第1章 MLOps概览

机器学习（machine learning，ML）作为从数据中学习和提取规律的强效工具，其价值已得到充分验证。过去十年间，得益于海量数据的生成、存储与处理能力的突破，以及计算资源的易获取性，该领域相继涌现出图像识别、语言翻译、大语言模型（large language model，LLM）等突破性进展，诸如BERT、DALL-E、ChatGPT等代表性成果相继问世。

如今机器学习已走出学术实验室的象牙塔，在商业领域获得规模化应用。企业通过部署ML技术解决现实业务难题，借助客户体验优化、成本控制、运营效率提升等路径构建竞争优势，最终实现行业变革。麦肯锡《2021年人工智能发展现状》报告[①]显示，全球各区域企业持续加速AI/ML技术采纳进程，这一趋势的核心驱动力源自AI对企业经营效益产生的实质性影响。

AI/ML的技术价值已无须赘言，当前企业决策层更关注的核心命题在于：如何将AI/ML技术高效融入业务流程与产品体系，实现商业价值的最大化释放。这要求企业建立可持续迭代、流程规范、安全可控的机器学习工程化体系，确保技术落地过程兼具效率与可预期性。

---

① Global Survey: The State of AI Adoption 2021 – www.mckinsey.com/capabilities/quantumblack/our-insights/global-survey-the-state-of-ai-in-2021

Gartner[①]与 VentureBeat[②]等机构的研究数据表明，机器学习项目的工程化落地是一项复杂的系统工程，需要构建涵盖模型开发、部署实施、持续运维的全链路标准化流程，并配套相应技术能力作为支撑。这正是 MLOps（机器学习运维）方法论的价值所在。

1.1 MLOps 体系解析

软件工程的核心使命是为企业创造业务价值，而价值的真正兑现始于软件在生产环境的成功部署。部署效率直接决定价值实现速度，这一认知推动了 DevOps 方法论在全球范围内的普及。通过消除开发与运维的壁垒，建立协同工作机制，DevOps 借助持续集成（CI）、持续交付（CD）和自动化部署体系，确保大规模软件能够快速、稳定地投入生产环境，这已成为现代软件工程的黄金标准。

机器学习项目同样以价值创造为目标，但其价值闭环的实现路径更为复杂。只有当训练完成的模型及其特征工程体系进入生产环境，并建立有效的监控机制后，项目才真正开始产生投资回报率。值得注意的是，机器学习项目的回报周期存在显著波动性：当模型针对具体业务痛点（如客户流失预测）时，可能在部署初期就能显现价值；但完整的投资回报率往往需要模型深度整合进业务流程，并经过长期稳定运行，方能充分释放。这种特性要求项目团队必须建立持续的模型优化机制和适应性监控体系。

深入辨析机器学习项目与传统软件工程的差异至关重要。二者的核心区别主要体现在哪些维度？DevOps 经验能否直接迁移？这些问题的答案将帮助我们构建对 MLOps 技术体系的完整认知。

---

① Our Top Data and Analytics Predicts for 2019 – https://blogs.gartner.com/andrew_white/2019/01/03/our-top-data-and-analytics-predicts-for-2019/

② Why do 87% of data science projects never make it into production? – https://venturebeat.com/ai/why-do-87-of-data-science-projects-never-make-it-into-production/

注意

DevOps 已在众多软件开发组织中广泛应用，这套方法论不仅能提升软件质量与可靠性，还可显著缩短产品的上市周期。其本质包含双重革新：既是对传统软件开发组织社会协作模式与技术架构的范式变革，也是贯穿软件全生命周期的持续自动化实践。

DevOps 的核心在于构建持续交付管道，涵盖从开发、集成、部署到监控的全流程闭环。这种机制保障了软件版本的高效迭代，使频繁发布既快速又可靠。

采用 DevOps 思维的工程师需突破传统角色边界，不仅要精于代码编写，更要深度参与软件的部署与生产环境维护，形成端到端的质量责任意识。

1.1.1　机器学习项目

虽然机器学习项目遵循特定的开发周期，但其科学探索属性导致项目周期呈现高度迭代特征。如图1.1所示，由于模型训练依赖大量实验验证且对数据质量极度敏感，其开发流程并非传统软件工程的线性推进模式，而是形成包含模型迭代优化、超参数调校与性能增强的螺旋式演进体系。

