数据科学与大数据技术

Effective 数据科学基础设施

［芬］维莱·图洛斯(Ville Tuulos) 著

郭 涛 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：01-2023-2922

Ville Tuulos
Effective Data Science Infrastructure
EISBN: 9781617299193

图书在版编目(CIP)数据

Effective 数据科学基础设施 / (芬)维莱·图洛斯著；郭涛译. —北京：清华大学出版社，2023.7
(数据科学与大数据技术)
ISBN 978-7-302-64186-5

Ⅰ. ①E…　Ⅱ. ①维…　②郭…　Ⅲ. ①数据管理－研究　Ⅳ. ①TP274

中国国家版本馆 CIP 数据核字(2023)第 145536 号

责任编辑：王　军
封面设计：孔祥峰
版式设计：思创景点
责任校对：马遥遥
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：北京同文印刷有限责任公司
经　　销：全国新华书店
开　　本：170mm×240mm　　印　　张：18.5　　字　　数：427 千字
版　　次：2023 年 8 月第 1 版　　印　　次：2023 年 8 月第 1 次印刷
定　　价：98.00 元

产品编号：097431-01

译 者 序

近年来，机器学习、深度学习和强化学习等人工智能技术已日趋成熟，逐渐从学术界走向了产业界，在计算机视觉、推荐系统和自然语言处理等方面产生了巨大价值。在生产中部署、运维和管理机器学习系统，已逐渐成为学术界和产业界面临的难题。“最后一公里”已成为近几年学者和工程师所要解决的关键技术，MLOps 概念、设计理念和产品也应运而生。MLOps 是一种 ML 工程文化和实践，旨在生产中统一 ML 模型开发(Dev)和 ML 模型运维(Ops)。MLOps 的主要目标是简化管理流程，在生产环境中大规模自动部署机器学习和深度学习模型，便于模型需求、业务需求和监管要求保持一致。从另一个角度看，MLOps 理念提倡数据科学家、数据工程团队、软件工程团队和运维团队之间更好地合作。MLOps 工程实践涉及 3 个学科：机器学习、软件工程(尤其是 DevOps)和数据工程。MLOps 理念能够落地，是人工智能和现代软件工程的有效结合。特别是近十年来，云计算高速发展，微服务、DevOps 等技术快速成熟，给机器学习系统的设计和实现提供了良好的基础。从价值驱动角度看，AI 模型的业务需求极大，催生了 MLOps 的落地。

MLOps 是 ML 基础设施的一部分，涉及自动化机器学习(AutoML)、ML 流程、分布式训练、智能监控和管理的方方面面，形成了端到端的机器学习工作流，持续集成和持续交付(CI/CD)，在模型开发和生成部署之间搭起了重要的桥梁。MLOps 使用高效且运行鲁棒的机器学习模型，专注于机器学习模型的规模化开发、部署、管理和运维的过程，以优化可伸缩工业环境中的 ML 生命周期，确保 ML 项目具备可重复性和可追溯性。MLOps 根据 ML 项目的生命周期，充分考虑业务目标、数据准备、模型实验、模型工程、模型评估、模型部署、模型预测和模型监控等步骤。此外，MLOps 还为人工智能的透明度和可解释性奠定了基础。

现代软件工程和人工智能的发展催生了很多专业术语，如 DataOps、AIOps 等。与 MLOps 相比，DataOps 的核心应用对象是数据应用，涉及数据生命周期内的所有步骤，包括数据收集、处理、分析和报告。DataOps 的目标主要是提高数据的质量和可靠性，同时尽可能缩短提供数据应用的周期。AIOps 的目标是自动发现日常 IT 运维中的问题，并利用 AI 技术主动做出智能反应和预警。总之，AIOps 是 AI 在 Ops 领域的应用，应用的主角是 Ops；而 MLOps 则是 Ops 在 ML 领域的应用，主角是 ML。

现在，很多高校、科研机构和 AI 企业已投身于 MLOps 产品和工具的研发，相继出现了 MLflow、Metaflow、Apache Airflow 和 Kubeflow 等。MLflow 是一个机器学习生命周期的开源平台，其核心包括 4 个模块：MLflow 追踪、MLflow 项目管理、MLflow 项目和 MLflow 模型注册，主要支持 Python、R 和 Java API。Metaflow 是为数据科学家(非机器)而建，旨在提高数据科学家的生产力，适用项目范围广泛，包括经典统计到最先进的深度学习，主要支持 Python

和 R。Apache Airflow 是一款开源的分布式任务调度框架，可以将一个具有上下级依赖关系的工作流组装成一个有向无环图。Kubeflow 是 Kubernetes 的机器学习包，是一个简单、可组合、可携式、可伸缩的机器学习技术栈，有助于促进机器学习的工作流部署，是机器学习和云计算的桥梁。这些平台整体上都是围绕 MLOps 理念展开设计的，各有利弊，需要根据业务场景、特定需求和搭建原则进行选择。

MLOps 平台的设计要足够灵活，且能实现模块化，打破开发语言和框架的限制，但也不能是“万金油”，不能“什么都能做，但什么都做不好”。要让 MLOps 具备持续的生命力，就必须不断完善设计理念，持续注入活力，促进技术新陈代谢。既要做到能够及时兼容已有的框架，如 PyTorch、TensorFlow 和 Scikit-Learn 等，又要能够兼顾新框架，如 JAX、Pyro。另外，还要做到不冗余，不笨重，能够根据技术发展实现精简、轻量级和学习成本低的产品。

本书由 Netflix 工程师 Ville Tuulos 撰写，以 Metaflow 为对象，介绍了数据科学所需要的基础设施，囊括数据准备、特征工程、模型训练、模型部署、服务和持续监控等环节。Metaflow 专注于构建生产流程，更适合具有深厚工程和 DevOps 技能的大型专业数据科学团队。本书的目标读者为数据科学家、机器学习工程师、IT 技术人员和 MLOps 工程师。数据科学家在人工智能和算法方面非常精通，但软件开发能力通常不足。他们渴望有一套方法论和工具来促进从构建到部署的迭代过程，从而落实自己的想法。数据科学家不在意在一个“孤岛”上开展数据挖掘和分析工作，他们更希望能够在离线、实时和批处理等场景中落实项目。IT 技术人员对机器学习算法理论和模型细节并不了解，他们渴望本书能够提供一个机器学习流程的全貌，便于他们进行任务编排。此外，一些企业的中高层管理人员可通过本书获取 MLOps 管理理念，为制定 AI 项目管理和 KPI 考核提供参考依据。总之，与传统的软件工程师技能要求相比，MLOps 工程师除了需要具备现代软件工程所要求的强大能力，还需要具备 ML 专业知识，具体包括 ML 模型训练、模型部署、模型监控和帮助企业落实架构、系统设计和故障排除等能力。

总之，利用开源组件或者自研 MLOps 是很有必要且很有价值的。译者使用一些开源组件为自己的场景构建过一些 MLOps 产品，得出以下经验：(1)生产型 ML 与云计算完美适配，MLOps 云端是趋势；(2)实现自动化的端到端 MLOps 产品还需要一些时间，需要进一步与大数据 Apache Spark、Flink 等深度融合，按照批量、实时等场景构建端到端的服务；(3)要以系统论的思想建立一个成本低、效率高、版本可控制、可复制的管理系统，以大量的工程化抽象出功能。本书为自研 MLOps 和 Metaflow 提供了一些经验和设计参考。相信本书能够为你释疑解惑，不会让你失望！

在翻译本书的过程中，我得到了很多人的帮助。其中，对外经济贸易大学英语学院的许瀚、吉林大学外国语学院的吴禹林等对本书译文进行了审校，感谢她们所做的工作。最后，要感谢清华大学出版社的编辑，他们为本书做了大量的编辑与校对工作，使得本书符合出版要求。

