# 理解深度学习

[美] 西蒙 • J. D. 普林斯(Simon J. D. Prince) 著
张亚东 马 壮 译

清華大學出版社
北 京

北京市版权局著作权合同登记号　图字：01-2024-5730
Simon J.D. Prince
Understanding Deep Learning
EISBN: 9780262048644


图书在版编目(CIP)数据

理解深度学习/(美) 西蒙·J.D.普林斯(Simon J. D. Prince)著;
张亚东, 马壮译. -- 北京 : 清华大学出版社, 2025. 6.
ISBN 978-7-302-69330-7

I. TP181

中国国家版本馆CIP数据核字第2025MR5980号

责任编辑：王　军
封面设计：高娟妮
版式设计：恒复文化
责任校对：成风进
责任印制：刘　菲

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.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：大厂回族自治县彩虹印刷有限公司
经　　销：全国新华书店
开　　本：170mm×240mm　　印　　张：29.25　　字　　数：574 千字
版　　次：2025 年 7 月第 1 版　　印　　次：2025 年 7 月第 1 次印刷
定　　价：188.00 元

产品编号：104408-01

# 行业名家力荐

人工智能的发展浩荡前行，累计出现过三次大的浪潮。第一次浪潮的符号主义困于逻辑的桎梏，第二次浪潮的连接主义受限于数据的藩篱，而作为第三次浪潮的深度学习技术则散发出魅力，为学术和产业界追捧，并持续酝酿、发酵及推动更猛烈的智能革命，被人们誉为第四次浪潮的大语言模型也正是基于深度学习。

由Simon J. D. Prince所著并由张亚东、马壮翻译的《理解深度学习》一书，从基础概念到核心技术再到最新进展，以全局视角系统性地梳理了深度学习的技术及应用发展全景，同时匹配了代码资源，是学术界和工业界有志于从事深度学习技术研究的新人上手实践不可多得的案头读物。

全书围绕深度神经网络的设计与应用展开，开篇聚焦网络训练策略以及性能评估，以奠定相关基础；继而深入解析卷积神经网络、残差连接与Transformer等关键架构的创新原理及其在各类任务中的适用性；随后剖析生成式对抗网络、变分自编码器、标准化流与扩散模型等现代生成机制，并概述深度强化学习的核心思想；结尾则从理论层面探讨深度网络的可训练性、泛化能力与参数冗余等基础问题，并从伦理角度反思深度学习技术的社会影响，内容上兼具理论深度与实践广度。

Simon教授因其具有工业界和学术界的双重经历，或许更了解那些苦于既不掌握技术细节，又难以动手实践的读者的困扰，因此本书写作风格深入浅出，配有大量的图与表以清楚地解释原本晦涩的概念，偶尔出现的公式看似枯燥，实际是有高等数学和线性代数基础读者可以轻松掌握的，也绝对是理论的有力补充。

深度学习技术及其应用已深入我们的工作与生活，如果你希望能够在这样一个智能革命的时代做一个弄潮儿，从这本书开始是个不错的选择。

李鑫

科大讯飞AI研究院副院长，科研部部长

在AI技术飞速演进的今天，我们正迎来“从模型走向应用”的时代。无论是产业数字化升级，还是开发者的新一轮创业浪潮，人工智能的落地能力，正在成为一家公司能否穿越周期的关键变量。极客邦科技在 2025 年将“AI 应用落地”定为年度主题，正是基于这样的判断：时代的新窗口已经打开，技术理解力决定了实践行动力。

在这个背景下，我非常高兴看到Simon J.D. Prince的经典著作Understanding Deep Learning推出中文版《理解深度学习》。这本书并不只是一本理论教材，它是一部帮助读者“真正理解”深度学习的入门指南。在我们长期运营InfoQ极客传媒、极客时间和TGO鲲鹏会的过程中，经常听到工程师、架构师和CTO们吐槽：“讲 Transformer 的人太多，能讲明白的人太少；讲推理框架的人不少，但从头带你理解原理的人几乎没有。”而这本书，恰恰解决了这样的问题。

《理解深度学习》不是快餐式教学，而是一场系统思维的启蒙。它通过严谨而通俗的语言，从线性代数、概率基础一步步引导你走进深度神经网络的世界。与市面上许多“照公式堆概念”的 AI 教程不同，本书强调WHY胜于WHAT，强调“理解”胜于“记忆”。它告诉我们，只有真正明白深度学习是如何工作的，才可能在未来的工程场景中创造出具有价值的应用。

对我个人而言，本书的意义不仅在于技术层面，更在于方法论上的启发。它体现了极客邦科技在内容和教育产品设计中始终坚持的价值观：技术的传播，应该是“实践驱动”的，应该追求可理解、可推理、可迁移，而不仅仅是堆叠信息。

随着AIGC、Agent、Agentic AI等技术生态加速演进，我们越来越看到，未来的开发者既要“理解底层”，又要“善用高阶工具”；既要具备对模型机制的抽象理解，也要能把技术变成具体的生产力。这本书，是打好底层认知根基的首选。

最后，我想特别感谢清华大学出版社引进这本作品。在喧嚣的当下，需要有人静下心来，从底层逻辑入手，抽丝剥茧，为世人排除通往未来的道路上的重重障碍。所以说，《理解深度学习》不仅是一本技术书籍，更是一把钥匙，帮助我们打开AI世界的思维之门。我相信，对于每一个想在 AI 应用时代中抓住机会的读者来说，这都将是一次重要的认知启程。

霍太稳

极客邦科技创始人兼CEO

《理解深度学习》是一本来自MIT的重磅教材，结构清晰，内容全面覆盖人工智能领域从基础到前沿的各种关键主题。作者是一位在学术界和工业界都有贡献的计算机科学家。他以直观的方式讲解复杂概念，同时结合了数学推导和可视化图示，帮助读者实现理论与实践的对接。无论是初学者，还是希望系统梳理AI知识的从业者，都能从中受益。

李烨
微软AI亚太区首席应用科学家

由深度学习引领的人工智能技术正在为各行各业带来巨大的范式创新。《理解深度学习》一书以精炼的文笔对深度学习这一复杂的理论体系抽丝剥茧，简明不失深刻，前沿不失实用，是系统掌握深度学习的佳作。

李建忠
全球机器学习技术大会主席，CSDN高级副总裁

《理解深度学习》是一部极具时代价值的经典教材。与传统教材不同，Simon J.D. Prince教授巧妙地在理论深度与实用性之间找到了平衡点，既避免了纯理论的晦涩，又不陷入代码实现的琐碎细节。我尤其欣赏本书对Transformer和扩散模型等前沿架构的深入剖析，这正是当前大模型技术的核心基石。书中通过精心设计的图表和直观解释，将复杂概念化繁为简，让读者真正"理解"而非仅仅"使用"深度学习。张亚东、马壮两位专家的翻译精准流畅，既保留原著的学术精髓，又贴合中文表达，有专门的章节探讨AI伦理与社会责任。对想在飞速迭代的AI浪潮中抓住技术本质的中文读者，本书是首选的入门与进阶指南。

王昊奋
OpenKG轮值主席，同济大学特聘研究员

在人工智能重塑人类文明边界的今天，Simon教授的《理解深度学习》犹如一盏明灯，既照亮了技术进化的底层逻辑，又烛照了伦理思考的精神维度。这部权威著作完美平衡了学术深度与教学温度，堪称深度学习领域的"概念罗盘"。无论你是渴望夯实理论根基的开发者，还是探寻智能本质的思想者，都能在这部兼具学术严谨性与思维启发性的作品中获得双重提升。在这个算法重构世界的时代，本书不仅是通向技术内核的通行证，更是一份关于智能文明的责任宣言。正如作者的观点："我们或许尚未真正理解深度学习，但绝不能停止理解它的努力。"这部著作，正是这场认知远征的最佳向导。

茹炳晟
腾讯Tech Lead，腾讯研究院特约研究员，
中国计算机学会CCF TF研发效能SIG主席

作为一名长期致力于集成电路领域教学与研究的教育工作者，我非常荣幸地向各位郑重推荐这本名为《理解深度学习》的优秀著作。

集成电路是现代科技的基石，尤其在人工智能时代，高性能、低功耗的芯片更是驱动AI技术发展的核心引擎。深度学习作为人工智能领域的核心技术之一，正以前所未有的速度渗透到集成电路的各个环节，从面向AI应用的芯片架构设计、智能化的设计、自动化工具开发，到利用AI进行电路性能优化和故障诊断，再到基于新型存储器实现高效的AI计算，都展现出巨大潜力。可以预见，未来的集成电路发展将更加紧密地契合AI领域的需求。

《理解深度学习》一书系统而深入地介绍了深度学习的基本原理和核心架构，如卷积神经网络、循环神经网络，以及近年来备受瞩目的生成式对抗网络、扩散模型和Transformer等。这些内容不仅能帮助集成电路领域的学生和研究人员快速掌握深度学习的关键技术，更能启发工程师们思考如何设计出更高效、更智能的芯片，以满足日益增长的AI算力需求，并探索将这些先进的算法直接部署到芯片上的可能性，实现软硬件的深度融合。

我相信，这本书的出版将成为连接集成电路领域与深度学习领域的桥梁，促进跨学科的交流与合作，加速人工智能技术在集成电路领域的创新应用，并推动设计出更智能、更强大的AI芯片。我诚挚地希望《理解深度学习》能够成为你探索人工智能奥秘、赋能集成电路事业的得力助手。

叶乐
北京大学集成电路学院/博雅教授

作为多年从事人工智能领域研究的教育工作者，我非常荣幸能为《理解深度学习》这本杰出的著作撰写推荐序。

作为人工智能领域的核心技术，深度学习(尤其是大语言模型)近年来取得了令人瞩目的成就，深刻地改变了我们的生活和工作方式。然而，深度学习的理论基础和内部机制相对复杂，对于初学者来说，往往存在一定的学习门槛。

《理解深度学习》这本书的出现，恰好弥补了这一缺憾。本书以清晰易懂的语言，深入浅出地介绍了深度学习(包括大语言模型)的基本概念、原理和方法。作者通过丰富的、深入浅出的讲解，让零基础的读者也能像搭积木一样，一步步掌握深度学习的核心魔法，并逐步掌握深度学习的核心技术。

本书的亮点总结如下。

- 系统全面的知识体系：本书涵盖深度学习的各个方面，从基础知识到各类模型应用方向，构建了一个完整的知识体系，帮助读者全面了解深度学习。
- 深入浅出的讲解方式：作者采用通俗易懂的语言，配合丰富的图表和公式，将复杂的概念和理论转化为易于理解的知识，降低了学习难度。
- 理论与实践相结合：本书不仅介绍了深度学习的理论知识，还提供了一定量的习题，帮助读者更好地掌握所学知识和理论。

本书的中文翻译版，更是为广大中文读者提供了一个便捷的学习途径。我相信，无论是初学者还是专业人士，都能从本书中获益匪浅。

我强烈推荐《理解深度学习》这本书，相信它将成为你学习深度学习的得力助手。

郭艳卿
大连理工大学未来技术学院/人工智能学院副院长

# 译者序

尊敬的读者：

首先，由衷地感谢你拿起我翻译的这本《理解深度学习》(Understanding Deep Learning)。作为本书的译者，能将这本杰作呈现在中文读者面前，我备感荣幸，内心充满感激之情。

我要向本书的作者西蒙·J. D.普林斯教授致以最诚挚的谢意。感谢他创作了这样一本深入浅出、系统全面、充满洞见的深度学习著作。本书不仅揭开了深度学习的神秘面纱，更以清晰的逻辑和丰富的实例，引导读者逐步理解和掌握这一前沿技术的核心概念和应用方法。 在翻译过程中，我一次次地被作者的深刻思考所折服，也愈发体会到这本书对于想要入门或深入理解深度学习、大模型学习的读者来说，是多么宝贵和不可或缺。

感谢编辑王军老师，一次偶遇让我有了圆梦的机会！深度学习(尤其是大语言模型)作为人工智能领域的核心驱动力，正深刻改变着世界。将这样一本优秀的著作译成中文，使其能够惠及广大的中文读者，帮助朋友们理解并掌握这方面的相关原理与技术，是我的荣幸。

翻译过程既是一次充满挑战的智力探险，也是一段收获满满的学习之旅。我力求在忠实原文的基础上，尽可能地使译文流畅自然，易于理解。确保术语的准确性和表达的规范性，尽量保证精准翻译每个细节。如果有任何不确切或者谬误之处，请原谅译者的才具不足。

在此，衷心感谢在本书翻译和出版过程中给予我帮助和支持的所有朋友们。感谢付裕从头到尾给予的指点与支持。感谢合作译者马壮的支持，他完成了另一半的翻译工作。感谢朋友王志晨校对了每一句中文，让工科背景人的译作也向“信、达、雅”靠近了！感谢所有为本书的出版辛勤付出的其他人士，使得本书的中文版能以高质量的面貌呈现在大家面前。

衷心希望本书能帮助更多读者踏入深度学习、大语言模型的大门，理解其精

髓，掌握其应用，并在未来的学习和工作中取得更大成就。如果我能为中国的深度学习技术发展贡献绵薄之力，那将是我最大的荣幸。

愿你在AI技术这一激动人心的探索之旅中，收获满满！

译者　张亚东

无论是在求学阶段，还是在步入职场后，很多人都会向我提出这样一个问题：“我准备进入人工智能领域进行学习或研究，你能推荐一些合适的学习资料吗？”

在当今这个信息爆炸的时代，获取学习资料的渠道并不匮乏，各类书籍和视频资源可谓琳琅满目。但我始终没有找到一本合适的“人工智能入门指南”。一部分书籍罗列了许多新颖的算法，但缺乏深入剖析，浮于表面；它们更适合作为科普读物，对于希望深入学习的读者来说，内容深度是远远不够的。一部分书籍从最基础的数学原理入手，详细列出相关公式，但学习门槛较高；而且即便完全掌握了这些原理，若想将算法落地，仍需要补充对应的编程技能。还有一部分书籍侧重实践应用，提供了大量示例代码和详细讲解，却容易让读者陷入“知其然”而“不知其所以然”的困境。

直到阅读了亚东老师推荐的Understanding Deep Learning后，我意识到终于找到理想的“指南书”了。本书从最基础的浅层神经网络入手，逐步深入，直至扩散模型，内容包罗万象。在基础模型章节中，Simon教授对每个问题的建模和公式推导都进行了细致入微的讲解；在高级模型章节中，阐述原理，还深入探讨模型的应用场景及前沿研究趋势。如果你潜心研读每一章节、每一个公式，会发现Simon教授倾注了大量心血，力求将复杂的概念以最清晰的方式呈现给读者。深度学习的难点之一在于如何理解高维空间中的问题，Simon教授通过降维和可视化手段，将梯度下降、迭代求解等抽象过程直观展现出来，极大地降低了理解门槛。本书讲解算法原理，还附有每个模型的Python Notebook源代码。无论你是具备一定数学基础的本科生，还是希望深入了解人工智能技术的软件工程师，本书都是你的最佳选择。

非常感谢亚东老师让我参与本书的翻译工作。翻译不仅是将文字从一种语言转换为另一种语言的过程，更是一次重构和深化知识体系的宝贵机会。在翻译过程中，原本零散的知识碎片被重新串联起来，形成更完整的知识框架，这对我来说是一次极为珍贵的经历。

由于时间仓促且本人学识有限，译文中难免存在疏漏与不足之处，恳请各位读者批评指正。同时，建议有能力的读者直接阅读Simon教授的原文，相信你一定会从中获得更多启发与收获。

译者　马壮

# 序言

这是一本最新的权威深度学习入门书籍，内容通俗易懂。本书涵盖从机器学习基础知识到高级模型的全部内容，精选了机器学习领域的核心要点和前沿课题，并直观地凝练了知识要点。

- 涵盖前沿课题，如 Transformer 和扩散模型。
- 用通俗易懂的语言阐述复杂的概念，并辅以数学公式和可视化图表。
- 使读者能够实现简单的模型。
- 提供了全面的在线资料，含 Python Notebook 中的编程练习。
- 适合任何具有应用数学基础的人学习。