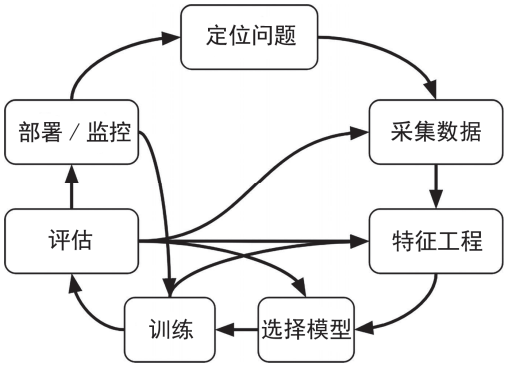

图 1.1　机器学习开发流程

机器学习项目往往服务于具有量化指标的商业或产品目标。项目启动之初，清晰界定问题范畴与目标优先级，是后续各环节有序推进的基石。当模型评估显示预测精度未

达预期或实验数据揭示现有方案的改进空间时，数据科学家往往需要回溯至前期步骤。无论是补充数据采集维度，还是重构特征工程流程，这种螺旋式迭代恰恰是机器学习研发的常态。

真正成功的机器学习项目，体现在研发团队能高效完成开发周期迭代，通过持续整合实验洞察优化数据管道与算法架构，最终锻造出预测性能卓越的工程化模型。其核心诉求在于，当面对未知数据时，模型能稳定输出符合业务预期的精准预测。

尽管机器学习开发生命周期具有循环迭代特性，但其核心架构可归纳为五个主要阶段：

- 数据采集与清洗。
- 特征工程。
- 模型训练。
- 模型部署。
- 模型监控。

1.1.2 机器学习项目的输入和输出

在传统软件开发范式下，工程师通过编码实现预设逻辑，如图 1.2 所示，其核心是确定性的输入/输出转换。

图 1.2 传统软件开发流程

而在机器学习领域，数据科学家聚焦于特征工程与模型研发两大维度。理解这两个关键环节的输入和输出要素，是把握机器学习项目本质的重要切入点。

特征的数量和质量直接决定模型性能上限。如图 1.3 所示，数据科学家超过 60%的精力投入在数据探索与特征工程环节，通过代码将原始数据转化为蕴含预测价值的特征矩阵，以便于训练 ML 模型。

图 1.3　ML 模型训练流程

当特征集通过验证后，研发进入模型训练阶段。该过程需要灵活运用算法库、调整超参数组合，并通过大量对比实验寻找最优解。若模型验证指标未达阈值，研发团队可能需要重新审视特征选择策略，甚至引入新的数据源进行补充。

完整的机器学习项目将产出四大成果：

- 经过清洗的结构化数据集。
- 从原始数据到特征的逻辑。
- 模型训练代码与参数配置。
- 可投入生产的 ML 模型。

ML 模型往往需要重新训练，这源于多种驱动因素，包括新业务需求的产生、新增数据源、机器学习库的可用性提升、模型性能出现衰减等情形。因此，针对 ML 成果，如图 1.4 所示，实施有效的版本控制与管理机制至关重要。

ML 模型 = 数据 + ML 算法 + 超参数

图 1.4　ML 成果

机器学习项目与传统软件工程项目存在显著差异，其独特性可归纳为以下核心特征：

- 模型训练依赖历史数据，这使得机器学习项目需要投入大量数据治理工作，包括数据采集和标注、输入数据的统计分析及可视化处理等环节。
- 模型研发具有高度探索性和迭代性，需持续进行实验验证。
- 当新数据的统计分布与训练数据产生偏移时，模型性能会随时间推移逐渐衰减。
- 项目成功依赖跨领域协作，要求数据科学家、工程师与领域专家密切配合，实现技术能力与专业知识的有机融合。

业界常将 MLOps 类比为机器学习领域的 DevOps 实践。MLOps 通过建立技术规范与管理流程的最佳实践体系，致力于帮助企业实现机器学习项目的高效部署与规模化应用。