由于本书涉及知识的广度和深度较大，加上译者翻译水平有限，在翻译过程中难免有不足，若读者在阅读过程中发现问题，欢迎批评指正。

译者

推 荐 序

2012 年，我第一次见到本书作者 Ville Tuulos，当时我正试图追逐 Hadoop 的热潮，而 Ville 正在开发 Disco；Disco 是一种基于 Erlang 的 map-reduce 解决方案，可使与 Python 的交互变得更容易。那时，我和 Peter Wang 刚刚创办了 Continuum Analytics 公司，Ville 的研究极大地鼓舞我们发布了 Python 分发版的大数据平台，即 Anaconda。

作为 NumPy 和 Anaconda 的创始人之一，我有幸目睹了在机器学习带来的巨大机遇下，在过去的六到七年中，MLOps 工具经历的爆炸式发展。目前，MLOps 工具琳琅满目，开发商们都费尽心思为自己的 MLOps 工具做宣传。我在 Quansight 和 OpenTeams 的团队持续评估新的工具和方法，从而推荐给我们的客户。

我感到非常荣幸，能够遇见像 Ville 这样值得信任的人，能够在 Netflix 和 Outerbounds 公司遇见创建和维护 Metaflow 的团队。本书详细介绍 Metaflow，并充分说明数据基础设施和机器学习操作在数据世界中的重要性，我对本书的出版感到十分兴奋。无论你使用何种 MLOps 框架，我相信，通过阅读本书，你都能学会如何使机器学习操作更高效、更多产。

——Travis Oliphant

NumPy 作者，

PyData、NumFOCUS 和 Anaconda 创始人

作 者 简 介

20 多年来，**Ville Tuulos** 一直为机器学习开发基础设施。Ville 曾涉足学术界、专注于数据和机器学习的初创企业，以及两家全球企业。他在 Netflix 领导机器学习基础设施团队时，创建了本书中介绍的开源框架 Metaflow。Tuulos 还是 Outerbounds 的首席执行官和联合创始人之一，Outerbounds 是一家初创公司，专注于以人为中心的数据科学基础设施。

致　谢

感谢 Netflix 和许多其他公司的数据科学家和工程师耐心解释了他们的痛点，分享了反馈，并允许我参考他们的项目。没有他们，本书就不可能出版。非常感谢！请继续提供反馈。

Metaflow 受益于 Savin Goyal、Romain Cledat、David Berg、Oleg Avdeev、Ravi Kiran Chirravuri、Valay Dave、Ferras Hamad、Jason Ge、Rob Hilton、Brett Rose、Abhishek Kapatkar 等才华横溢、充满激情且具有很强共情能力的工程师。本书中到处都是你们留下的痕迹！和你们共事是一种荣幸和乐趣。此外，我还要感谢 Kurt Brown、Julie Amundson、Ashish Rastogi、Faisal Siddiqi 和 Prasanna Padmanabhan，他们自项目启动以来，一直给予大力支持。

我想和 Manning 团队一起完成本书，因为他们以出版高质量的技术书籍而闻名。果然如此！我很荣幸能与一位经验丰富的编辑(Doug Rudder)合作，他助我精进写作，使长达一年半的写作过程精彩纷呈。非常感谢 Nick Watts 和 Al Krinker 提出的深刻的技术评论，感谢所有在早期试读本书期间提供反馈的读者和审稿人。

致所有审稿人：Abel Alejandro Coronado Iruegas、Alexander Jung、 David Patschke、 David Yakobovitch、 Edgar Hassler、Fibinse Xavier、Hari Ravindran、Henry Chen、Ikechukwu Okonkwo、Jesús A. Juárez Guerrero、 Matthew Copple、Matthias Busch、 Max Dehaut、 Mikael Dautrey、Ninoslav Cerkez、 Obiamaka Agbaneje、Ravikanth Kompella、 Richard Vaughan、Salil Athalye、Sarah Catanzaro、 Sriram Macharla、Tuomo Kalliokoski 和 Xiangbo Mao，你们的建议使本书变得更优秀。

最后，我要感谢我的妻子和孩子们，感谢他们给予我无限的耐心和支持。孩子们，我想说，我欠你们一个冰淇淋！

前　言

我在十几岁时，对人工智能产生了浓厚的兴趣。13 岁时，我训练了我的第一个人工神经网络。我从零开始，用 C 和 C++实现了简单的训练算法，这是 20 世纪 90 年代时探索该领域的唯一方法。此后，我继续学习了计算机科学、数学和心理学，以更好地理解这一庞大主题的基础。当时，机器学习(数据科学这个术语还不存在)的应用方式有时似乎更像魔术，而不是真正的科学或原理工程。

后来，我从学术界转向大公司和初创公司，此后，我一直在构建支持机器学习的系统。Linux 等开源项目和当时新兴的 Python 数据生态系统对我的影响很大。Python 数据生态系统提供了 NumPy 等包，与 C 或 C++相比，这些包使得构建高性能代码更容易。除了开源的技术优势，我还发现围绕这些项目形成了十分创新、充满活力且广受欢迎的领域。

当我在 2017 年加入 Netflix，受命从零开始构建新的机器学习基础设施时，我秉持着 3 个原则。首先，我们需要对全栈有一个原则性的理解——数据科学和机器学习不是魔术，而需要成为一门真正的工程学科。其次，无论是出于技术角度还是因为其庞大的包容性领域，我都确信 Python 是新平台的基础。最后，归根结底，数据科学和机器学习是人类使用的工具。使用工具的唯一目的是提高工作效率，成功的工具还可提供令人愉悦的使用体验。

工具是由孕育其诞生的文化塑造的。我创建了开源工具 Metaflow 后，Netflix 的文化对其产生了巨大的影响，该工具后来成为一个强劲的开源项目。Netflix 的发展压力确保了 Metaflow 和我们对整个数据科学堆栈的理解都是基于数据科学家的实际需求。

Netflix 给予其数据科学家高度的自主权，而这些数据科学家通常不是经过训练的软件工程师。这使我们要仔细考虑数据科学家在开发项目并最终将其部署到生产中时面临的所有挑战。Netflix 的顶尖工程团队已使用云计算超过十年，已充分了解了云计算的优缺点，我们对堆栈的理解也深受他们的影响。

我撰写本书旨在与更多人共享这些经历。无论是开源领域、深有远见又无私共享的个人，还是聪明绝顶的数据科学家，都教会了我很多，我觉得我有义务回馈他们。本书肯定不是我学习旅程的终点，只是一个里程碑。因此，我很期待反馈。不要犹豫，赶快与我联系吧，分享你的经历、想法和反馈！

关 于 本 书

如果将作为驱动的软硬件全栈考虑在内，那么机器学习和数据科学应用程序可以称得上是人类构建的最复杂的工程工件之一。鉴于此，即使是在今天，构建这样的应用程序也并不容易，这一点其实毫不令人奇怪。

机器学习和数据科学具有长期发展的前景。先进的数据驱动技术支持各类应用程序，在各个行业中的运用越来越广泛。因此，我们需要让此类应用程序的构建过程和运行过程更简单、有序。正如 Alfred Whitehead 所说：“当人类可以不假思索地完成更多的重要行动时，文明也就随之进步了。”

本书将展示如何构建一个高效的数据科学基础设施，以方便用户试验创新应用程序，将其部署到生产环境中并不断改进，而不必顾虑太多技术细节。由于不存在普适的方法，因此本书将重点介绍一般的基本原则和组件，读者可以在自己的环境中合理地实现这些原则和组件。

本书读者对象

本书主要面向以下两类读者。

- 数据科学家：希望了解在真实业务环境中有效开发和部署数据科学应用程序的全栈系统。即使你不具备基础设施工程、DevOps 或一般软件工程的背景知识，也可通过本书全面了解所涉及的内容，并学习新内容。
- 基础设施工程师：负责建立基础设施，为数据科学家提供帮助。即使你有 DevOps 或系统工程方面的经验，也可通过本书全面了解数据科学与传统软件工程在需求方面的区别。另外，还可了解如何以及为何需要不同的基础设施堆栈才能提高数据科学家的工作效率。