Simon J. D. Prince 是巴斯大学计算机科学系荣誉教授，是《计算机视觉：模型、学习和推断》一书的作者。他是一位专注于人工智能和深度学习研究的科学家，曾在Borealis AI 等公司担任研究员。

这是一部学术与视觉双重精妙的杰作。它以简洁而清晰的方式传递核心思想，并通过精心设计的插图加以诠释，堪称当今最出色的深度学习入门著作。

——Kevin Murphy

Google DeepMind研究科学家

*Probabilistic Machine learning: Advanced Topics*作者

# 前言

在学术界，深度学习的发展史极不寻常。一小群科学家坚持不懈地在一个看似没有前途的领域工作了25年，最终使一个领域发生了技术革命并极大地影响了人类社会。研究者持续探究学术界或工程界中深奥且难以解决的问题，通常情况下这些问题无法得到根本性解决。但深度学习领域是个例外，尽管广泛的怀疑仍然存在，但Yoshua Bengio、Geoff Hinton 和Yann LeCun 等人的系统性努力最终取得了成效。

本书的书名是“理解深度学习”，这意味着它更关注深度学习背后的原理，而不侧重于编程实现或实际应用。本书的前半部分介绍深度学习模型并讨论了如何训练它们，评估它们的表现并做出改进。后半部分讨论专用于图像、文本、图数据的模型架构。只要学习过线性代数、微积分和概率论的二年级本科生都能掌握这些章节的知识。对于后续涉及生成模型和强化学习的章节，则需要更多的概率论和微积分知识，它们面向更高年级的学生。

这个书名在一定程度上也是一个玩笑——在撰写本书时，没有人能够真正理解深度学习。目前深度神经网络学习的分段线性函数的数量比宇宙中的原子数还多，可用远少于模型参数数量的样本进行训练。现在我们既无法找到可靠地拟合这些函数的方法，又不能保证能很好地描述新数据。第20章讨论了上述问题和其他尚未完全理解的问题。无论如何，深度学习都将或好或坏地改变这个世界。最后一章讨论了人工智能伦理，并呼吁从业者更多地考虑所从事的工作带来的伦理问题。

本书配套资源丰富，提供教师课件、习题及Python Notebook编程练习。

你的时间是宝贵的，为了保证你能高效理解深度学习相关知识，本书的内容都经过我的精心整理。每一章都对最基本思路进行简明描述，并附有插图。附录回顾了所有数学原理。对于希望深入研究的读者，可认真研究每章列出的问题、Python Notebook资源和背景说明。

# 致谢

本书的出版离不开以下人士的无私帮助和建议：Kathryn Hume、Kevin Murphy、Christopher Bishop、Peng Xu、Yann Dubois、Justin Domke、Chris Fletcher、Yanshuai Cao、Wendy Tay、Corey Toler-Franklin、Dmytro Mishkin、Guy McCusker、Daniel Worrall、Paul McIlroy、Roy Amoyal、Austin Anderson、Romero Barata de Morais、Gabriel Harrison、Peter Ball、Alf Muir、David Bryson、Vedika Parulkar、Patryk Lietzau、Jessica Nicholson、Alexa Huxley、Oisin Mac Aodha、Giuseppe Castiglione、Josh Akylbekov、Alex Gougoulaki、Joshua Omilabu、Alister Guenther、Joe Goodier、Logan Wade、Joshua Guenther、Kylan Tobin、Benedict Ellett、Jad Araj、Andrew Glennerster、Giorgos Sfikas、Diya Vibhakar、Sam Mansat-Bhattacharyya、Ben Ross、Ivor Simpson、Gaurang Aggarwal、Shakeel Sheikh、Jacob Horton、Felix Rammell、Sasha Luccioni、Akshil Patel、Mark Hudson、Alessandro Gentilini、Kevin Mercier、Krzysztof Lichocki、Chuck Krapf、Brian Ha、Chris Kang、Leonardo Viotti、Kai Li、Himan Abdollahpouri、Ari Pakman、Giuseppe Antonio Di Luna、Dan Oneată、Conrad Whiteley、Joseph Santarcangelo、Brad Shook、Gabriel Brostow、Lei He、Ali Satvaty、Romain Sabathé、Qiang Zhou、Prasanna Vigneswaran、Siqi Zheng、Stephan Grein、Jonas Klesen、Giovanni Stilo、Huang Bokai、Bernhard Pfahringer、Joseph Santarcangelo、Kevin McGuinness、Qiang Sun、Zakaria Lotfi、Yifei Lin、Sylvain Bouix、Alex Pitt、Stephane Chretien、Robin Liu、Bian Li、Adam Jones、Marcin Świerkot、Tommy Löfstedt、Eugen Hotaj、Fernando Flores Mangas、Tony Polichroniadis、Pietro Monticone、Rohan Deepak Ajwani、Menashe Yarden Einy、Robert Gevorgyan、Thilo Stadelmann、Gui JieMiao、Botao Zhu、Mohamed Elabbas、Satya Krishna Gorti、James Elder、Helio Perroni Filho、Xiaochao Qu、Jaekang Shin、Joshua Evans、Robert Dobson、Shibo Wang、Edoardo Zorzi、

Joseph Santarcangelo、Stanisław Jastrzębski、Pieris Kalligeros、Matt Hewitt和Zvika Haramaty.

感谢 Daniyar Turmukhambetov、Amedeo Buonanno、Tyler Mills、Andrea Panizza和Bernhard Pfahringer 对多个章节给出的详细修改建议。尤其感谢Andrew Fitzgibbon和Konstantinos Derpanis 通读全书，你们的热情使我有动力完成本书的撰写工作。同时，感谢Neill Campbell和Özgür Şimşek邀请我到巴斯大学讲课，在那里我基于这些材料第一次开设了一门课程。最后，尤其感谢编辑 Elizabeth Swayze 提出的中肯建议。

第12章(变换器)和第17章(变分自编码器)最初发布在Borealis AI的博客中，改编后的版本在获得许可的情况下在此转载。非常感谢他们对我工作的支持。第16章(标准化流)大致基于Kobyzev等人在2002年发表的文章，我是该文章的合著者。我很幸运能够和来自达尔豪斯大学的Travis LaCroix合作撰写第21章。他既有趣又平易近人，他承担了大部分工作。

# 目录

# 第1章

# 引言

人工智能(artificial intelligence，AI)是一种旨在构建模拟人类智能行为的系统。它涵盖了多种方法，包括基于逻辑、搜索和概率推理的方法。机器学习是AI的一个子集，通过将数学模型拟合到观察到的数据上来学习做出决策。这一领域已经经历了爆炸性增长，导致现在机器学习几乎被误认为AI的同义词了。

深度神经网络是一种机器学习模型，当它拟合到数据时，被称为深度学习。在撰写本书时，深度网络是最强大、最实用的机器学习模型，经常出现在日常生活中。例如，使用自然语言处理算法翻译文本、使用计算机视觉系统在互联网上搜索特定物体的图像，或者通过语音识别界面与数字助手对话，都是司空见惯的事。所有这些应用都是由深度学习驱动的。

本书旨在帮助那些刚接触这个领域的读者理解深度学习背后的原理。本书既不追求理论化(没有数学上的证明过程)，也不追求极致的实用化(几乎没有代码)。目标是解释深度学习的核心思想；读完本书后，读者能够将深度学习应用于那些没有现成的成功方法的新场景中。

机器学习方法大致可以分为三个领域：监督学习、无监督学习和强化学习。在撰写本书时，这三个领域的前沿方法都依赖于深度学习(图1.1)。本章对这三个领域进行了概述，这种分类也大致反映了本书的结构。无论我们喜欢与否，深度学习都将改变世界，但这种改变并不全是积极的。因此，本章还将简要介绍AI伦理。最后，将提供一些如何充分利用本书的建议。

## 1.1 监督学习

监督学习模型定义了从输入数据到输出预测的映射。接下来将讨论输入、输

出、模型本身，以及“学习”这个动作对一个模型究竟意味着什么。

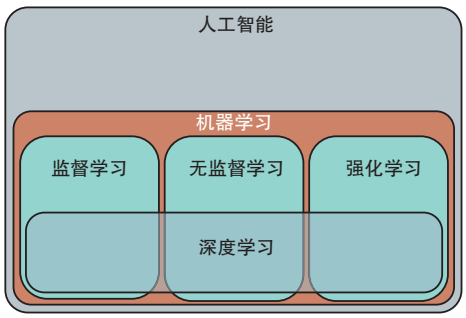



图1.1 机器学习是人工智能中的一个子领域，将数学模型拟合到观测数据上。机器学习大致可分为监督学习、无监督学习和强化学习。深度神经网络对这些领域的每一个都有贡献

### 1.1.1 回归和分类问题

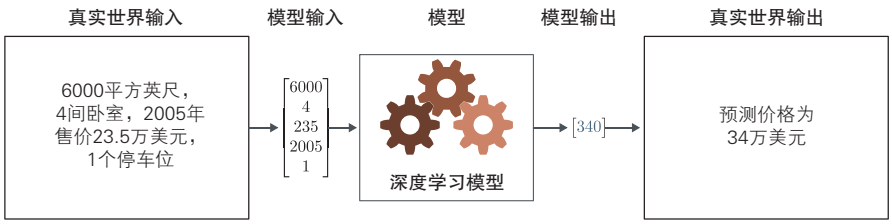
图1.2展示了几个回归和分类问题。在每个案例中，都有一个有意义的实际输入(句子、声音文件、图片等)，并将其编码为一个数字向量。这个向量构成了模型的输入。模型将输入映射到一个输出向量，然后将其“转换”并返回一个有意义的实际预测。目前，我们专注于输入和输出，并将模型视为黑盒，它接收一个数字向量并返回另一个数字向量。

图1.2(a)中的模型基于输入特征(如房屋的平方英尺数和卧室数量)预测房价。因为模型返回一个连续数字(而不是一个类别分配)，所以这是回归问题。相比之下，图1.2(b)中的模型以一个分子的化学结构为输入，并预测它的熔点和沸点。因为它预测不止一个数字，所以是二元分类问题。

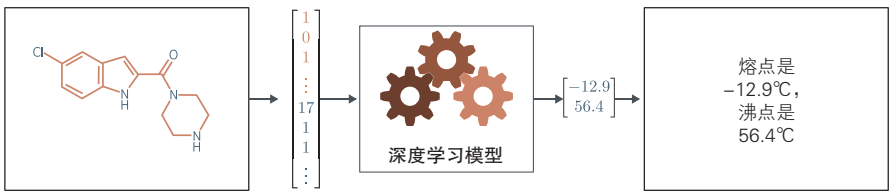
图1.2(c)中的模型接收一个包含餐厅评论的文本字符串作为输入，并预测评论是正面的还是负面的。因为模型试图将输入分配到两个类别中的一个，所以这是二元分类问题。输出向量包含输入属于每个类别的概率。图1.2(d)和1.2(e)展示了多分类问题。这里，模型将输入分配到$N>2$个类别中的一个。在图1.2(d)的案例中，输入是一个音频文件，模型预测它包含的音乐类型。在图1.2(e)的案例中，输入是一张图片，模型预测它包含的物体。每种情况下，模型都返回一个大小为$N$的向量，其中包含$N$个类别的概率。

### 1.1.2　输入

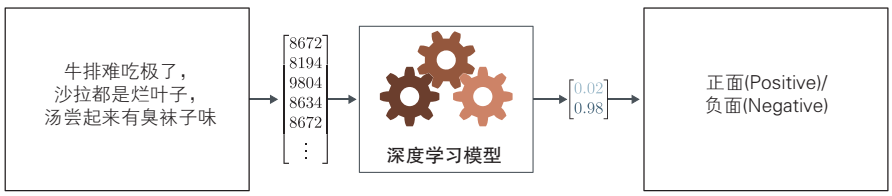
图1.2中的输入数据的类型差异很大。在房价预测例子中，输入的是包含了表征房屋特征值的固定长度的向量。这是一组表格数据，因为这组数据没有内部结构；如果我们改变输入的顺序并构建一个新模型，我们希望模型的预测结果保持不变。



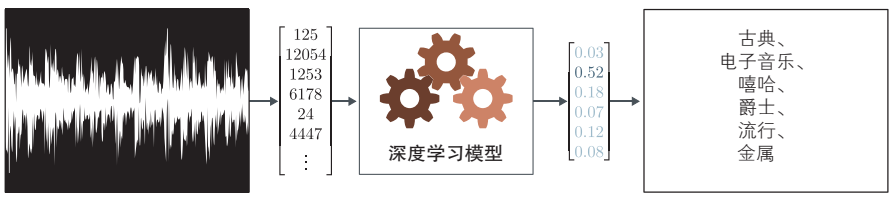

(a) 该回归模型接收一组描述属性的数字，并预测其价格



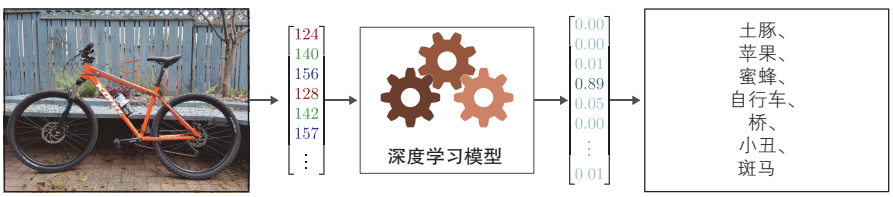

(b) 该多元回归模型接收化学分子的结构，并预测其熔点和沸点




(c) 该二元分类模型接收一条餐厅评论，并将其分类为正面或负面




(d) 该多类别分类问题将一段音频片段归类为$N$个类型之一




(e) 第二个多类别分类问题，模型根据图像可能包含的$N$种物体之一对其进行分类

图1.2　回归和分类问题

相反，在餐厅评论的例子中，输入是一段文字。输入的长度可能根据评论中的单词数量而变化，在该例中输入顺序就很重要；“我的妻子吃了鸡肉”与“鸡肉吃了我的妻子”的含义完全不同。文本必须在传递给模型之前编码成数字形式。在该例中，使用大小为10 000的固定词汇表，并简单地将单词的索引串联起来。

对于音乐分类的例子，输入向量可能是固定大小的(如10秒的片段)，但维度非常高(包含许多条目)。数字音频通常以44.1kHz的频率采样并用16位整数表示，因此10秒的片段将由441 000个整数组成。显然，监督学习模型必须能够处理相当庞大的输入数据。在图像分类的例子中的输入(由每个像素处的RGB值组成)也很庞大。此外，它的结构天然是二维的；即使在输入向量中不相邻，上下相邻的两个像素也是密切相关的。

最后，考虑预测分子的熔点和沸点的模型输入。一个分子可能包含不同数量的原子。另外，这些原子还可通过不同方式进行连接。所以这种情况下，模型必须同时将分子的几何结构和组成分子的原子作为输入。

### 1.1.3 机器学习模型

到目前为止，我们可将机器学习模型视作黑盒，它接收输入向量并返回输出向量。但这个黑盒里面究竟是什么呢？联想一下根据年龄预测孩子身高的模型(图1.3)。机器学习模型是一个数学函数，它描述了平均身高如何随年龄变化(图1.3中的青色曲线)。将年龄代入这个函数时，它就会返回身高。例如，如果年龄是10岁，预测身高将是139厘米。

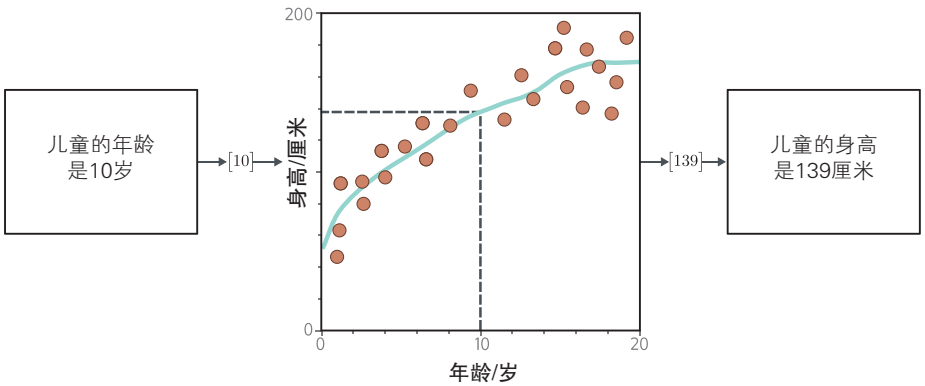



图1.3 机器学习模型。该模型代表了一系列关系，将输入(儿童的年龄)与输出(儿童的身高)关联在一起。具体关系是通过训练数据集来选择的，这些训练数据由输入/输出对(橙色点)组成。当我们训练模型时，会在可能的关系中搜索一个能很好地描述数据的关系。这里，训练后的模型是青色曲线，可用来计算任何年龄的儿童的身高