1.2 MLOps 的价值定位

随着全球企业日益认识到 AI/ML 技术的价值，它们纷纷投入预算将其应用于提升业务价值、增强竞争力等场景，在此背景下，如何量化机器学习项目的投资回报率（ROI）便成为关键考量。值得注意的是，真正的投资回报率只能产生于 ML 模型部署至生产环境并深度融入企业产品或业务流程之后。那么，推动 ML 模型高效落地的核心要素究竟是什么？经过多年的实践积累与行业探索，机器学习从业者已形成共识——答案正是 MLOps。

本节将剖析机器学习项目落地过程中的典型障碍，并阐释 MLOps 的应对策略。

1.2.1 机器学习项目实施的挑战

Gartners 与 NewVantage Partners 的多项研究显示，尽管企业普遍试图通过机器学习项目提升业务价值，但在实际部署环节却屡屡受挫。由于难以实现模型在生产环境中的快速落地、高效运行和持续迭代，这些项目的投资回报率往往低于预期。NewVantage Partners 2020[①]年的调研数据尤为触目，仅有 15%的头部企业真正实现了 AI 能力的规模化应用。

如今业界已形成共识，机器学习项目的工程化落地与传统软件部署存在本质差异，其复杂程度远超预期。

本节将剖析机器学习领域常见的实施困境，追溯其形成机理。需要说明的是，诸如专业人才短缺、商业目标模糊等非技术性挑战，虽同样是项目折戟的重要因素，但不在本书探讨范畴内。

1. 应用机器学习

在实际 ML 项目中，其核心任务已形成共识，即通过机器学习实现数据驱动的决策与

① AI Stats News: Only 14.6% Of Firms Have Deployed AI Capabilities In Production – www.forbes.com/sites/ gilpress/2020/01/13/ai-stats-news-only-146-of-firms-have-deployedai-capabilities-in-production/

产品创新。

相较于传统软件工程，机器学习作为一门学科，本质上具有更强的实验属性和迭代特征。

具体实施时，首先需要基于数据集训练模型，随后在训练集和独立测试集上验证性能。由于初版模型往往难以达到预期效果，这个过程通常需要反复迭代——每次尝试可能涉及不同的算法架构、超参数配置或特征工程方案。

项目初期，精准预测何种算法组合能带来高性能模型极具挑战。因此，探索性实验与快速迭代成为筛选最优方案、淘汰低效路径的关键环节。

与其他科研领域相似，机器学习实验需要系统记录输入参数、方法路径和输出结果。通过横向对比多组实验结果，能够有效加速分析进程，为后续优化指明方向。

值得注意的是，机器学习领域仍处于高速发展阶段，新技术、新方法和工具库持续涌现。从业者必须保持技术敏感度，主动尝试并整合这些创新来提升模型表现。

这里的核心在于迭代速度：若因流程缺失或工具低效导致实验迭代受阻，ML 模型的投产将难以实现，项目投资回报周期也将大幅延长。

2. 输入垃圾，输出垃圾

计算机领域“输入垃圾，输出垃圾”（garbage in, garbage out）的经典格言，揭示了有缺陷的输入数据必然导致有缺陷的输出结果。这一原理在机器学习领域尤为重要，因为模型训练效果高度依赖于输入数据的质量。

ML 模型的训练过程是将标注数据输入算法并使其学习数据内在规律的过程。业内共识表明，模型性能与训练数据的质量呈正相关。近期，多位知名机器学习专家开始倡导“以数据为中心的 AI”的方法论，强调高质量训练数据对模型性能的决定性作用。

> **注意**　殊途同归的两种 AI 方法论：以模型为中心与以数据为中心
>
> DevOps 已在众多软件开发组织中广泛应用，这套方法论不仅能提升软件质量与可靠性，还能显著缩短产品上市周期。其本质包含双重革新：既是对传统软件开发组织社会协作模式与技术架构的范式变革，也是贯穿软件全生命周期的持续自动化实践。

除了数据质量，数据时效性与统计特征变化等数据特性，同样对模型性能产生显著影响。

若缺乏完善的数据基础设施、严谨的数据工程规范以及专业团队支持，将直接影响模型效果，延缓机器学习项目的投产进程。

传统的“以模型为中心的 AI”方法论聚焦于调整超参数、优化模型架构及算法，通过反复调试以达到预期指标，这种思路长期主导行业实践。

与之目标相同但路径相异的“以数据为中心的 AI”方法论，则保持超参数和模型架构固定，通过基于错误分析的数据迭代持续提升模型性能。根据 Data-centric AI Resource Hub website[①]，该方法论是“系统化构建 AI 系统数据工程体系的学科”，由吴恩达在《MLOps 对话：从以模型为中心到以数据为中心的 AI 转型》专题讲座中首次提出并推广[②]。

