少训练误差，就可能产生过拟合现象。模型选择的方法有正则化与交叉验证。学习方法泛化能力的分析是机器学习理论研究的重要课题。

5. 分类问题、标注问题和回归问题都是监督学习的重要问题。本书第 1 篇介绍的机器学习方法包括感知机、$k$ 近邻法、朴素贝叶斯法、决策树、逻辑斯谛回归与最大熵模型、支持向量机、Boosting、EM 算法、隐马尔可夫模型和条件随机场。这些方法是主要的分类、标注以及回归方法。它们又可以归类为生成方法与判别方法。

## 继续阅读

关于机器学习或机器学习方法一般介绍的书籍可以参阅文献 [1] ～文献 [8]。

## 习　题

**1.1** 说明伯努利模型的极大似然估计以及贝叶斯估计中的机器学习方法三要素。伯努利模型是定义在取值为 0 与 1 的随机变量上的概率分布。假设观测到伯努利模型 $n$ 次独立的数据生成结果，其中 $k$ 次的结果为 1，这时可以用极大似然估计或贝叶斯估计来估计结果为 1 的概率。

**1.2** 通过经验风险最小化推导极大似然估计。证明模型是条件概率分布，当损失函数是对数损失函数时，经验风险最小化等价于极大似然估计。

## 参考文献


[1] HASTIE T, TIBSHIRANI R, FRIEDMAN J. The elements of statistical learning: data mining, inference, and prediction[M]. 范明，柴玉梅，昝红英，等译. Springer, 2001.

[2] BISHOP M. Pattern recognition and machine learning[M]. Springer, 2006.

[3] DAPHNE K, NIR F. Probabilistic graphical models: principles and techniques[M]. MIT Press, 2009.

[4] IAN G, YOSHUA B, AARON C, et al. Deep learning[M]. MIT Press, 2016.

[5] TOM M M. Machine learning[M]. 曾华军，张银奎，等译. McGraw-Hill Companies, Inc. 1997.

[6] DAVID B. Bayesian reasoning and machine learning[M]. Cambridge University Press, 2012.

[7] RICHARD S S, ANDREW G B. Reinforcement learning: an introduction[M]. MIT Press, 1998.

[8] 周志华. 机器学习 [M]. 北京：清华大学出版社，2016.

# 第 2 章　感　知　机

感知机（perceptron）是二类分类的线性分类模型，其输入为实例的特征向量，输出为实例的类别，取 $+1$ 和 $-1$ 二值。感知机对应于输入空间（特征空间）中将实例划分为正负两类的分离超平面，属于判别模型。感知机学习旨在求出将训练数据进行线性划分的分离超平面，为此，导入基于误分类的损失函数，利用梯度下降法对损失函数进行极小化，求得感知机模型。感知机学习算法具有简单而易于实现的优点，分为原始形式和对偶形式。感知机预测是用学习得到的感知机模型对新的输入实例进行分类。感知机在 1957 年由 Rosenblatt 提出，是神经网络与支持向量机的基础。

本章首先介绍感知机模型；然后叙述感知机的学习策略，特别是损失函数；最后介绍感知机学习算法，包括原始形式和对偶形式，并证明算法的收敛性。

## 2.1　感知机模型

**定义 2.1（感知机）**　假设输入空间（特征空间）是 $\mathcal{X} \subseteq \boldsymbol{R}^n$，输出空间是 $\mathcal{Y} = \{+1, -1\}$。输入 $x \in \mathcal{X}$ 表示实例的特征向量，对应于输入空间（特征空间）的点；输出 $y \in \mathcal{Y}$ 表示实例的类别。由输入空间到输出空间的如下函数

$$f(x) = \text{sign}(w \bullet x + b) \tag{2.1}$$

称为感知机。其中，$w$ 和 $b$ 为感知机模型参数，$w \in \boldsymbol{R}^n$ 叫作权值（weight）或权值向量（weight vector），$b \in \boldsymbol{R}$ 叫作偏置（bias），$w \bullet x$ 表示 $w$ 和 $x$ 的内积。sign 是符号函数，即

$$\text{sign}(x) = \begin{cases} +1, & x \geqslant 0 \\ -1, & x < 0 \end{cases} \tag{2.2}$$

感知机是一种线性分类模型，属于判别模型。感知机模型的假设空间是定义在特征空间中的所有线性分类模型（linear classification model）或线性分类器（linear classifier），即函数集合 $\{f | f(x) = w \bullet x + b\}$。

感知机有如下几何解释：线性方程

$$w \bullet x + b = 0 \tag{2.3}$$

对应于特征空间 $\boldsymbol{R}^n$ 中的一个超平面 $S$，其中 $w$ 是超平面的法向量，$b$ 是超平面的截距。这

个超平面将特征空间划分为两个部分。位于两部分的点（特征向量）分别被分为正、负两类。因此，超平面 $S$称为分离超平面（separating hyperplane），如图 2.1 所示。

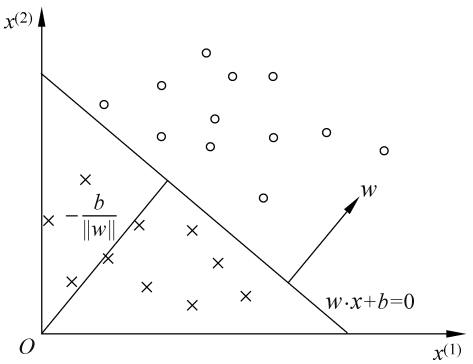


图 2.1 感知机模型

感知机学习由训练数据集（实例的特征向量及类别）

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$i = 1, 2, \cdots, N$，求得感知机模型（式（2.1））， 即求得模型参数 $w, b$。感知机预测通过学习得到的感知机模型，对新的输入实例给出其对应的输出类别。

## 2.2 感知机学习策略

### 2.2.1 数据集的线性可分性

**定义 2.2（数据集的线性可分性）** 给定一个数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$i = 1, 2, \cdots, N$，如果存在某个超平面 $S$

$$w \bullet x + b = 0$$

能够将数据集的正实例点和负实例点完全正确地划分到超平面的两侧，即对所有 $y_i = +1$ 的实例 $i$，有 $w \bullet x_i + b > 0$，对所有 $y_i = -1$ 的实例 $i$，有 $w \bullet x_i + b < 0$，则称数据集 $T$ 为线性可分数据集（linearly separable data set）；否则，称数据集 $T$ 线性不可分。

### 2.2.2 感知机学习策略

假设训练数据集是线性可分的，感知机学习的目标是求得一个能够将训练集正实例点和负实例点完全正确分开的分离超平面。为了找出这样的超平面，即确定感知机模型参数 $w, b$，需要确定一个学习策略，即定义（经验）损失函数并将损失函数极小化。

损失函数的一个自然选择是误分类点的总数。但是，这样的损失函数不是参数 $w, b$ 的连续可导函数，不易优化。损失函数的另一个选择是误分类点到超平面 $S$ 的总距离，这是感知机所采用的。为此，首先写出输入空间 $\boldsymbol{R}^n$ 中任一点 $x_0$ 到超平面 $S$ 的距离：

$$\frac{1}{\|w\|}|w \bullet x_0 + b|$$

这里，$\|w\|$ 是 $w$ 的 $L_2$ 范数。

其次，对于误分类的数据 $(x_i, y_i)$，

$$-y_i(w \bullet x_i + b) > 0$$

成立。因为当 $w \bullet x_i + b > 0$ 时，$y_i = -1$；而当 $w \bullet x_i + b < 0$ 时，$y_i = +1$。因此，误分类点 $x_i$ 到超平面 $S$ 的距离是

$$-\frac{1}{\|w\|}y_i(w \bullet x_i + b)$$

假设超平面 $S$ 的误分类点集合为 $M$，那么所有误分类点到超平面 $S$ 的总距离为

$$-\frac{1}{\|w\|}\sum_{x_i \in M} y_i(w \bullet x_i + b)$$

不考虑 $\frac{1}{\|w\|}$，就得到感知机学习的损失函数[①]。

给定训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$i = 1, 2, \cdots, N$。感知机 $\text{sign}(w \bullet x + b)$ 学习的损失函数定义为

$$L(w, b) = -\sum_{x_i \in M} y_i(w \bullet x_i + b) \tag{2.4}$$

其中，$M$ 为误分类点的集合。这个损失函数就是感知机学习的经验风险函数。

显然，损失函数 $L(w, b)$ 是非负的。如果没有误分类点，损失函数值是 0。而且，误分类点越少，误分类点离超平面越近，损失函数值就越小。一个特定的样本点的损失函数在误分类时是参数 $w, b$ 的线性函数，在正确分类时是 0。因此，给定训练数据集 $T$，损失函数 $L(w, b)$ 是 $w, b$ 的连续可导函数。

感知机学习的策略是在假设空间中选取使损失函数式（2.4）最小的模型参数 $w$, $b$, 即感知机模型。

## 2.3 感知机学习算法

感知机学习问题转化为求解损失函数式（2.4）的最优化问题，最优化的方法是随机梯度下降法。本节叙述感知机学习的具体算法，包括原始形式和对偶形式，并证明在训练数据线

① 第 7 章中会介绍 $y(w \bullet x + b)$ 称为样本点的函数间隔。

性可分条件下感知机学习算法的收敛性。

### 2.3.1 感知机学习算法的原始形式

感知机学习算法是对以下最优化问题的算法。给定一个训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{-1, 1\}$，$i = 1, 2, \cdots, N$，求参数 $w, b$，使其为以下损失函数极小化问题的解：

$$\min_{w,b} L(w, b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b) \tag{2.5}$$

其中，$M$ 为误分类点的集合。

感知机学习算法是误分类驱动的，具体采用随机梯度下降法（stochastic gradient descent）。首先，任意选取一个超平面 $w_0$, $b_0$，然后用梯度下降法不断地极小化目标函数（式（2.5））。极小化过程中不是一次使 $M$ 中所有误分类点的梯度下降，而是一次随机选取一个误分类点使其梯度下降。

假设误分类点集合 $M$ 是固定的，那么损失函数 $L(w, b)$ 的梯度由

$$\nabla_w L(w, b) = -\sum_{x_i \in M} y_i x_i$$

$$\nabla_b L(w, b) = -\sum_{x_i \in M} y_i$$

给出。

随机选取一个误分类点 $(x_i, y_i)$，对 $w, b$ 进行更新：

$$w \leftarrow w + \eta y_i x_i \tag{2.6}$$

$$b \leftarrow b + \eta y_i \tag{2.7}$$

式中 $\eta\,(0 < \eta \leqslant 1)$ 是步长，在机器学习中又称为学习率（learning rate）。这样，通过迭代可以期待损失函数 $L(w, b)$ 不断减小，直到为 0。综上所述，得到如下算法：

**算法 2.1（感知机学习算法的原始形式）**

输入：训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，其中 $x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$，$i = 1, 2, \cdots, N$；学习率 $\eta\,(0 < \eta \leqslant 1)$。

输出：$w$,$b$；感知机模型 $f(x) = \text{sign}(w \cdot x + b)$。

（1）选取初值 $w_0, b_0$；

（2）在训练集中选取数据 $(x_i, y_i)$；

（3）如果 $y_i(w \cdot x_i + b) \leqslant 0$，则

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

（4）转至步骤（2），直至训练集中没有误分类点。 ■

这种学习算法直观上有如下解释：当一个实例点被误分类，即位于分离超平面的错误一侧时，则调整 $w, b$ 的值，使分离超平面向该误分类点的一侧移动，以减少该误分类点与超平面间的距离，直至超平面越过该误分类点使其被正确分类。

算法 2.1 是感知机学习的基本算法，对应于后面的对偶形式，称为原始形式。感知机学习算法简单且易于实现。

**例 2.1** 如图 2.2 所示的训练数据集，其正实例点是 $x_1 = (3,3)^{\mathrm{T}}$，$x_2 = (4,3)^{\mathrm{T}}$，负实例点是 $x_3 = (1,1)^{\mathrm{T}}$，试用感知机学习算法的原始形式求感知机模型 $f(x) = \mathrm{sign}(w \cdot x + b)$。这里，$w = (w^{(1)}, w^{(2)})^{\mathrm{T}}$，$x = (x^{(1)}, x^{(2)})^{\mathrm{T}}$。

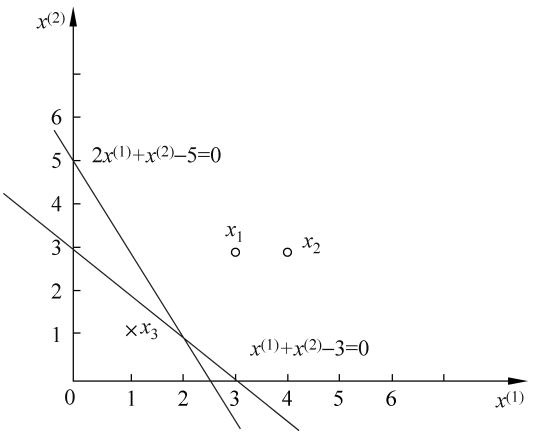


图 2.2 感知机示例

**解** 构建最优化问题：

$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b)$$

按照算法 2.1 求解 $w, b$，$\eta = 1$。

（1）取初值 $w_0 = 0$，$b_0 = 0$；

（2）对 $x_1 = (3,3)^{\mathrm{T}}$，$y_1(w_0 \cdot x_1 + b_0) = 0$，未能被正确分类，更新 $w$，$b$：

$$w_1 = w_0 + y_1 x_1 = (3,3)^{\mathrm{T}},\ b_1 = b_0 + y_1 = 1$$

得到线性模型：

$$w_1 \cdot x + b_1 = 3x^{(1)} + 3x^{(2)} + 1$$

（3）对 $x_1, x_2$，显然，$y_i(w_1 \cdot x_i + b_1) > 0$，被正确分类，不修改 $w$，$b$；对 $x_3 = (1,1)^{\mathrm{T}}$，$y_3(w_1 \cdot x_3 + b_1) < 0$，被误分类，更新 $w$，$b$：

$$w_2 = w_1 + y_3 x_3 = (2,2)^{\mathrm{T}},\ b_2 = b_1 + y_3 = 0$$

得到线性模型：

$$w_2 \cdot x + b_2 = 2x^{(1)} + 2x^{(2)}$$

如此继续下去，直到

$$w_7 = (1,1)^{\mathrm{T}},\ b_7 = -3$$

$$w_7 \bullet x + b_7 = x^{(1)} + x^{(2)} - 3$$

对所有数据点 $y_i(w_7 \bullet x_i + b_7) > 0$，没有误分类点，损失函数达到极小。

分离超平面为 $x^{(1)} + x^{(2)} - 3 = 0$，感知机模型为 $f(x) = \mathrm{sign}(x^{(1)} + x^{(2)} - 3)$。 ■

迭代过程见表 2.1。

**表 2.1　例 2.1 求解的迭代过程**

| 迭代次数 | 误分类点 | $w$ | $b$ | $w \bullet x + b$ |
|---|---|---|---|---|
| 0 | | 0 | 0 | 0 |
| 1 | $x_1$ | $(3,3)^{\mathrm{T}}$ | 1 | $3x^{(1)} + 3x^{(2)} + 1$ |
| 2 | $x_3$ | $(2,2)^{\mathrm{T}}$ | 0 | $2x^{(1)} + 2x^{(2)}$ |
| 3 | $x_3$ | $(1,1)^{\mathrm{T}}$ | $-1$ | $x^{(1)} + x^{(2)} - 1$ |
| 4 | $x_3$ | $(0,0)^{\mathrm{T}}$ | $-2$ | $-2$ |
| 5 | $x_1$ | $(3,3)^{\mathrm{T}}$ | $-1$ | $3x^{(1)} + 3x^{(2)} - 1$ |
| 6 | $x_3$ | $(2,2)^{\mathrm{T}}$ | $-2$ | $2x^{(1)} + 2x^{(2)} - 2$ |
| 7 | $x_3$ | $(1,1)^{\mathrm{T}}$ | $-3$ | $x^{(1)} + x^{(2)} - 3$ |
| 8 | 0 | $(1,1)^{\mathrm{T}}$ | $-3$ | $x^{(1)} + x^{(2)} - 3$ |