更准确地说，该模型表示一系列将输入映射到输出的函数(即，不同的青色曲线族)。特定的函数(曲线)是使用训练数据(输入/输出对)选择的。在图1.3中，这些对用橙色点表示，可以看到用该模型(即青线)来描述这些数据就非常合理。当谈论训练或拟合一个模型时，我们的意思是在可能的函数(可能的青色曲线)之间搜索，来找到那个可以最准确地描述训练数据的函数。

因此，图1.2中的模型需要用标记的“输入/输出对”进行训练。例如，音乐分类模型需要大量音频片段，其中人类专家已经标记了每个片段的流派。这些输入/输出对在训练过程中扮演教师或监督者的角色，由此产生了“监督学习”这个术语。

### 1.1.4　深度神经网络

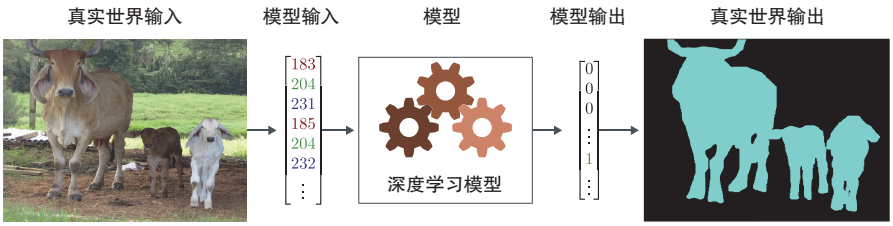
本书将重点介绍深度神经网络，这是一种特别实用的机器学习模型，也是函数，可以表示输入和输出之间极其广泛的关系族，并可通过遍历这个关系族找到训练数据之间的关系。

深度神经网络可以处理非常大、长度可变且包含各种内部结构的输入。它们可以输出单个实数值(回归)、多个数值(多元回归)或两个及更多类别的概率(分别为二元分类和多元分类)。正如我们将在下一节看到的，它们的输出也可能非常大、长度可变且包含内部结构。想象具有这些性质的函数可能十分困难，但你现在应该尝试暂时放下怀疑。

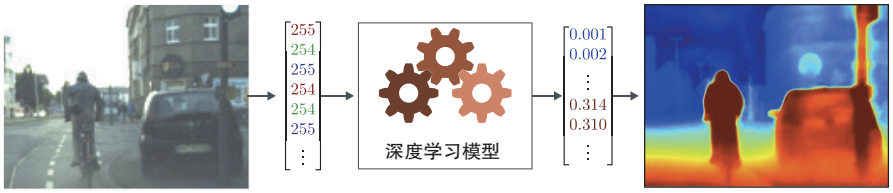
### 1.1.5　结构化输出

图1.4(a)描绘了一个用于语义分割的多元分类模型。这里，输入图像的每个像素都被分配一个二元标签，指示它属于牛本身还是属于背景。图1.4(b)展示了一个单目深度估计模型，该模型的输入是一幅街景图像，输出是每个像素的深度。这两种情况下，输出都是高维且结构化的。然而，这种结构与输入密切相关，并且可供利用；如果一个像素被标记为“牛”，那么与其具有相似RGB值的邻居可能也有相同的标签。

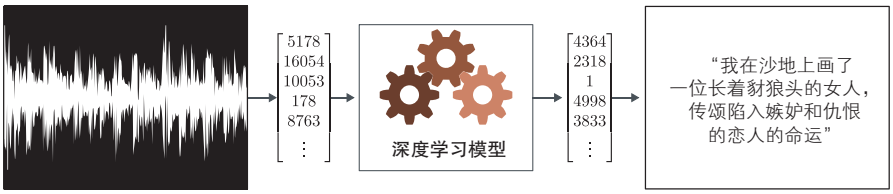
图1.4(c)~(e)描绘了三组具有复杂结构的输出但与输入联系不那么密切的模型。图1.4(c)展示的模型的输入是音频文件，输出是该文件中语音的文字转录。图1.4(d)是翻译模型，输入是中文文本，而输出是法文文本。图1.4(e)描述了一种极具挑战性的任务，其输入是描述性文本，而模型的输出是必须与此描述相符的图像。



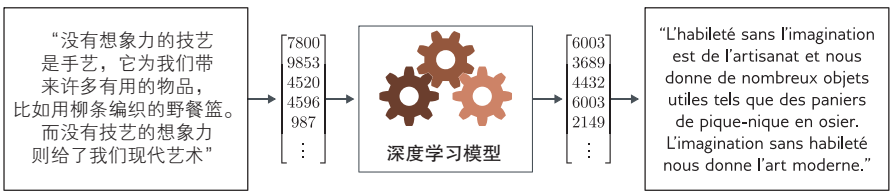

(a) 这个语义分割模型将RGB图像映射到一个二值图像，指示每个像素属于背景还是牛(Noh等，2015)



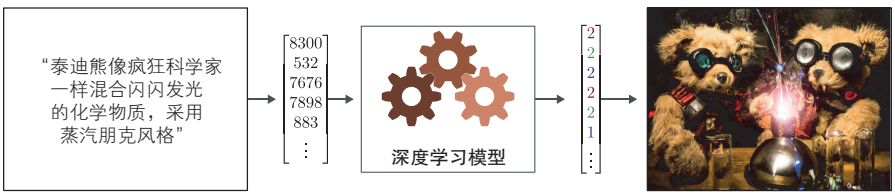

(b) 这个单目深度估计模型将RGB图像映射到一个输出图像，其中每个像素代表深度(Cordts等，2016)




(c) 这个音频转录模型将音频样本映射到文字转录




(d) 这个翻译模型将中文文本映射为法文文本




(e) 这个图像合成模型将标题映射到图像

图1.4 具有结构化输出的监督学习任务。在每个案例中，输出都有复杂的内部结构或语法。某些情况下，许多输出与输入兼容

原则上，后三个任务都可在标准的监督学习框架下实现，但由于下面两个原因使它们变得更加困难。首先，输出可能确实是模糊的；从一个中文句子到法文句子有多种有效的翻译方式，任何标题都可能与多种图像相符。其次，输出包含

相当多的结构；并不是所有单词或字符都能构成有效的中文和法文句子，也不是所有的RGB的组合都能构成合理的图像。除了学习映射，我们还必须遵循输出的“语法”。

幸运的是，这种“语法”可在不需要输出标签的情况下进行学习。例如，可通过学习大量文本数据的统计信息来学习如何形成有效的中文句子。这为后文中的无监督学习模型打下了基础。

## 1.2　无监督学习

不使用输入数据对应的输出数据标签来构建模型的过程称为无监督学习；没有输出标签意味着没有“监督”。无监督学习的目标并非是学习从输入到输出的映射，而是描述或理解输入数据的结构。就像监督学习一样，数据可能具有非常不同的特征：可能是离散的或连续的，低维的或高维的，长度固定的或可变的。

### 1.2.1　生成模型

本书关注的是生成无监督模型，这类模型能合成新的数据样本，这些样本在统计学上与训练数据无法区分。一些生成模型明确描述了输入数据的概率分布，并从这个分布中采样生成新的样本。另一些模型只是学习生成新样本的机制，而不是描述它们的分布。

最先进的生成模型可以合成极其逼真但与训练样本不同的样本。它们在生成图像(图1.5)和文本(图1.6)方面特别成功。它们还可以在一定约束条件下合成数据，即预先设定某些输出(称为条件生成)，例如图像修复(图1.7)和文本补全(图1.8)。事实上，现代文本生成模型如此强大，以至于它们看起来是智能的。给定一段文本然后提出一个问题，模型通常可以通过生成文档最可能的结尾部分来“填补”缺失的答案。然而，实际上，模型只是了解语言的统计信息，并不理解答案的意义。

图1.5　针对图像的生成模型。左边两张图像是由训练有猫图片的模型生成的。这些不是真实的猫，而是概率模型的样本。右边两张图像是由训练有建筑物图像的模型生成的(Karras等，2020b)

The moon had risen by the time I reached the edge of the forest, and the light that filtered through the trees was silver and cold. I shivered, though I was not cold, and quickened my pace. I had never been so far from the village before, and I was not sure what to expect. I had been walking for hours, and I was tired and hungry. I had left in such a hurry that I had not thought to pack any food, and I had not thought to bring a weapon. I was unarmed and alone in a strange place, and I did not know what I was doing.

I had been walking for so long that I had lost all sense of time, and I had no idea how far I had come. I only knew that I had to keep going. I had to find her. I was getting close. I could feel it. She was nearby, and she was in trouble. I had to find her and help her, before it was too late.

图1.6 文本数据生成模型合成的短篇故事。该模型描述了一个概率分布，为每个输出字符串分配一个概率。从模型中抽样可以创建遵循训练数据(这里是短篇故事)统计特征但之前从未见过的字符串

图1.7 图像修复。在原始图像(左)中，男孩被金属电缆遮挡。这些不希望出现的区域(中)被移除，生成模型合成了一个新图像(右)，在这个过程中保持剩余像素不变(Saharia等，2022a)

I was a little nervous before my first lecture at the University of Bath. It seemed like there were hundreds of students and they looked intimidating. I stepped up to the lectern and was about to speak when something bizarre happened.

Suddenly, the room was filled with a deafening noise, like a giant roar. It was so loud that I couldn't hear anything else and I had to cover my ears. I could see the students looking around, confused and frightened. Then, as quickly as it had started, the noise stopped and the room was silent again.

I stood there for a few moments, trying to make sense of what had just happened. Then I realized that the students were all staring at me, waiting for me to say something. I tried to think of something witty or clever to say, but my mind was blank. So I just said, "Well, that was strange,' and then I started my lecture.

图1.8 条件文本合成。给定一段初始文本(黑色部分)，文本的生成模型可以通过合成"缺失"的剩余部分来可信地延续字符串。由GPT3生成(Brown等，2020)

### 1.2.2　潜变量

一些(但并非全部)生成模型利用了这样一个观察结果：数据的内在维度可能比观察到的变量维度总数所暗示的要少。例如，有效且有意义的英语句子数量远少于通过随机抽取单词并排列组合生成的字符串数量。同样，真实世界的图像只是通过对每个像素随机抽取 RGB 值而创建的图像中极少的一部分，这是因为图像是由物理过程产生的(见图1.9)。

图1.9　人脸的变化。人脸大约有42块肌肉，因此可用大约42个数字来描述同一个人在相同光照下的图像中的大部分变化。一般来说，图像、音乐和文本的数据集可以用较少的潜变量来描述，尽管将这些变量与特定的物理机制联系起来通常更困难。图片来自Dynamic FACES数据库(Holland等，2019)

这引出了一个观点，即可用更少数量的潜变量来描述每个数据样本。这里，深度学习的作用是描述这些潜变量与数据之间的映射。获得的潜变量可以有一个简单的概率分布。通过从这个分布中抽样并传递给深度学习模型，可以创建新的样本(图1.10)。

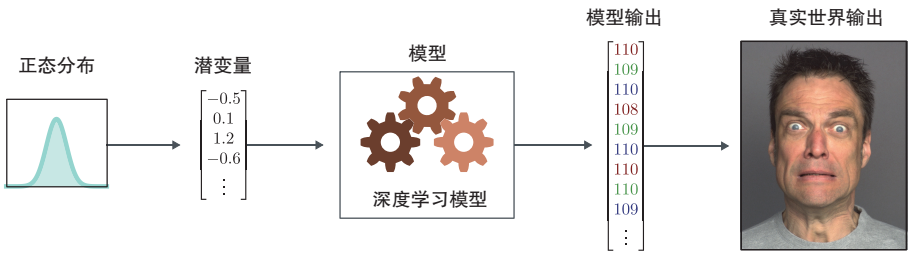



图1.10　潜变量。许多生成模型使用深度学习模型来描述低维“潜”变量与观察到的高维数据之间的关系。潜变量具有简单的概率分布。因此，可通过从潜变量的简单分布中采样，然后使用深度学习模型将样本映射到观察数据空间来生成新的样本

这些模型开启了操纵真实数据的全新方法。例如，考虑找出支撑两个真实样本的潜变量。可通过在它们的潜在表示之间插值，并将中间位置映射回数据空间来实现这些样本之间的插值(图1.11)。

图1.11 图像插值。在每一行中，左右的图像是真实的，中间的三个图像代表由生成模型创建的一系列插值。这些插值的基础生成模型已经学会所有图像都可以由一组潜变量创建。通过找到这两个真实图像的潜变量，在它们之间插值，然后使用这些中间变量创建新图像，从而生成视觉上可信又混合了两个原始图像特征的中间结果(Sauer等，2022；Ramesh等，2022)

### 1.2.3 联系监督学习和无监督学习

具有潜变量的生成模型也可促进监督学习模型的发展，特别是在输出具有结构时(见图1.4)。例如，考虑学习如何预测与标题对应的图像。可以不直接将文本输入映射到图像上，而是学习解释文本的潜变量与解释图像的潜变量之间的关系。

这样做有三个优点。首先，由于输入和输出的维度较低，我们可能需要更少的文本/图像对来学习这种映射。其次，我们更可能生成看起来合理的图像；任何合理的潜变量值都应该产生像合理样本的东西。最后，如果在两组潜变量之间的映射或从潜变量到图像的映射中引入随机性，那么可生成多个都能被标题很好地描述的图像(见图1.12)。

图1.12 由标题“时代广场上玩滑板的泰迪熊”生成的多幅图像。由DALL-E 2生成(Ramesh等，2022)

## 1.3　强化学习

机器学习的最后一个领域是强化学习。这一范式引入了“代理”概念，代理存在于一个世界中，可在每个时间步骤执行特定的动作。这些动作会改变系统的状态，但这种改变并不总是确定的。执行动作也会产生奖励，强化学习的目标是让“代理”学会选择能带来高奖励的动作。

一个难点是奖励可能在动作执行后一段时间才出现，因此奖励与动作的关联并不直接。这称为时间信用分配问题。随着代理的学习，它必须在探索和利用已有知识之间进行权衡；也许代理已经学会了如何获得适度的奖励；它应该遵循这种策略(利用已知知识)，还是应该尝试用不同的动作来检验是否有进一步提升的可能性(探索其他机会)?