3. 发展历程

机器学习最初被视作独立的科研实验，主要由数据科学家单独完成。数据科学家深耕机器学习领域，主要承担模型构建与训练工作。

这种工作模式导致数据科学家往往忽视模型训练之外的工程环节，包括自动化数据管道的建设、高质量代码的开发规范、端到端训练流程的自动化实现，以及将模型集成到生产系统的工程实践。

企业级机器学习项目不同于一次性科研实验，它要求所有软件工程相关环节都必须实现自动化管控、版本追踪、运行监控和可重复验证。若不能引导数据科学家建立工程化思维，或未能提供配套的工具链与基础设施支持，将严重影响机器学习项目的落地效率。更深层次而言，这需要企业整体向产品化思维转型。

4. 团队协作

从模型开发到最终集成至数据决策产品的完整流程，涉及数据工程、机器学习、软件工程和 DevOps 等多个专业领域。这个跨学科的复杂过程本质上是团队协作的过程，需要明

① Data-centric AI Resource Hub – https://datacentricai.org/

② A Chat with Andrew on MLOps: From Model-centric to Data-centric AI – www.youtube.com/watch?v=06-AZXmwHjo

确责任划分、加强跨部门协作，才能确保产品稳定性、持续迭代性，以及最重要的商业价值延续性。

典型的团队协作项目需要明确的角色分工与责任划分，而机器学习生产化是典型的团队协作，因此有必要梳理机器学习开发生命周期中各环节的典型角色职责。

表 1.1 展示了机器学习开发的核心活动，其内容并非穷尽式列举。

表 1.1　机器学习核心活动

活动	角色
数据准备	数据工程师
特征工程	数据科学家
模型训练	数据科学家
模型部署	ML 工程师
模型监控	ML 工程师

部分企业可能会在机器学习实施过程中纳入其他角色，例如业务线负责人或数据治理专员。

大中型企业通常为每个角色配置专职人员，而在初创公司或中小型企业中，往往存在一人兼任多个角色的情况。

论文“MLOps：概述、定义与架构”[①]详细阐释了各角色间的协作关系，具体如图 1.5 所示。

成功的团队协作需要完备的人员配置、清晰的职责划分以及顺畅的协作机制。同理，企业要实现机器学习的高效落地，必须构建具备跨学科背景的复合型团队，建立标准化流程、明确的沟通机制与责任边界，确保各团队协调一致，在关键节点按时交付成果并顺利完成工作交接。

① Machine Learning Operations (MLOps): Overview, Definition, and Architecture - https://arxiv.org/ftp/arxiv/papers/2205/2205.02302.pdf