这是在计算中误分类点先后取 $x_1$,$x_3$,$x_3$,$x_3$,$x_1$,$x_3$,$x_3$ 得到的分离超平面和感知机模型。如果在计算中误分类点依次取 $x_1$,$x_3$,$x_3$,$x_3$,$x_2$,$x_3$,$x_3$,$x_3$,$x_1$,$x_3$,$x_3$，那么得到的分离超平面是 $2x^{(1)} + x^{(2)} - 5 = 0$。

可见，感知机学习算法由于采用不同的初值或选取不同的误分类点，解可以不同。

### 2.3.2 算法的收敛性

现在证明，对于线性可分数据集，感知机学习算法原始形式收敛，即经过有限次迭代可以得到一个将训练数据集完全正确划分的分离超平面及感知机模型。

为了便于叙述与推导，将偏置 $b$ 并入权重向量 $w$，记作 $\hat{w} = (w^{\mathrm{T}}, b)^{\mathrm{T}}$，同样也将输入向量加以扩充，加进常数 1，记作 $\hat{x} = (x^{\mathrm{T}}, 1)^{\mathrm{T}}$。这样，$\hat{x} \in \boldsymbol{R}^{n+1}$，$\hat{w} \in \boldsymbol{R}^{n+1}$。显然，$\hat{w} \bullet \hat{x} = w \bullet x + b$。

**定理 2.1（Novikoff）** 设训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ 是线性可分的，其中 $x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$，$i = 1, 2, \cdots, N$，则

（1）存在满足条件 $\|\hat{w}_{\mathrm{opt}}\| = 1$ 的超平面 $\hat{w}_{\mathrm{opt}} \bullet \hat{x} = w_{\mathrm{opt}} \bullet x + b_{\mathrm{opt}} = 0$ 将训练数据集完全正确分开；且存在 $\gamma > 0$，对所有 $i = 1, 2, \cdots, N$，有

$$y_i(\hat{w}_{\mathrm{opt}} \bullet \hat{x}_i) = y_i(w_{\mathrm{opt}} \bullet x_i + b_{\mathrm{opt}}) \geqslant \gamma \tag{2.8}$$

（2）令 $R=\max\limits_{1\leqslant i\leqslant N}\|\hat{x}_i\|$，则感知机算法 2.1 在训练数据集上的误分类次数 $k$ 满足不等式

$$k\leqslant\left(\frac{R}{\gamma}\right)^2 \tag{2.9}$$

**证明** （1）由于训练数据集是线性可分的，按照定义 2.2，存在超平面可将训练数据集完全正确分开，取此超平面为 $\hat{w}_{\text{opt}}\bullet\hat{x}=w_{\text{opt}}\bullet x+b_{\text{opt}}=0$，使 $\|\hat{w}_{\text{opt}}\|=1$。由于对有限的 $i=1,2,\cdots,N$，均有

$$y_i(\hat{w}_{\text{opt}}\bullet\hat{x}_i)=y_i(w_{\text{opt}}\bullet x_i+b_{\text{opt}})>0$$

所以存在

$$\gamma=\min_i\{y_i(w_{\text{opt}}\bullet x_i+b_{\text{opt}})\}$$

使

$$y_i(\hat{w}_{\text{opt}}\bullet\hat{x}_i)=y_i(w_{\text{opt}}\bullet x_i+b_{\text{opt}})\geqslant\gamma$$

（2）感知机算法从 $\hat{w}_0=0$ 开始，如果实例被误分类，则更新权重。令 $\hat{w}_{k-1}$ 是第 $k$ 个误分类实例之前的扩充权重向量，即

$$\hat{w}_{k-1}=(w_{k-1}^{\mathrm{T}},b_{k-1})^{\mathrm{T}}$$

则第 $k$ 个误分类实例的条件是

$$y_i(\hat{w}_{k-1}\bullet\hat{x}_i)=y_i(w_{k-1}\bullet x_i+b_{k-1})\leqslant 0 \tag{2.10}$$

若 $(x_i,y_i)$ 是被 $\hat{w}_{k-1}=(w_{k-1}^{\mathrm{T}},b_{k-1})^{\mathrm{T}}$ 误分类的数据，则 $w$ 和 $b$ 的更新是

$$w_k\leftarrow w_{k-1}+\eta y_i x_i$$
$$b_k\leftarrow b_{k-1}+\eta y_i$$

即

$$\hat{w}_k=\hat{w}_{k-1}+\eta y_i\hat{x}_i \tag{2.11}$$

下面推导不等式 (2.12) 及不等式 (2.13)：

$$\hat{w}_k\bullet\hat{w}_{\text{opt}}\geqslant k\eta\gamma \tag{2.12}$$

由式 (2.11) 及式 (2.8) 得：

$$\begin{aligned}\hat{w}_k\bullet\hat{w}_{\text{opt}}&=\hat{w}_{k-1}\bullet\hat{w}_{\text{opt}}+\eta y_i\hat{w}_{\text{opt}}\bullet\hat{x}_i\\&\geqslant\hat{w}_{k-1}\bullet\hat{w}_{\text{opt}}+\eta\gamma\end{aligned}$$

由此递推即得不等式 (2.12)：

$$\hat{w}_k\bullet\hat{w}_{\text{opt}}\geqslant\hat{w}_{k-1}\bullet\hat{w}_{\text{opt}}+\eta\gamma\geqslant\hat{w}_{k-2}\bullet\hat{w}_{\text{opt}}+2\eta\gamma\geqslant\cdots\geqslant k\eta\gamma$$

$$\|\hat{w}_k\|^2\leqslant k\eta^2R^2 \tag{2.13}$$

由式 (2.11) 及式 (2.10) 得：

$$\begin{aligned}\|\hat{w}_k\|^2 &= \|\hat{w}_{k-1}\|^2 + 2\eta y_i \hat{w}_{k-1} \bullet \hat{x}_i + \eta^2 \|\hat{x}_i\|^2 \\ &\leqslant \|\hat{w}_{k-1}\|^2 + \eta^2 \|\hat{x}_i\|^2 \\ &\leqslant \|\hat{w}_{k-1}\|^2 + \eta^2 R^2 \\ &\leqslant \|\hat{w}_{k-2}\|^2 + 2\eta^2 R^2 \leqslant \cdots \\ &\leqslant k\eta^2 R^2\end{aligned}$$

结合不等式 (2.12) 及式 (2.13) 即得：

$$\begin{aligned}&k\eta\gamma \leqslant \hat{w}_k \bullet \hat{w}_{\text{opt}} \leqslant \|\hat{w}_k\| \ \|\hat{w}_{\text{opt}}\| \ \leqslant \sqrt{k}\eta R \\ &k^2\gamma^2 \leqslant kR^2\end{aligned}$$

于是

$$k \leqslant \left(\frac{R}{\gamma}\right)^2$$

■

定理表明，误分类的次数 $k$ 是有上界的，经过有限次搜索可以找到将训练数据完全正确分开的分离超平面。也就是说，当训练数据集线性可分时，感知机学习算法原始形式迭代是收敛的。但是例 2.1 说明，感知机学习算法存在许多解，这些解既依赖于初值的选择，也依赖于迭代过程中误分类点的选择顺序。为了得到唯一的超平面，需要对分离超平面增加约束条件。这就是第 7 章将要讲述的线性支持向量机的想法。当训练集线性不可分时，感知机学习算法不收敛，迭代结果会发生振荡。

### 2.3.3 感知机学习算法的对偶形式

现在考虑感知机学习算法的对偶形式。感知机学习算法的原始形式和对偶形式与第 7 章中支持向量机学习算法的原始形式和对偶形式相对应。

对偶形式的基本想法是：将 $w$ 和 $b$ 表示为实例 $x_i$ 和标记 $y_i$ 的线性组合的形式，通过求解其系数而求得 $w$ 和 $b$。不失一般性，在算法 2.1 中可假设初始值 $w_0, b_0$ 均为 0。对误分类点 $(x_i, y_i)$ 通过

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

逐步修改 $w, b$，设修改 $n$ 次，则 $w, b$ 关于 $(x_i, y_i)$ 的增量分别是 $\alpha_i y_i x_i$ 和 $\alpha_i y_i$，这里 $\alpha_i = n_i \eta, n_i$ 是点 $(x_i, y_i)$ 被误分类的次数。这样，从学习过程不难看出，最后学习到的 $w, b$ 可以分别表示为

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{2.14}$$

$$b = \sum_{i=1}^{N} \alpha_i y_i \tag{2.15}$$

这里，$\alpha_i \geqslant 0$，$i = 1, 2, \cdots, N$，当 $\eta = 1$ 时，表示第 $i$ 个实例点由于误分而进行更新的次数。实例点更新次数越多，意味着它距分离超平面越近，也就越难正确分类。换句话说，这样的实例对学习结果影响最大。

下面对照原始形式来叙述感知机学习算法的对偶形式。

**算法 2.2（感知机学习算法的对偶形式）**

输入：线性可分的数据集 $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，其中 $x_i \in \boldsymbol{R}^n$，$y_i \in \{-1, +1\}$，$i = 1, 2, \cdots, N$；学习率 $\eta$（$0 < \eta \leqslant 1$）。

输出：$\alpha, b$；感知机模型 $f(x) = \operatorname{sign}\left(\sum\limits_{j=1}^{N} \alpha_j y_j x_j \cdot x + b\right)$，其中 $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_N)^{\mathrm{T}}$。

（1）$\alpha \leftarrow 0$，$b \leftarrow 0$；

（2）在训练集中选取数据 $(x_i, y_i)$；

（3）如果 $y_i \left(\sum\limits_{j=1}^{N} \alpha_j y_j x_j \cdot x_i + b\right) \leqslant 0$，则

$$\alpha_i \leftarrow \alpha_i + \eta$$
$$b \leftarrow b + \eta y_i$$

（4）转至步骤（2）直到没有误分类数据。 ■

对偶形式中训练实例仅以内积的形式出现。为了方便，可以预先将训练集中实例间的内积计算出来并以矩阵的形式存储，这个矩阵就是所谓的 Gram 矩阵（Gram matrix）：

$$G = [x_i \cdot x_j]_{N \times N}$$

**例 2.2** 数据同例 2.1，正样本点是 $x_1 = (3, 3)^{\mathrm{T}}$，$x_2 = (4, 3)^{\mathrm{T}}$，负样本点是 $x_3 = (1, 1)^{\mathrm{T}}$，试用感知机学习算法对偶形式求感知机模型。

**解** 按照算法 2.2，

（1）取 $\alpha_i = 0$，$i = 1, 2, 3$，$b = 0$，$\eta = 1$；

（2）计算 Gram 矩阵：

$$G = \begin{bmatrix} 18 & 21 & 6 \\ 21 & 25 & 7 \\ 6 & 7 & 2 \end{bmatrix}$$

（3）误分条件

$$y_i \left(\sum_{j=1}^{N} \alpha_j y_j x_j \cdot x_i + b\right) \leqslant 0$$

参数更新：

$$\alpha_i \leftarrow \alpha_i + 1, \ b \leftarrow b + y_i$$

（4）迭代，过程从略，结果列于表 2.2；

（5）

$$w = 2x_1 + 0x_2 - 5x_3 = (1,1)^{\mathrm{T}}$$

$$b = -3$$

分离超平面为

$$x^{(1)} + x^{(2)} - 3 = 0$$

感知机模型为

$$f(x) = \mathrm{sign}(x^{(1)} + x^{(2)} - 3)$$ ■

**表 2.2　例 2.2 求解的迭代过程**

| $k$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| | | $x_1$ | $x_3$ | $x_3$ | $x_3$ | $x_1$ | $x_3$ | $x_3$ |
| $\alpha_1$ | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 2 |
| $\alpha_2$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $\alpha_3$ | 0 | 0 | 1 | 2 | 3 | 3 | 4 | 5 |
| $b$ | 0 | 1 | 0 | $-1$ | $-2$ | $-1$ | $-2$ | $-3$ |

对照例 2.1，结果一致，迭代步骤也是互相对应的。

与原始形式一样，感知机学习算法的对偶形式迭代是收敛的，存在多个解。

## 本章概要

1. 感知机是根据输入实例的特征向量 $x$ 对其进行二类分类的线性分类模型：

$$f(x) = \mathrm{sign}(w \cdot x + b)$$

感知机模型对应于输入空间（特征空间）中的分离超平面 $w \cdot x + b = 0$。

2. 感知机学习的策略是极小化损失函数：

$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b)$$

损失函数对应于误分类点到分离超平面的总距离。

3. 感知机学习算法是基于随机梯度下降法的对损失函数的最优化算法，有原始形式和对偶形式。算法简单且易于实现。原始形式中，首先任意选取一个超平面，然后用梯度下降法不断极小化目标函数。在这个过程中一次随机选取一个误分类点使其梯度下降。

4. 当训练数据集线性可分时，感知机学习算法是收敛的。感知机算法在训练数据集上的误分类次数 $k$ 满足不等式

$$k \leqslant \left(\frac{R}{\gamma}\right)^2$$

当训练数据集线性可分时，感知机学习算法存在无穷多个解，其解由于不同的初值或不同的迭代顺序而可能有所不同。

## 继续阅读

感知机最早在 1957 年由 Rosenblatt 提出 [1]。Novikoff [2]，Minsky 与 Papert [3] 等人对感知机进行了一系列理论研究。感知机的扩展学习方法包括口袋算法（pocket algorithm）[4]、表决感知机（voted perceptron）[5]、带边缘感知机（perceptron with margin）[6]。关于感知机的介绍可进一步参考文献 [7] 和文献 [8]。

## 习 题

**2.1** Minsky 与 Papert 指出：感知机因为是线性模型，所以不能表示复杂的函数，如异或（XOR）。验证感知机为什么不能表示异或。

**2.2** 模仿例题 2.1，构建从训练数据集求解感知机模型的例子。

**2.3** 证明以下定理：样本集线性可分的充分必要条件是正实例点集所构成的凸壳①与负实例点集所构成的凸壳互不相交。

## 参考文献

[1] ROSENBLATT F. The perceptron: a probabilistic model for information storage and organization in the Brain[J]. Psychological Review, 1958, 65 (6): 386–408.

[2] NOVIKOFF A B. On convergence proofs on perceptrons[C]//Polytechnic Institute of Brooklyn. Proceedings of the Symposium on the Mathematical Theory of Automata. 1962: 615–622.

[3] MINSKY M L, Papert S A. Perceptrons[M]. Cambridge, MA: MIT Press. 1969.

[4] GALLANT S I. Perceptron-based learning algorithms[J]. IEEE Transactions on Neural Networks, 1990, 1(2): 179–191.

[5] FREUND Y, SCHAPIRE R E. Large margin classification using the perceptron algorithm[C]// Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT' 98). ACM Press, 1998.

[6] LI Y Y, Zaragoza H, Herbrich R, et al. The perceptron algorithm with uneven margins[C]// Proceedings of the 19th International Conference on Machine Learning. 2002: 379–386.

[7] WIDROW B, LEHR M A. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation[J]. Proceedings of the IEEE, 1990, 78(9): 1415–1442.

[8] CRISTIANINI N, SHAWE-TAYLOR J. An introduction to support vector machines and other kernel-based learning methods[M]. Cambridge University Press, 2000.