### 两个例子

第一个例子考虑训练一个人形机器人的移动。机器人在任何时候都可执行有限数量的动作(移动各个关节)，这些动作会改变机器人的世界状态(它的姿势)。我们可能会因为机器人到达障碍赛中的检查点奖励它。要到达每个检查点，它必须执行许多动作，且当它收到奖励时，不清楚哪些动作与奖励有关，哪些是无关的。这就是时间信用分配问题的一个例子。

第二个例子是学习下棋。同样，代理在任何时候都有一组有效的动作(移动棋子)。然而，这些动作以非确定方式改变系统状态；对于选择的任何动作，对手都可能做出许多不同的反馈。这里，可能会在吃掉棋子时设置奖励，或仅在游戏结束时获得单一奖励来设置奖励结构。在后一种情况下，时间信用分配问题是极端的；系统必须学习它所执行的大量动作，并了解哪些对成功或失败起到了关键作用。

探索-利用权衡在这两个例子中也很明显。机器人可能已经发现，它可以通过侧卧并用一条腿推的方式向前移动。这种策略会推动机器人，并带来奖励，但比最佳解决方案(双腿平衡行走)要慢得多。因此，它面临着两种选择：利用它已经知道的东西(如何笨拙地滑过地板)或探索更多的动作空间(可能加快移动速度)。同样，在国际象棋的例子中，代理可能学会了一系列合理的开局。它应该利用这些知识还是探索不同的开局顺序？

或许，深度学习如何融入强化学习框架并不明显。有几种可能的方法，但一种技术是使用深度网络构建从观察到的世界状态到动作的映射。这称为策略网络。在机器人例子中，策略网络将学习从其传感器测量数据到关节运动的映射。在国际象棋例子中，策略网络将学习从棋盘的当前状态到移动选择的映射(图1.13)。

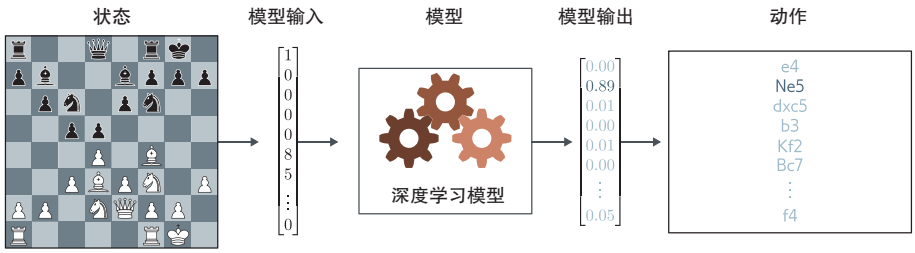



图1.13 强化学习的策略网络。将深度神经网络融入强化学习的一种方法是使用它们定义从状态(棋盘上的位置)到动作(可能的移动)的映射。这种映射被称为策略(Pablok，2017)

## 1.4 伦理学

不讨论人工智能的伦理影响将是不负责任的。这种强大的技术将至少像电力、内燃机、晶体管或互联网那样改变世界。在医疗保健、设计、娱乐、交通、教育和几乎所有商业领域，存在巨大的潜在好处。然而，科学家和工程师对他们工作的成果往往过于乐观，而人工智能(AI)潜在的危害却同样巨大。以下几段将重点阐述五个令人担忧的问题。

(1) 偏见与公平。如果我们基于历史数据，训练一个系统来预测个人的薪资水平，那么这个系统将复制历史上的偏见；例如，它可能预测女性应该比男性获得更低的薪酬。已经有几个此类案例成为国际新闻：一个用于超分辨率人脸图像的AI系统使非白人看起来更白；一个用于生成图像的系统在被要求合成律师的图片时只生成了男性的图片。AI算法决策的不慎应用，可能加剧或加深现有的偏见(Binns，2018)。

(2) 可解释性。深度学习系统可以做出决策，但是我们通常无法确切知道它是如何做出的，或者是基于哪些信息的。它们可能包含数十亿个参数，我们无法仅通过检查参数来理解它们的工作方式。这催生了可解释人工智能这一子领域。在这个领域，可进行局部解释；我们无法解释整个系统，但可以产生一个可解释的描述，说明如何做出了特定决策。然而，构建完全透明的复杂决策系统，使开发者和使用者都能完全理解其运作方式，目前仍是一个未知领域(Grennan等，2022)。

(3) 武器化AI。所有重要的技术都直接或间接地应用于战争。可悲的是，暴力冲突似乎是人类行为的一个不可避免的特征。人工智能无疑是有史以来人类构建的最强大的技术，并且无疑会在军事领域中得到广泛应用。事实上，这已经发生了(Heikkilä，2022)。

(4) 权力集中。世界上最强大的公司之所以大举投资人工智能，并非出于改

善人类境况的善意。他们知道这些技术将带来巨额利润。像任何先进技术一样，深度学习可能将权力集中在控制它的少数组织手中。自动化目前由人类完成的工作，将改变经济环境，对技能较少的低薪工人的生计产生不成比例的影响。乐观主义者认为，类似的颠覆事件在工业革命期间也曾发生过，并导致了工作时间的缩短。事实是，我们根本无法预知大规模采用人工智能会对社会产生哪些影响( David，2015)。

(5) 生存威胁。对人类构成重大生存威胁的因素都源于科技。气候变化是由工业化驱动的。核武器来自物理研究。由于交通、农业和建筑领域的创新使人口更加庞大、人群更密集和人与人之间的联系更加紧密，大流行病也更容易发生并更迅速传播。AI带来了新的生存威胁。我们应该非常谨慎地构建比人类更强大和更可扩展的系统。在最乐观的情况下，AI将把巨大的权力交到所有者手中。在最悲观的情况下，我们将无法控制它，甚至无法理解它的动机(Tegmark，2018)。

这个列表远未详尽。人工智能还可能导致监控、虚假信息、隐私侵犯、欺诈和金融市场操纵，训练人工智能系统所需的能源也会加剧气候变化。此外，这些担忧并非臆测；已经有很多违背伦理的人工智能应用案例(Dao，2021)。此外，互联网的近期历史表明，新技术会以意想不到的方式造成伤害。20世纪80年代和90年代网络普及之初，人们无法预测之后会出现假新闻、垃圾邮件、网络骚扰、欺诈、网络欺凌、政治操纵、泄露个人信息、网络激进化和报复色情的泛滥。

所有研究人工智能的人士都应该思考科学家对其技术应用负有多大责任。我们应该考虑到资本是推动人工智能发展的主要力量，针对法律进步和社会福祉的部署可能会大大滞后。科学家和工程师应该思考是否可能控制这个领域的进步并减少潜在的危害。应该考虑愿意为哪种类型的组织工作，组织对减少人工智能潜在危害的承诺有多认真？他们只是在“洗白”道德以降低声誉风险，还是真的在实施机制阻止可疑的项目？

鼓励所有读者进一步调查这些问题。网站https://ethics-of-ai.mooc.fi/上的在线课程是一个有用的入门资源。如果你使用本书授课，那么鼓励你与学生讨论这些问题。如果你是参加没有讨论这些问题的课程的学生，那么请游说你的教授这样做。如果你在企业环境中部署或研究人工智能，则鼓励你仔细审查雇主的价值观。

## 1.5　本书编排方式

本书第2~9章逐步介绍监督学习，会介绍浅层和深度神经网络，并讨论如何训练它们、评估和提高它们的性能。第10~13章介绍深度神经网络的常见架构变化，

包括卷积网络、残差网络和变换器(Transformer)。这些架构被广泛应用于监督学习、无监督学习和强化学习中。

第14~18章介绍深度神经网络如何进行无监督学习。将介绍4种现代深度生成模型：生成式对抗网络、变分自编码器、标准化流和扩散模型。第19章是对深度强化学习的简要介绍。这是一个足以单独成书的主题，因此本书的介绍必然是浅尝辄止的。然而，对于不熟悉该领域的朋友来说，它仍然是一个很好的入门。

尽管本书的书名是“理解深度学习”，但深度学习仍有一些方面没有被很好地理解。第20章提出了几个基本问题。为什么深度网络如此容易训练？为什么它们泛化得这么好？为什么它们需要这么多参数？它们需要很深的网络吗？在此过程中，探讨一些意想不到的现象，如损失函数的结构、双重下降、顿悟现象和彩票假设。最后，在第21章讨论深度学习的伦理。

## 1.6 其他书籍

本书是独立成篇的，但仅限于深度学习领域的内容。它旨在成为《深度学习》(Goodfellow等，2016)的续作。《深度学习》是一部极好的资源，但没有涵盖近期的进展。对于更广泛的机器学习领域，最新且全面的资源是《概率机器学习》(Murphy，2022，2023)。然而，《模式识别与机器学习》(Bishop，2006)仍然是一本优秀且极具参考价值的书籍。

如果你喜欢这本书，那么我的上一部作品《计算机视觉：模型、学习和推理》仍然值得一读。尽管部分内容稍显过时，但它包含了完整的概率学入门知识，介绍贝叶斯方法，以及潜变量模型、计算机视觉的几何学、高斯过程和图形模型。该书使用与本书相同的符号，并可在网上找到。关于图形模型的详细论述，可以参考《概率图形模型：原理与技术》(Koller和Friedman，2009)，高斯过程则可以参考《机器学习中的高斯过程》(Williams和Rasmussen，2006)。

对于数学背景知识，可参考《机器学习的数学》(Deisenroth等，2020)。关于编程实践，可以参考《深入深度学习》(Zhang等，2023)。关于计算机视觉，最佳的综述是*Szeliski*，还有即将出版的《计算机视觉基础》(Torralba 等，2024)。学习图神经网络的一个良好的起点是《图表示学习》(Hamilton，2020)。关于强化学习的权威著作是《强化学习：导论》(Sutton 和 Barto，2018)。一个很好的入门学习资源是《深度强化学习基础》(Graesser 和 Keng，2019)。

## 1.7　如何阅读本书

本书大部分章节都包含正文、注释部分和问题集。正文旨在独立成篇，同时，尽可能多地融入数学背景知识。然而，对于那些会分散主旨论述的较大主题，相关的背景材料会被放在附录中。本书中的大多数符号都是标准的。然而，也会使用一些没有被广泛使用的约定，建议读者在继续之前先参阅附录A。

正文包含许多关于深度学习模型和运行结果的新颖插图和可视化图表。我努力提供对现有概念的新解释，而不仅仅是整理他人的成果。深度学习是一个新领域，有时现象是很难理解的。我尽力阐明哪些部分属于这种情况，以及何时应该谨慎对待我的解释。

只有在描绘结果的地方，才会将参考文献包含在章节的正文中。相反，通常在章节末尾的“注释”部分可以找到参考文献。在正文中，我通常不会尊重历史先例；如果一个当前技术的先例已不再有用，将不会提及它。但是，注释部分会介绍该领域的发展史。注释应该可以帮助读者认识相应的子领域，并了解它与机器学习的其他部分的关系。与正文相比，注释部分不那么独立。根据你的背景知识和兴趣，你可能会觉得这些部分更有用或者没那么有用。

每章都有一系列问题。正如George Pólya所指出的：“数学，你看，不是一项观赏运动。”他是对的，我强烈推荐你在阅读时尝试解决这些问题。某些情况下，它们可以帮助你更好地理解正文。如果问题标有星号，则表明答案可以在相关网站上找到。此外，有助于你理解本书中的观点的Python代码也可通过网站获取(网址为https://udlbook.github.io/udlbook/)。事实上，如果你感觉生疏，现在就通过这些Python 程序来复习一下数学背景知识也是不错的选择。

遗憾的是，AI 研究的快速发展使得本书的编写不可避免地成为一个持续的过程。如果你发现书中有某些部分难以理解，有明显的遗漏，或者有某些部分多余了，请通过相关网站与我联系。我们可以共同使下一版变得更好。

# 第2章

# 监督学习

监督学习模型定义了从一个或多个输入到一个或多个输出的映射。例如，输入可以是二手丰田普锐斯的车龄和里程数，输出可以是汽车的估价(美元)。

这个模型只是一个数学函数；当输入通过这个函数时，它会计算输出，这称为推理。模型函数也包含参数。不同的参数值会改变计算结果；模型函数描述了输入和输出之间的一组可能的关系，而模型参数则指定了特定的关系。

当训练模型时，目标是寻找能描述输入和输出之间真实关系的参数。学习算法会取一个输入/输出对作为训练集，并调整参数，直到输入尽可能准确地预测其对应的输出。如果模型在这些训练对的数据上表现良好，那么我们希望它能对新输入数据(即真实输出未知的情况)做出良好的预测。

本章的目标是扩展这些概念。首先，将更正式地描述这个框架并引入一些符号。然后，通过一个简单示例演示如何用一条直线来描述输入与输出之间的关系。这个线性模型既为人熟知又易于可视化，但仍能很好地阐明监督学习的所有主要思想。

## 2.1 监督学习概述

在监督学习中，我们的目标是构建一个模型，它可以接收一个输入$\boldsymbol{x}$并输出一个预测$\boldsymbol{y}$。为简单起见，假设输入$\boldsymbol{x}$和输出$\boldsymbol{y}$都是内容已预先确定且大小固定的向量，并且每个向量的元素始终按照相同的方式排列；在前述的普锐斯示例中，输入$\boldsymbol{x}$总会按顺序包含汽车的车龄，然后是里程数。这类数据被称为结构化数据或表格数据。

为了做出预测，我们需要一个模型$\boldsymbol{f}[\bullet]$，它接收输入$\boldsymbol{x}$并返回输出$\boldsymbol{y}$，所以：

$$\boldsymbol{y} = \boldsymbol{f}[\boldsymbol{x}] \tag{2.1}$$

我们把用输入$\boldsymbol{x}$计算预测结果$\boldsymbol{y}$的过程称为推理。

模型只是一个固定形式的数学函数，代表了输入和输出之间一系列不同的关系。模型也包含参数$\boldsymbol{\phi}$。参数的选择决定了输入和输出之间的具体关系，所以应该更准确地写成：

$$\boldsymbol{y} = \boldsymbol{f}[\boldsymbol{x}, \boldsymbol{\phi}] \tag{2.2}$$

当谈论学习或训练模型时，我们的目的是试图找到能够通过输入做出合理输出预测的参数$\boldsymbol{\phi}$。我们使用一个包含$I$对输入和输出示例的训练数据集$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$来学习获得这些参数。目标是选择参数，尽可能准确地将每个训练输入数据映射到其相应的输出数据。用损失函数$\boldsymbol{L}$来量化这种映射的不匹配程度。损失是一个标量值，概括了在给定参数$\boldsymbol{\phi}$的情况下，模型从对应的输入中预测训练输出的误差。

可将损失$\boldsymbol{L}$视为参数$\boldsymbol{\phi}$的函数$\boldsymbol{L}[\boldsymbol{\phi}]$。当训练模型时，我们就是在寻找能使这个损失函数值最小的参数$\hat{\boldsymbol{\phi}}$：[1]

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\left[\boldsymbol{L}[\boldsymbol{\phi}]\right] \tag{2.3}$$

如果在完成这一最小化过程后损失函数的值很小，我们就已经找到了能准确预测输入$\boldsymbol{x}_i$到输出$\boldsymbol{y}_i$的模型参数。

训练完一个模型后，必须评估其性能；在独立的测试数据上运行模型，观测它在训练期间未观察到的样本上的泛化能力。如果性能令人满意，就可以准备部署模型了。

## 2.2　线性回归示例

现在，让我们通过一个简单例子来具体化这些概念。设想一个模型$\boldsymbol{y} = \boldsymbol{f}[\boldsymbol{x}, \boldsymbol{\phi}]$，它通过一个输入$\boldsymbol{x}$预测一个输出$\boldsymbol{y}$。接下来，我们会设计一个损失函数，最后讨论模型的训练过程。

### 2.2.1　一维(1D)线性回归模型

一维线性回归模型用一条直线描述输入$\boldsymbol{x}$和输出$\boldsymbol{y}$之间的关系：

1　更准确地说，损失函数还依赖于训练数据$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$，所以应该写作$\boldsymbol{L}[\{\boldsymbol{x}_i, \boldsymbol{y}_i\}, \boldsymbol{\phi}]$，但这样表示会显得相当繁杂。

$$\begin{aligned} y &= f[x,\phi] \\ &= \phi_0 + \phi_1 x \end{aligned} \tag{2.4}$$

该模型有两个参数 $\phi = [\phi_0, \phi_1]^T$，其中 $\phi_0$ 是直线的$y$轴截距，$\phi_1$ 是斜率。不同的$y$轴截距和斜率选择会决定输入和输出之间的不同关系(图2.1)。因此，式(2.4)定义了一组可能的输入-输出关系(所有可能的直线)，参数的选择决定了这一组中的具体成员(特定的直线)。

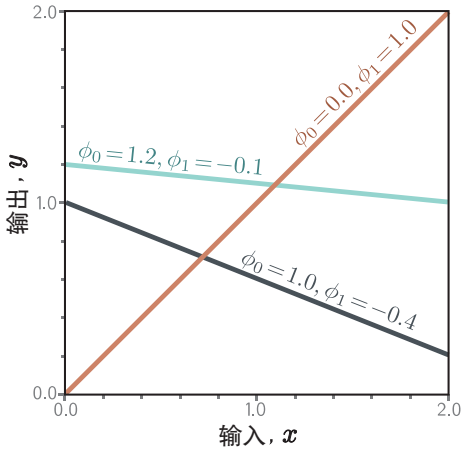



图2.1 线性回归模型。对于给定的参数 $\phi = [\phi_0, \phi_1]^T$，模型根据输入($x$轴)对输出($y$轴)进行预测。不同的截距 $\phi_0$ 和斜率 $\phi_1$ 的选择会改变这些预测结果(青色、橙色和灰色直线)。线性回归模型(式2.4)定义了一组输入/输出的关系(直线)，而参数决定了该组中的具体成员(特定的直线)