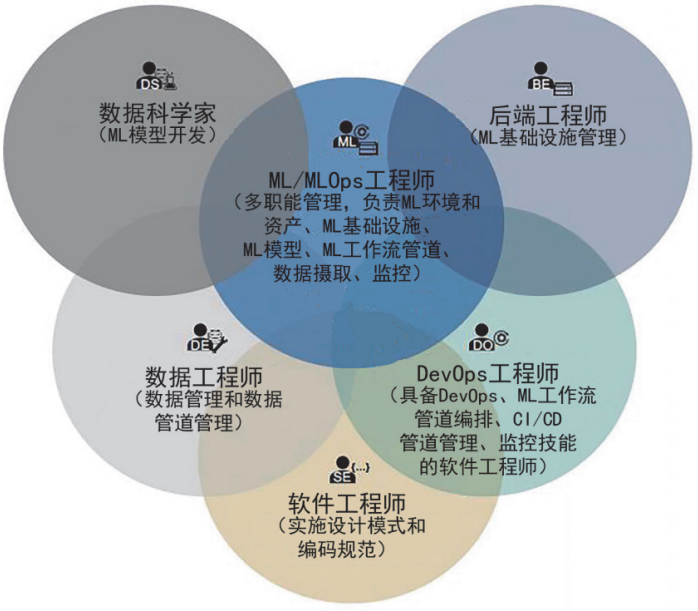

图 1.5　MLOps 的角色协同示意图

5. 挑战综述

虽然持续高效地规模化实施机器学习存在诸多挑战，但其可行性已获验证，且实施成果往往物超所值。

对于积极布局该领域的企业而言，机器学习技术已展现出显著的变革潜力——既能有效降低成本、提升运营效率，又能切实改善企业盈利水平。

本小节将挑战归纳为三个核心维度，即自动化、复现及监控，这些正是 MLOps 技术框架着力解决的重点方向。

1）自动化

如“应用机器学习”一节所述，机器学习研发具有高度实验性与迭代性特征。这意味着在研发流程中，几乎所有环节都能通过自动化获得效率提升。传统人工操作模式存在明显缺陷：错误率高、耗时漫长、结果波动大且难以复现。

自动化技术的应用可显著加速研发进程，使数据科学家能够快速完成开发周期迭代。

正如“输入垃圾，输出垃圾”原则所揭示的，数据相关环节对模型性能具有决定性影响。通过实现数据管道自动化运行与监控等关键环节的自动化处理，将有效保障数据质量与时效性，从而为模型性能优化奠定基础。

2）复现

复杂的机器学习项目往往需要多位数据科学家协同验证假设与训练实验。要实现高效协作，团队成员必须能够快速复现既有实验，这要求先前实验使用的数据、代码及参数配置等关键信息均完整且可追溯。

实践中，基于现有模型迭代开发新版本是常见工作模式。业务需求变更、新增训练数据、用户行为演化等因素都会驱动模型迭代。在此过程中，若能快速复现前期工作成果，将大幅提升新版本模型的开发效率。

3）监控

业界常说，ML 模型的部署只是万里长征第一步，持续维护其运行性能才是真正的考验。这是因为生产环境中的模型性能常常出现退化，这种退化不仅影响用户体验，更可能对企业运营造成实质损害。

性能衰退的诱因复杂多样，例如预测所用特征的质量缺陷、用户行为模式变迁、突发环境变量（如疫情）等都会产生影响。因此，对已部署模型进行持续性能监控，并在关键指标跌破阈值时触发预警机制，就成为数据科学家的必要任务。

这种监控机制同样需要覆盖数据管道，即为模型训练和预测特征提供数据支持的基础设施。

通过对数据质量、特征工程和模型表现的立体化监控，数据科学家和工程师能够建立早期预警系统，在问题萌芽阶段及时干预，确保模型在全生命周期内保持高效运行。

1.2.2　MLOps 的愿景与价值

在深入剖析机器学习项目的开发生命周期、核心要素及落地挑战后，我们需要在先前总结的三大挑战维度框架下，重新审视 MLOps 的价值。

当前业界对 MLOps 的定义存在细微分歧，但其核心诉求高度一致——通过建立系统工程方法，破解机器学习落地难题。

要深入理解 MLOps 的核心内涵，我们可以借鉴剥洋葱的方式逐层剖析。MLOps 体系包含三个基础层面：方法论范式、工程实践准则和核心原则。这三个层面相互支撑，共同构成完整的 MLOps 框架，本节将逐一展开详细论述。通过系统分析每个层面的特性及其相互作用机制，我们能够全面把握支撑 MLOps 体系的理论基础与实践要义，进而理解其在提升机器学习全生命周期管理效能方面的重要价值。MLOps 体系架构如图 1.6 所示。

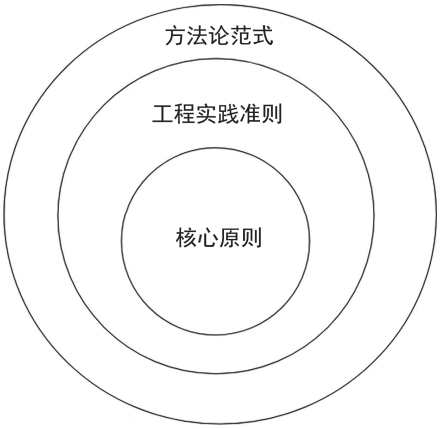

图 1.6　MLOps 体系架构：方法论范式、工程实践准则和核心原则

1. 方法论范式

MLOps 标志着企业机器学习应用的根本性范式转变。这种新范式将机器学习从实验室中的研究课题，提升为企业级的技术资产，需要与传统软件系统同等强度的工程化管理。

领先企业的实践表明，成功的关键在于实现双重转变。首先，将模型及相关制品视为核心软件资产，建立完整的版本控制和质量管理体系；其次，培养工程化的运维思维，在系统设计阶段就融入可靠性、可扩展性等运维要素。这种范式革新，使得机器学习真正成为驱动业务增长的核心引擎。