① 设集合 $S \subset \boldsymbol{R}^n$ 是由 $\boldsymbol{R}^n$ 中的 $k$ 个点所组成的集合，即 $S = \{x_1, x_2, \cdots, x_k\}$，定义 $S$ 的凸壳 $\mathrm{conv}(S)$ 为

$$\mathrm{conv}(S) = \left\{ x = \sum_{i=1}^{k} \lambda_i x_i \,\middle|\, \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geqslant 0,\ i = 1, 2, \cdots, k \right\}$$

# 第 3 章　$k$ 近 邻 法

$k$ 近邻法（$k$-nearest neighbor，$k$-NN）是一种基本分类与回归方法。本书只讨论分类问题中的 $k$ 近邻法。$k$ 近邻法的输入为实例的特征向量，对应于特征空间的点；输出为实例的类别，可以取多类。$k$ 近邻法假设给定一个训练数据集，其中的实例类别已定。分类时，对新的实例，根据其 $k$ 个最近邻的训练实例的类别，通过多数表决等方式进行预测。因此，$k$ 近邻法不具有显式的学习过程。$k$ 近邻法实际上利用训练数据集对特征向量空间进行划分，并作为其分类的“模型”。$k$ 值的选择、距离度量及分类决策规则是 $k$ 近邻法的三个基本要素。$k$ 近邻法在 1968 年由 Cover 和 Hart 提出。

本章首先叙述 $k$ 近邻算法，然后讨论 $k$ 近邻法的模型及三个基本要素，最后讲述 $k$ 近邻法的一个实现方法 —— $kd$ 树，介绍构造 $kd$ 树和搜索 $kd$ 树的算法。

## 3.1　$k$ 近邻算法

$k$ 近邻算法简单、直观：给定一个训练数据集，对新的输入实例，在训练数据集中找到与该实例最邻近的 $k$ 个实例，这 $k$ 个实例的多数属于某个类，就把该输入实例分为这个类。下面先叙述 $k$ 近邻算法，然后再讨论其细节。

**算法 3.1（$k$ 近邻法）**

输入：训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} \subseteq \boldsymbol{R}^n$ 为实例的特征向量，$y_i \in \mathcal{Y} = \{c_1, c_2, \cdots, c_K\}$ 为实例的类别，$i = 1, 2, \cdots, N$。

输出：实例 $x$ 所属的类 $y$。

（1）根据给定的距离度量，在训练集 $T$ 中找出与 $x$ 最邻近的 $k$ 个点，涵盖这 $k$ 个点的 $x$ 的邻域记作 $N_k(x)$；

（2）在 $N_k(x)$ 中根据分类决策规则（如多数表决）决定 $x$ 的类别 $y$：

$$y = \arg\max_{c_j} \sum_{x_i \in N_k(x)} I(y_i = c_j), \quad i = 1, 2, \cdots, N, j = 1, 2, \cdots, K \tag{3.1}$$

式 (3.1) 中，$I$ 为指示函数，即当 $y_i = c_j$ 时 $I$ 为 1，否则 $I$ 为 0。 ■

$k$ 近邻法的特殊情况是 $k = 1$ 的情形，称为最近邻算法。对于输入的实例点（特征向量）$x$，最近邻法将训练数据集中与 $x$ 最邻近点的类作为 $x$ 的类。

$k$ 近邻法没有显式的学习过程。

## 3.2 $k$ 近邻模型

$k$ 近邻法使用的模型实际上对应于对特征空间的划分。模型由三个基本要素 —— 距离度量、$k$ 值的选择和分类决策规则决定。

### 3.2.1 模型

$k$ 近邻法中，当训练集、距离度量（如欧氏距离）、$k$ 值及分类决策规则（如多数表决）确定后，对于任何一个新的输入实例，它所属的类唯一地确定。这相当于根据上述要素将特征空间划分为一些子空间，确定子空间里的每个点所属的类。这一事实从最近邻算法中可以看得很清楚。

特征空间中，对每个训练实例点 $x_i$，距离该点比其他点更近的所有点组成一个区域，叫作单元（cell）。每个训练实例点拥有一个单元，所有训练实例点的单元构成对特征空间的一个划分。最近邻法将实例 $x_i$ 的类 $y_i$ 作为其单元中所有点的类标记（class label）。这样，每个单元的实例点的类别是确定的。图 3.1 是二维特征空间划分的一个例子。

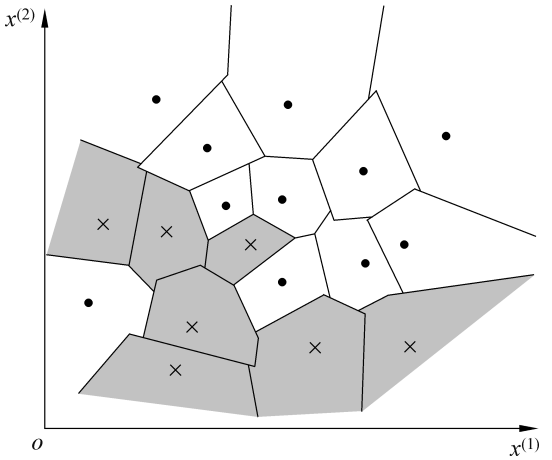


图 3.1 $k$ 近邻法的模型对应特征空间的一个划分

### 3.2.2 距离度量

特征空间中两个实例点的距离是两个实例点相似程度的反映。$k$ 近邻模型的特征空间一般是 $n$ 维实数向量空间 $\boldsymbol{R}^n$。使用的距离是欧氏距离，但也可以是其他距离，如更一般的 $L_p$ 距离（$L_p$ distance）或 Minkowski 距离（Minkowski distance）。

设特征空间 $\mathcal{X}$ 是 $n$ 维实数向量空间 $\boldsymbol{R}^n$，$x_i, x_j \in \mathcal{X}$，$x_i = (x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(n)})^{\mathrm{T}}$，$x_j = (x_j^{(1)}, x_j^{(2)}, \cdots, x_j^{(n)})^{\mathrm{T}}$，$x_i, x_j$ 的 $L_p$ 距离定义为

$$L_p(x_i, x_j) = \left( \sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|^p \right)^{\frac{1}{p}} \tag{3.2}$$

这里 $p \geqslant 1$。当 $p = 2$ 时，称为欧氏距离（Euclidean distance），即

$$L_2(x_i, x_j) = \left( \sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|^2 \right)^{\frac{1}{2}} \tag{3.3}$$

当 $p = 1$ 时，称为曼哈顿距离（Manhattan distance），即

$$L_1(x_i, x_j) = \sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}| \tag{3.4}$$

当 $p = \infty$ 时，它是各个坐标距离的最大值，即

$$L_\infty(x_i, x_j) = \max_l |x_i^{(l)} - x_j^{(l)}| \tag{3.5}$$

图 3.2 给出了二维空间中 $p$ 取不同值时，与原点的 $L_p$ 距离为 1（$L_p = 1$）的点的图形。

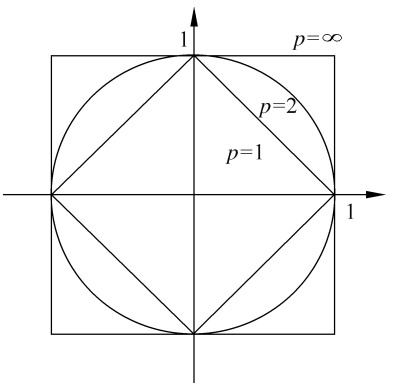


图 3.2 $L_p$ 距离间的关系

下面的例子说明，由不同的距离度量所确定的最近邻点是不同的。

**例 3.1** 已知二维空间的 3 个点 $x_1 = (1, 1)^{\mathrm{T}}$, $x_2 = (5, 1)^{\mathrm{T}}$, $x_3 = (4, 4)^{\mathrm{T}}$，试求在 $p$ 取不同值时，$L_p$ 距离下 $x_1$ 的最近邻点。

**解** 因为 $x_1$ 和 $x_2$ 只有第一维的值不同，所以 $p$ 为任何值时，$L_p(x_1, x_2) = 4$。而

$$L_1(x_1, x_3) = 6, \quad L_2(x_1, x_3) = 4.24, \quad L_3(x_1, x_3) = 3.78, \quad L_4(x_1, x_3) = 3.57$$

于是得到：$p = 1$ 或 2 时，$x_2$ 是 $x_1$ 的最近邻点；$p \geqslant 3$ 时，$x_3$ 是 $x_1$ 的最近邻点。 ■

### 3.2.3 $k$ 值的选择

$k$ 值的选择会对 $k$ 近邻法的结果产生重大影响。

如果选择较小的 $k$ 值，就相当于用较小的邻域中的训练实例进行预测，“学习”的近似误差（approximation error）会减小，只有与输入实例较近的（相似的）训练实例才会对预测结

果起作用。但缺点是“学习”的估计误差（estimation error）会增大，预测结果会对近邻的实例点非常敏感 [2]。如果邻近的实例点恰巧是噪声，预测就会出错。换句话说，$k$ 值的减小就意味着整体模型变得复杂，容易发生过拟合。

如果选择较大的 $k$ 值，就相当于用较大邻域中的训练实例进行预测。其优点是可以减少学习的估计误差，但缺点是学习的近似误差会增大。这时与输入实例较远的（不相似的）训练实例也会对预测起作用，使预测发生错误。$k$ 值的增大就意味着整体模型变得简单。

如果 $k = N$，那么无论输入实例是什么，都将简单地预测它属于在训练实例中最多的类。这时，模型过于简单，完全忽略训练实例中的大量有用信息，是不可取的。

在应用中，$k$ 值一般取一个比较小的数值。通常采用交叉验证法来选取最优的 $k$ 值。

### 3.2.4 分类决策规则

$k$ 近邻法中的分类决策规则往往是多数表决，即由输入实例的 $k$ 个邻近的训练实例中的多数类决定输入实例的类。

多数表决规则（majority voting rule）有如下解释：如果分类的损失函数为 0-1 损失函数，分类函数为

$$f : \boldsymbol{R}^n \to \{c_1, c_2, \cdots, c_K\}$$

那么误分类的概率是

$$P(Y \neq f(X)) = 1 - P(Y = f(X))$$

对给定的实例 $x \in \mathcal{X}$，其最近邻的 $k$ 个训练实例点构成集合 $N_k(x)$。如果涵盖 $N_k(x)$ 的区域的类别是 $c_j$，那么误分类率是

$$\frac{1}{k} \sum_{x_i \in N_k(x)} I(y_i \neq c_j) = 1 - \frac{1}{k} \sum_{x_i \in N_k(x)} I(y_i = c_j)$$

要使误分类率最小即经验风险最小，就要使 $\sum\limits_{x_i \in N_k(x)} I(y_i = c_j)$ 最大，所以多数表决规则等价于经验风险最小化。

## 3.3 $k$ 近邻法的实现：$kd$ 树

实现 $k$ 近邻法时，主要考虑的问题是如何对训练数据进行快速 $k$ 近邻搜索。这点在特征空间的维数大及训练数据容量大时尤其必要。

$k$ 近邻法最简单的实现方法是线性扫描（linear scan），这时要计算输入实例与每一个训练实例的距离。当训练集很大时，计算非常耗时，这种方法是不可行的。

为了提高 $k$ 近邻搜索的效率，可以考虑使用特殊的结构存储训练数据，以减少计算距离的次数。具体方法很多，下面介绍其中的 $kd$ 树（$kd$ tree）方法①。

① $kd$ 树是存储 $k$ 维空间数据的树结构，这里的 $k$ 与 $k$ 近邻法的 $k$ 意义不同，为了与习惯一致，本书仍用 $kd$ 树的名称。

### 3.3.1 构造 $kd$ 树

$kd$ 树是一种对 $k$ 维空间中的实例点进行存储以便对其进行快速检索的树形数据结构。$kd$ 树是二叉树，表示对 $k$ 维空间的一个划分（partition）。构造 $kd$ 树相当于不断地用垂直于坐标轴的超平面将 $k$ 维空间切分，构成一系列的 $k$ 维超矩形区域。$kd$ 树的每个结点对应于一个 $k$ 维超矩形区域。

构造 $kd$ 树的方法如下：构造根结点，使根结点对应于 $k$ 维空间中包含所有实例点的超矩形区域；通过下面的递归方法，不断地对 $k$ 维空间进行切分，生成子结点。在超矩形区域（结点）上选择一个坐标轴和在此坐标轴上的一个切分点，确定一个超平面，这个超平面通过选定的切分点并垂直于选定的坐标轴，将当前超矩形区域切分为左、右两个子区域（子结点），这时实例被分到两个子区域。这个过程直到子区域内没有实例时终止（终止时的结点为叶结点）。在此过程中，将实例保存在相应的结点上。

通常，依次选择坐标轴对空间切分，选择训练实例点在选定坐标轴上的中位数（median）[①]为切分点，这样得到的 $kd$ 树是平衡的。注意，平衡的 $kd$ 树搜索时的效率未必是最优的。

下面给出构造 $kd$ 树的算法。

**算法 3.2（构造平衡 $kd$ 树）**

输入：$k$ 维空间数据集 $T=\{x_1,x_2,\cdots,x_N\}$，其中 $x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(k)})^{\mathrm{T}}$，$i=1,2,\cdots,N$。

输出：$kd$ 树。

（1）开始：构造根结点，根结点对应于包含 $T$ 的 $k$ 维空间的超矩形区域。

选择 $x^{(1)}$ 为坐标轴，以 $T$ 中所有实例的 $x^{(1)}$ 坐标的中位数为切分点，将根结点对应的超矩形区域切分为两个子区域。切分由通过切分点并与坐标轴 $x^{(1)}$ 垂直的超平面实现。

由根结点生成深度为 1 的左、右子结点：左子结点对应坐标 $x^{(1)}$ 小于切分点的子区域，右子结点对应坐标 $x^{(1)}$ 大于切分点的子区域。

将落在切分超平面上的实例点保存在根结点。

（2）重复：对深度为 $j$ 的结点，选择 $x^{(l)}$ 为切分的坐标轴，$l=j(\mathrm{mod}\ k)+1$，以该结点的区域中所有实例的 $x^{(l)}$ 坐标的中位数为切分点，将该结点对应的超矩形区域切分为两个子区域。切分由通过切分点并与坐标轴 $x^{(l)}$ 垂直的超平面实现。

由该结点生成深度为 $j+1$ 的左、右子结点：左子结点对应坐标 $x^{(l)}$ 小于切分点的子区域，右子结点对应坐标 $x^{(l)}$ 大于切分点的子区域。

将落在切分超平面上的实例点保存在该结点。

（3）直到两个子区域没有实例存在时停止，从而形成 $kd$ 树的区域划分。 ■

**例 3.2** 给定一个二维空间的数据集

$$T=\{(2,3)^{\mathrm{T}},(5,4)^{\mathrm{T}},(9,6)^{\mathrm{T}},(4,7)^{\mathrm{T}},(8,1)^{\mathrm{T}},(7,2)^{\mathrm{T}}\}$$

构造一个平衡 $kd$ 树[②]。

① 一组数据按大小顺序排列起来，处在中间位置的一个数或最中间两个数的平均值。

② 取自 Wikipedia。

**解** 根结点对应包含数据集 $T$ 的矩形，选择 $x^{(1)}$ 轴，6 个数据点的 $x^{(1)}$ 坐标的中位数是 7 ①，以平面 $x^{(1)}=7$ 将空间分为左、右两个子矩形（子结点）；接着，左矩形以 $x^{(2)}=4$ 分为两个子矩形，右矩形以 $x^{(2)}=6$ 分为两个子矩形，如此递归，最后得到如图 3.3 所示的特征空间划分和如图 3.4 所示的 $kd$ 树。 ■

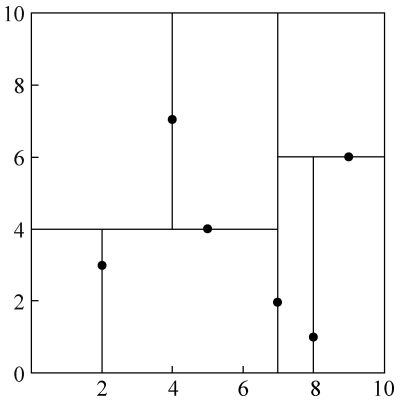


图 3.3 特征空间划分

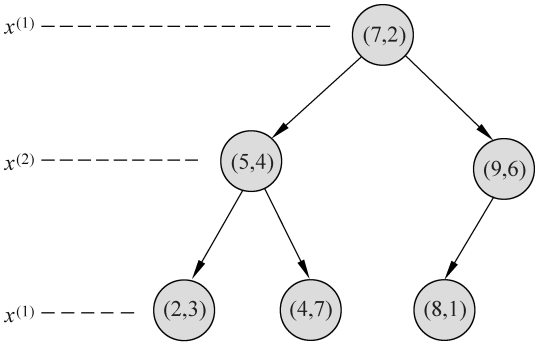


图 3.4 $kd$ 树示例

### 3.3.2 搜索 $kd$ 树

下面介绍如何利用 $kd$ 树进行 $k$ 近邻搜索。可以看到，利用 $kd$ 树可以省去对大部分数据点的搜索，从而减少搜索的计算量。这里以最近邻为例加以叙述，同样的方法可以应用到 $k$ 近邻。