此外，数据科学和平台工程组织的领导者可以参阅本书，因为基础设施与组织相辅相成。

本书结构：阅读指南

本书围绕数据科学基础设施的完整堆栈展开。堆栈具有结构化的特点，底部为最基础的、面向工程的层；顶部为与数据科学相关的更高级别的层。本书将从下往上遍历堆栈，大致内容如下。

- 第 1 章首先解释数据科学基础设施的存在意义，并推荐以人为中心的基础设施方法。

- 第 2 章介绍基础知识，包括数据科学家的日常工作，以及如何优化其工作环境的人机工程学。
- 第 3 章介绍开源框架 Metaflow，我们将使用该框架说明有效基础设施的概念。
- 第 4 章关注可伸缩计算：所有数据科学应用程序都需要执行计算，规模大小不一。我们将使用云实现可伸缩计算。
- 第 5 章关注性能：众所周知，过早优化并不理想。更好的方法是逐步优化代码，仅在需要时增加复杂性。
- 第 6 章讨论生产部署：原型开发和生产环境之间存在一些关键差异，但在两者之间迁移并不太难。
- 第 7 章深入探讨数据科学的另一个基本问题：数据。我们将研究与现代数据仓库和数据工程团队集成的有效方法。
- 第 8 章讨论数据科学在相关业务系统中的应用。数据科学不应成为一座孤岛，我们将学习如何将数据科学连接到其他系统，以产生真正的业务价值。
- 第 9 章介绍一个现实的、端到端的深度学习应用程序，将堆栈的所有层联系在一起。
- 附录包括有关安装和配置 Metaflow 的 Conda 包管理器的说明。

从第 3 章开始，各章将使用小型但现实的机器学习应用程序来说明相关概念。阅读本书不需要有机器学习或数据科学的预备知识，我们使用这些技术仅用于说明。许多优秀的书籍已深入讲解了机器学习和数据科学技术，本书将不再赘述，重点仅在于支持这些应用程序的基础设施。

阅读完前 3 章后，可以随意跳过与你无关的章节。例如，如果只需要处理小规模数据，可以重点阅读第 3、6 和 8 章。

关于代码和 Links 文件的下载

本书的所有示例均使用了开源 Python 框架 Metaflow。然而，书中提出的概念和原则并非只针对 Metaflow。特别是第 4～8 章介绍的概念和原则，也可轻松适用于其他框架。你可通过扫描本书封底的二维码下载书中的所有源代码。

可以在 OS X 或 Linux 计算机上执行示例，只需要一个代码编辑器和一个终端窗口。如下以# python 开头的代码行

```
# python taxi_regression_model.py --environment=conda run
```

需要在终端上执行。根据第 4 章的说明，许多示例都可以借助 AWS 账户实现。

在此要说明的是，读者在阅读本书时会看到一些有关链接的编号，形式是编码加方括号，例如，链接[1]表示读者可扫描本书封底的二维码下载 Links 文件(书中提到的所有链接都放在了该文件中)，找到对应章节中[1]指向的链接。

目　录

第 1 章

数据科学基础设施介绍

本章内容

- 数据科学基础设施对公司的重要意义
- 数据科学和机器学习的基础设施堆栈
- 成功数据科学基础设施的组成要素

20 世纪 50 年代，机器学习和人工智能在学术界横空出世。从技术上讲，在不考虑时间和成本的前提下，本书中的所有内容均可在数十年内实现。然而，在过去的 70 年里，进展并不顺利，可谓处处碰壁。

正如许多公司所经历的那样，想要构建机器学习驱动的应用程序，需要具备专业知识的大型工程师团队，工程师通常需要工作多年，才能提供优化的解决方案。如果回顾计算的历史，就会发现，大多数社会层面的变化都不是发生在不可能变成可能时，而是发生在可能变得容易时。弥合可能和容易之间的差距需要高效的基础设施，这就是本书要讨论的主题。

字典将基础设施定义为“一个国家、地区或组织正常运行所需的基本设备和结构(如道路和桥梁)。”本书涵盖了数据科学应用程序正常运行所需的设备和结构的基本堆栈。阅读本书后，你将能设置和定制一个基础设施，帮助你的组织更快、更轻松地开发和交付数据科学应用程序。

术语解释

如今所说的“数据科学”一词诞生于 21 世纪初。如前所述，在此之前，“机器学习”和“人工智能”这两个术语已使用了几十年，其他相关术语，如“数据挖掘”或“专家系统”等也是如此。这些术语曾一度流行。

相关领域的研究者对于这些术语的确切含义没有达成共识，这具有挑战性。这些领域的专业人士认识到数据科学、机器学习和人工智能之间存在细微差别，但它们之间的界限是

有争议且模糊的，20 世纪 70 年代对“模糊逻辑”感兴趣的人一定会为此感到高兴！

本书的目标是将现代数据科学、机器学习和人工智能领域结合起来。为了简洁起见，我们选用“数据科学”一词来描述这一组合。术语的选择旨在体现包容性，本书不排除任何特定的方法或方法集。

就本书而言，这些领域之间的差异并不显著。在一些特定情况下，当需要强调差异时，我们将使用更具体的术语，如“深度神经网络”。总之，无论本书在何处使用“数据科学”这一术语，只要使用其他你喜欢的术语能够使文本更易于理解，你就可以进行替换。

如果你问该领域的专业人员，数据科学家的工作是什么，你可能很快就会得到一个答案：他们的工作是建立模型。虽然这个答案没有错，但有点狭隘。人们越来越期望数据科学家和工程师可以为业务问题构建端到端的解决方案，模型是方案中一个很小但很重要的部分。因为本书专注于端到端的解决方案，所以我们认为数据科学家的工作是构建“数据科学应用程序”。鉴于此，当你在本书中看到“数据科学应用程序”时，请将其理解为“端到端解决方案所需的模型等物”。

1.1 选择数据科学基础设施的原因

目前，已有许多优秀的书籍向我们介绍了什么是数据科学，数据科学的意义，以及如何在各种环境中应用数据科学。本书重点讨论与基础设施相关的问题。在详细讨论为什么我们需要专门用于数据科学的基础设施之前，首先简要讨论需要基础设施的原因。

想象一下，在 20 世纪工业化规模农业出现之前，人们是如何生产和消费牛奶的。许多家庭有一两头牛，可以生产牛奶以满足家庭的迫切需要。养牛需要一些专业知识，但不需要太多技术基础设施。当某家人想扩大乳品经营时，如果不投资于大规模饲料生产、人力资源和产品储存，则会难以进行。简言之，他们能够以最少的基础设施经营一家小型乳品企业，但大规模生产所需要的投资要比再购买一头奶牛多得多。

即使农场能够养活更多的奶牛，也需要将多余的牛奶进行出售。这带来了一个速度问题：如果农户运送牛奶的速度不够快，其他农户就可能会抢先出售，使市场饱和。更糟糕的是，牛奶可能会变质，从而破坏产品的有效性。

也许一位友好的邻居能够帮忙将牛奶运送到附近的城镇。目光敏锐且有创业精神的农户可能会发现当地市场的生牛奶供应过剩，但顾客需要的是各种精制乳制品，如酸奶、奶酪甚至冰淇淋。虽然农户非常想为这些顾客提供服务(并赚取他们的钱)，但很明显，该农户的生意无法应对这种复杂的市场。

随着时间的推移，一套相互关联的系统出现了，可以满足市场需求，并形成了如今的现代乳制品基础设施：工业规模的农场针对产量进行了优化。冷藏、巴氏杀菌和物流保障了向乳制品工厂输送优质牛奶所需的速度，乳制品工厂随后生产出各种各样的产品，并分销到食品市场。请注意，乳制品基础设施并没有取代所有的小农生产：有机农场、微型农场、家庭农场产出的

特定产品仍有相当大的市场，但这种劳动密集型的方式无法满足所有需求。

Michael Stonebraker 教授最初使用“数量、速度和多样性”这 3 个维度对大数据数据库系统进行分类。有效性与数据科学高度相关，因此我们添加“有效性”作为第 4 个维度。此处有一个问题值得思考：考虑在你的业务环境中，哪一个维度最重要？大多数情况下，高效的数据科学基础设施应该在这 4 个维度之间取得平衡。