MLOps 范式包含一整套最佳实践、核心概念与协作机制，这些要素将在后续“工程实践准则”和“核心原则”中深入探讨。

2. 工程实践准则

随着全球企业加速应用机器学习解决商业问题，MLOps 逐渐发展成为融合三大领域精髓的新兴领域——MLOps 有机整合了机器学习、数据工程与 DevOps 的工程实践方法与核心原则，如图 1.7 所示。

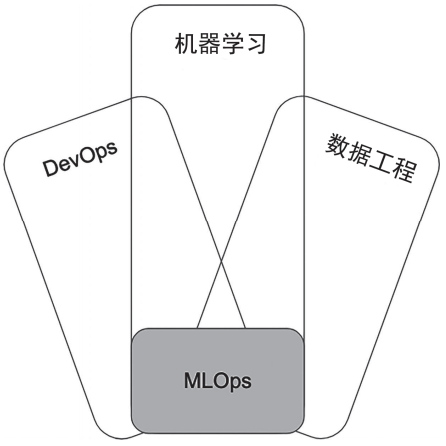

图 1.7　MLOps 工程实践——三大支柱领域的融合

作为系统工程方法论，MLOps 致力于将工程化思维贯穿 ML 模型的开发、部署、监控与全生命周期管理。其核心目标在于建立标准化流程，使企业能够以高效率、高频率、可扩展且可持续的方式实现 ML 模型的工业化部署。简言之，就是通过系统工程方法最大限度缩短从概念验证到生产落地的周期，同时确保与现有软件系统的无缝集成。

1）*数据工程*

在人工智能领域，数据质量与规模直接决定 ML 模型的上限，数据是 AI/ML 的血液。数据工程对 MLOps 有三大贡献：

- 构建标准化数据预处理框架，为模型训练提供高质量原料。
- 搭建异构数据处理基础设施，支持多源异构数据的采集、存储与消费。
- 通过自动化数据管道实现质量监控，确保训练数据满足准确性、完整性和时效性要求。

2）机器学习

企业级机器学习应用的核心竞争力，在于对算法技术的深刻理解与创新应用。

机器学习对 MLOps 有三大贡献：

- 数据分析与特征洞察能力，确保训练数据集的代表性与公平性。
- 算法选型与超参优化方法论，构建面向新数据的强泛化模型。
- 性能评估与迭代优化机制，使模型指标精准对接业务目标。

3）DevOps

MLOps 继承并扩展了 DevOps 的成熟方法论。相较于传统软件开发，MLOps 需要管理更多类型的工程要素，这种复杂性要求对 DevOps 实践进行适应性改造。

DevOps 对 MLOps 有三大贡献：

- 建立跨职能协作、知识共享机制。MLOps 比 DevOps 团队更为庞大，跨职能协作对 MLOps 的成功尤其重要。
- 构建自动化交付管道，通过持续集成/持续部署（CI/CD）实现模型的高频可靠发布，显著缩短从实验到生产的转化周期，并通过自动化重训练维持模型性能。
- 建立持续测试、质量保障、实时监测、日志追踪与反馈闭环。由于 ML 模型性能高度依赖动态变化的训练数据与预测数据，因而必须对数据统计特征和模型表现进行持续监测，这是确保模型稳定运行、降低用户体验风险的核心保障。

数据工程、机器学习与 DevOps 的既有经验为 MLOps 奠定基础，但 MLOps 特有的实验驱动开发模式、跨职能团队协作要求，以及数据－代码－模型三元体系，催生出以下差异化：

- 测试体系升级：除传统单元/集成测试，需增加数据质量验证、模型性能评估及泛化能力验证。
- 部署流程重构：构建自动化训练－部署管道，实现特征库的在线动态更新。
- 生产环境监测：通过追踪预测数据分布变化与线上模型指标波动，主动预警性能衰减。
- 持续训练机制：当数据漂移或代码、模型更新时，在安全边界内自动触发模型迭代。