给定一个目标点，搜索其最近邻。首先找到包含目标点的叶结点；然后从该叶结点出发，依次回退到父结点；不断查找与目标点最邻近的结点，当确定不可能存在更近的结点时终止。这样搜索就被限制在空间的局部区域上，效率大为提高。

包含目标点的叶结点对应包含目标点的最小超矩形区域。以此叶结点的实例点作为当前最近点，目标点的最近邻一定在以目标点为中心并通过当前最近点的超球体的内部（参阅

① $x^{(1)}=6$ 是中位数，但 $x^{(1)}=6$ 上没有数据点，故选 $x^{(1)}=7$。

图 3.5)。然后返回当前结点的父结点，如果父结点的另一子结点的超矩形区域与超球体相交，那么在相交的区域内寻找与目标点更近的实例点。如果存在这样的点，将此点作为新的当前最近点。算法转到更上一级的父结点，继续上述过程。如果父结点的另一子结点的超矩形区域与超球体不相交，或不存在比当前最近点更近的点，则停止搜索。

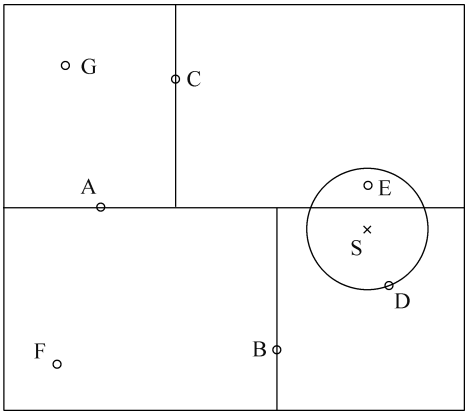


图 3.5　通过 $kd$ 树搜索最近邻

下面叙述用 $kd$ 树的最近邻搜索算法。

**算法 3.3（用 $kd$ 树的最近邻搜索）**

输入：已构造的 $kd$ 树，目标点 $x$。

输出：$x$ 的最近邻。

（1）在 $kd$ 树中找出包含目标点 $x$ 的叶结点：从根结点出发，递归地向下访问 $kd$ 树。若目标点 $x$ 当前维的坐标小于切分点的坐标，则移动到左子结点，否则移动到右子结点，直到子结点为叶结点为止。

（2）以此叶结点为“当前最近点”。

（3）递归地向上回退，在每个结点进行以下操作：

（a）如果该结点保存的实例点比当前最近点距离目标点更近，则以该实例点为“当前最近点”。

（b）当前最近点一定存在于该结点一个子结点对应的区域。检查该子结点的父结点的另一子结点对应的区域是否有更近的点。具体地，检查另一子结点对应的区域是否与以目标点为球心、以目标点与“当前最近点”间的距离为半径的超球体相交。如果相交，可能在另一个子结点对应的区域内存在距目标点更近的点，移动到另一个子结点。接着，递归地进行最近邻搜索。如果不相交，向上回退。

（4）当回退到根结点时，搜索结束。最后的“当前最近点”即为 $x$ 的最近邻点。 ■

如果实例点是随机分布的，$kd$ 树搜索的平均计算复杂度是 $O(\log N)$，这里 $N$ 是训练实例数。$kd$ 树更适用于训练实例数远大于空间维数时的 $k$ 近邻搜索。当空间维数接近训练实例数时，它的效率会迅速下降，几乎接近线性扫描。

下面通过一个例题来说明搜索方法。

**例 3.3**　给定一个如图 3.5 所示的 $kd$ 树，根结点为 A，其子结点为 B，C 等。树上共存储 7 个实例点，另有一个输入目标实例点 S，求 S 的最近邻。

**解** 首先在 $kd$ 树中找到包含点 S 的叶结点 D（图中的右下区域），以点 D 作为近似最近邻。真正最近邻一定在以点 S 为中心通过点 D 的圆的内部。然后返回结点 D 的父结点 B，在结点 B 的另一子结点 F 的区域内搜索最近邻。结点 F 的区域与圆不相交，不可能有最近邻点。继续返回上一级父结点 A，在结点 A 的另一子结点 C 的区域内搜索最近邻。结点 C 的区域与圆相交，该区域在圆内的实例点有点 E，点 E 比点 D 更近，成为新的最近邻近似。最后得到点 E 是点 S 的最近邻。 ■

## 本章概要

1. $k$ 近邻法是基本且简单的分类与回归方法。$k$ 近邻法的基本做法是：对给定的训练实例点和输入实例点，首先确定输入实例点的 $k$ 个最近邻训练实例点，然后利用这 $k$ 个训练实例点的类的多数来预测输入实例点的类。

2. $k$ 近邻模型对应于基于训练数据集对特征空间的一个划分。$k$ 近邻法中，当训练集、距离度量、$k$ 值及分类决策规则确定后，其结果唯一确定。

3. $k$ 近邻法三要素：距离度量、$k$ 值的选择和分类决策规则。常用的距离度量是欧氏距离及更一般的 $L_p$ 距离。$k$ 值小时，$k$ 近邻模型更复杂；$k$ 值大时，$k$ 近邻模型更简单。$k$ 值的选择反映了对近似误差与估计误差之间的权衡，通常由交叉验证选择最优的 $k$。常用的分类决策规则是多数表决，对应于经验风险最小化。

4. $k$ 近邻法的实现需要考虑如何快速搜索 $k$ 个最近邻点。$kd$ 树是一种便于对 $k$ 维空间中的数据进行快速检索的数据结构。$kd$ 树是二叉树，表示对 $k$ 维空间的一个划分，其每个结点对应于 $k$ 维空间划分中的一个超矩形区域。利用 $kd$ 树可以省去对大部分数据点的搜索，从而减少搜索的计算量。

## 继续阅读

$k$ 近邻法由 Cover 与 Hart 提出 [1]。$k$ 近邻法相关的理论在文献 [2] 和文献 [3] 中已有论述。$k$ 近邻法的扩展可参考文献 [4]。$kd$ 树及其他快速搜索算法可参考文献 [5]。关于 $k$ 近邻法的介绍可参考文献 [2]。

## 习 题

**3.1** 参照图 3.1，在二维空间中给出实例点，画出 $k$ 为 1 和 2 时的 $k$ 近邻法构成的空间划分，并对其进行比较，体会 $k$ 值选择与模型复杂度及预测精确率的关系。

**3.2** 利用例题 3.2 构造的 $kd$ 树求点 $x = (3, 4.5)^{\mathrm{T}}$ 的最近邻点。

**3.3** 参照算法 3.3，写出输出为 $x$ 的 $k$ 近邻的算法。

# 参考文献

[1] COVER T, HART P. Nearest neighbor pattern classification[J]. IEEE Transactions on Information Theory, 1967, 13(1): 21–27.

[2] HASTIE T, TIBSHIRANI R, FRIEDMAN J. The elements of statistical learning: data mining, inference, and prediction[M]. 范明，柴玉梅，昝红英，等译. Springer, 2001.

[3] FRIEDMAN J. Flexible metric nearest neighbor classification[J]. Technical Report, 1994.

[4] WEINBERGER K Q, BLITZER J, SAUL L K. Distance metric learning for large margin nearest neighbor classification[C]//Proceedings of the NIPS. 2005.

[5] SAMET H. The design and analysis of spatial data structures[M]. Reading, MA: Addison-Wesley, 1990.

# 第 4 章　朴素贝叶斯法

朴素贝叶斯（naïve Bayes）法是基于贝叶斯定理与特征条件独立假设的分类方法[①]。对于给定的训练数据集，首先基于特征条件独立假设学习输入输出的联合概率分布；然后基于此模型，对给定的输入 $x$，利用贝叶斯定理求出后验概率最大的输出 $y$。朴素贝叶斯法实现简单，学习与预测的效率都很高，是一种常用的方法。

本章叙述朴素贝叶斯法，包括朴素贝叶斯法的学习与分类、朴素贝叶斯法的参数估计算法。

## 4.1　朴素贝叶斯法的学习与分类

### 4.1.1　基本方法

设输入空间 $\mathcal{X} \subseteq \boldsymbol{R}^n$ 为 $n$ 维向量的集合，输出空间为类标记集合 $\mathcal{Y} = \{c_1, c_2, \cdots, c_K\}$。输入为特征向量 $x \in \mathcal{X}$，输出为类标记（class label）$y \in \mathcal{Y}$。$X$ 是定义在输入空间 $\mathcal{X}$ 上的随机向量，$Y$ 是定义在输出空间 $\mathcal{Y}$ 上的随机变量。$P(X, Y)$ 是 $X$ 和 $Y$ 的联合概率分布。训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

由 $P(X, Y)$ 独立同分布产生。

朴素贝叶斯法通过训练数据集学习联合概率分布 $P(X, Y)$。具体地，学习以下先验概率分布及条件概率分布。先验概率分布

$$P(Y = c_k), \quad k = 1, 2, \cdots, K \tag{4.1}$$

条件概率分布

$$P(X = x|Y = c_k) = P(X^{(1)} = x^{(1)}, \cdots, X^{(n)} = x^{(n)}|Y = c_k), \quad k = 1, 2, \cdots, K \tag{4.2}$$

于是学习到联合概率分布 $P(X, Y)$。

条件概率分布 $P(X = x|Y = c_k)$ 有指数级数量的参数，其估计实际是不可行的。事实上，假设 $x^{(j)}$ 可取值有 $S_j$ 个，$j = 1, 2, \cdots, n$，$Y$ 可取值有 $K$ 个，那么参数个数为 $K\prod_{j=1}^{n} S_j$。

① 注意：朴素贝叶斯法与贝叶斯估计（Bayesian estimation）是不同的概念。

朴素贝叶斯法对条件概率分布作了条件独立性的假设。由于这是一个较强的假设，朴素贝叶斯法也由此得名。具体地，条件独立性假设是

$$\begin{aligned} P(X=x|Y=c_k) &= P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)}|Y=c_k) \\ &= \prod_{j=1}^{n} P(X^{(j)}=x^{(j)}|Y=c_k) \end{aligned} \tag{4.3}$$

朴素贝叶斯法实际上学习到生成数据的机制，所以属于生成模型。条件独立假设等于是说用于分类的特征在类确定的条件下都是条件独立的。这一假设使朴素贝叶斯法变得简单，但有时会牺牲一定的分类准确率。

朴素贝叶斯法分类时，对给定的输入 $x$，通过学习到的模型计算后验概率分布 $P(Y=c_k|X=x)$，将后验概率最大的类作为 $x$ 的类输出。后验概率计算根据贝叶斯定理进行：

$$P(Y=c_k|X=x)=\frac{P(X=x|Y=c_k)P(Y=c_k)}{\sum_k P(X=x|Y=c_k)P(Y=c_k)} \tag{4.4}$$

将式 (4.3) 代入式 (4.4)，有

$$P(Y=c_k|X=x)=\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)}|Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)}|Y=c_k)},\quad k=1,2,\cdots,K \tag{4.5}$$

这是朴素贝叶斯法分类的基本公式。于是，朴素贝叶斯分类器可表示为

$$y=f(x)=\arg\max_{c_k}\frac{P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)}|Y=c_k)}{\sum_k P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)}|Y=c_k)} \tag{4.6}$$

注意到，在式 (4.6) 中分母对所有 $c_k$ 都是相同的，所以，

$$y=\arg\max_{c_k} P(Y=c_k)\prod_j P(X^{(j)}=x^{(j)}|Y=c_k) \tag{4.7}$$

### 4.1.2 后验概率最大化的含义

朴素贝叶斯法将实例分到后验概率最大的类中，这等价于期望风险最小化。假设选择 0-1 损失函数：

$$L(Y,f(X))=\begin{cases}1, & Y\neq f(X)\\ 0, & Y=f(X)\end{cases}$$

式中 $f(X)$ 是分类决策函数。这时，期望风险函数为

$$R_{\exp}(f)=E[L(Y,f(X))]$$

期望是对联合分布 $P(X,Y)$ 取的。由此取条件期望

$$R_{\exp}(f)=E_X\sum_{k=1}^{K}[L(c_k,f(X))]P(c_k|X)$$

为了使期望风险最小化，只需对 $X=x$ 逐个极小化，由此得到：

$$\begin{aligned}f(x)&=\arg\min_{y\in\mathcal{Y}}\sum_{k=1}^{K}L(c_k,y)P(c_k|X=x)\\&=\arg\min_{y\in\mathcal{Y}}\sum_{k=1}^{K}P(y\neq c_k|X=x)\\&=\arg\min_{y\in\mathcal{Y}}(1-P(y=c_k|X=x))\\&=\arg\max_{y\in\mathcal{Y}}P(y=c_k|X=x)\end{aligned}$$

这样一来，根据期望风险最小化准则就得到了后验概率最大化准则：

$$f(x)=\arg\max_{c_k}P(c_k|X=x)$$

即朴素贝叶斯法所采用的原理。

## 4.2 朴素贝叶斯法的参数估计

### 4.2.1 极大似然估计

在朴素贝叶斯法中，学习意味着估计 $P(Y=c_k)$ 和 $P(X^{(j)}=x^{(j)}|Y=c_k)$。可以应用极大似然估计法估计相应的概率。先验概率 $P(Y=c_k)$ 的极大似然估计是

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},\quad k=1,2,\cdots,K \tag{4.8}$$

设第 $j$ 个特征 $x^{(j)}$ 可能取值的集合为 $\{a_{j1},a_{j2},\cdots,a_{jS_j}\}$，条件概率 $P(X^{(j)}=a_{jl}|Y=c_k)$ 的极大似然估计是

$$P(X^{(j)}=a_{jl}|Y=c_k)=\frac{\sum_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)}{\sum_{i=1}^{N}I(y_i=c_k)},$$

$$j=1,2,\cdots,n,\quad l=1,2,\cdots,S_j,\quad k=1,2,\cdots,K \tag{4.9}$$

式中，$x_i^{(j)}$ 是第 $i$ 个样本的第 $j$ 个特征；$a_{jl}$ 是第 $j$ 个特征可能取的第 $l$ 个值；$I$ 为指示函数。

### 4.2.2 学习与分类算法

下面给出朴素贝叶斯法的学习与分类算法。

**算法 4.1（朴素贝叶斯算法（naïve Bayes algorithm））**

输入：训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$，其中 $x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$，$x_i^{(j)}$ 是第 $i$ 个样本的第 $j$ 个特征，$x_i^{(j)}\in\{a_{j1},a_{j2},\cdots,a_{jS_j}\}$，$a_{jl}$ 是第 $j$ 个特征可能取的第 $l$ 个值，$j=1,2,\cdots,n$，$l=1,2,\cdots,S_j$，$y_i\in\{c_1,c_2,\cdots,c_K\}$；实例 $x$。

输出：实例 $x$ 的分类。

（1）计算先验概率及条件概率

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},\quad k=1,2,\cdots,K$$

$$P(X^{(j)}=a_{jl}|Y=c_k)=\frac{\sum_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)}{\sum_{i=1}^{N}I(y_i=c_k)},$$

$$j=1,2,\cdots,n,\quad l=1,2,\cdots,S_j,\quad k=1,2,\cdots,K$$

（2）对于给定的实例 $x=(x^{(1)},x^{(2)},\cdots,x^{(n)})^{\mathrm{T}}$，计算

$$P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}|Y=c_k),\quad k=1,2,\cdots,K$$

（3）确定实例 $x$ 的类

$$y=\arg\max_{c_k}P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}|Y=c_k)$$ ■