数据科学项目的生命周期

在过去的 70 年中，可以说大多数数据科学应用程序都是手工创建的，即由一个高级软件工程师团队从零开始构建整个应用程序。与乳制品的情况一样，手工并不意味着“不好”——通常恰恰相反。手工方式通常用于实验前沿创新或创建高度专业化应用。

与乳制品情况一样，随着行业的成熟，我们需要产品在数量、速度、有效性和多样性方面实现优化提高，因此，在一个公共基础设施上构建许多(如果无法构建大多数)应用程序已成为合理需求。你可能对生牛奶如何转换为奶酪以及工业规模的奶酪生产需要何种基础设施有大致的了解，但对于数据科学呢？图 1.1 说明了一个典型的数据科学项目。

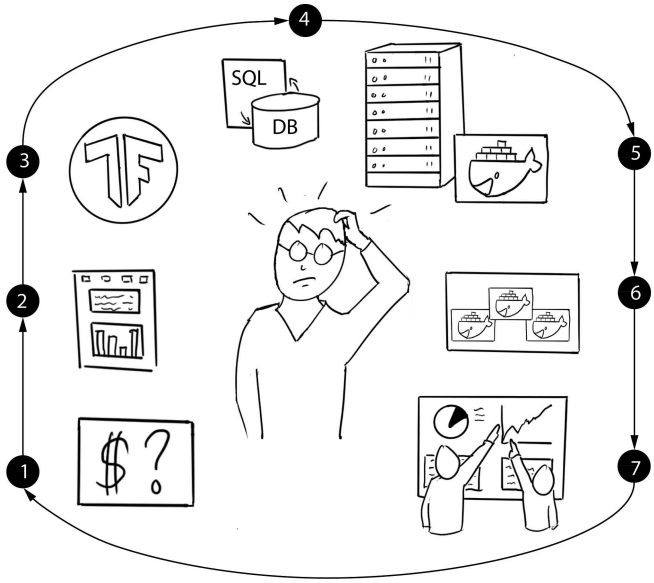

图 1.1　数据科学项目的生命周期

1. 图的中心有一位数据科学家，他需要解决一个业务问题，例如，要创建一个模型来评估客户的终身价值；或者创建一个系统，以在电子邮件通信中生成个性化的产品推荐。

2. 数据科学家通过提出假设和实验来启动项目。他们可以使用自己最喜欢的工具来测试想法，包括 Jupyter Notebooks、R 或 Julia 等专用语言，或者 MATLAB 或 Mathematica 等软件包。

3. 当涉及机器学习或统计模型的原型开发时，可使用有效的开源软件包，如 Scikit-Learn、PyTorch、TensorFlow、Stan 等。在许多情况下，借助一些优秀的在线文档和教程，使用这些软

件包构建一个初始原型并不需要太长时间。

4. 然而，每个模型都需要数据。数据库中可能存在合适的数据。提取原型的静态数据样本通常很简单，但处理更大的数据集(如几十吉字节)可能会变得更复杂。这种情况下，数据科学家甚至不需要担心如何使数据自动更新，这一问题需要更复杂的架构和工程。

5. 数据科学家在哪里运行笔记？也许他们可以在计算机上运行，但如何共享结果呢？如果同事想测试原型，但没有足够强大的计算机，该怎么办？在云中的共享服务器上执行实验很方便，所有合作者都可轻松访问该服务器。但是，需要有人先设置这一环境，并确保服务器上有所需的工具、库和数据。

6. 数据科学家需要解决一个业务问题。很少有公司使用笔记或其他数据科学工具来开展业务。为了证明原型的价值，仅将原型保存在笔记或其他数据科学环境中是不够的，还需要将它集成到周围的业务基础设施中。也许系统被组织为微服务，所以如果新模型也可部署为微服务，就会有所帮助。前提是需要在基础设施工程方面具有丰富的经验和充足的知识储备。

7. 最后，在原型与周围系统集成后，利益相关者(产品经理和企业所有者)评估结果，并向数据科学家提供反馈。可能会出现两种结果：一是利益相关者可能对结果持乐观态度，向数据科学家提出进一步的改进要求；二是安排数据科学家解决其他更有前途的业务问题。值得注意的是，无论结果如何，下一步骤都是相同的：重新开始整个周期，或是专注于改进结果，或是致力于解决新问题。

生命周期的细节自然会因公司和项目而异：为客户终身价值开发预测模型与建造自动驾驶汽车有很大区别。然而，所有数据科学和机器学习项目都具有以下关键要素：

1. 从技术角度来看，所有项目的基础都涉及数据和计算。

2. 本书侧重于技术的实际应用，而不是纯粹的研究，因此我们期望所有项目最终都需要解决一个问题：将结果集成到生产系统。解决这一问题通常涉及大量的软件工程。

3. 最后，从人类的角度看，所有项目都涉及实验和迭代，许多人认为这是数据科学的中心活动。

尽管个人、公司或团队完全可以定制自己的流程和实践，从而开发数据科学项目，但一个通用基础设施有助于增加同时执行的项目数量(数量)、加快上市速度、确保结果的鲁棒性(有效性)，并可以支持更多种类(多样性)的项目。

注意，项目的规模，即数据集或模型的大小，是一个正交问题。特别是，认为只有大型项目需要基础设施的想法是极其错误的。通常，情况恰恰相反。

我是否适合阅读本书

本书适用于对与数据科学项目生命周期相关的问题和潜在解决方案感兴趣的读者。如果你是一名数据科学家，可能亲身经历过一些挑战。如果你是一名基础设施工程师，希望设计和构建系统来帮助数据科学家，则对于这些困扰已久的问题，你可能想找到可伸缩的、鲁棒性的解决方案。

我们将系统地介绍构成数据科学现代化、有效基础设施的系统堆栈。本书所涵盖的原则

并不仅限于任何特定的实现，我们将使用开源框架 Metaflow 来展示如何将这些想法付诸实践。或者，可使用其他现有的库来定制自己的解决方案。本书将帮助你为特定任务选择合适的工具。

值得注意的是，在本书不适用的情况下，也可能存在完全有效的重要场景。如果你处于以下情况，本书和一般数据科学基础设施可能并不适用：

- 你专注于理论研究，而不在实际用例中应用方法和结果。
- 你正处于应用的第一个数据科学项目的早期阶段(如前所述的步骤 1~4)，且一切进展顺利。
- 你正在开发一个非常具体、成熟的应用程序，因此优化项目的数量、速度和多样性并不成问题。

在这些情况下，你可以等到出现更多的项目，或者开始遇到上述数据科学家遇到的难题时，再阅读本书。如果你未处于这些情况下，请继续阅读本书！在下一节中，我们将介绍一个基础设施堆栈，它为后文讨论的所有内容提供了总体框架。

1.2　什么是数据科学基础设施

新的基础设施是如何出现的？在 20 世纪 90 年代的万维网早期，除了原始的网络浏览器和服务器，没有任何基础设施存在。在互联网繁荣时期，建立一家电子商务商店是一项重大的技术壮举，需要有团队人员、大量定制的 C 或 C++代码，以及财力雄厚的风险投资家作为后盾。

在接下来的十年里，爆炸式增长的 Web 框架开始汇聚到 LAMP(Linux、Apache、MySQL、PHP/Perl/Python)等常见的基础设施堆栈中。到 2020 年，许多组件，如操作系统、Web 服务器和数据库，使用起来已得心应手，进而使大多数开发人员能够使用 ReactJS 等高级框架，专注于研究面向用户的应用层。

数据科学的基础设施也在经历类似的演变。原始机器学习和优化库已存在了几十年，除此之外没有太多其他基础设施。现在，在 21 世纪 20 年代初，我们正在经历数据科学库、框架和基础设施的爆炸式增长。这种增长通常由商业利益驱动，类似于互联网繁荣期间和其后发生的情况。如果历史可以为证，这种分片化的环境中将诞生广泛共享的模式，这将成为数据科学通用开源基础设施堆栈的基础。