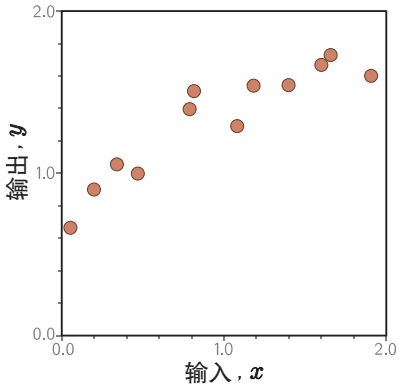
## 2.2.2 损失

对于这个模型，训练数据集(图2.2(a))由$I$个输入/输出对 $\{x_i, y_i\}$ 构成。图2.2(b)~(d)显示了由三组参数定义的三条直线。图2.2(d)中的绿色直线比其他两条线更准确地描述了数据，因为它更接近数据点。然而，我们需要一个有依据的方法来决定哪个参数 $\phi$ 比其他参数更好。为此，给每个参数分配一个数值，用该数值来量化模型与数据之间的不匹配程度。将这个值称为损失；更低的损失意味着更好的拟合效果。

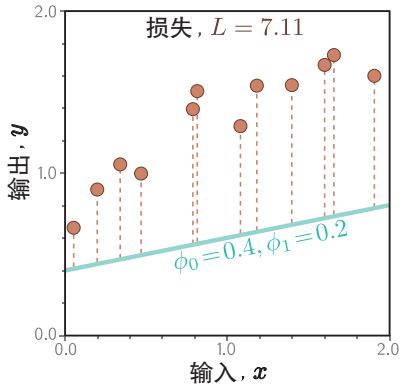
这种不匹配由模型的预测 $f[x_i, \phi]$ (在 $x_i$ 处线上方的高度) 和实际输出 $y_i$ 之间的偏差所捕获。这些偏差在图2.2(b)~(d)中以橙色虚线表示。将所有$I$个训练对的偏差的平方和称为总的不匹配、训练误差或损失：

$$\begin{aligned} L[\boldsymbol{\phi}] &= \sum_{i=1}^{I} (\boldsymbol{f}[\boldsymbol{x}_i, \boldsymbol{\phi}] - \boldsymbol{y}_i)^2 \\ &= \sum_{i=1}^{I} (\phi_0 + \phi_1 \boldsymbol{x}_i - \boldsymbol{y}_i)^2 \end{aligned} \tag{2.5}$$

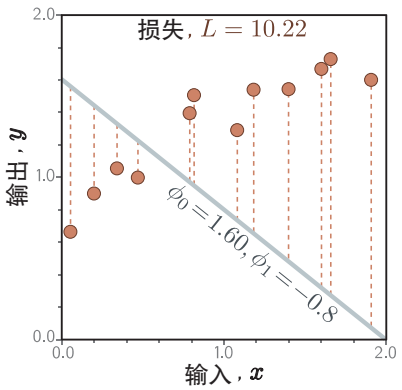
由于最佳参数会最小化这个式子的值，所以我们称之为最小二乘损失。平方计算意味着偏差的方向(即直线是在数据上方还是下方)无关重要。第5章中将再次讨论选择这样做的理论依据。



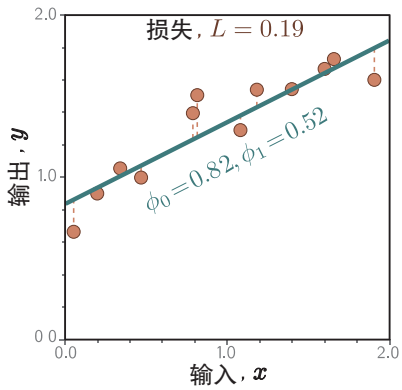

(a) 训练数据(橙色点)由$I$=12组输入/输出对$\{x_i, y_i\}$组成




(b) 图中定义直线的参数具有较大的损失$L$=7.11，是因为模型拟合得很差




(c) 图中定义直线的参数具有较大的损失$L$ =10.22，是因为模型拟合得很差



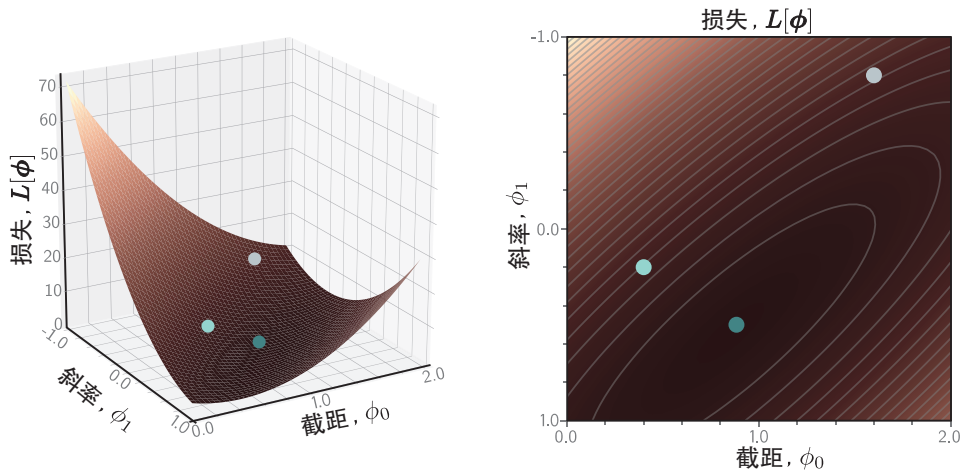

(d) 图中的损失$L$=0.19较小，是因为模型拟合得很好；实际上，这是所有可能直线中损失最小的，所以这是最优参数

图2.2　线性回归的训练数据、模型和损失。图(b)~(d)分别展示了具有不同参数的线性回归模型。根据截距和斜率参数$\boldsymbol{\phi} = [\phi_0, \phi_1]^T$的选择，模型误差(橙色虚线)可能更大或更小。损失$L$是这些误差平方的总和

损失$\boldsymbol{L}$是参数$\boldsymbol{\phi}$的函数；当模型拟合较差时(图2.2(b)和(c))损失更大，当拟合良好时(图2.2(d))损失更小。从这个角度看，将$\boldsymbol{L}[\boldsymbol{\phi}]$称为损失函数或代价函数。训练模型的目标是找到最小化这个值的参数$\hat{\boldsymbol{\phi}}$：

$$\begin{aligned}\hat{\boldsymbol{\phi}} &= \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\left[\boldsymbol{L}[\boldsymbol{\phi}]\right] \\ &= \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\left[\sum_{i=1}^{I}(\boldsymbol{f}[\boldsymbol{x}_i,\boldsymbol{\phi}]-\boldsymbol{y}_i)^2\right] \\ &= \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\left[\sum_{i=1}^{I}(\phi_0+\phi_1\boldsymbol{x}_i-\boldsymbol{y}_i)^2\right]\end{aligned} \tag{2.6}$$

只有两个参数($y$轴截距 $\phi_0$ 和斜率 $\phi_1$)，因此我们可计算每种参数值组合的损失，并将损失函数可视化为一个曲面(图2.3)。“最佳”参数位于该曲面的最低点。




(a) 每个参数组合 $\boldsymbol{\phi}=[\phi_0,\phi_1]^T$ 都有相应的损失。所得的损失函数 $\boldsymbol{L}[\boldsymbol{\phi}]$ 可以被可视化为一个曲面。三个圆圈对应于图2.2(b)~(d)中的三条直线

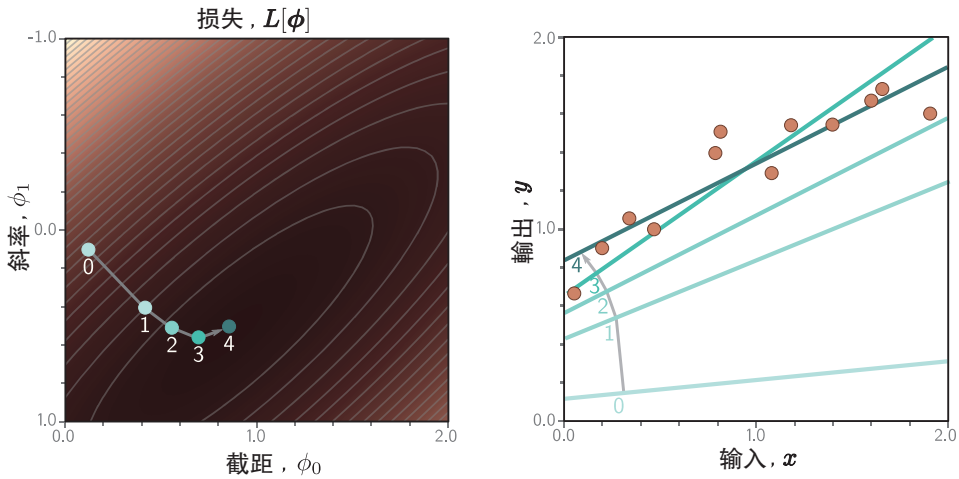
(b) 损失也可以被可视化为热力图，其中较亮的区域代表较大的损失；这里从上方直接看向(a)中的曲面，灰色椭圆代表等高线。最佳拟合直线(见图2.2(d))具有最小损失的参数(绿色圆圈)

图2.3 针对图2.2(a)的数据集的线性回归模型的损失函数

## 2.2.3 训练

寻找使损失最小化的参数的过程称为模型拟合、训练或学习。基本方法是随机选择初始参数，然后“沿着”损失函数“下降”的方向改进它们，直至到达最低点(图2.4)。一种方法是测量当前点所在曲面的梯度，并向最陡峭的下坡方向迈出一步。此后重复这个过程，直到梯度变得平坦，无法进一步改进为止[1]。

1 对于线性回归模型来说，这种迭代方法实际上并不是必要的。在这里，可找到参数的闭式表达式。然而，梯度下降法适用于更复杂的模型，这些模型中没有闭式解，并且参数太多，无法评估每种值组合的损失。




(a) 迭代训练算法随机初始化参数，然后通过“下坡”来改进它们，直到无法进一步改进为止。这里，从位置0开始，向下移动一定距离(垂直于等高线方向)到达位置1。然后重新计算下坡方向，移到位置2。最终，到达函数的最小值(位置4)

(b) 图(a)中的每个位置0~4对应于不同的截距和斜率，因此代表了不同的直线。随着损失的减少，这些直线会越来越贴合数据

图2.4　线性回归训练。训练目标是找到对应于最小损失的截距和斜率参数

### 2.2.4　测试

完成模型训练后，我们需要知道它在现实世界中的表现如何。通过在单独的测试数据集上计算损失来评估这一点。预测准确性会在何种程度上泛化到测试数据部分取决于训练数据的代表性和完整性，也取决于模型的表达能力。一个简单模型(如一条直线)可能无法捕捉输入和输出之间的真实关系，这称为欠拟合。相反，一个表达能力很强的模型可能描述训练数据的一些非典型的统计特性，同时会引起异常的预测。这被称为过拟合。

## 2.3　本章小结

监督学习模型是一个函数 $\boldsymbol{y}=\boldsymbol{f}[\boldsymbol{x},\boldsymbol{\phi}]$，将输入$\boldsymbol{x}$与输出$\boldsymbol{y}$联系起来。特定的关系由参数 $\boldsymbol{\phi}$ 决定。为训练模型，我们在训练数据集 $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$ 上定义了一个损失函数 $\boldsymbol{L}[\boldsymbol{\phi}]$。它将模型预测 $\boldsymbol{f}[\boldsymbol{x}_i,\boldsymbol{\phi}]$ 和观察到的输出 $\boldsymbol{y}_i$ 之间的误差量化为参数 $\boldsymbol{\phi}$ 的函数。然后搜索使损失最小化的参数。在不同的测试数据集上评估模型，观察它对新输入的泛化能力如何。

第3~9章将扩展这些概念。首先，研究模型本身；线性回归的明显缺点是它只能将输入和输出之间的关系描述成一条直线。浅层神经网络(第 3 章)比线性回归复杂不了多少，但可以描述更大范围的输入/输出关系。深度神经网络(第4章)具有同样的表达能力，但可以用更少的参数描述复杂函数，并且在实践中效果更好。

第5章研究不同任务的损失函数，并揭示最小二乘损失的理论基础。第 6 章和第7章讨论训练过程。第 8 章讨论如何衡量模型性能。第 9 章探讨旨在提高性能的正则化技术。

## 2.4 注释

**损失函数与代价函数：**在机器学习的大部分领域以及本书中，术语“损失函数”和“代价函数”可以互换使用。然而，更确切地说，损失函数是与单个数据点相关的项(即式(2.5)右侧的每一个平方项)，而代价函数是需要最小化的整体量(即式(2.5)右侧的表达式)。代价函数可以包含与单个数据点无关的附加项(见第9.1节)。更一般地说，目标函数是任何一个需要被最大化或最小化的函数。

**生成模型与判别模型：**本章中的 $\boldsymbol{y}=\boldsymbol{f}[\boldsymbol{x},\boldsymbol{\phi}]$ 是判别模型。它们根据现实世界的测量值$\boldsymbol{x}$做出输出预测$\boldsymbol{y}$。另一种方法是构建生成模型 $\boldsymbol{x}=\boldsymbol{g}[\boldsymbol{y},\boldsymbol{\phi}]$，其中现实世界的测量值$\boldsymbol{x}$是输出$\boldsymbol{y}$的函数。

生成方法的缺点是它不能直接预测$\boldsymbol{y}$。为进行推理，必须将生成函数反转为 $\boldsymbol{y}=\boldsymbol{g}^{-1}[\boldsymbol{x},\boldsymbol{\phi}]$，这可能很难。然而，生成模型的优点是可将关于数据生成过程的先验知识构建到其中。例如，如果想预测图像$\boldsymbol{x}$中汽车的三维位置和方向$\boldsymbol{y}$，那么可将关于汽车形状、三维几何和光线传输的知识纳入函数 $\boldsymbol{x}=\boldsymbol{g}[\boldsymbol{y},\boldsymbol{\phi}]$ 中。

这似乎是一个好主意，但事实上，判别模型主导了现代机器学习；利用生成模型中的先验知识获得的优势通常被利用大量训练数据学习出的更灵活的判别模型所超越。

## 2.5 问题

**问题2.1** 为在损失函数(式(2.5))上“向下走”，我们需要计算其相对于参数 $\phi_0$ 和 $\phi_1$ 的梯度。计算斜率 $\partial L/\partial\phi_0$ 和 $\partial L/\partial\phi_1$ 的表达式。

**问题2.2** 证明可通过将问题2.1的导数表达式设置为零并通过解出 $\phi_0$ 和 $\phi_1$，以闭式形式找到损失函数的最小值。注意，这对于线性回归是有效的，但不适用于更复杂的模型；这就是为什么要使用类似梯度下降这样的迭代模型拟合方法(图2.4)。

问题2.3*　考虑将线性回归重新表述为生成模型，即 $\boldsymbol{x}=\boldsymbol{g}[\boldsymbol{y},\boldsymbol{\phi}]=\phi_0+\phi_1\boldsymbol{y}$。新的损失函数是什么？找到一个用于推断的逆函数 $\boldsymbol{y}=\boldsymbol{g}^{-1}[\boldsymbol{x},\boldsymbol{\phi}]$ 的表达式。对于给定的训练数据集 $\{\boldsymbol{x}_i,\boldsymbol{y}_i\}$，这个模型是否会做出与判别版本相同的预测？一种方法是编写代码来确认它，用这两种方法分别拟合一条通过三个数据点的直线，并查看结果是否相同。