**例 4.1** 试由表 4.1 的训练数据学习一个朴素贝叶斯分类器并确定 $x=(2,S)^{\mathrm{T}}$ 的类标记 $y$。表中 $X^{(1)}$，$X^{(2)}$ 为特征，取值的集合分别为 $A_1=\{1,2,3\}$，$A_2=\{S,M,L\}$，$Y$ 为类标记，$Y\in C=\{1,-1\}$。

**表 4.1 训练数据**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $X^{(1)}$ | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |
| $X^{(2)}$ | $S$ | $M$ | $M$ | $S$ | $S$ | $S$ | $M$ | $M$ | $L$ | $L$ | $L$ | $M$ | $M$ | $L$ | $L$ |
| $Y$ | $-1$ | $-1$ | 1 | 1 | $-1$ | $-1$ | $-1$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $-1$ |

**解** 根据算法 4.1，由表 4.1 容易计算下列概率：

$$P(Y=1)=\frac{9}{15},\quad P(Y=-1)=\frac{6}{15}$$

$$P(X^{(1)}=1|Y=1)=\frac{2}{9},\quad P(X^{(1)}=2|Y=1)=\frac{3}{9},\quad P(X^{(1)}=3|Y=1)=\frac{4}{9}$$

$$P(X^{(2)}=S|Y=1)=\frac{1}{9},\quad P(X^{(2)}=M|Y=1)=\frac{4}{9},\quad P(X^{(2)}=L|Y=1)=\frac{4}{9}$$

$$P(X^{(1)}=1|Y=-1)=\frac{3}{6},\quad P(X^{(1)}=2|Y=-1)=\frac{2}{6},\quad P(X^{(1)}=3|Y=-1)=\frac{1}{6}$$

$$P(X^{(2)}=S|Y=-1)=\frac{3}{6},\quad P(X^{(2)}=M|Y=-1)=\frac{2}{6},\quad P(X^{(2)}=L|Y=-1)=\frac{1}{6}$$

对于给定的 $x=(2,S)^{\mathrm{T}}$，计算

$$P(Y=1)P(X^{(1)}=2|Y=1)P(X^{(2)}=S|Y=1)=\frac{9}{15}\cdot\frac{3}{9}\cdot\frac{1}{9}=\frac{1}{45}$$

$$P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1)=\frac{6}{15}\cdot\frac{2}{6}\cdot\frac{3}{6}=\frac{1}{15}$$

由于 $P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1)$ 最大，所以 $y=-1$。 ■

### 4.2.3 贝叶斯估计

用极大似然估计可能会出现所要估计的概率值为 0 的情况。这时会影响后验概率的计算结果，使分类产生偏差。解决这一问题的方法是采用贝叶斯估计。具体地，条件概率的贝叶斯估计是

$$P_\lambda(X^{(j)}=a_{jl}|Y=c_k)=\frac{\sum\limits_{i=1}^{N}I(x_i^{(j)}=a_{jl},y_i=c_k)+\lambda}{\sum\limits_{i=1}^{N}I(y_i=c_k)+S_j\lambda} \tag{4.10}$$

式中 $\lambda\geqslant 0$，等价于在随机变量各个取值的频数上赋予一个正数 $\lambda>0$。当 $\lambda=0$ 时就是极大似然估计。常取 $\lambda=1$，这时称为拉普拉斯平滑（Laplacian smoothing）。显然，对任何 $l=1,2,\cdots,S_j$，$k=1,2,\cdots,K$，有

$$P_\lambda(X^{(j)}=a_{jl}|Y=c_k)>0$$

$$\sum_{l=1}^{S_j}P_\lambda(X^{(j)}=a_{jl}|Y=c_k)=1$$

表明式 (4.10) 确为一种概率分布。同样, 先验概率的贝叶斯估计是

$$P_\lambda(Y=c_k)=\frac{\sum\limits_{i=1}^{N}I(y_i=c_k)+\lambda}{N+K\lambda} \tag{4.11}$$

**例 4.2** 问题同例 4.1，按照拉普拉斯平滑估计概率，即取 $\lambda=1$。

**解** $A_1=\{1,2,3\}$，$A_2=\{S,M,L\}$，$C=\{1,-1\}$。按照式 (4.10) 和式 (4.11) 计算下列

概率：

$$P(Y=1)=\frac{10}{17},\quad P(Y=-1)=\frac{7}{17}$$

$$P(X^{(1)}=1|Y=1)=\frac{3}{12},\quad P(X^{(1)}=2|Y=1)=\frac{4}{12},\quad P(X^{(1)}=3|Y=1)=\frac{5}{12}$$

$$P(X^{(2)}=S|Y=1)=\frac{2}{12},\quad P(X^{(2)}=M|Y=1)=\frac{5}{12},\quad P(X^{(2)}=L|Y=1)=\frac{5}{12}$$

$$P(X^{(1)}=1|Y=-1)=\frac{4}{9},\quad P(X^{(1)}=2|Y=-1)=\frac{3}{9},\quad P(X^{(1)}=3|Y=-1)=\frac{2}{9}$$

$$P(X^{(2)}=S|Y=-1)=\frac{4}{9},\quad P(X^{(2)}=M|Y=-1)=\frac{3}{9},\quad P(X^{(2)}=L|Y=-1)=\frac{2}{9}$$

对于给定的 $x=(2,S)^{\mathrm{T}}$，计算

$$P(Y=1)P(X^{(1)}=2|Y=1)P(X^{(2)}=S|Y=1)=\frac{10}{17}\cdot\frac{4}{12}\cdot\frac{2}{12}=\frac{5}{153}=0.0327$$

$$P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1)=\frac{7}{17}\cdot\frac{3}{9}\cdot\frac{4}{9}=\frac{28}{459}=0.0610$$

由于 $P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1)$ 最大，所以 $y=-1$。 ■

## 本章概要

1. 朴素贝叶斯法是典型的生成学习方法。生成方法由训练数据学习联合概率分布 $P(X,Y)$，然后求得后验概率分布 $P(Y|X)$。具体来说，利用训练数据学习 $P(X|Y)$ 和 $P(Y)$ 的估计，得到联合概率分布：

$$P(X,Y)=P(Y)P(X|Y)$$

概率估计方法可以是极大似然估计或贝叶斯估计。

2. 朴素贝叶斯法的基本假设是条件独立性：

$$\begin{aligned}P(X=x|Y=c_k)&=P(X^{(1)}=x^{(1)},\cdots,X^{(n)}=x^{(n)}|Y=c_k)\\&=\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}|Y=c_k)\end{aligned}$$

这是一个较强的假设。由于这一假设，模型包含的条件概率的数量大为减少，朴素贝叶斯法的学习与预测大为简化。因而朴素贝叶斯法高效且易于实现，其缺点是分类的性能不一定很高。

3. 朴素贝叶斯法利用贝叶斯定理与学到的联合概率模型进行分类预测。

$$P(Y|X)=\frac{P(X,Y)}{P(X)}=\frac{P(Y)P(X|Y)}{\sum\limits_{Y}P(Y)P(X|Y)}$$

将输入 $x$ 分到后验概率最大的类 $y$。

$$y = \arg\max_{c_k} P(Y = c_k) \prod_{j=1}^{n} P(X_j = x^{(j)} | Y = c_k)$$

后验概率最大等价于 0-1 损失函数时的期望风险最小化。

## 继续阅读

朴素贝叶斯法的介绍可见文献 [1] 和文献 [2]。朴素贝叶斯法中假设输入变量都是条件独立的，如果假设它们之间存在概率依存关系，模型就变成了贝叶斯网络，参见文献 [3]。

## 习 题

**4.1** 用极大似然估计法推出朴素贝叶斯法中的概率估计公式 (4.8) 及公式 (4.9)。

**4.2** 用贝叶斯估计法推出朴素贝叶斯法中的概率估计公式 (4.10) 及公式 (4.11)。

## 参考文献

[1] MITCHELL T M. Machine Learning[M]. Engineering, 2005.

[2] HASTIE T, TIBSHIRANI R, FRIEDMAN J. The elements of statistical learning: data mining, inference, and prediction[M]. 范明，柴玉梅，昝红英，等译. Springer, 2001.

[3] BISHOP C. Pattern recognition and machine learning[M]. Springer, 2006.

# 第 5 章　决　策　树

决策树（decision tree）是一种基本的分类与回归方法。本章主要讨论用于分类的决策树。决策树模型呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程。它可以认为是 if-then 规则的集合，也可以认为是定义在特征空间与类空间上的条件概率分布。其主要优点是模型具有可读性，分类速度快。学习时，利用训练数据，根据损失函数最小化的原则建立决策树模型。预测时，对新的数据利用决策树模型进行分类。决策树学习通常包括 3 个步骤：特征选择、决策树的生成和决策树的修剪。这些决策树学习的思想主要来源于由 Quinlan 在 1986 年提出的 ID3 算法和 1993 年提出的 C4.5 算法，以及由 Breiman 等人在 1984 年提出的 CART 算法。

本章首先介绍决策树的基本概念，然后通过 ID3 算法和 C4.5 算法介绍特征的选择、决策树的生成以及决策树的修剪，最后介绍 CART 算法。

## 5.1　决策树模型与学习

### 5.1.1　决策树模型

**定义 5.1（决策树）**　分类决策树模型是一种描述对实例进行分类的树形结构。决策树由结点（node）和有向边（directed edge）组成。结点有两种类型：内部结点（internal node）和叶结点（leaf node）。内部结点表示一个特征或属性，叶结点表示一个类。

用决策树分类，从根结点开始，对实例的某一特征进行测试，根据测试结果，将实例分配到其子结点，这时，每一个子结点对应该特征的一个取值。如此递归地对实例进行测试并分配，直至达到叶结点。最后将实例分到叶结点的类中。

图 5.1 是一个决策树模型，图中圆和方框分别表示内部结点和叶结点。

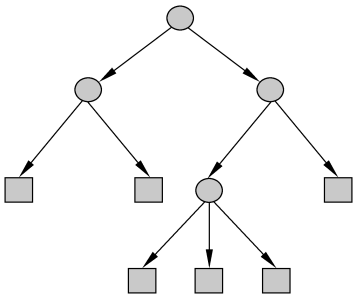

图 5.1　决策树模型

### 5.1.2 决策树与 if-then 规则

可以将决策树看成一个 if-then 规则的集合。将决策树转换成 if-then 规则的过程如下：由决策树的根结点到叶结点的每一条路径构建一条规则；路径上内部结点的特征对应规则的条件，而叶结点的类对应规则的结论。决策树的路径或其对应的 if-then 规则集合具有一个重要的性质：互斥并且完备。这就是说，每一个实例都被一条路径或一条规则所覆盖，而且只被一条路径或一条规则所覆盖。这里所谓覆盖是指实例的特征与路径上的特征一致或实例满足规则的条件。

### 5.1.3 决策树与条件概率分布

决策树还表示给定特征条件下类的条件概率分布，这一条件概率分布定义在特征空间的一个划分（partition）上。将特征空间划分为互不相交的单元（cell）或区域（region），并在每个单元定义一个类的概率分布就构成了一个条件概率分布。决策树的一条路径对应于划分中的一个单元。决策树所表示的条件概率分布由各个单元给定条件下类的条件概率分布组成。假设 $X$ 为表示特征的随机变量，$Y$ 为表示类的随机变量，那么这个条件概率分布可以表示为 $P(Y|X)$。$X$ 取值于给定划分下单元的集合，$Y$ 取值于类的集合。各叶结点（单元）上的条件概率往往偏向某一个类，即属于某一类的概率较大。决策树分类时将该结点的实例强行分到条件概率大的那一类去。

图 5.2（a）示意地表示特征空间的一个划分。图中的大正方形表示特征空间。这个大正方形被若干个小矩形分割，每个小矩形表示一个单元。特征空间划分上的单元构成了一个集合，$X$ 取值为单元的集合。为简单起见，假设只有两类：正类和负类，即 $Y$ 取值为 +1 和 –1。小矩形中的数字表示单元的类。图 5.2（b）示意地表示特征空间划分确定时，特征（单元）给定条件下类的条件概率分布。图 5.2（b）中条件概率分布对应于图 5.2（a）的划分。当某个单元 $c$ 的条件概率满足 $P(Y=+1|X=c)>0.5$ 时，则认为这个单元属于正类，即落在这个单元的实例都被视为正例。图 5.2（c）为对应于图 5.2（b）中条件概率分布的决策树。

### 5.1.4 决策树学习

假设给定训练数据集

$$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$$

其中，$x_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$ 为输入实例（特征向量），$n$ 为特征个数，$y_i\in\{1,2,\cdots,K\}$ 为类标记，$i=1,2,\cdots,N$，$N$ 为样本容量。决策树学习的目标是根据给定的训练数据集构建一个决策树模型，使它能够对实例进行正确的分类。

决策树学习本质上是从训练数据集中归纳出一组分类规则。与训练数据集不相矛盾的决策树（即能对训练数据进行正确分类的决策树）可能有多个，也可能一个都没有。我们需要的是一个与训练数据矛盾较小的决策树，同时具有很好的泛化能力。从另一个角度看，决策树学习是由训练数据集估计条件概率模型。基于特征空间划分的类的条件概率模型有无穷多

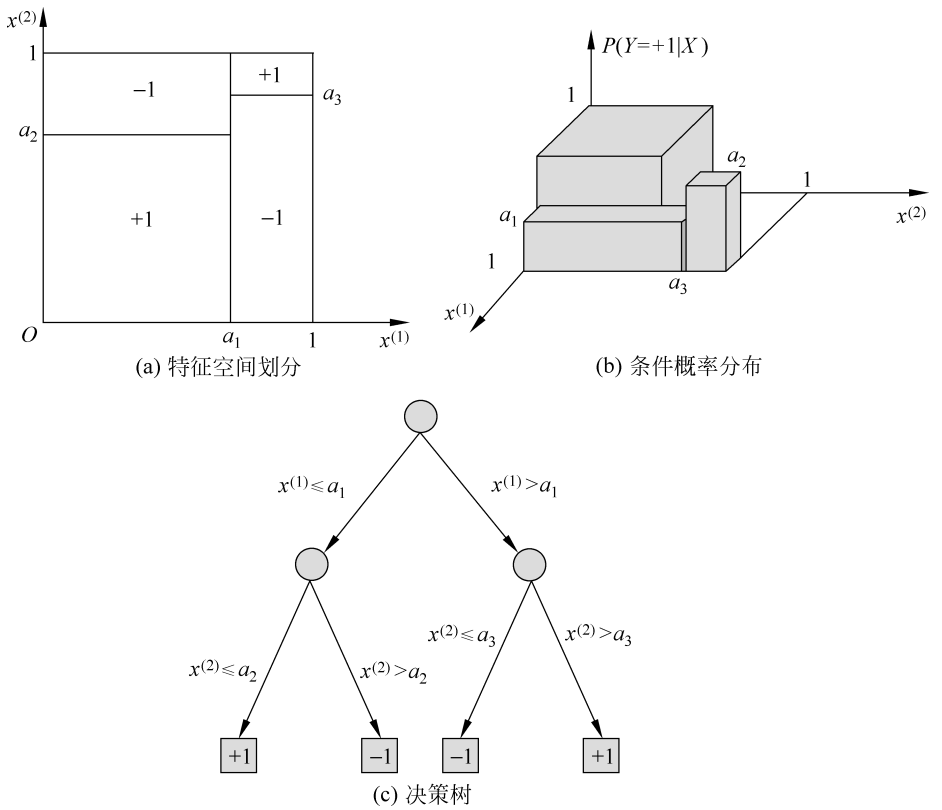


图 5.2 决策树对应于条件概率分布

个。我们选择的条件概率模型应该不仅对训练数据有很好的拟合，而且对未知数据有很好的预测。

决策树学习用损失函数表示这一目标。如下所述，决策树学习的损失函数通常是正则化的极大似然函数。决策树学习的策略是以损失函数为目标函数的最小化。

当损失函数确定以后，学习问题就变为在损失函数意义下选择最优决策树的问题。因为从所有可能的决策树中选取最优决策树是 NP 完全问题，所以现实中决策树学习算法通常采用启发式方法，近似求解这一最优化问题。这样得到的决策树是次最优（sub-optimal）的。