在构建任何基础设施时，请记住，基础设施只是达到目的的手段，而不是目的本身。在我们的案例中，我们希望构建基础设施，使数据科学项目和负责这些项目的数据科学家更加成功，如图 1.2 所示。

下一节将介绍堆栈的目标是解读 4 个 V(即 4 个维度)：堆栈应该能够以更快的速度交付更多数量和更多多样性的项目，而不会影响结果的有效性。然而，堆栈本身并不能交付项目。成功的项目由数据科学家交付，而堆栈有望大幅提高数据科学家的生产效率。

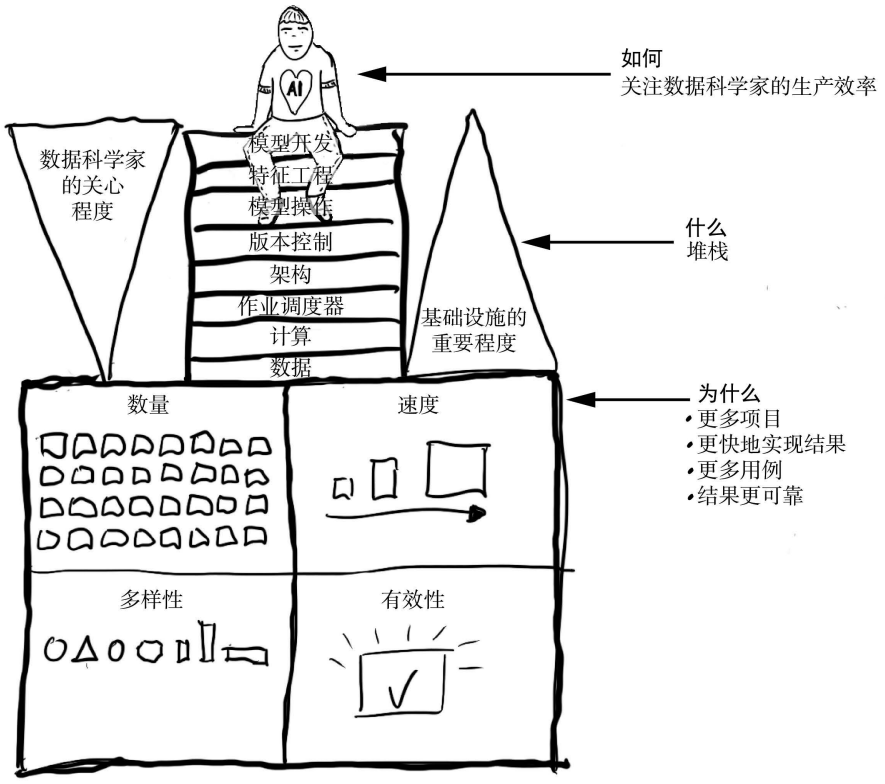

图 1.2　本书的主要关注点

1.2.1　数据科学基础设施堆栈

数据科学基础设施堆栈的具体元素是什么？得益于硅谷和全球公司之间的开源和相对自由的技术信息共享文化，我们能够观察和收集数据科学项目和基础设施组件中的常见模式。尽管实施细节各不相同，但在大多数项目中，主要基础设施层是相对统一的。本书的目的是详细描述这些层，以及这些层为数据科学构建的基础设施堆栈。

图 1.3 所示的堆栈不是构建数据科学基础设施的唯一有效方法。但这种方法是合理有效的：如果从第一原则开始，而不解决堆栈的所有层，很难成功执行数据科学项目。作为练习，你可以挑战堆栈的任何层，并询问如果该层不存在会发生什么。

每一层都能以各种方式实现，由环境和用例的特定需求驱动，但总体情况非常一致。

应该组织构建数据科学的基础设施堆栈，使得最基本的通用组件位于堆栈的底部，而让堆栈的顶层专门针对数据科学。

堆栈是将本书各章耦合在一起的关键心智模型。当你读到最后一章时，将能够回答一些问题，例如，为什么需要堆栈、每一层的用途，以及如何在堆栈的每一层做出适当的技术选择。你将能够构建具有一致愿景和架构的基础设施，从而为使用这一基础设施的数据科学家提供连贯、愉快的体验。为了让你了解这些层的含义，我们将从下到上逐层进行介绍。

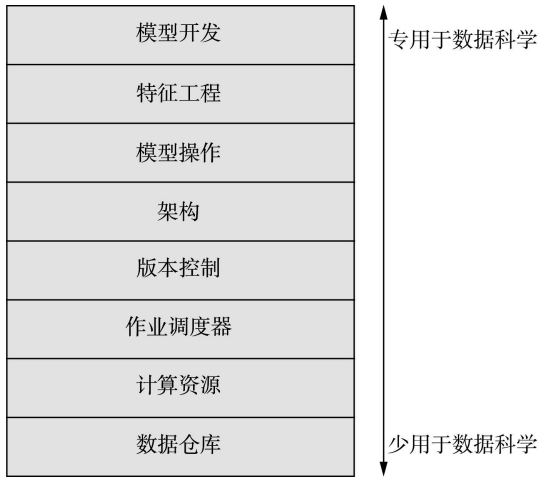

图 1.3　数据科学的基础设施堆栈

数据仓库

数据仓库存储应用程序使用的输入数据。一般来说，有效做法是依靠一个中心化的数据仓库作为事实的共同来源，而不是专门为数据科学构建一个单独的仓库，因为这样很容易导致数据和定义出现分歧。第 7 章将专门讨论这一广泛而深入的主题。

计算资源

原始数据本身不会做任何事情，你需要运行计算，如数据转换或模型训练，以将原始数据转换为更有价值的东西。与软件工程的其他领域相比，数据科学对计算的需求往往更多。数据科学家使用的算法有多种形式和大小。有些需要很多 CPU 内核，有些需要数个 GPU，还有一些需要大量内存。我们需要一个能够平滑扩展的计算层，以处理不同类型的工作负载。第 4 章和第 5 章将讨论这些主题。

作业调度器

可以说，数据科学中没有一次性的操作：模型应该定期重新训练，并根据需要生成预测。数据科学应用程序可被视为一个不断运作的引擎，通过模型源源不断地推送数据流。调度层的工作是确保机器以预期节奏运行。此外，调度器有助于将应用程序作为计算中关联步骤的工作流来构建和执行。第 2、3 和 6 章将讨论有关作业调度和工作流编排的主题。

版本控制

实验和迭代是数据科学项目的典型特征。因此，应用程序总是会发生更改。然而，线性过程很少。通常，我们无法预知应用程序的哪个版本比其他版本有所改进。要正确判断版本，就需要并行运行多个版本，作为 A/B 实验。为了确保开发和实验快速且有条理，我们需要一个鲁棒的版本控制层来确保工作的有序性。第 3 章和第 6 章将讨论与版本控制相关的主题。

架构

除了核心的数据科学工作，我们还需要大量的软件工程来构建一个鲁棒的、可用于生产的

数据科学应用程序。越来越多的公司发现：赋能未经软件工程训练的数据科学家，使其自主构建应用程序，同时用鲁棒的基础设施来支持应用程序，将为公司带来益处。基础设施堆栈必须为数据科学家提供软件基础和指南，确保他们生成的代码遵循架构最佳实践。我们将在第 3 章介绍 Metaflow。这是一个开源框架，可以对许多类似实践进行编码。

模型操作

数据科学应用程序没有固有价值，仅在连接到其他系统(如产品用户界面或决策支持系统)时才有价值。一旦部署了应用程序，使其成为产品体验或业务运营的关键部分，它就能在各种条件下运行，并提供正确的结果。就像所有生产系统偶尔发生故障那样，如果应用程序发生故障，那么系统必须用于快速检测、排除故障和修复错误。我们可以从传统软件工程的最佳实践中学到很多相关知识，但数据和概率模型具有不断更改的本质，从而赋予了数据科学操作一种特殊的意义，这将是第 6 章和第 8 章讨论的主题。