# 第3章

# 浅层神经网络

第2章使用一维线性回归公式介绍了监督学习。然而，这个模型只能将输入/输出关系描述为一条直线。本章介绍浅层神经网络(shallow neural networks)。它能够描述的是分段线性函数，并且具有足够强的表达能力来近似任意复杂的多维输入和输出之间的关系。

## 3.1 神经网络示例

浅层神经网络的表现形式是函数$y=f[x,\phi]$，通过使用参数$\phi$将多变量输入$x$映射到多变量输出$y$。第3.4节将提供完整定义，并使用一个示例网络$f[x,\phi]$来介绍主要概念，该网络将标量输入$x$映射到标量输出$y$，并具有十个参数$\phi=\{\phi_0,\phi_1,\phi_2,\phi_3,\theta_{10},\theta_{11},\theta_{20},\theta_{21},\theta_{30},\theta_{31}\}$：

$$\begin{aligned} y &= f[x,\phi] \\ &= \phi_0+\phi_1 a[\theta_{10}+\theta_{11}x]+\phi_2 a[\theta_{20}+\theta_{21}x]+\phi_3 a[\theta_{30}+\theta_{31}x] \end{aligned} \tag{3.1}$$

可将这个计算分解成三个部分。首先计算输入数据的三个线性函数$(\theta_{10}+\theta_{11}x$、$\theta_{20}+\theta_{21}x$和$\theta_{30}+\theta_{31}x)$。然后将这三个结果传递给激活函数$a[\bullet]$。最后将三个激活结果与$\phi_1$、$\phi_2$和$\phi_3$加权获得激活值，再将它们求和后加上偏移量$\phi_0$。

为完成上面的计算，必须定义激活函数$a[\bullet]$。它有很多种可能的选择，但最常见的是整流线性单元(rectified linear unit，ReLU)：

$$a[z]=\mathrm{ReLU}[z]=\begin{cases}0 & z<0\\ z & z\geqslant 0\end{cases} \tag{3.2}$$

当输入值为正时返回输入值，否则返回零(图3.1)。

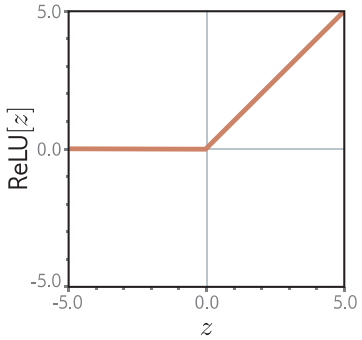



图3.1　激活函数ReLU在输入值小于零时返回零，否则直接返回输入值。换句话说，它将负值截断为零。请注意，还有其他很多可能的激活函数选择(见图3.13)，但ReLU是最常用的，也是最容易理解的

可能不能直观地看出式(3.1)代表了哪个输入/输出关系。然而，上一章的所有概念在这里都适用。式(3.1)表示一组函数，该组中的特定成员取决于十个参数 $\phi$。如果知道了这些参数，就可以使用式子对给定输入$x$进行推理(预测$y$)。给定一个训练数据集 $\{x_i, y_i\}_{i=1}^{I}$，就可以定义一个最小二乘损失函数 $L[\phi]$，并用它来衡量任何给定的参数值 $\phi$ 的模型表述这个数据集的效果。训练模型，就是寻找最小化这个损失的参数值 $\hat{\phi}$。

### 3.1.1　神经网络的直观理解

实际上，式(3.1)表示一个最多具有四个线性区域的连续分段线性函数系列(图3.2)。现在将分解式(3.1)并分析它如何描述这个函数系列。为便于理解，将函数分成两部分。引入中间量：

$$\begin{aligned} h_1 &= a[\theta_{10} + \theta_{11}x] \\ h_2 &= a[\theta_{20} + \theta_{21}x] \\ h_3 &= a[\theta_{30} + \theta_{31}x] \end{aligned} \tag{3.3}$$

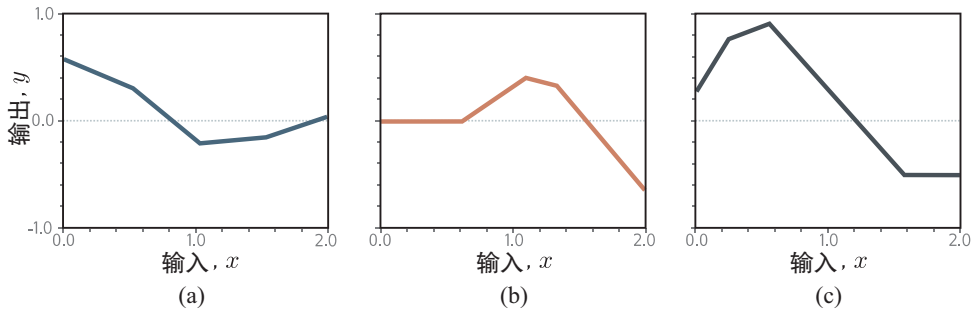



图3.2　由式(3.1)定义的函数系列。(a)~(c)表示三种不同参数 $\phi$ 对应的函数。每种情况下，输入/输出关系都是分段线性的。然而，连接点的位置、连接点之间线性区域的斜率及整体高度都有所不同

这里称其中的 $h_1$、$h_2$ 和 $h_3$ 为隐藏单元。我们通过将这些隐藏单元与一个线性函数结合来计算输出：[1]

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3 \tag{3.4}$$

图3.3显示了创建图3.2(a)中函数的计算流程。每个隐藏单元都包含输入的线性函数 $\theta_{\bullet 0} + 0_{\bullet 1} x$，并且该直线零以下的部分被ReLU 函数$a[\bullet]$截断了。三条线与零相交的位置成为最终输出中的三个“关节”。然后，这三条被截断的线分别被 $\phi_1$、$\phi_2$ 和 $\phi_3$ 加权。最后，加上偏移 $\phi_0$，它控制着最终函数的整体高度。

图3.3(j)中的每个线性区域对应于隐藏单元中不同的激活模式。当一个单元被截断时，我们称它为非活跃状态；当它没有被截断时，我们称它为活跃状态。例如，阴影区域接收来自 $h_1$ 和 $h_3$ 的贡献(它们是活跃的)，但不接收来自 $h_2$ 的贡献(它是非活跃的)。每个线性区域的斜率由以下几点决定：①该区域活跃输入的原始斜率 $\theta_{\bullet 1}$；②随后应用的权重 $\phi_\bullet$。例如，阴影区域的斜率(见问题3.3)是 $\theta_{11}\phi_1 + \theta_{31}\phi_3$，其中第一项是子图(g)中的斜率，第二项是子图(i)中的斜率。

每个隐藏单元为函数贡献一个“关节”，因此具有三个隐藏单元的函数可以有四个线性区域。然而，这些区域的斜率只有三个是独立的；第四个要么是零(如果在这个区域所有隐藏单元都是非活跃的)，要么是其他区域斜率的总和。

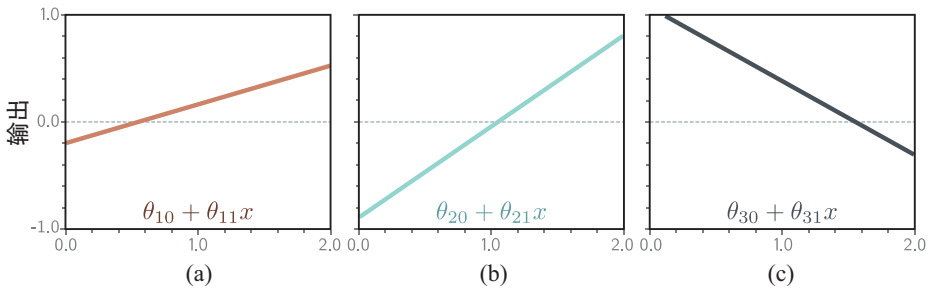



图3.3 图3.2(a)所示函数的计算过程。(a)~(c)中，输入$x$传递给三个线性函数，每个函数具有不同的$y$轴截距 $\theta_{\bullet 0}$ 和斜率 $\theta_{\bullet 1}$。(d)~(f)中，每条线都通过ReLU激活函数进行处理，将负值截断为零。(g)~(i)中，三条被截断的线分别被 $\phi_1$、$\phi_2$ 和 $\phi_3$ 加权(缩放)。(j)中，将这些截断且加权后的函数相加，再加上一个控制高度的偏移量 $\phi_0$。四个线性区域中的每一个都对应于隐藏单元中不同的激活模式。在阴影区域中，$h_2$ 是非活跃的(被截断)，但 $h_1$ 和 $h_3$ 都是活跃的

1 本书使用的线性函数的形式为 $z' = \phi_0 + \sum_i \phi_i z_i$。其他任何类型的函数都是非线性的。如ReLU函数(式(3.2))和包含它的示例神经网络(式(3.1))都是非线性的。有关进一步说明，请参考本章末尾的注释。

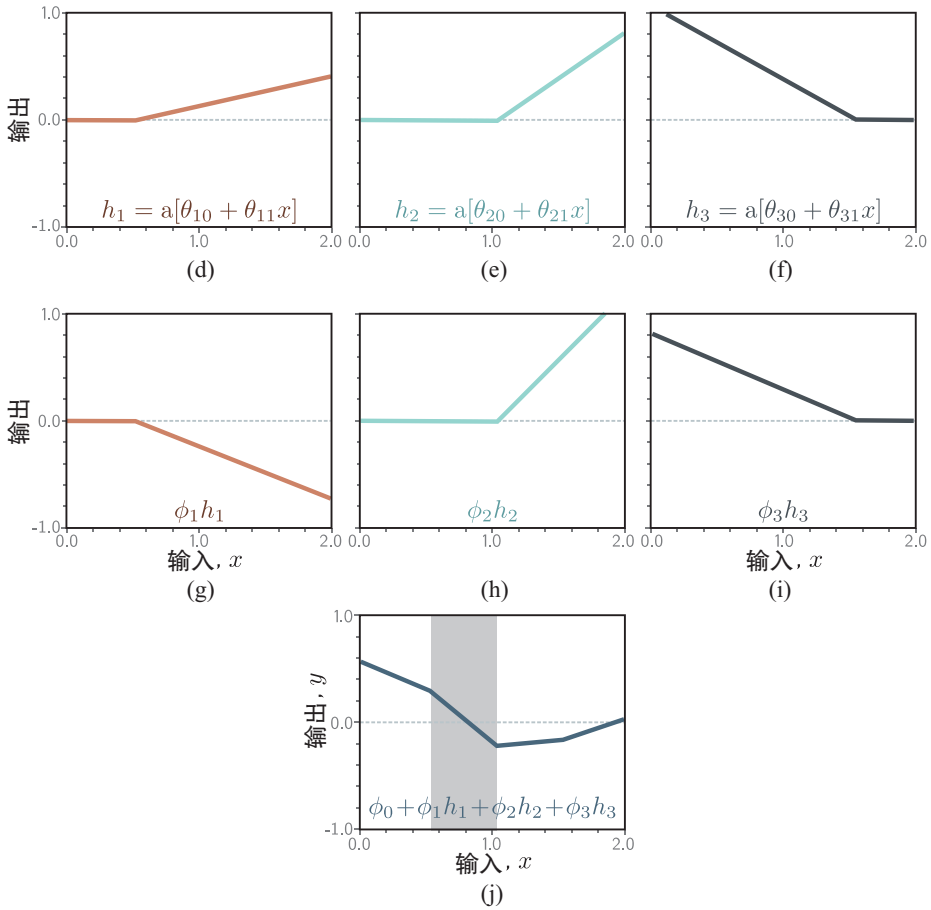
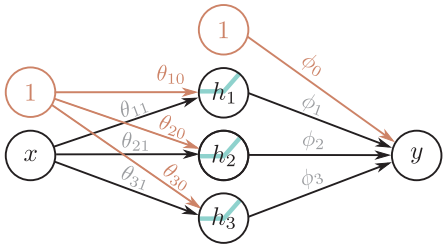


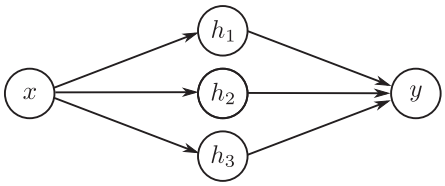

图3.3　(续)

## 3.1.2　描绘神经网络

我们一直在讨论一个具有一个输入、一个输出和三个隐藏单元的神经网络。图3.4(a)中显示了这个网络。输入在左侧，隐藏单元在中间，输出在右侧。每个连接代表十个参数中的一个。为简化这种表示，通常不会画出截距参数，因此该网络通常如图3.4(b)所示。




(a) 输入$x$在左侧，隐藏单元$h_1$、$h_2$和$h_3$在中间，输出$y$在右侧。计算流程从左到右。输入用于计算隐藏单元，这些隐藏单元被组合起来生成输出。十条箭头中的每一条都代表一个参数(橙色表示截距，黑色表示斜率)。每个参数将其自身乘以来源并将结果加到目标上。例如，将参数$\phi_1$乘以来源$h_1$，然后加到$y$上。我们引入额外的节点(包含数字1，橙色圆圈)以将偏移量纳入这个方案中，所以将$\phi_0$乘以1(没有效果)，然后加到$y$上。ReLU函数应用于隐藏单元




(b) 更常见的是，省略了截距、ReLU函数和参数名称；这个更简单的描述仍然代表了相同的网络

图3.4 描述神经网络

## 3.2 通用逼近定理

在上一节中，我们介绍了一个具有一个输入、一个输出、一个ReLU激活函数和三个隐藏单元的示例神经网络。现在将这一概念稍微推广一下，考虑具有$D$个隐藏单元的情况，其中第$d$个隐藏单元为：

$$h_d = a\left[\theta_{d0} + \theta_{d1}x\right] \tag{3.5}$$

这些隐藏单元通过线性组合输出为：

$$y = \phi_0 + \sum_{d=1}^{D} \phi_d h_d \tag{3.6}$$

浅层网络中隐藏单元的数量是网络容量的衡量标准。使用ReLU激活函数，具有$D$个隐藏单元的网络的输出最多有$D$个关节，因此它是一个最多有$D$+1个线性区域的分段线性函数。随着我们添加更多的隐藏单元，模型可以近似更复杂的函数。

事实上，具有足够容量(隐藏单元)的浅层网络可以任意精度描述任何定义在实数线上紧凑子集的连续一维函数。要理解这一点，可以考虑在每次添加一个隐藏单元时，都为函数添加一个新的线性区域。随着这些线性区域逐渐增多，它们代表的函数部分不断缩短，就可用线段越来越逼近目标曲线(图3.5)。神经网络逼

近任何连续函数的能力可以通过数学推理来证明，这称为通用逼近定理。

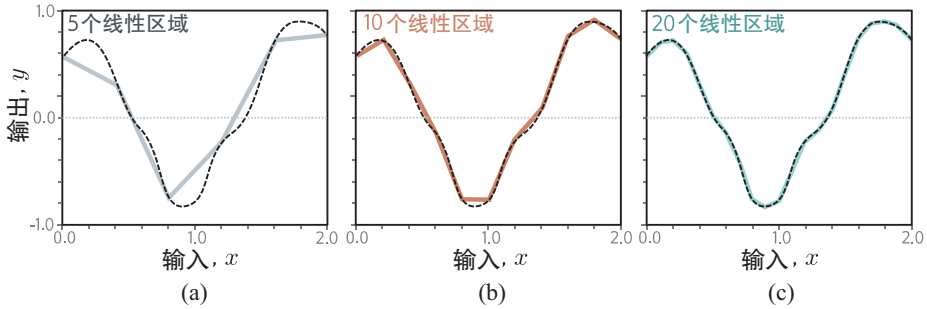



图3.5　通过分段线性模型逼近一维函数(虚线)。(a)~(c)中，随着区域数量的增加，模型越来越接近连续函数。具有标量输入的神经网络每增加一个隐藏单元就创建一个额外的线性区域。由通用逼近定理可知，只要有足够的隐藏单元，浅层神经网络就能以任意精度描述在 $\mathbb{R}^D$ 的紧凑子集上定义的任何连续函数