决策树学习的算法通常是一个递归地选择最优特征，并根据该特征对训练数据进行分割，使得对各个子数据集有一个最好的分类的过程。这一过程对应对特征空间的划分，也对应决策树的构建。首先，构建根结点，将所有训练数据都放在根结点。选择一个最优特征，按照这一特征将训练数据集分割成子集，使得各个子集有一个在当前条件下最好的分类。如果这些子集已经能够被基本正确分类，那么构建叶结点，并将这些子集分到所对应的叶结点中去；如果还有子集不能被基本正确分类，那么就对这些子集选择新的最优特征，继续对其进行分割，构建相应的结点。如此递归地进行下去，直至所有训练数据子集被基本正确分类，或者没有合适的特征为止。最后每个子集都被分到叶结点上，即都有了明确的类。这就生成了一棵决策树。

以上方法生成的决策树可能对训练数据有很好的分类能力，但对未知的测试数据未必有

很好的分类能力，即可能发生过拟合现象。我们需要对已生成的树自下而上进行剪枝，将树变得更简单，从而使它具有更好的泛化能力。具体地，就是去掉过于细分的叶结点，使其回退到父结点，甚至更高的结点，然后将父结点或更高的结点改为新的叶结点。

如果特征数量很多，也可以在决策树学习开始的时候对特征进行选择，只留下对训练数据有足够分类能力的特征。

可以看出，决策树学习算法包含特征选择、决策树的生成与决策树的剪枝过程。由于决策树表示一个条件概率分布，所以深浅不同的决策树对应不同复杂度的概率模型。决策树的生成对应于模型的局部选择，决策树的剪枝对应于模型的全局选择。决策树的生成只考虑局部最优，相对地，决策树的剪枝则考虑全局最优。

决策树学习常用的算法有 ID3、C4.5 与 CART，下面结合这些算法分别叙述决策树学习的特征选择、决策树的生成和剪枝过程。

## 5.2 特征选择

### 5.2.1 特征选择问题

特征选择在于选取对训练数据具有分类能力的特征，这样可以提高决策树学习的效率。如果利用一个特征进行分类的结果与随机分类的结果没有很大差别，则称这个特征是没有分类能力的。经验上扔掉这样的特征对决策树学习的精度影响不大。通常特征选择的准则是信息增益或信息增益比。

首先通过一个例子来说明特征选择问题。

**例 5.1**[①] 表 5.1 是一个由 15 个样本组成的贷款申请训练数据。数据包括贷款申请人的 4 个特征（属性）：第 1 个特征是年龄，有 3 个可能值：青年，中年，老年；第 2 个特征是有工作，有两个可能值：是，否；第 3 个特征是有自己的房子，有两个可能值：是，否；第 4 个特征是信贷情况，有 3 个可能值：非常好，好，一般。表的最后一列是类别，是否同意贷款，取两个值：是，否。

希望通过所给的训练数据学习一个贷款申请的决策树，用以对未来的贷款申请进行分类，即当新的客户提出贷款申请时，根据申请人的特征利用决策树决定是否批准贷款申请。■

特征选择是决定用哪个特征来划分特征空间。

图 5.3 表示从表 5.1 数据学习到的两个可能的决策树，分别由两个不同特征的根结点构成。图 5.3（a）所示的根结点的特征是年龄，有 3 个取值，对应于不同的取值有不同的子结点。图 5.3（b）所示的根结点的特征是有工作，有两个取值，对应于不同的取值有不同的子结点。两个决策树都可以从此延续下去。问题是：究竟选择哪个特征更好些？这就要求确定选择特征的准则。直观上，如果一个特征具有更好的分类能力，或者说，按照这一特征将训练数据集分割成子集，使得各个子集在当前条件下有最好的分类，那么就更应该选择这个特征。信息增益（information gain）就能够很好地表示这一直观的准则。

① 此例取自参考文献 [5]。

**表 5.1 贷款申请样本数据表**

| ID | 年龄 | 有工作 | 有自己的房子 | 信贷情况 | 类别 |
|---|---|---|---|---|---|
| 1 | 青年 | 否 | 否 | 一般 | 否 |
| 2 | 青年 | 否 | 否 | 好 | 否 |
| 3 | 青年 | 是 | 否 | 好 | 是 |
| 4 | 青年 | 是 | 是 | 一般 | 是 |
| 5 | 青年 | 否 | 否 | 一般 | 否 |
| 6 | 中年 | 否 | 否 | 一般 | 否 |
| 7 | 中年 | 否 | 否 | 好 | 否 |
| 8 | 中年 | 是 | 是 | 好 | 是 |
| 9 | 中年 | 否 | 是 | 非常好 | 是 |
| 10 | 中年 | 否 | 是 | 非常好 | 是 |
| 11 | 老年 | 否 | 是 | 非常好 | 是 |
| 12 | 老年 | 否 | 是 | 好 | 是 |
| 13 | 老年 | 是 | 否 | 好 | 是 |
| 14 | 老年 | 是 | 否 | 非常好 | 是 |
| 15 | 老年 | 否 | 否 | 一般 | 否 |

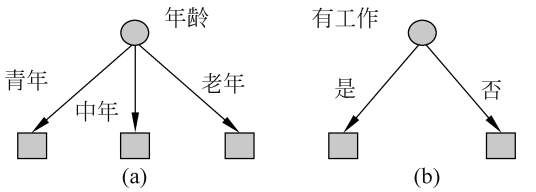


图 5.3 不同特征决定的不同决策树

### 5.2.2 信息增益

为了便于说明，先给出熵与条件熵的定义。

在信息论与概率统计中，熵（entropy）是表示随机变量不确定性的度量。设 $X$ 是一个取有限个值的离散随机变量，其概率分布为

$$P(X = x_i) = p_i, \quad i = 1, 2, \cdots, n$$

则随机变量 $X$ 的熵定义为

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i \tag{5.1}$$

在式 (5.1) 中，若 $p_i = 0$，则定义 $0 \log 0 = 0$。通常，式 (5.1) 中的对数以 2 为底或以 e 为底（自然对数），这时熵的单位分别称作比特（bit）或纳特（nat）。由定义可知，熵只依赖于 $X$ 的分布，而与 $X$ 的取值无关，所以也可将 $X$ 的熵记作 $H(p)$，即

$$H(p) = -\sum_{i=1}^{n} p_i \log p_i \tag{5.2}$$

熵越大，随机变量的不确定性就越大。从定义可验证

$$0 \leqslant H(p) \leqslant \log n \tag{5.3}$$

当随机变量只取两个值，如 1，0 时，即 $X$ 的分布为

$$P(X=1)=p, \quad P(X=0)=1-p, \quad 0 \leqslant p \leqslant 1$$

熵为

$$H(p)=-p\log_2 p-(1-p)\log_2(1-p) \tag{5.4}$$

这时，熵 $H(p)$ 随概率 $p$ 变化的曲线如图 5.4 所示（单位为比特）。

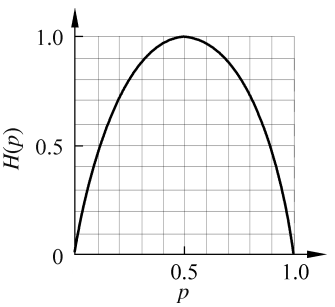


图 5.4 分布为伯努利分布时熵与概率的关系

当 $p=0$ 或 $p=1$ 时 $H(p)=0$，随机变量完全没有不确定性。当 $p=0.5$ 时，$H(p)=1$，熵取值最大，随机变量不确定性最大。

设有随机变量 $(X,Y)$，其联合概率分布为

$$P(X=x_i,Y=y_j)=p_{ij}, \quad i=1,2,\cdots,n, \quad j=1,2,\cdots,m$$

条件熵 $H(Y|X)$ 表示在已知随机变量 $X$ 的条件下随机变量 $Y$ 的不确定性。随机变量 $X$ 给定的条件下随机变量 $Y$ 的条件熵（conditional entropy）$H(Y|X)$ 定义为 $X$ 给定条件下 $Y$ 的条件概率分布的熵对 $X$ 的数学期望：

$$H(Y|X)=\sum_{i=1}^{n} p_i H(Y|X=x_i) \tag{5.5}$$

这里，$p_i=P(X=x_i)$，$i=1,2,\cdots,n$。

当熵和条件熵中的概率由数据估计（特别是极大似然估计）得到时，所对应的熵与条件熵分别称为经验熵（empirical entropy）和经验条件熵（empirical conditional entropy）。此时，如果有 0 概率，令 $0\log 0=0$。

信息增益（information gain）表示得知特征 $X$ 的信息而使得类 $Y$ 的信息的不确定性减少的程度。

**定义 5.2（信息增益）** 特征 $A$ 对训练数据集 $D$ 的信息增益 $g(D,A)$ 定义为集合 $D$ 的经验熵 $H(D)$ 与特征 $A$ 给定条件下 $D$ 的经验条件熵 $H(D|A)$ 之差，即

$$g(D,A)=H(D)-H(D|A) \tag{5.6}$$

一般地，熵 $H(Y)$ 与条件熵 $H(Y|X)$ 之差称为互信息（mutual information）。决策树学习中的信息增益等价于训练数据集中类与特征的互信息。

决策树学习应用信息增益准则选择特征。给定训练数据集 $D$ 和特征 $A$，经验熵 $H(D)$ 表示对数据集 $D$ 进行分类的不确定性。而经验条件熵 $H(D|A)$ 表示在特征 $A$ 给定的条件下对数据集 $D$ 进行分类的不确定性。那么它们的差，即信息增益，就表示由于特征 $A$ 而使得对数据集 $D$ 的分类的不确定性减少的程度。显然，对于数据集 $D$ 而言，信息增益依赖于特征，不同的特征往往具有不同的信息增益。信息增益大的特征具有更强的分类能力。

根据信息增益准则的特征选择方法是：对训练数据集（或子集）$D$，计算其每个特征的信息增益，并比较它们的大小，选择信息增益最大的特征。

设训练数据集为 $D$，$|D|$ 表示其样本容量，即样本个数。设有 $K$ 个类 $C_k$，$k=1,2,\cdots,K$，$|C_k|$ 为属于类 $C_k$ 的样本个数，$\sum_{k=1}^{K}|C_k| = |D|$。设特征 $A$ 有 $n$ 个不同的取值 $\{a_1,a_2,\cdots,a_n\}$，根据特征 $A$ 的取值将 $D$ 划分为 $n$ 个子集 $D_1,D_2,\cdots,D_n$，$|D_i|$ 为 $D_i$ 的样本个数，$\sum_{i=1}^{n}|D_i| = |D|$。记子集 $D_i$ 中属于类 $C_k$ 的样本的集合为 $D_{ik}$，即 $D_{ik}=D_i\cap C_k$，$|D_{ik}|$ 为 $D_{ik}$ 的样本个数。于是信息增益的算法如下。

**算法 5.1（信息增益的算法）**

输入：训练数据集 $D$ 和特征 $A$。

输出：特征 $A$ 对训练数据集 $D$ 的信息增益 $g(D,A)$。

（1）计算数据集 $D$ 的经验熵 $H(D)$

$$H(D)=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}\log_2\frac{|C_k|}{|D|} \tag{5.7}$$

（2）计算特征 $A$ 对数据集 $D$ 的经验条件熵 $H(D|A)$

$$H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|} \tag{5.8}$$

（3）计算信息增益

$$g(D,A)=H(D)-H(D|A) \tag{5.9}$$

■

**例 5.2** 对表 5.1 所给的训练数据集 $D$，根据信息增益准则选择最优特征。

**解** 首先计算经验熵 $H(D)$：

$$H(D)=-\frac{9}{15}\log_2\frac{9}{15}-\frac{6}{15}\log_2\frac{6}{15}=0.971$$

然后计算各特征对数据集 $D$ 的信息增益。分别以 $A_1$，$A_2$，$A_3$，$A_4$ 表示年龄、有工作、有自己的房子和信贷情况 4 个特征，则

$$
\begin{aligned}
g(D, A_1) &= H(D) - \left[\frac{5}{15}H(D_1) + \frac{5}{15}H(D_2) + \frac{5}{15}H(D_3)\right] \\
&= 0.971 - \left[\frac{5}{15}\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) + \right. \\
&\quad \left. \frac{5}{15}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) + \frac{5}{15}\left(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}\right)\right] \\
&= 0.971 - 0.888 = 0.083
\end{aligned}
$$

这里 $D_1$，$D_2$，$D_3$ 分别是 $D$ 中 $A_1$（年龄）取值为青年、中年和老年的样本子集。类似地，

$$
\begin{aligned}
g(D, A_2) &= H(D) - \left[\frac{5}{15}H(D_1) + \frac{10}{15}H(D_2)\right] \\
&= 0.971 - \left[\frac{5}{15}\times 0 + \frac{10}{15}\left(-\frac{4}{10}\log_2\frac{4}{10} - \frac{6}{10}\log_2\frac{6}{10}\right)\right] = 0.324 \\
g(D, A_3) &= 0.971 - \left[\frac{6}{15}\times 0 + \frac{9}{15}\left(-\frac{3}{9}\log_2\frac{3}{9} - \frac{6}{9}\log_2\frac{6}{9}\right)\right] \\
&= 0.971 - 0.551 = 0.420 \\
g(D, A_4) &= 0.971 - 0.608 = 0.363
\end{aligned}
$$

最后，比较各特征的信息增益值。由于特征 $A_3$（有自己的房子）的信息增益值最大，所以选择特征 $A_3$ 作为最优特征。 ■

### 5.2.3 信息增益比

以信息增益作为划分训练数据集的特征，存在偏向于选择取值较多的特征的问题。使用信息增益比（information gain ratio）可以对这一问题进行校正。这是特征选择的另一准则。

**定义 5.3（信息增益比）** 特征 $A$ 对训练数据集 $D$ 的信息增益比 $g_{\mathrm{R}}(D, A)$ 定义为其信息增益 $g(D, A)$ 与训练数据集 $D$ 关于特征 $A$ 的值的熵 $H_A(D)$ 之比，即

$$
g_{\mathrm{R}}(D, A) = \frac{g(D, A)}{H_A(D)} \tag{5.10}
$$

其中，$H_A(D) = -\sum_{i=1}^{n}\frac{|D_i|}{|D|}\log_2\frac{|D_i|}{|D|}$，$n$ 是特征 $A$ 取值的个数。

## 5.3 决策树的生成

本节将介绍决策树学习的生成算法。首先介绍 ID3 的生成算法，然后再介绍 C4.5 中的生成算法。这些都是决策树学习的经典算法。

### 5.3.1 ID3 算法

ID3 算法的核心是在决策树各个结点上应用信息增益准则选择特征，递归地构建决策树。具体方法是：从根结点（root node）开始，对结点计算所有可能的特征的信息增益，选择信息增益最大的特征作为结点的特征，由该特征的不同取值建立子结点；再对子结点递归地调用以上方法，构建决策树；直到所有特征的信息增益均很小或没有特征可以选择为止，最后得到一棵决策树。ID3 相当于用极大似然法进行概率模型的选择。

**算法 5.2（ID3 算法）**

输入：训练数据集 $D$，特征集 $A$ 阈值 $\varepsilon$。

输出：决策树 $T$。

（1）若 $D$ 中所有实例属于同一类 $C_k$，则 $T$ 为单结点树，并将类 $C_k$ 作为该结点的类标记，返回 $T$；

（2）若 $A=\varnothing$，则 $T$ 为单结点树，并将 $D$ 中实例数最大的类 $C_k$ 作为该结点的类标记，返回 $T$；

（3）否则，按算法 5.1 计算 $A$ 中各特征对 $D$ 的信息增益，选择信息增益最大的特征 $A_g$；

（4）如果 $A_g$ 的信息增益小于阈值 $\varepsilon$，则置 $T$ 为单结点树，并将 $D$ 中实例数最大的类 $C_k$ 作为该结点的类标记，返回 $T$；