特征工程

数据科学的核心关注点位于面向工程的层上。首先，数据科学家必须找到合适的原始数据，确定其期望的子集，开发数据转换，并决定如何将结果特征输入模型。设计此类流程是数据科学家日常工作的主要部分。从人类生产力和计算复杂性的角度看，我们应该努力使该流程尽可能高效。有效的解决方案通常只针对某一特定问题领域，因此基础设施应该能够支持第 7 章和第 9 章中讨论的各种特征工程方法。

模型开发

最后，在堆栈的最顶层是模型开发层：寻找并描述一个数学模型，将特征转换为预期输出。我们希望这一层在数据科学家的专业领域内稳定存在，因此基础设施不需要对建模方法过于武断。我们应该能够支持各种现有的库，以便科学家灵活选择适合工作的最佳工具。

让许多行业新人感到惊讶的是，端到端机器是一个高效的数据科学应用程序，而模型开发只占端到端机器的一小部分。模型开发层相当于人脑，人脑只占人体总体重的 2%～3%。

1.2.2 支持数据科学项目的整个生命周期

基础设施堆栈的目标是在整个生命周期支持一个典型的数据科学项目，从其开始部署一直到无数次的增量改进迭代。之前，我们确定了大多数数据科学项目常见的三个共同主题，如下所示。图 1.4 显示了这些主题映射到堆栈的过程。

(1) 很容易看出，无论问题领域如何，每个数据科学项目都需要处理数据和计算，因此这些层构成了基础设施。这些层不知道具体的执行内容。

(2) 中间层定义了单个数据科学应用程序的软件架构：执行内容及方式——算法、数据流程、部署策略和结果分布。大部分工作都与集成现有软件组件相关。

(3) 堆栈的顶部是数据科学领域：定义一个数学模型以及如何将原始输入转换为模型可以处理的内容。在一个典型的数据科学项目中，当数据科学家用不同的方法进行实验时，这些层可以快速演变。

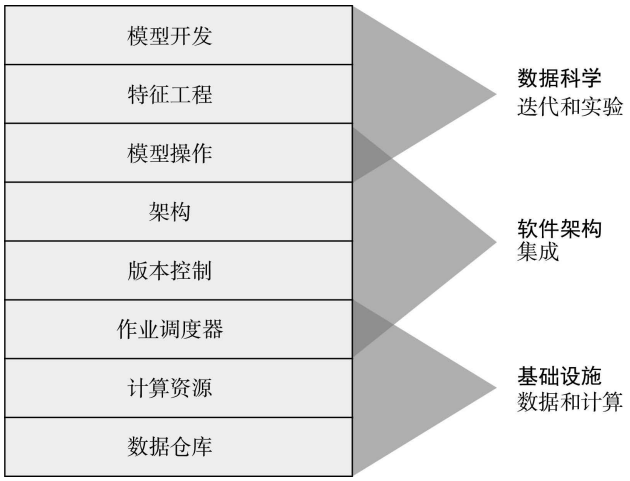

图 1.4　映射到基础设施层的数据科学项目的关注点

注意，层和主题之间不存在一对一的映射。关注点彼此重叠。我们使用堆栈作为设计和构建基础设施的蓝图，但用户不必对此过多关注。特别是，用户不应该触及层与层之间的接缝，而应该将堆栈用作一个高效的数据科学基础设施。

下一章将介绍 Metaflow。该框架通过示例展示了如何在实践中实现上述内容。也可以通过组合框架来定制自己的解决方案，这些框架需要遵循后面几章中列出的一般性原则，以此解决堆栈的不同部分。

1.2.3　不能以偏概全

如果你的公司需要高度专业化的数据科学应用程序(自动驾驶汽车、高频交易系统或可部署在资源受限的物联网设备上的小型模型)，该怎么办？当然，对于这样的应用程序而言，基础设施堆栈也存在巨大差异。

假设贵公司希望向市场提供最先进的无人驾驶飞机。整个公司都在开发一个数据科学应用程序：无人机。当然，这样一个复杂的项目涉及许多子系统，但最终的结果是产生一个应用程序，因此，数量或多样性不是首要考虑的问题。毫无疑问，速度和有效性很重要，但公司可能觉得核心业务需要高度定制的解决方案。

可使用图 1.5 所示的象限来评估你的公司需要的是高度定制的解决方案还是通用的基础设施。

无人机公司的这个特殊的应用程序对通用基础设施的多样性和数量没有需求，因此公司可能会专注于构建一个定制应用程序。同样，如果一家小型初创企业需要使用预测模型对二手车进行估价，则可以快速组装一个基本的应用程序来完成这项工作，而不必重新构建基础设施。

相比之下，一家大型跨国银行则拥有数百个数据科学应用程序，涵盖从信用评级到风险分析和交易等多个领域。每个应用程序都可使用容易理解的模型来解决(尽管很复杂，但“通用”在这种情况下并不意味着简单或不先进)，因此选择一个通用基础设施合情合理。而生物信息

学研究所可能有许多高度专业化的应用程序，因此需要高度定制的基础设施。

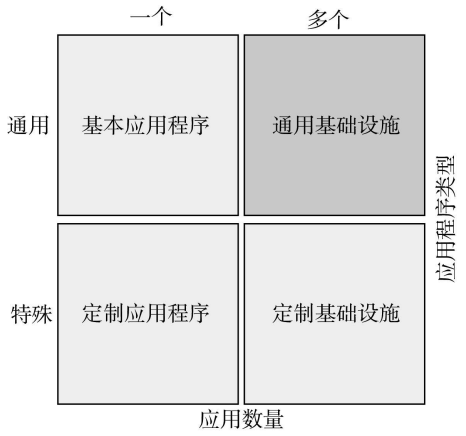

图 1.5　基础设施类型

随着时间的推移，无论从哪里开始，公司都倾向于使用通用基础设施。一家最初拥有定制应用程序的无人机公司，最终都需要其他数据科学应用程序来支持销售、市场营销、客户服务或其他产品线。公司可以为自己的核心技术保留一个专门的应用程序，甚至是定制的基础设施，同时为其他业务使用通用的基础设施。

提示　在决定基础设施策略时，请考虑最广泛的用例集，包括新的具有实验性的应用程序。一种常见的错误是根据几个最常见的应用程序的需求来设计基础设施，这并不能代表大多数(未来)用例的需求。事实上，最常见的应用程序可能需要一种可以与通用基础设施共存的定制方法。

定制应用程序在规模(如谷歌搜索)或性能(如必须在几毫秒内提供预测的高频交易应用程序)方面可能有独特的需求。像这样的应用程序通常需要人工定制：需要由经验丰富的工程师精心创建，还可能需要使用特定的硬件。定制应用程序存在的缺点是，通常很难在速度和数量方面进行优化(所需的特殊技能限制了可使用该类应用程序的人数)，且无法在设计上支持各种应用程序。

仔细考虑你需要构建或支持什么类型的应用程序。如今，大多数数据科学应用程序都可以由通用基础设施支持，这就是本书的主题。这是一件好事，可以帮助你对数量、速度、多样性和有效性进行优化。如果你的某个应用程序有特殊需求，则可能需要定制的方法。对于这种情况，较好的做法是将特殊应用程序视为特殊情况，同时仍对其他应用程序使用通用基础设施。

1.3　良好基础设施的重要性

当我们浏览基础设施堆栈的各层时，可以看到构建现代数据科学应用程序所需的各种技术

组件。事实上，大规模的机器学习应用程序是人类有史以来构建的最复杂的机器，比如 YouTube 的个性化推荐，或者另一个更为普通的例子：实时优化横幅广告的复杂模型。这类应用程序涉及数百个子系统和数千万行代码。

按照我们最初的例子，相比于许多生产级数据科学应用程序，为乳制品行业构建基础设施的复杂性可能要低一个数量级。许多复杂性并不体现在表面上，但当出现故障时肯定会变得明显。

为了说明复杂性，想象一下，前面提到的八层堆栈为数据科学项目提供了动力。记住，单个项目可以涉及多台相互连接的机器，每台机器代表一个复杂的模型。不断有新的数据流并且可能是大量的数据通过这些机器。这些机器由一个计算平台提供动力，该平台需要同时管理数千台不同大小的机器。机器由一个作业调度器进行编排，确保数据在机器之间正确流动，并且每台机器各自在正确的时刻执行自己的操作。