## 3.3　多变量输入和输出

在上例中，网络有一个标量输入$x$和一个标量输出$y$。然而，通用逼近定理也适用于更一般的情况，即网络将多变量输入 $\boldsymbol{x}=[x_1, x_2,..., x_{D_i}]^{\mathrm{T}}$ 映射到多变量输出预测 $\boldsymbol{y}=[y_1, y_2,..., y_{D_o}]^{\mathrm{T}}$。我们首先探讨如何通过扩展模型来预测多变量输出。然后考虑多变量输入。最后，在第 3.4 节中，将给出浅层神经网络的通用定义。

### 3.3.1　可视化多变量输出

为将网络扩展到多变量输出$\boldsymbol{y}$，只需要为每个输出使用隐藏单元的不同线性函数。因此，一个具有标量输入$x$、四个隐藏单元 $h_1$、$h_2$、$h_3$ 和 $h_4$，以及二维多变量输出 $\boldsymbol{y}=[y_1, y_2]^{\mathrm{T}}$ 的网络可以被定义为：

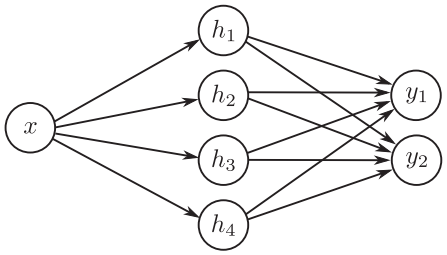
$$\begin{aligned} h_1 &= a[\theta_{10}+\theta_{11}x] \\ h_2 &= a[\theta_{20}+\theta_{21}x] \\ h_3 &= a[\theta_{30}+\theta_{31}x] \\ h_4 &= a[\theta_{40}+\theta_{41}x] \end{aligned} \tag{3.7}$$

和

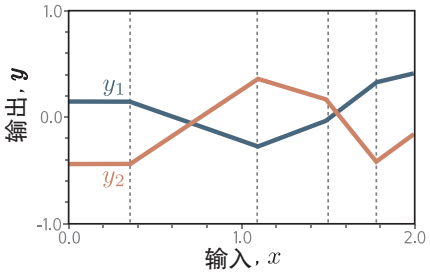
$$\begin{aligned} y_1 &= \phi_{10}+\phi_{11}h_1+\phi_{12}h_2+\phi_{13}h_3+\phi_{14}h_4 \\ y_2 &= \phi_{20}+\phi_{21}h_1+\phi_{22}h_2+\phi_{23}h_3+\phi_{24}h_4 \end{aligned} \tag{3.8}$$

这两个输出是隐藏单元的两个不同的线性函数。

正如在图3.3 中看到的，分段函数中的“关节”取决于隐藏单元中初始线性函数 $\theta_{\bullet 0}+\theta_{\bullet 1}x$ 被 ReLU 函数$a[\bullet]$截断的位置。由于两个输出$y_1$和$y_2$都是相同的四个隐藏单元的不同线性函数，每个输出中的四个“关节”必须位于相同的位置。然而，线性区域的斜率和整体垂直偏移可以有所不同(图3.6)。




(a) 网络结构的可视化表示




(b) 这个网络产生了两个分段线性函数，$y_1[x]$和$y_2[x]$。这些函数的四个“关节”(在垂直虚线处)因为共享相同的隐藏单元，所以必须位于相同的位置，但斜率和整体高度可能不同

图3.6 具有一个输入、四个隐藏单元和两个输出的网络

## 3.3.2 可视化多变量输入

为处理多变量输入$\boldsymbol{x}$，我们扩展了输入和隐藏单元之间的线性关系。因此，具有两个输入$\boldsymbol{x}=[x_1, x_2]^{\mathrm{T}}$和一个标量输出$y$的网络(图3.7)可能有以下三个隐藏单元：

$$\begin{aligned} h_1 &= a[\theta_{10}+\theta_{11}x_1+\theta_{12}x_2] \\ h_2 &= a[\theta_{20}+\theta_{21}x_1+\theta_{22}x_2] \\ h_3 &= a[\theta_{30}+\theta_{31}x_1+\theta_{32}x_2] \end{aligned} \tag{3.9}$$

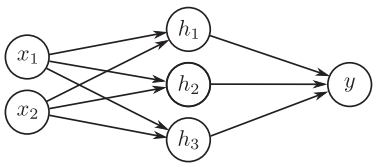



图3.7 具有二维多变量输入$\boldsymbol{x}=[x_1, x_2]^{\mathrm{T}}$和标量输出$y$的神经网络的可视化表示

其中每个输入都有一个斜率参数。隐藏单元按通常的组合方式生成输出：

$$y=\phi_0+\phi_1h_1+\phi_2h_2+\phi_3h_3 \tag{3.10}$$

图3.8说明了网络的处理过程。每个隐藏单元接收两个输入的线性组合，从而形成3D输入/输出空间中的一个定向平面。激活函数将这些平面的负值截取为零。然后，将截取的平面在第二个线性函数(式(3.10))中重新组合，创建一个由凸多边形区域组成的连续分段线性曲面(见图3.8(j))。每个区域对应不同的激活模式。例如，在中央三角形区域，第一个和第三个隐藏单元是活动的，第二个是无效的。

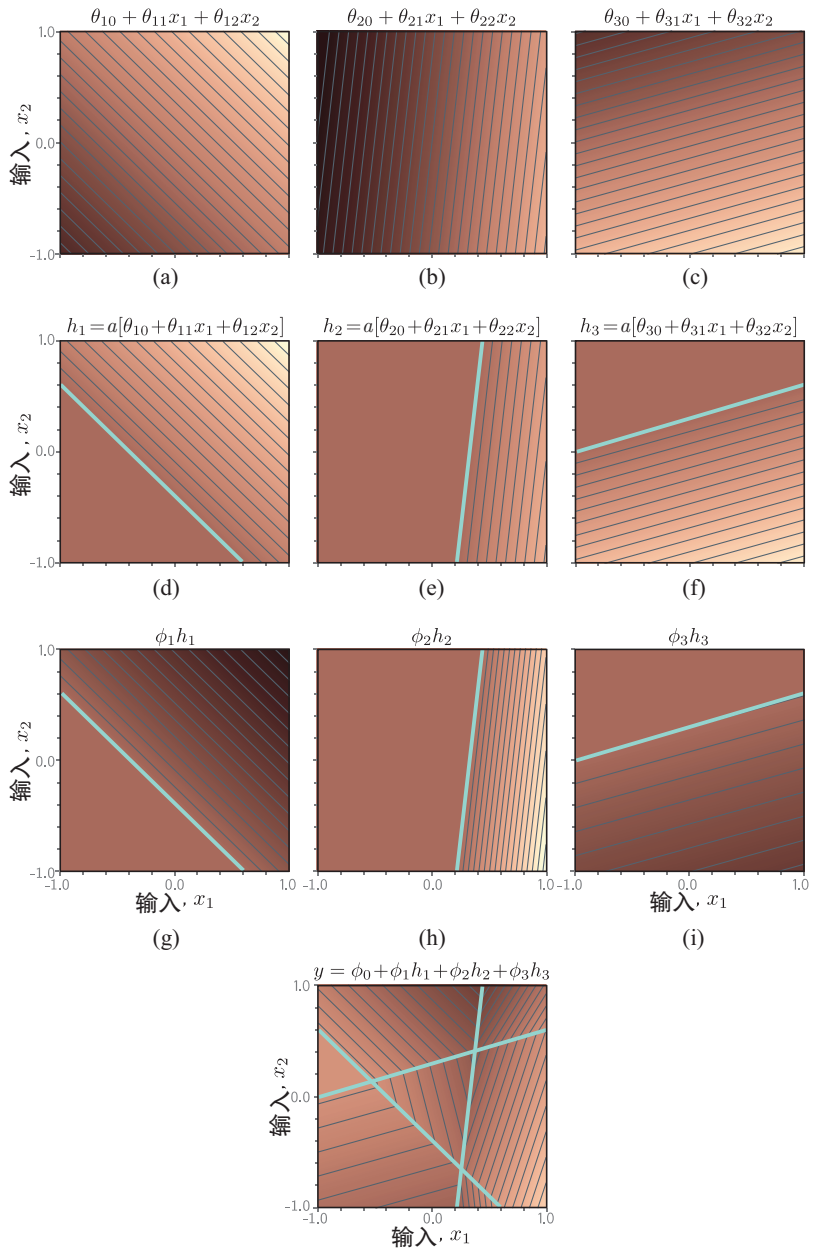
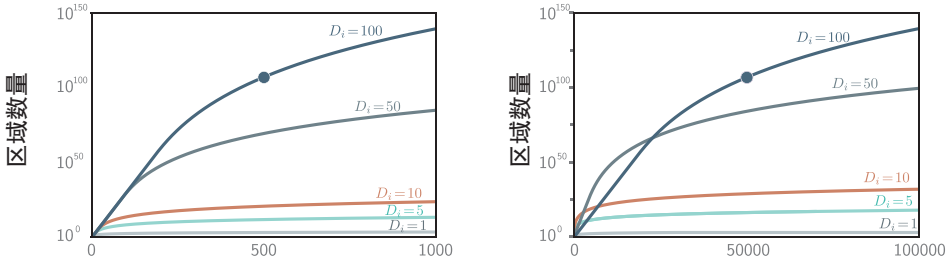



图3.8　展示了具有两个输入 $\boldsymbol{x}=[x_1, x_2]^{\mathrm{T}}$、三个隐藏单元 $h_1$, $h_2$, $h_3$ 和一个输出 $y$的网络处理过程。(a)~(c)中，每个隐藏单元的输入是两个输入的线性函数，对应于一个定向平面。亮度表示函数输出。例如，在(a)中，亮度代表 $\theta_{10}+\theta_{11}x_1+\theta_{12}x_2$。细线是等高线。(d)~(f)中，每个平面都被ReLU激活函数(青色线相当于图3.3(d)~(f)中的“关节”)截断。(g)~(i)然后对截断后的平面进行加权。(j)求和，加上一个偏移量，这个偏移量决定了曲面的整体高度。结果是一个由凸分段线性多边形区域组成的连续曲面

当模型有多于两个输入时，就很难被可视化了。然而，解释是类似的。输出是输入的连续分段线性函数，其中线性区域现在是多维输入空间中的凸多面体。

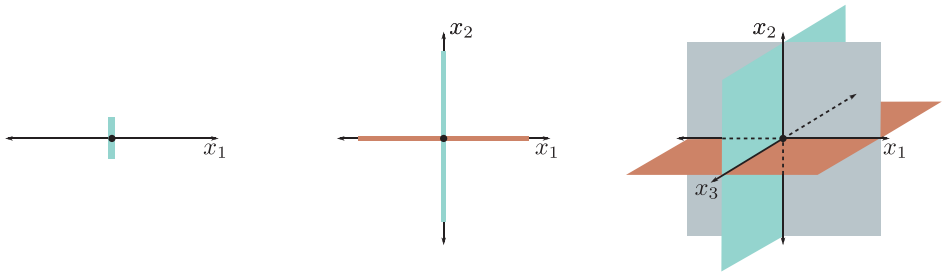
注意，随着输入维度的增加，线性区域的数量也迅速增加(见图3.9)。为了更好地理解其迅速增加的方式，请考虑每个隐藏单元都定义了一个超平面，该超平面划分了空间中该单元处于活动状态的部分和不处于活动状态的部分(3.8(d)~(f)中的青色线)。若有与输入维度 $D_i$ 相同数量的隐藏单元，可将每个超平面与一个坐标轴对齐(图3.10)。对于两个输入维度，这将把空间分成4个象限。对于三个维度，这将创建8个象限，对于 $D_i$ 个维度，这将创建 $2^{D_i}$ 个正交区域。浅层神经网络通常具有比输入维度更多的隐藏单元，因此它们通常会创建超过 $2^{D_i}$ 个线性区域。




(a) 对于5种不同的输入维度 $D_i = \{1, 5, 10, 50, 100\}$，最大可能的区域数与隐藏单元数量的函数关系。在高维情景中，区域数量迅速增加；当 $D=500$ 个单元和输入大小 $D_i=100$ 时，可以有超过 $10^{107}$ 个区域(实心圆)

(b) 同样的数据按照参数数量绘制。实心圆代表与图(a)中相同的模型，具有 $D=500$ 个隐藏单元。这个网络有51 001个参数，以现代标准看，通常被认为规模是非常小的

图3.9 线性区域与隐藏单元的关系




(a) 只有一个输入维度时，具有一个隐藏单元的模型创建一个关节，将轴分成两个线性区域

(b) 有两个输入维度时，具有两个隐藏单元的模型可使用两条线(这里与轴对齐)来划分输入空间，创建四个区域

(c) 有三个输入维度时，具有三个隐藏单元的模型可以使用三个平面(同样与轴对齐)来划分输入空间，创建8个区域。继续这个论点，可以推出，具有 $D_i$ 个输入维度和 $D_i$ 个隐藏单元的模型可用 $D_i$ 个超平面划分输入空间，创建 $2^{D_i}$ 个线性区域

图3.10 线性区域与输入维度的数量对比

## 3.4　浅层神经网络：一般情形

我们已经描述了几个浅层网络的例子，以帮助大家直观地了解它们的工作原理。现在，为浅层神经网络 $y=f[x,\phi]$ 定义一个通用函数，它使用 $h\in\mathbb{R}^D$ 个隐藏单元将多维输入 $x\in\mathbb{R}^{D_i}$ 映射到多维输出 $\boldsymbol{y}\in\mathbb{R}^D$。每个隐藏单元的计算方式为：

$$h_d = a\left[\theta_{d0}+\sum_{i=1}^{D_i}\theta_{di}x_i\right] \tag{3.11}$$

然后这些隐藏单元通过线性组合形成输出：

$$y_j = \phi_{j0}+\sum_{d=1}^{D}\phi_{jd}h_d \tag{3.12}$$

其中 $a[\bullet]$ 是非线性激活函数。模型的参数为 $\phi=\{\theta_{\bullet\bullet},\phi_{\bullet\bullet}\}$。图3.11显示了一个具有三个输入、三个隐藏单元和两个输出的示例。

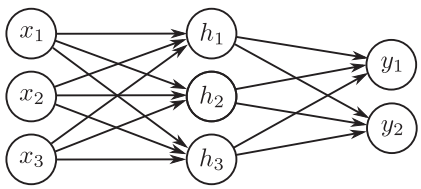



图3.11　一个具有三个输入和两个输出的神经网络的可视化表示。这个网络有20个参数、15个斜率(由箭头表示)和5个偏移量(未显示)

激活函数使模型能够描述输入和输出之间的非线性关系，因此它本身必须是非线性的；如果没有激活函数或使用线性激活函数，则从输入到输出的整体映射将被限制为线性的。业界已经尝试了许多不同的激活函数，但最常见的选择是ReLU(图3.1)，它具有易于解释的优点。使用ReLU激活时，网络将输入空间划分为由ReLU函数中的“连接点”计算出的超平面交点定义的凸多面体。每个凸多面体包含一个不同的线性函数，对每个输出，这些凸多面体都是相同的，但它们包含的线性函数可以不同。

## 3.5　术语

本章最后介绍一些术语。遗憾的是，神经网络有很多相关的术语，它们通常用层来描述。图3.12 的左侧是输入层，中间是隐藏层，右侧是输出层。图3.12 中的网络有一个隐藏层，包含四个隐藏单元。隐藏单元本身有时被称为神经元。将数据通过网络传递时，隐藏层输入的值(即在 ReLU 函数应用之前)称为预激活值。隐藏层的值(即在应用 ReLU 函数之后)称为激活值。

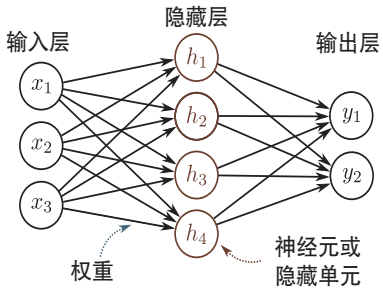



图3.12 一个浅层网络由输入层、隐藏层和输出层组成。每一层通过前向连接(箭头)与下一层相连。因此，这些模型被称为前馈网络(feed-forward networks)。当一层中的每个变量与下一层的每个变量相连时，称为全连接网络(fully connected network)。每个连接在底层函数中代表一个斜率参数，这些参数被称为权重(weight)。隐藏层中的变量被称为神经元或隐藏单元。输入隐藏单元的值被称为预激活值，隐藏单元中的值(即应用ReLU函数后的值)被称为激活值