（5）否则，对 $A_g$ 的每一可能值 $a_i$，依 $A_g=a_i$ 将 $D$ 分割为若干非空子集 $D_i$，将 $D_i$ 中实例数最大的类作为标记，构建子结点，由结点及其子结点构成树 $T$，返回 $T$；

（6）对第 $i$ 个子结点，以 $D_i$ 为训练集，以 $A-\{A_g\}$ 为特征集，递归地调用步骤 (1)～步骤 (5)，得到子树 $T_i$，返回 $T_i$。 ■

**例 5.3** 对表 5.1 的训练数据集，利用 ID3 算法建立决策树。

**解** 利用例 5.2 的结果，由于特征 $A_3$（有自己的房子）的信息增益值最大，所以选择特征 $A_3$ 作为根结点的特征。它将训练数据集 $D$ 划分为两个子集 $D_1$（$A_3$ 取值为“是”）和 $D_2$（$A_3$ 取值为“否”）。由于 $D_1$ 只有同一类的样本点，所以它成为一个叶结点，结点的类标记为“是”。

对 $D_2$ 则需从特征 $A_1$（年龄），$A_2$（有工作）和 $A_4$（信贷情况）中选择新的特征。计算各个特征的信息增益：

$$
\begin{aligned}
g(D_2,A_1)&=H(D_2)-H(D_2|A_1)=0.918-0.667=0.251\\
g(D_2,A_2)&=H(D_2)-H(D_2|A_2)=0.918\\
g(D_2,A_4)&=H(D_2)-H(D_2|A_4)=0.474
\end{aligned}
$$

选择信息增益最大的特征 $A_2$（有工作）作为结点的特征。由于 $A_2$ 有两个可能取值，从这一结点引出两个子结点：一个对应“是”（有工作）的子结点，包含 3 个样本，它们属于同一类，所以这是一个叶结点，类标记为“是”；另一个是对应“否”（无工作）的子结点，包含 6 个样本，它们也属于同一类，所以这也是一个叶结点，类标记为“否”。

这样生成一棵如图 5.5 所示的决策树，该决策树只用了两个特征（有两个内部结点）。 ■

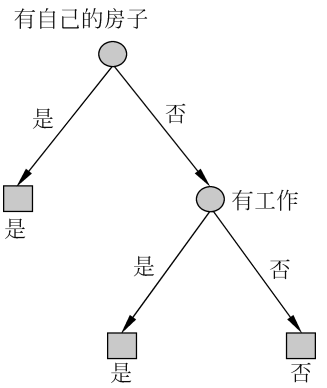


图 5.5 决策树的生成

ID3 算法只有树的生成，所以该算法生成的树容易产生过拟合。

### 5.3.2 C4.5 的生成算法

C4.5 算法与 ID3 算法相似，C4.5 算法对 ID3 算法进行了改进。C4.5 算法在生成的过程中，用信息增益比来选择特征。

**算法 5.3（C4.5 的生成算法）**

输入：训练数据集 $D$，特征集 $A$ 阈值 $\varepsilon$。

输出：决策树 $T$。

（1）如果 $D$ 中所有实例属于同一类 $C_k$，则置 $T$ 为单结点树，并将 $C_k$ 作为该结点的类，返回 $T$；

（2）如果 $A=\varnothing$，则置 $T$ 为单结点树，并将 $D$ 中实例数最大的类 $C_k$ 作为该结点的类，返回 $T$；

（3）否则，按式 (5.10) 计算 $A$ 中各特征对 $D$ 的信息增益比，选择信息增益比最大的特征$A_g$；

（4）如果 $A_g$ 的信息增益比小于阈值 $\varepsilon$，则置 $T$ 为单结点树，并将 $D$ 中实例数最大的类 $C_k$ 作为该结点的类，返回 $T$；

（5）否则，对 $A_g$ 的每一可能值 $a_i$，依 $A_g=a_i$ 将 $D$ 分割为若干非空子集 $D_i$，将 $D_i$ 中实例数最大的类作为标记，构建子结点，由结点及其子结点构成树 $T$，返回 $T$；

（6）对结点 $i$，以 $D_i$ 为训练集，以 $A-\{A_g\}$ 为特征集，递归地调用步骤 (1)～步骤 (5)，得到子树 $T_i$，返回 $T_i$。 ■

## 5.4 决策树的剪枝

决策树生成算法递归地产生决策树，直到不能继续下去为止。这样产生的树往往对训练数据的分类很准确，但对未知的测试数据的分类却没有那么准确，即出现过拟合现象。过拟合的原因在于学习时过多地考虑如何提高对训练数据的正确分类，从而构建出过于复杂的决策树。解决这个问题的办法是考虑决策树的复杂度，对已生成的决策树进行简化。

在决策树学习中将已生成的树进行简化的过程称为剪枝（pruning）。具体地，剪枝从已生成的树上裁掉一些子树或叶结点，并将其根结点或父结点作为新的叶结点，从而简化分类树模型。

本节介绍一种简单的决策树学习的剪枝算法。

决策树的剪枝往往通过极小化决策树整体的损失函数（loss function）或代价函数（cost function）来实现。设树 $T$ 的叶结点个数为 $|T|$，$t$ 是树 $T$ 的叶结点，该叶结点有 $N_t$ 个样本点，其中 $k$ 类的样本点有 $N_{tk}$ 个，$k=1,2,\cdots,K$，$H_t(T)$ 为叶结点 $t$ 上的经验熵，$\alpha \geqslant 0$ 为参数，则决策树学习的损失函数可以定义为

$$C_\alpha(T)=\sum_{t=1}^{|T|} N_t H_t(T)+\alpha|T| \tag{5.11}$$

其中经验熵为

$$H_t(T)=-\sum_k \frac{N_{tk}}{N_t}\log\frac{N_{tk}}{N_t} \tag{5.12}$$

在损失函数中，将式 (5.11) 右端的第 1 项记作

$$C(T)=\sum_{t=1}^{|T|} N_t H_t(T)=-\sum_{t=1}^{|T|}\sum_{k=1}^{K} N_{tk}\log\frac{N_{tk}}{N_t} \tag{5.13}$$

这时有

$$C_\alpha(T)=C(T)+\alpha|T| \tag{5.14}$$

式 (5.14) 中，$C(T)$ 表示模型对训练数据的预测误差，即模型与训练数据的拟合程度，$|T|$ 表示模型复杂度，参数 $\alpha \geqslant 0$ 控制两者之间的影响。较大的 $\alpha$ 促使选择较简单的模型（树），较小的 $\alpha$ 促使选择较复杂的模型（树）。$\alpha=0$ 意味着只考虑模型与训练数据的拟合程度，不考虑模型的复杂度。

剪枝就是当 $\alpha$ 确定时，选择损失函数最小的模型，即损失函数最小的子树。当 $\alpha$ 值确定时，子树越大，往往与训练数据的拟合越好，但是模型的复杂度就越高；相反，子树越小，模型的复杂度就越低，但是往往与训练数据的拟合不好。损失函数正好表示了对两者的平衡。

可以看出，决策树生成只考虑了通过提高信息增益（或信息增益比）对训练数据进行更好的拟合。而决策树剪枝通过优化损失函数还考虑了减小模型复杂度。决策树生成学习局部的模型，而决策树剪枝学习整体的模型。

式（5.11）或式（5.14）定义的损失函数的极小化等价于正则化的极大似然估计。所以，利用损失函数最小原则进行剪枝就是用正则化的极大似然估计进行模型选择。

图 5.6 表示决策树剪枝过程。下面介绍剪枝算法。

**算法 5.4（树的剪枝算法）**

输入：生成算法产生的整个树 $T$，参数 $\alpha$。

输出：修剪后的子树 $T_\alpha$。

（1）计算每个结点的经验熵。

（2）递归地从树的叶结点向上回缩。

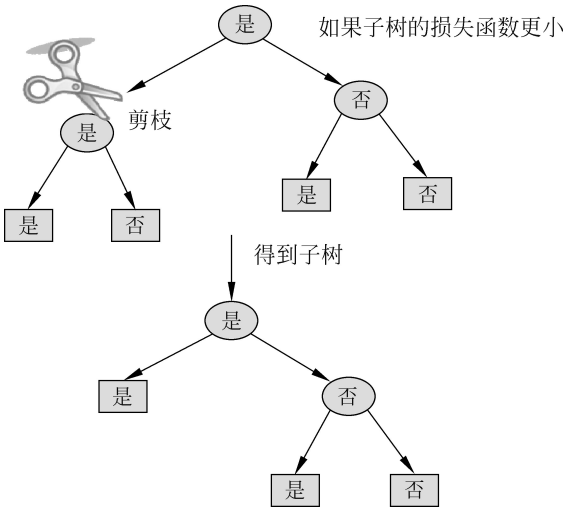


图 5.6 决策树的剪枝

设一组叶结点回缩到其父结点之前与之后的整体树分别为 $T_B$ 与 $T_A$，其对应的损失函数值分别是 $C_\alpha(T_B)$ 与 $C_\alpha(T_A)$，如果

$$C_\alpha(T_A) \leqslant C_\alpha(T_B) \tag{5.15}$$

则进行剪枝，即将父结点变为新的叶结点。

（3）返回步骤（2），直至不能继续为止，得到损失函数最小的子树 $T_\alpha$。 ■

注意，式（5.15）只需考虑两个树的损失函数的差，其计算可以在局部进行。所以，决策树的剪枝算法可以由一种动态规划的算法实现。类似的动态规划算法可参考文献 [10]。

## 5.5 CART 算 法

分类与回归树（classification and regression tree，CART）模型由 Breiman 等人在 1984 年提出，是应用广泛的决策树学习方法。CART 同样由特征选择、树的生成及剪枝组成，既可以用于分类也可以用于回归。以下将用于分类与回归的树统称为决策树。

CART 是在给定输入随机变量 $X$ 条件下输出随机变量 $Y$ 的条件概率分布的学习方法。CART 假设决策树是二叉树，内部结点特征的取值为“是”和“否”，左分支是取值为“是”的分支，右分支是取值为“否”的分支。这样的决策树等价于递归地二分每个特征，将输入空间即特征空间划分为有限个单元，并在这些单元上确定预测的概率分布，也就是在输入给定的条件下输出的条件概率分布。

CART 算法由以下两步组成：

（1）决策树生成：基于训练数据集生成决策树，生成的决策树要尽量大。

（2）决策树剪枝：用验证数据集对已生成的树进行剪枝并选择最优子树，这时用损失函数最小作为剪枝的标准。

### 5.5.1 CART 生成

决策树的生成就是递归地构建二叉决策树的过程。对回归树用平方误差最小化准则，对分类树用基尼指数（Gini index）最小化准则，进行特征选择，生成二叉树。

1. 回归树的生成

假设 $X$ 与 $Y$ 分别为输入和输出变量，并且 $Y$ 是连续变量，给定训练数据集

$$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

考虑如何生成回归树。

一棵回归树对应着输入空间（即特征空间）的一个划分以及在划分的单元上的输出值。假设已将输入空间划分为 $M$ 个单元 $R_1, R_2, \cdots, R_M$，并且在每个单元 $R_m$ 上有一个固定的输出值 $c_m$，于是回归树模型可表示为

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{5.16}$$

当输入空间的划分确定时，可以用平方误差 $\sum\limits_{x_i \in R_m} (y_i - f(x_i))^2$ 来表示回归树对训练数据的预测误差，用平方误差最小的准则求解每个单元上的最优输出值。易知，单元 $R_m$ 上的 $c_m$ 的最优值 $\hat{c}_m$ 是 $R_m$ 上的所有输入实例 $x_i$ 对应的输出 $y_i$ 的均值，即

$$\hat{c}_m = \text{ave}(y_i | x_i \in R_m) \tag{5.17}$$

问题是怎样对输入空间进行划分。这里采用启发式的方法，选择第 $j$ 个变量 $x^{(j)}$ 和它取的值 $s$ 作为切分变量（splitting variable）和切分点（splitting point），并定义两个区域：

$$R_1(j, s) = \{x | x^{(j)} \leqslant s\}, \quad R_2(j, s) = \{x | x^{(j)} > s\} \tag{5.18}$$

然后寻找最优切分变量 $j$ 和最优切分点 $s$。具体地，求解

$$\min_{j,s} \left[ \min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \tag{5.19}$$

对固定输入变量 $j$ 可以找到最优切分点 $s$。

$$\hat{c}_1 = \text{ave}(y_i | x_i \in R_1(j, s)), \quad \hat{c}_2 = \text{ave}(y_i | x_i \in R_2(j, s)) \tag{5.20}$$

遍历所有输入变量，找到最优的切分变量 $j$，构成一个对 $(j, s)$。依此将输入空间划分为两个区域。接着，对每个区域重复上述划分过程，直到满足停止条件为止。这样就生成一棵回归树。这样的回归树通常称为最小二乘回归树（least squares regression tree），现将算法叙述如下。

**算法 5.5（最小二乘回归树生成算法）**

输入：训练数据集 $D$。

输出：回归树 $f(x)$。

在训练数据集所在的输入空间中，递归地将每个区域划分为两个子区域并决定每个子区域上的输出值，构建二叉决策树：

（1）选择最优切分变量 $j$ 与切分点 $s$，求解

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right] \tag{5.21}$$

遍历变量 $j$，对固定的切分变量 $j$ 扫描切分点 $s$，选择使式 (5.21) 达到最小值的对 $(j,s)$。

（2）用选定的对 $(j,s)$ 划分区域并决定相应的输出值：

$$R_1(j,s)=\{x|x^{(j)}\leqslant s\},\quad R_2(j,s)=\{x|x^{(j)}>s\}$$

$$\hat{c}_m=\frac{1}{N_m}\sum_{x_i\in R_m(j,s)}y_i,\quad x\in R_m,\quad m=1,2$$

（3）继续对两个子区域调用步骤 (1) 和步骤 (2)，直至满足停止条件。

（4）将输入空间划分为 $M$ 个区域 $R_1,R_2,\cdots,R_M$，生成决策树：

$$f(x)=\sum_{m=1}^{M}\hat{c}_mI(x\in R_m)$$ ■

2. 分类树的生成

分类树用基尼指数选择最优特征，同时决定该特征的最优二值切分点。

**定义 5.4（基尼指数）** 分类问题中，假设有 $K$ 个类，样本点属于第 $k$ 类的概率为 $p_k$，则概率分布的基尼指数定义为

$$\text{Gini}(p)=\sum_{k=1}^{K}p_k(1-p_k)=1-\sum_{k=1}^{K}p_k^2 \tag{5.22}$$

对于二类分类问题，若样本点属于第 1 个类的概率是 $p$，则概率分布的基尼指数为

$$\text{Gini}(p)=2p(1-p) \tag{5.23}$$

对于给定的样本集合 $D$，其基尼指数为

$$\text{Gini}(D)=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2 \tag{5.24}$$

这里，$C_k$ 是 $D$ 中属于第 $k$ 类的样本子集，$K$ 是类的个数。

如果样本集合 $D$ 根据特征 $A$ 是否取某一可能值 $a$ 被分割成 $D_1$ 和 $D_2$ 两部分，即

$$D_1=\{(x,y)\in D|A(x)=a\},\quad D_2=D-D_1$$

则在特征 $A$ 的条件下，集合 $D$ 的基尼指数定义为

$$\text{Gini}(D,A)=\frac{|D_1|}{|D|}\text{Gini}(D_1)+\frac{|D_2|}{|D|}\text{Gini}(D_2) \tag{5.25}$$

基尼指数 Gini($D$) 表示集合 $D$ 的不确定性，基尼指数 Gini($D, A$) 表示经 $A = a$ 分割后集合 $D$ 的不确定性。基尼指数值越大，样本集合的不确定性也就越大，这一点与熵相似。

图 5.7 显示二类分类问题中基尼指数 Gini($p$)、熵（单位比特）之半 $H(p)/2$ 和分类误差率的关系。横坐标表示概率 $p$，纵坐标表示损失。可以看出基尼指数和熵之半的曲线很接近，都可以近似地代表分类误差率。