我们有一个数据科学家团队研究这些机器，每位科学家都在快速迭代中测试所分配机器的不同版本。我们希望确保每个版本都产生有效的结果，希望通过并行执行每个版本来实时评估。每个版本都需要自己的独立环境，以确保版本之间不会相互干扰。

这个场景应该会让人想起工厂车间的样子，工厂里有工人和数百台不停运作、嗡嗡作响的机器。与工业时代的工厂不同，这个工厂不是只建一次，而是在不断发展，每天都会有些变化。软件不受物理世界的限制，但必然会产生不断增长的业务价值。

故事不会就此结束。一家大型或中型现代公司并非只有一个工厂，或是一个数据科学应用程序，而是可以拥有任意数量的应用程序。应用程序的庞大数量带来了操作负担，但主要挑战来自多样性：每个现实世界的问题领域都需要不同的解决方案，每个解决方案都有自己的需求和特点，从而导致需要支持的应用程序种类繁多。应用程序通常相互依赖，从而进一步加剧了复杂性。

举个具体的例子，假设有一家中型电子商务商店。这家店有一个定制的推荐引擎(“这些产品推荐给你！”)；衡量营销活动有效性的模型(“在康涅狄格州，Facebook 广告的效果似乎比谷歌广告的效果更好”)；物流优化模型(“投放 B 类货物比保留库存更有效”)；以及用于估计客户流失的财务预测模型(“购买 X 的客户似乎流失更少”)。这 4 个应用程序每个都是一个独立的问题。这可能涉及多个模型、多个数据流程、多人和多个版本。

1.3.1　管理复杂性

现实生活中数据科学应用程序的复杂性给基础设施带来了许多挑战。对于这个问题没有简单、精巧的技术解决方案。我们没有将复杂性视为可被清除或模糊处理的麻烦，而是将管理复杂性作为有效基础设施的关键目标。我们在多个方面应对挑战，具体如下：

- **实现**——设计和实现可以处理这种复杂性的基础设施是一项非常重要的任务。我们稍后将讨论应对工程挑战的策略。
- **可用性**——尽管涉及复杂性，但这是有效基础设施的一个关键挑战，以使数据科学家

富有成效。后文将介绍以人为中心的基础设施，可用性是此类基础设施的关键动机。

- **操作**——如何在最少的人为干预下保持机器运转？减少数据科学应用程序的操作负担是基础设施的另一个关键目标，这是本书各章的共同主题。

在所有这些情况下，我们必须避免引入偶然复杂性，或问题本身并不需要而所用方法也不需要的复杂性。偶然复杂性对于现实世界的数据科学来说是一个巨大的问题，因为我们必须处理高度的固有复杂性，这使得很难区分真实问题和想象问题。

你可能听说过样板代码(该代码的存在只是为了满足框架)、意大利面流程(系统间的关系缺乏组织性)或依赖性地狱(难以管理不断更改的第三方库的图)。除了这些技术问题，我们还要面对人类组织带来的偶然复杂性：有时我们不得不在系统之间引入复杂的界面，但这并不是出于技术需要，而是因为界面要遵循组织边界，如数据科学家和数据工程师之间的边界。要了解更多相关信息，请参阅论文“Hidden Technical Debt in Machine Learning Systems”，该论文于 2015 年由谷歌出版，广受引用。

高效的基础设施有助于公开并管理固有复杂性(固有复杂性是我们所处世界的自然状态)，同时有意避免引入偶然复杂性。为此，需要不断进行判断。幸运的是，我们可以使用简单性控制偶然复杂性。简单性是一种久经考验的启发式方法。“一切都应该尽可能简单，但不能过分简单”是适用于所有高效数据科学基础设施的核心设计原则。

1.3.2 利用现有平台

如前几节所述，我们的工作就是基于八层堆栈构建高效的通用数据科学基础设施。我们希望以此管理现实世界的复杂性，同时将基础设施本身带来的额外复杂性降至最低。这听起来似乎是一项艰巨的任务。

很少有公司负担得起专门的大型工程师团队，为数据科学构建和维护基础设施。小型公司可能有一两名工程师专门负责这项工作，而大型公司可能有一个小团队。最终，公司希望用数据科学应用程序创造业务价值。基础设施是实现这一目标的手段，而不是目标本身，因此，公司应该确定相应的基础设施投资的规模。总之，在建设和维护基础设施方面，我们能花费的时间和精力是有限的。

幸运的是，正如本章开头所述，本书中的所有内容在技术上都可以在几十年内实现，因此我们不必从零开始。我们的工作不是发明新的硬件、操作系统或数据仓库，而是要利用现有的最佳平台，将其集成，并使其易于对数据科学应用程序进行原型开发和产品化。

工程师常常低估了“可能”和“容易”之间的差距，如图 1.6 所示。在“可能”一侧，以各种方式重新实现事物很容易，而不用真正回答如何从根本上使事情变得更容易的问题。然而，正是“容易”一侧使我们能够最大限度地实现数据科学应用程序

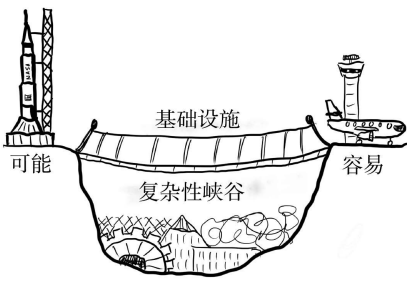

图 1.6 基础设施使事情变得容易

的 4 个维度——数量、速度、多样性和有效性，所以我们不应该在左侧(即“可能”一侧)花费太多时间。

本书首先帮助你尽可能地利用现有组件构建桥梁，这本身是一项非常重要的任务。由于堆栈各层并不相同，因此对于每个组件都有其他团队和公司进行研究。随着时间的推移，如果我们发现某一组件有问题，则可以在不干扰用户的情况下，用更好的替代品取代它们。

奇思妙想

云计算是解决方案的最好例子，尽管并不总是那么容易，但可以使许多事情在技术上成为可能。公共云(如 Amazon Web Services、Google Compute Platform 和 Microsoft Azure)允许任何人访问以前只有大型公司才能使用的基础层，从而大幅改变了基础设施的格局。这些服务不仅在技术上可用，而且在认真使用后，成本效益也非常高。

除了普及底层基础设施，云从本质上改变了我们构建基础设施的方式。以前，为高性能计算构建系统时面临的许多挑战都与资源管理有关，例如，如何保护和限制对有限计算和存储资源的访问，以及如何使资源的使用尽可能高效。

云改变了我们的思维方式。所有的云都提供了一个数据层，如 Amazon S3，提供了几乎无限量的存储，具有接近完美的耐久性和高可用性。同理，云还提供了几乎无限的、弹性伸缩的计算资源，如 Amazon Elastic Compute Cloud(Amazon EC2)和以此为基础构建的各种抽象概念。在构建系统时，我们可以假设拥有大量可用的计算资源和存储空间，并转而关注成本效益和生产效率。

本书假设你可以访问与云类似的基础设施。到目前为止，满足这一要求的最简单方法是在某家云供应商创建一个账户。你可以花几百美元构建并测试堆栈，也可以通过云提供的免费层进行构建和测试。或者，你可以构建或使用现有的私有云环境。然而，如何构建私有云则超出了本书的讨论范围。

云还为数据科学提供了更高级别的产品，如 Azure Machine Learning(ML)Studio 和 Amazon SageMaker。通常，可通过最少的定制将这些产品用作端到端平台，或者，也可将其部分集成到自己的系统中。本书采用后一种方法：你将学习如何利用云提供的各种服务以及使用开源框架，来构建自己的堆栈。尽管这种方法需要更大的工作量，但将为你提供更大的灵活性，结果可能更易于使用，而且定制堆栈也可能更具成本效益。在接下来的章节中，你将了解其中的原因。