由于历史原因，任何具有至少一个隐藏层的神经网络也被称为多层感知器，简称 MLP。具有一个隐藏层的网络(如本章所述)有时被称为浅层神经网络。具有多个隐藏层的网络(如下一章所述)被称为深度神经网络。连接形成无环图(即没有循环的图，如本章中所有的示例)的神经网络称为前馈网络。如果一层中的每个元素都分别连接到下一层中的每个元素(如本章中的所有示例)，则网络是全连接的。这些连接代表底层函数中的斜率参数，称为网络权重(network weight)。偏置参数(图3.12中未显示)称为偏置(bias)。

## 3.6 本章小结

浅层神经网络只有一个隐藏层。它们的作用是：①计算输入的几个线性函数，②将每个结果通过激活函数，③对这些激活值进行线性组合以形成输出。浅层神经网络通过将输入空间分为分段线性区域的连续曲面，基于输入$x$做出预测$y$。只要有足够多的隐藏单元(神经元)，浅层神经网络可以任意精度近似模拟任何连续函数。

第4章将讨论深度神经网络，它通过添加更多隐藏层扩展了本章的模型。第 5~7章将描述如何训练这些模型。

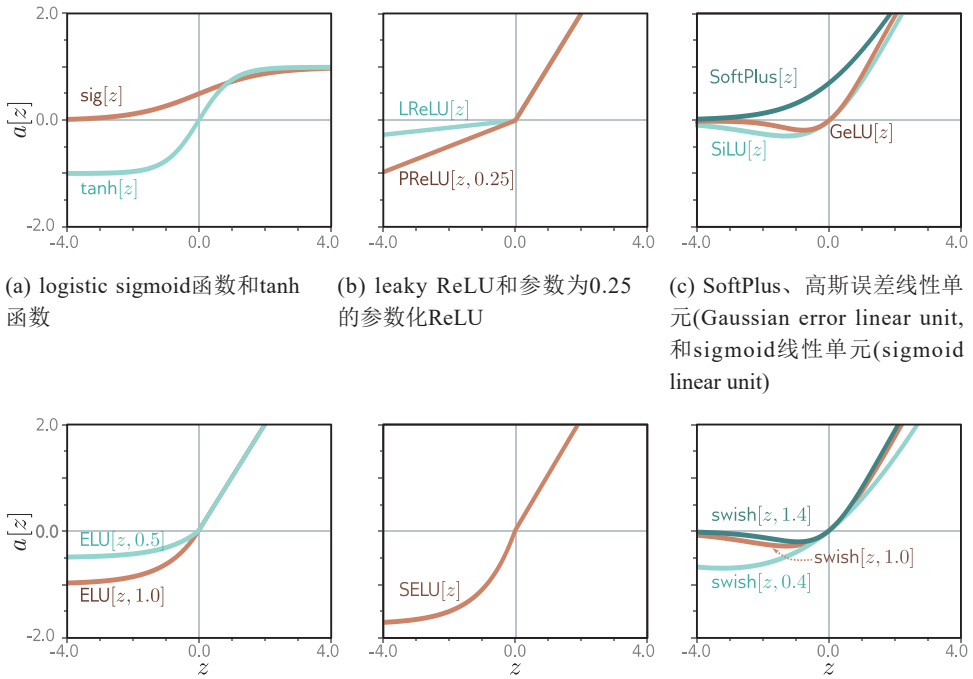
## 3.7 注释

**“神经”网络**：如果本章的模型只是函数，为什么它们被称为“神经网络”？遗憾的是，二者的联系有点牵强。图3.12展示了节点(输入、隐藏单元和输出)，节点之间密集地相互连接。这与哺乳动物大脑中的神经元看起来具有相似

性，后者也有密集的连接。然而，几乎没有证据表明大脑的计算方式与神经网络的工作方式相同，而且将生物学思维带入未来的研究是无益的。

**神经网络的历史：**McCulloch 和 Pitts (1943) 最先提出了人工神经元的概念，将输入组合起来生成输出，但该模型没有实用的学习算法。Rosenblatt (1958) 开发了感知机，该感知机以线性方式组合输入，然后阈值化以作出是/否决定；他还提供了一个算法从数据中学习权重。Minsky和Papert (1969) 认为线性函数不足以解决一般分类问题，但添加具有非线性激活函数的隐藏层(因此称为多层感知器)允许学习更一般的输入/输出关系。然而，他们得出结论，Rosenblatt的算法无法学习这类模型的参数。直到 1980 年，才开发了一种实用算法(反向传播，参见第 7 章)，神经网络的重要研究工作才得以恢复。由Kurenkov (2020)、Sejnowski (2018)和Schmidhuber (2022) 书写了神经网络的发展历史。

**激活函数：**Fukushima 早在1969年就已经使用过ReLU 函数。然而，在早期的神经网络中，更常使用 logistic sigmoid 或 tanh 激活函数(图3.13(a))。ReLU 被 Jarrett 等人 (2009)、Nair & Hinton (2010) 和 Glorot 等人 (2011) 重新推广而得以流行起来，成为现代神经网络成功的重要组成部分。ReLU具有一个很好的特性，即对于大于零的输入，输出相对于输入的导数始终为 1。这对训练的稳定性和效率都很友好(参见第 7 章)，与 sigmoid 激活函数的导数相比，后者在较大的正输入和负输入时会饱和(接近零)。




(a) logistic sigmoid函数和tanh函数

(b) leaky ReLU和参数为0.25的参数化ReLU

(c) SoftPlus、高斯误差线性单元(Gaussian error linear unit, 和sigmoid线性单元(sigmoid linear unit)

(d) 参数为0.5和1.0的指数线性单元(exponential linear unit)

(e) 缩放指数线性单元(scaled exponential linear unit)

(f) 参数为0.4、1.0和1.4的swish函数

图3.13　激活函数

然而，ReLU 函数的缺点是它的导数对于负输入为零。如果所有的训练样本对给定的ReLU函数产生负输入，那么我们在训练期间将无法改善输入ReLU的参数。相对于传入权重的梯度，在局部是平的，所以我们无法“下坡”。这称为死亡ReLU问题。为了解决这个问题，业界提出了许多ReLU的变体(见图3.13(b))。例如：①leaky ReLU(Maas等，2013)，对于负值也有一个较小斜率(0.1)的线性输出；②参数化ReLU(He等，2015)，将负部分的斜率视为一个未知参数；③串联ReLU(Shang等，2016)，生成两个输出，其中一个在零以下截断(就像典型的ReLU)，另一个在零以上截断。

业界还研究了各种平滑函数(见图3.13(c)~(d))，包括SoftPlus函数(Glorot等，2011)、高斯误差线性单元(Gaussian error linear unit)(Hendrycks & Gimpel，2016)、sigmoid线性单元(sigmoid linear unit)(Hendrycks & Gimpel，2016)和指数线性单元(exponential linear unit)(Clevert等，2015)。这些大多数是为了在限制负值的梯度的同时避免死亡神经元问题。Klambauer等人(2017)引入了缩放指数线性单元(scaled exponential linear unit)(见图3.13(e))，它特别有趣，有助于在输入方差有限的情况下稳定激活的方差(见第7.5节)。Ramachandran等人(2017)采取了经验方法来选择激活函数，在可能的函数空间中搜索，以找到在各种监督学习任务上表现最佳的函数；发现的最佳函数是 $a[x]=x/(1+\exp[-\beta x])$，其中$\beta$是一个学习得到的参数(见图3.13(f))。他们将这个函数称为swish。有趣的是，这是对之前由Hendrycks & Gimpel(2016)和Elfwing等人(2018)提出的激活函数的重新发现。Howard等人(2019)通过HardSwish函数近似swish，函数形式非常相似，但HardSwish的计算速度更快：

$$\text{HardSwish}[z]=\begin{cases}0 & z<-3\\ z(z+3)/6 & -3\leqslant z\leqslant 3\\ z & z>3\end{cases} \tag{3.13}$$

目前还没有确定的答案来说明这些激活函数中的哪一个在实际使用中更优秀。然而，leaky ReLU、参数化ReLU以及许多连续函数在特定情况下可以显示出比ReLU略好的性能。在本书的其余部分，将重点放在使用基本ReLU函数的神经网络上，因为很容易通过线性区域的数量来描述它们创建的函数。

**通用逼近定理：**这个定理的通用版本指出，一个包含有限数量隐藏单元和一个激活函数的单隐藏层网络，可在 $\mathbb{R}^n$ 的紧凑子集上以任意精度逼近任何连续函数。这一点已由Cybenko(1989)针对Sigmoid激活函数进行了证明，后来Hornik(1991)证明这一结论同样适用于更大类别的非线性激活函数。

**线性区域的数量：**考虑一个具有 $D_i\geqslant 2$ 维输入和$D$个隐藏单元的浅层网络。线性区域的数量由$D$个超平面的交点决定，这些超平面由ReLU函数中的“关节”

创建(例如，图3.8(d)~(f))。每个区域由ReLU函数的不同组合裁剪或不裁剪输入而创建。由$D$个超平面在$D_i \leqslant D$维输入空间中创建的区域数量被 Zaslavsky(1975)证明最多为$\sum_{j=0}^{D_i}\binom{D}{j}$(即，二项式系数的总和)。作为经验法则，浅层神经网络几乎始终拥有比输入维度$D_i$更多的隐藏单元数量$D$，并会创建$2^{D_i}\sim 2^{D}$个线性区域。

**线性、仿射和非线性函数：**从技术角度看，线性变换$f[\bullet]$是任何遵循叠加原理的函数，因此$f[a+b]=f[a]+f[b]$。这个定义意味着$f[2a]=2f[a]$。加权和$f[h_1,h_2,h_3]=\phi_1h_1+\phi_2h_2+\phi_3h_3$是线性的，但一旦加上偏置(bias)变成$f[h_1,h_2,h_3]=\phi_0+\phi_1h_1+\phi_2h_2+\phi_3h_3$，这就不再成立。要理解这一点，需要考虑当我们将前一个函数的参数加倍时，输出也会加倍。对于后一个函数则不是这样，它更恰当地被称为仿射函数。然而，在机器学习中，这些术语经常被混用。在本书中，我们遵循这一惯例，将两者都称为线性。我们将遇到的所有其他函数都是非线性的。

## 3.8　问题

**问题3.1**　如果式(3.1)中的激活函数是线性的，使$a[z]=\psi_0+\psi_1z$，那么会是什么样的输入到输出的映射？如果移除激活函数，使$a[z]=z$，会生成什么样的映射？

**问题3.2**　对于图3.3(j)中的4个线性区域，指出哪些隐藏单元是非活跃的(即会裁剪它们的输入)，哪些是活跃的(即不会裁剪它们的输入)。

**问题3.3***　推导出图3.3(j)中用10个参数$\phi$和输入$x$来表示的函数的“关节”位置的表达式。推导出四个线性区域的斜率的表达式。

**问题3.4**　绘制图3.3的一个新版本，其中第三个隐藏单元的$y$轴截距和斜率已经改变，如图3.14(c)所示。假设其余参数保持不变。

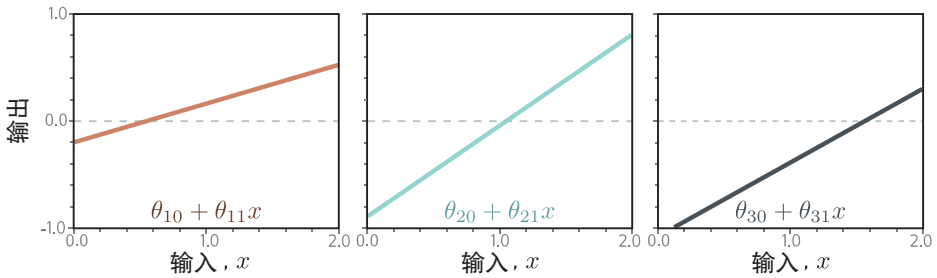



图3.14　在问题3.4网络的处理中有一个输入、三个隐藏单元与一个输出。(a)~(c)中，每个隐藏单元的输入是输入的线性函数。前两个与图3.3中的相同，但最后一个不同

**问题 3.5** 证明对于 $a \in \mathbb{R}^+$，以下性质对于ReLU函数成立：

$$\mathrm{ReLU}[a \cdot z] = a \cdot \mathrm{ReLU}[z] \tag{3.14}$$

这称为 ReLU 函数的非负齐次性。

**问题3.6** 在问题3.5的基础上，对于在式(3.3)和式(3.4)中定义的浅层网络，如果将参数 $\theta_{10}$ 和 $\theta_{11}$ 乘以正常数$\alpha$，并将斜率 $\phi_1$ 除以相同的参数$\alpha$，那么该浅层网络会发生什么变化？如果 $\alpha$ 是负数会发生什么？

**问题 3.7** 考虑使用最小二乘损失函数来拟合式(3.1)中的模型。这个损失函数是否有唯一最小值？也就是说，是否存在一组“最佳”参数？

**问题 3.8** 考虑用以下激活函数替换 ReLU 激活函数：①海维赛德阶跃函数(heaviside[$z$])，②双曲正切函数 tanh[$z$]，③矩形函数 rect[$z$]，④正弦函数 sin[$z$]。其中：

$$\mathrm{heaviside}[z] = \begin{cases} 0 & z < 0 \\ 1 & z \geqslant 0 \end{cases} \qquad \mathrm{rect}[z] = \begin{cases} 0 & z < 0 \\ 1 & 0 \leqslant z \leqslant 1 \\ 0 & z > 1 \end{cases} \tag{3.15}$$

为这些函数重新绘制图3.3 的版本。原始参数是：

$\phi = \{\phi_0, \phi_1, \phi_2, \phi_3, \phi_{10}, \phi_{11}, \phi_{20}, \phi_{21}, \phi_{30}, \phi_{31}\} = \{-0.23, -1.3, 1.3, 0.66, -0.2, 0.4, -0.9, 0.9, 1.1, -0.7\}$。对每个激活函数，简要描述神经网络(具有一个输入、三个隐藏单元和一个输出)可以创建的函数系列。

**问题 3.9*** 证明图3.3 中的第三个线性区域的斜率是第一个和第四个线性区域斜率的和。

**问题 3.10** 考虑一个具有一个输入、一个输出和三个隐藏单元的神经网络。图3.3 中的构造展示了如何创建四个线性区域。在什么情况下，这个网络可以生成少于四个线性区域的函数？

**问题 3.11*** 图3.6 中的模型有多少个参数？

**问题 3.12** 图3.7 中的模型有多少个参数？

**问题 3.13** 图3.8 中的7个区域的激活模式是什么？换句话说，每个区域中哪些隐藏单元是活跃的(传递输入)，哪些是非活跃的(截断输入)？

**问题 3.14** 写出定义图3.11 中网络的算式。应该有三个算式通过输入计算出三个隐藏单元，另有两个算式通过隐藏单元计算输出。

**问题 3.15*** 图3.11 中的网络最多可以创建多少个三维线性区域？

**问题 3.16** 写出一个具有两个输入、四个隐藏单元和三个输出的网络的算式。以图3.11 的风格绘制这个模型。

**问题 3.17*** 式(3.11)和式(3.12)定义了一个通用的神经网络，其中 $D_i$ 是输入维度，隐藏层包含$D$个隐藏单元，$D_o$ 是输出维度。根据 $D_i$、$D$和 $D_o$ 找出模型参数数量的表达式。

**问题 3.18*** 证明具有 $D_i$ = 二维输入，$D_o$ = 一维输出，$D$=3个隐藏单元的浅层网络创建的最大区域数是7个，如图3.8(j)所示。使用 Zaslavsky (1975) 的结果，即用$D$个超平面划分 $D_i$ 维空间创建的最大区域数是 $\sum_{j=0}^{D_i}\binom{D}{j}$。如果在这个模型中增加两个隐藏单元，使得$D$=5，那么最大区域数是多少？