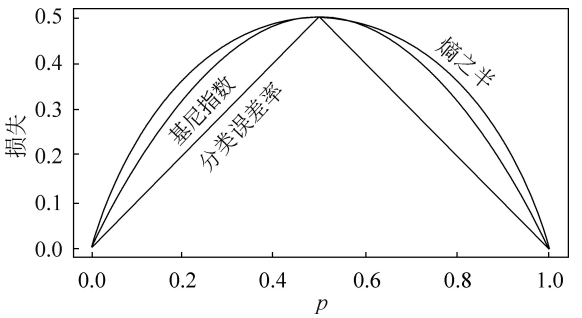


图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

**算法 5.6（CART 生成算法）**

输入：训练数据集 $D$，停止计算的条件。

输出：CART 决策树。

根据训练数据集，从根结点开始，递归地对每个结点进行以下操作，构建二叉决策树：

（1）设结点的训练数据集为 $D$，计算现有特征对该数据集的基尼指数。此时，对每一个特征 $A$，对其可能取的每个值 $a$，根据样本点对 $A = a$ 的测试为“是”或“否”将 $D$ 分割成 $D_1$ 和 $D_2$ 两部分，利用式（5.25）计算 $A = a$ 时的基尼指数。

（2）在所有可能的特征 $A$ 以及它们所有可能的切分点 $a$ 中，选择基尼指数最小的特征及其对应的切分点作为最优特征与最优切分点。依最优特征与最优切分点，从现结点生成两个子结点，将训练数据集依特征分配到两个子结点中去。

（3）对两个子结点递归地调用步骤（1）和步骤（2），直至满足停止条件。

（4）生成 CART 决策树。 ■

算法停止计算的条件是结点中的样本个数小于预定阈值或样本集的基尼指数小于预定阈值（样本基本属于同一类），或者没有更多特征。

**例 5.4**　根据表 5.1 所给训练数据集，应用 CART 算法生成决策树。

**解**　首先计算各特征的基尼指数，选择最优特征以及其最优切分点。仍采用例 5.2 的记号，分别以 $A_1$，$A_2$，$A_3$，$A_4$ 表示年龄、有工作、有自己的房子和信贷情况 4 个特征，并以 1，2，3 表示年龄的值为青年、中年和老年，以 1，2 表示有工作和有自己的房子的值为是和否，以 1，2，3 表示信贷情况的值为非常好、好和一般。

求特征 $A_1$ 的基尼指数:

$$\text{Gini}(D, A_1 = 1) = \frac{5}{15}\left[2 \times \frac{2}{5} \times \left(1 - \frac{2}{5}\right)\right] + \frac{10}{15}\left[2 \times \frac{7}{10} \times \left(1 - \frac{7}{10}\right)\right] = 0.44$$

$$\text{Gini}(D, A_1 = 2) = 0.48$$

$$\text{Gini}(D, A_1 = 3) = 0.44$$

由于 $\text{Gini}(D, A_1 = 1)$ 和 $\text{Gini}(D, A_1 = 3)$ 相等，且最小，所以 $A_1 = 1$ 和 $A_1 = 3$ 都可以选作 $A_1$ 的最优切分点。

求特征 $A_2$ 和 $A_3$ 的基尼指数：

$$\text{Gini}(D, A_2 = 1) = 0.32$$

$$\text{Gini}(D, A_3 = 1) = 0.27$$

由于 $A_2$ 和 $A_3$ 只有一个切分点，所以它们就是最优切分点。

求特征 $A_4$ 的基尼指数：

$$\text{Gini}(D, A_4 = 1) = 0.36$$

$$\text{Gini}(D, A_4 = 2) = 0.47$$

$$\text{Gini}(D, A_4 = 3) = 0.32$$

$\text{Gini}(D, A_4 = 3)$ 最小，所以 $A_4 = 3$ 为 $A_4$ 的最优切分点。

在 $A_1$，$A_2$，$A_3$，$A_4$ 几个特征中，$\text{Gini}(D, A_3 = 1) = 0.27$ 最小，所以选择特征 $A_3$ 为最优特征，$A_3 = 1$ 为其最优切分点。于是根结点生成两个子结点，一个是叶结点。对另一个结点继续使用以上方法在 $A_1$，$A_2$，$A_4$ 中选择最优特征及其最优切分点，结果是 $A_2 = 1$。依此计算得知，所得结点都是叶结点。 ■

对于本问题，按照 CART 算法所生成的决策树与按照 ID3 算法所生成的决策树完全一致。

### 5.5.2 CART 剪枝

CART 剪枝算法从“完全生长”的决策树的底端剪去一些子树，使决策树变小（模型变简单），从而能够对未知数据有更准确的预测。CART 剪枝算法由两步组成：首先从生成算法产生的决策树 $T_0$ 底端开始不断剪枝，直到 $T_0$ 的根结点，形成一个子树序列 $\{T_0, T_1, \cdots, T_n\}$；然后通过交叉验证法在独立的验证数据集上对子树序列进行测试，从中选择最优子树。

1. 剪枝，形成一个子树序列

在剪枝过程中，计算子树的损失函数：

$$C_\alpha(T) = C(T) + \alpha|T| \tag{5.26}$$

其中，$T$ 为任意子树，$C(T)$ 为对训练数据的预测误差（如基尼指数），$|T|$ 为子树的叶结点个数，$\alpha \geqslant 0$ 为参数，$C_\alpha(T)$ 为参数是 $\alpha$ 时的子树 $T$ 的整体损失。参数 $\alpha$ 权衡训练数据的拟合程度与模型的复杂度。

对固定的 $\alpha$，一定存在使损失函数 $C_\alpha(T)$ 最小的子树，将其表示为 $T_\alpha$。$T_\alpha$ 在损失函数 $C_\alpha(T)$ 最小的意义下是最优的。容易验证这样的最优子树是唯一的。当 $\alpha$ 大的时候，最优子树 $T_\alpha$ 偏小；当 $\alpha$ 小的时候，最优子树 $T_\alpha$ 偏大。极端情况：当 $\alpha = 0$ 时，整体树是最优的。当 $\alpha \to \infty$ 时，根结点组成的单结点树是最优的。

Breiman 等人证明：可以用递归的方法对树进行剪枝。将 $\alpha$ 从小增大，$0=\alpha_0<\alpha_1<\cdots<\alpha_n<+\infty$，产生一系列的区间 $[\alpha_i,\alpha_{i+1})$，$i=0,1,\cdots,n$；剪枝得到的子树序列对应着区间 $\alpha\in[\alpha_i,\alpha_{i+1})$，$i=0,1,\cdots,n$ 的最优子树序列 $\{T_0,T_1,\cdots,T_n\}$，序列中的子树是嵌套的。

具体地，从整体树 $T_0$ 开始剪枝。对 $T_0$ 的任意内部结点 $t$，以 $t$ 为单结点树的损失函数是

$$C_\alpha(t)=C(t)+\alpha \tag{5.27}$$

以 $t$ 为根结点的子树 $T_t$ 的损失函数是

$$C_\alpha(T_t)=C(T_t)+\alpha|T_t| \tag{5.28}$$

当 $\alpha=0$ 及 $\alpha$ 充分小时，有不等式

$$C_\alpha(T_t)<C_\alpha(t) \tag{5.29}$$

当 $\alpha$ 增大时，在某一 $\alpha$ 有

$$C_\alpha(T_t)=C_\alpha(t) \tag{5.30}$$

当 $\alpha$ 再增大时，不等式 (5.29) 反向。只要 $\alpha=\dfrac{C(t)-C(T_t)}{|T_t|-1}$，$T_t$ 与 $t$ 有相同的损失函数值，而 $t$ 的结点少，因此 $t$ 比 $T_t$ 更可取，对 $T_t$ 进行剪枝。

为此，对 $T_0$ 中每一内部结点 $t$，计算

$$g(t)=\frac{C(t)-C(T_t)}{|T_t|-1} \tag{5.31}$$

它表示剪枝后整体损失函数减少的程度。在 $T_0$ 中剪去 $g(t)$ 最小的 $T_t$，将得到的子树作为 $T_1$，同时将最小的 $g(t)$ 设为 $\alpha_1$。$T_1$ 为区间 $[\alpha_1,\alpha_2)$ 的最优子树。

如此剪枝下去，直至得到根结点。在这一过程中，不断地增加 $\alpha$ 的值，产生新的区间。

2. 在剪枝得到的子树序列 $T_0,T_1,\cdots,T_n$ 中通过交叉验证选取最优子树 $T_\alpha$

具体地，利用独立的验证数据集，测试子树序列 $T_0,T_1,\cdots,T_n$ 中各棵子树的平方误差或基尼指数。平方误差或基尼指数最小的决策树被认为是最优的决策树。在子树序列中，每棵子树 $T_1,T_2,\cdots,T_n$ 都对应一个参数 $\alpha_1,\alpha_2,\cdots,\alpha_n$。所以，当最优子树 $T_k$ 确定时，对应的 $\alpha_k$ 也确定了，即得到最优决策树 $T_\alpha$。

现在写出 CART 剪枝算法。

**算法 5.7（CART 剪枝算法）**

输入：CART 算法生成的决策树 $T_0$。

输出：最优决策树 $T_\alpha$。

（1）设 $k=0$，$T=T_0$。

（2）设 $\alpha=+\infty$。

（3）自下而上地对各内部结点 $t$ 计算 $C(T_t)$，$|T_t|$ 以及

$$g(t)=\frac{C(t)-C(T_t)}{|T_t|-1}$$

$$\alpha=\min(\alpha,g(t))$$

这里，$T_t$ 表示以 $t$ 为根结点的子树，$C(T_t)$ 是对训练数据的预测误差，$|T_t|$ 是 $T_t$ 的叶结点个数。

（4）对 $g(t)=\alpha$ 的内部结点 $t$ 进行剪枝，并对叶结点 $t$ 以多数表决法决定其类，得到树 $T$。

（5）设 $k=k+1$，$\alpha_k=\alpha$，$T_k=T$。

（6）如果 $T_k$ 不是由根结点及两个叶结点构成的树，则回到步骤 (2)；否则，令 $T_k=T_n$。

（7）采用交叉验证法在子树序列 $T_0,T_1,\cdots,T_n$ 中选取最优子树 $T_\alpha$。 ■

## 本章概要

1. 分类决策树模型是表示基于特征对实例进行分类的树形结构。决策树可以转换成一个 if-then 规则的集合，也可以看作是定义在特征空间划分上的类的条件概率分布。

2. 决策树学习旨在构建一个与训练数据拟合很好并且复杂度小的决策树。因为从可能的决策树中直接选取最优决策树是 NP 完全问题，现实中采用启发式方法学习次优的决策树。

决策树学习算法包括 3 个部分：特征选择、树的生成和树的剪枝。常用的算法有 ID3 算法、C4.5 算法和 CART 算法。

3. 特征选择的目的在于选取对训练数据能够分类的特征。特征选择的关键是其准则，常用的准则如下：

（1）样本集合 $D$ 对特征 $A$ 的信息增益（ID3）：

$$
\begin{aligned}
g(D,A)&=H(D)-H(D|A)\\
H(D)&=-\sum_{k=1}^{K}\frac{|C_k|}{|D|}\log_2\frac{|C_k|}{|D|}\\
H(D|A)&=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)
\end{aligned}
$$

其中，$H(D)$ 是数据集 $D$ 的熵，$H(D_i)$ 是数据集 $D_i$ 的熵，$H(D|A)$ 是数据集 $D$ 对特征 $A$ 的条件熵，$D_i$ 是 $D$ 中特征 $A$ 取第 $i$ 个值的样本子集，$C_k$ 是 $D$ 中属于第 $k$ 类的样本子集，$n$ 是特征 $A$ 取值的个数，$K$ 是类的个数。

（2）样本集合 $D$ 对特征 $A$ 的信息增益比（C4.5）：

$$
g_{\mathrm{R}}(D,A)=\frac{g(D,A)}{H_A(D)}
$$

其中，$g(D,A)$ 是信息增益，$H_A(D)$ 是 $D$ 关于特征 $A$ 的值的熵。

（3）样本集合 $D$ 的基尼指数（CART）：

$$
\mathrm{Gini}(D)=1-\sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2
$$

特征 $A$ 条件下集合 $D$ 的基尼指数：

$$\mathrm{Gini}(D,A)=\frac{|D_1|}{|D|}\mathrm{Gini}(D_1)+\frac{|D_2|}{|D|}\mathrm{Gini}(D_2)$$

4. 决策树的生成。通常使用信息增益最大、信息增益比最大或基尼指数最小作为特征选择的准则。决策树的生成往往通过计算信息增益或其他指标，从根结点开始，递归地产生决策树。这相当于用信息增益或其他准则不断地选取局部最优的特征，或将训练集分割为能够基本正确分类的子集。

5. 决策树的剪枝。由于生成的决策树存在过拟合问题，需要对它进行剪枝，以简化学到的决策树。决策树的剪枝往往从已生成的树上剪掉一些叶结点或叶结点以上的子树，并将其父结点或根结点作为新的叶结点，从而简化生成的决策树。

## 继续阅读

介绍决策树学习方法的文献很多，关于 ID3 可见文献 [1]，C4.5 可见文献 [2]，CART 可见文献 [3] 和文献 [4]。决策树学习的一般性介绍可见文献 [5]～文献 [7]。与决策树类似的分类方法还有决策列表（decision list）。决策列表与决策树可以相互转换 [8]，决策列表的学习方法可见文献 [9]。

## 习　题

**5.1**　根据表 5.1 所给的训练数据集，利用信息增益比（C4.5 算法）生成决策树。

**5.2**　已知如表 5.2 所示的训练数据，试用平方误差损失准则生成一个二叉回归树。

**表 5.2　训练数据表**

| $x_i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $y_i$ | 4.50 | 4.75 | 4.91 | 5.34 | 5.80 | 7.05 | 7.90 | 8.23 | 8.70 | 9.00 |

**5.3**　证明 CART 剪枝算法中，当 $\alpha$ 确定时，存在唯一的最小子树 $T_\alpha$ 使损失函数 $C_\alpha(T)$ 最小。

**5.4**　证明 CART 剪枝算法中求出的子树序列 $\{T_0,T_1,\cdots,T_n\}$ 分别是区间 $\alpha\in[\alpha_i,\alpha_{i+1})$ 的最优子树 $T_\alpha$，这里 $i=0,1,\cdots,n$，$0=\alpha_0<\alpha_1<\cdots<\alpha_n<+\infty$。

## 参考文献


[1] OLSHEN R A, QUINLAN J R. Induction of decision trees[J]. Machine Learning, 1986, 1(1): 81–106.

[2] OLSHEN R A, QUINLAN J R. C4.5: programs for machine learning[M]. Morgan Kaufmann, 1992.

[3] OLSHEN R A, BREIMAN L, FRIEDMAN J, et al. Classification and regression trees[M]. Wadsworth, 1984.

[4] RIPLEY B. Pattern recognition and neural networks[M]. Cambridge University Press, 1996.

[5] LIU B. Web data mining: Exploring hyperlinks, contents and usage data[M]. Springer-Verlag, 2006.

[6] HYAFIL L, RIVEST R L. Constructing optimal binary decision trees is NP-complete[J]. Information Processing Letters, 1976, 5(1): 15–17.

[7] HASTIE T, TIBSHIRANI R, FRIEDMAN J. The elements of statistical learning: data mining, inference, and prediction[M]. 范明，柴玉梅，咎红英，等译. Springer, 2001.

[8] YAMANISHI K. A learning criterion for stochastic rules[J]. Machine Learning, 1992, 9(2–3): 165–203.

[9] LI H, YAMANISHI K. Text classification using ESC-based stochastic decision lists[J]. Information Processing & Management, 2002, 38(3): 343–361.

[10] LI H, ABE N. Generalizing case frames using a thesaurus and the MDL principle[J]. Computational Linguistics, 1998, 24(2): 217–244.