总之，你可以利用云来解决低级、无差别的技术重任，从而有助于将有限的开发预算集中在独特的、差异化的业务需求上；最重要的是，应集中在优化组织中数据科学家的生产效率上。正如下一节所述，可以利用云逐渐将我们的重点从技术问题转移到人类问题上。

1.4　以人为中心的基础设施

基础设施旨在从多个方面最大化组织的生产力，可支持更多项目，交付更快，获得更可靠的结果，覆盖更多业务领域。为了更好地理解基础设施的完成方式，请考虑在没有高效的基础

设施时会出现的以下典型瓶颈：

- **数量**——我们无法支持更多的数据科学应用程序，因为没有足够的数据科学家来进行研究。现有的数据科学家都在忙于改进和支持现有的应用程序。
- **速度**——我们无法更快地交付结果，因为开发可投入生产的模型 X 是一项重大的工程任务。
- **有效性**——模型的原型在笔记中运行良好，但我们没有想到模型可能会收到类似 Y 的数据，从而在生产中失效。
- **多样性**——我们希望支持一个新的用例 Z，但我们的数据科学家只了解 Python，而 Z 周围的系统只支持 Java。

所有这些情况都存在一个共同的因素，即人类带来了瓶颈。除了一些高度专业化的应用程序，由于硬件或软件的基本限制导致项目无法交付的情况很少发生。而人类无法足够快地交付软件就是一个典型的瓶颈成因。即使能够足够快地运行代码，也可能忙于维护现有系统，这是另一项至关重要的人类活动。

这一观察结果有助于我们认识到，尽管“基础设施”听起来很具有技术性，但我们并不是在为机器构建基础设施。我们构建基础设施旨在提高人类的生产力。这一认识将从根本上影响我们应该如何为数据科学家思考并设计基础设施，而数据科学家是人类，并非机器。

例如，若假设人类时间比计算机时间更宝贵(对于大多数数据科学家来说确实如此)，就应该淘汰像 C++这样的低级语言，而使用像 Python 这样的语言，尽管它使工作负载的处理效率更低，但它具有高表达性、高生产力。我们将在第 5 章深入探讨这个问题。

注意在上一节中，我们希望最大限度地利用现有平台，并将其集成到基础设施堆栈中。我们想要以一种提供连贯用户体验的方式做到这一点，最大限度地减少用户必须独立理解和操作每一层时所需的认知开销。我们的假设是，通过减少与基础设施相关的认知开销，可以提高数据科学家在认知开销最重要的领域(即数据科学本身)中的生产效率。

1.4.1 自由与责任

流媒体视频公司 Netflix 以其独特文化而闻名，最近的一本书 *No Rules Rules：Netflix and the Culture of Reinvention*(Penguin 出版社，2020)中详细描述了该公司的独特文化，其作者是 Erin Meyer 和 Netflix 的联合创始人兼长期首席执行官 Reed Hastings。Netflix 的核心价值观之一是“自由与责任”，赋予所有员工高度自由来决定自己如何工作。另一方面，公司期望员工考虑做最符合公司利益的事情，并在工作中认真负责。一个以人为中心的数据科学基础设施框架 Metaflow 诞生于 Netflix，深受 Netflix 公司文化的影响，第 3 章将对其进行介绍。

我们可以将自由与责任的概念应用于数据科学家的工作以及一般的数据科学基础设施。我们期望数据科学家是自己领域内的专家，例如，与特征工程和模型开发相关的问题领域；但并不要求他们是系统工程或其他与基础设施相关的主题的专家。然而，如果数据科学基础设施可用，我们希望他们有足够的责任感来选择利用基础设施。这个想法可以映射到基础设施堆栈，

如图 1.7 所示。

图中左侧的三角形描绘了数据科学家的专业知识和兴趣领域。堆栈顶部最宽，专用于数据科学。在这几层中，基础设施应该赋予数据科学家最大的自由，允许他们根据自己的专业知识自由选择最佳的建模方法、库和特征。我们希望数据科学家稍后能够更自主地处理这一问题，因此他们应该关注模型操作、版本控制和架构，这是他们职责的一部分。

图 1.7　基础设施补充了数据科学家的兴趣

图中右侧的三角形描绘了基础设施团队的自由与责任。基础设施团队应该有最大的自由度来选择和优化堆栈的最底层，从技术角度看，最底层是至关重要的。可以在不过分限制数据科学家自由的情况下做到这一点。然而，随着层的升高，责任逐渐减轻。基础设施团队无法对模型本身负责，因为他们通常不具备专业知识，而且绝对不具备支持所有用例的规模。

这种安排具有双重目的：一方面，我们可以让数据科学家个人专注于他们喜欢的事情，赋予他们高度自由，从而最大限度地提高他们的生产效率和幸福感。另一方面，我们可以要求数据科学家负责地使用堆栈(包括他们可能不太感兴趣的部分)，来实现符合公司利益的 4 个维度。最终在公司的需求和数据科学家的幸福之间实现合理的平衡。

1.4.2　数据科学家自主性

到目前为止，我们已经浅谈了“基础设施团队”和“数据科学家”。然而，数据科学项目中的实际角色可以更加丰富多彩，如下所示：

- **数据科学家或机器学习研究人员**，负责开发机器学习或其他数据科学模型，并制作原型。
- **机器学习工程师**，以可伸缩的、可投入生产的方式实现模型。
- **数据工程师**，为输入和输出数据建立数据流程，包括数据转换。
- **DevOps 工程师**，在生产中部署应用程序，并确保所有系统完美运行。
- **应用程序工程师**，将模型与其他业务组件(如 Web 应用程序)集成，这些组件是模型的消费者。
- **基础设施或平台工程师**，为许多应用程序提供通用的基础设施，如数据仓库或计算平台。

除了技术团队，数据科学项目的参与者还可能包括：业务所有者，了解应用程序的业务背景；产品经理，将业务环境映射到技术需求；以及项目经理，帮助协调跨职能协作。

任何参与过涉及多方利益的项目的人都知道，保持项目向前推进需要大量的沟通和协调。除了协调开销，增加并发数据科学项目的数量也很有挑战性。原因很简单，因为没有足够的人来填补所有项目中的所有角色。出于上述原因以及其他许多原因，许多公司认为，只要项目的执行不受影响，就可以减少项目的参与者。

基础设施堆栈的目标是能够合并前 4 种技术角色，从而使数据科学家能够在项目中自主地实现所有这些功能。公司中可能仍然存在这些角色，但不是每个项目都需要，这种情况下可以将这些角色分配给几个关键项目，他们可能支持更多的横向、跨项目的工作。

总之，数据科学家不可能突然成为 DevOps 或数据工程方面的专家，但应该能够以可伸缩的方式实现模型，建立数据流程，并独立部署和监控生产中的模型。数据科学家应该为此投入最小的额外开销，从而保持对数据科学的关注。这是以人为中心的数据科学基础设施的关键价值主张。从下一章开始，我们将从零开始构建以人为中心的数据科学基础设施。

1.5 本章小结

- 尽管在没有专用基础设施的情况下，也可以开发和交付数据科学项目，但高效的基础设施可以在不影响结果有效性的情况下，以更快的速度开发更多数量和多样性的项目。
- 支持数据科学家需要一个完整的系统基础设施堆栈，包括从数据和计算等基础层到特征工程和模型开发等更高级别的问题。本书将系统介绍所有层。
- 数据科学中并不存在普适方法。有效做法是根据你的特定需求定制基础设施堆栈的各个层。本书将帮助你实现这一点。
- 现代数据科学应用程序是一个精密、复杂的机器，涉及许多移动工件。管理好这种固有复杂性，并避免在执行时引入任何不必要的复杂性，是数据科学基础设施面临的一个关键挑战。
- 利用现有的、久经考验的系统(如公共云)，可以有效控制复杂性。接下来的各章将帮助你在堆栈的每一层选择合适的系统。
- 最终，人类往往是数据科学项目的瓶颈来源。我们的重点应该是提高整个堆栈的可用性，从而提高数据科学家的生产效率。