3. 核心原则

我们详细探讨了 MLOps 所依赖的三个领域的最佳实践。本节旨在系统地提炼实践经验，并引入若干补充要素，最终形成一套适用于任何 MLOps 实施场景的核心原则。

这些原则将为组织的 MLOps 实践提供指导框架，而每条原则的具体执行力度与关注焦点，需结合该组织的人工智能战略、业务目标、具体用例及文化特质进行适配调整。

1）自动化

自动化旨在通过建立标准化流程和工具链，最大限度减少人工干预，系统性地完成机器学习开发生命周期中的关键环节。这涵盖数据、代码和 ML 模型等核心要素的执行、构建、测试、训练及部署过程。

在机器学习实践中，诸如数据管道处理、特征工程管道、模型训练管道等开发，往往需要按固定频率持续迭代。这类重复性工作正是自动化技术最能发挥价值的领域。

自动化需求通常会在三种场景下凸显：

- ML 模型数量达到人工运维的临界点，传统管理方式面临资源消耗过大的挑战。
- 开发团队（包含数据科学家、数据工程师、机器学习工程师等角色）规模超过 10 人时，人员协作成本呈指数级增长。
- 企业业务对 ML 模型价值的依赖度持续加深，模型交付效率直接影响商业竞争力。

通过建立自动化机制，项目参与者能够及时获得各环节的反馈数据，这种实时协同不仅提升了个体工作效率，更显著增强了团队整体研发效能。

2）版本控制

机器学习项目的三大核心要素是数据、代码和模型。遵循 MLOps 的核心准则，应当像 DevOps 对待代码那样，通过版本控制系统对这些要素进行全生命周期管理。

与软件开发规范相似，ML 模型的训练代码不仅需要版本控制，还必须纳入代码审查流程。这种双重机制既保证了训练过程的可追溯性，也确保了模型迭代的可重复性。

业界长期存在一个典型困境：当模型开发者离职后，由于原始训练代码和元数据未被妥善归档至版本控制系统，模型无法重新训练或复现。版本控制正是破解这一困局的关键。

实现训练数据版本控制的难点，主要源于海量数据存储带来的技术挑战。

3）实验跟踪

正如前文所述，机器学习开发本质上是一项具有高度迭代性和实验性的科研活动。为支持机器学习这一独特属性，帮助数据科学家高效开展实验、评估结果并与团队协作，我们需要建立完善的机制来追踪元数据，元数据包括实验参数、性能指标、模型谱系（model lineage）、数据及代码等信息。

追踪的价值不仅在于确保结果可复现，更重要的是构建完整的追溯体系。考虑到 ML 模型的探索与迭代过程需要大量时间和计算资源投入，任何能够优化这两个维度的措施，都将显著提升组织的研发效率。

4）可复现性

可复现性原则强调在机器学习全流程（涵盖特征工程、模型训练实验、部署等关键环节）中，给定相同输入条件时必须能够复现实验结果。

传统软件开发通过版本控制系统即可满足可复现性要求，但机器学习项目需要更复杂的保障机制。具体而言，必须系统性追踪各类数据管道、特征生成逻辑、训练代码版本、超参数配置（特别是数据集的版本管理），以及模型训练时的环境依赖。

这种严格的追溯体系在实际场景中具有重要价值。当项目发生人员交接（如数据科学家离职或调岗），或是线上模型出现影响业务的异常，需要排查时，完备的可复现性保障能大幅降低问题定位成本，确保符合行业监管要求。

5）测试

可以预见，数据工程师、数据科学家、机器学习工程师等从业者遵循测试原则时遇到的阻力最小。然而，机器学习项目固有的动态特性，以及多样化的输入数据和产物形态，使得相关测试不仅更具挑战性，其重要性也愈发凸显。

机器学习项目的有效测试需要覆盖多个维度。本小节重点讨论两方面内容：数据与模型。

（1）数据相关测试。数据质量及其统计特性直接影响 ML 模型在训练和预测阶段的性能表现。以下是关键的测试类型：

- 空值检测、异常分布检验及特征相关性分析。
- 验证二分类或多分类任务中目标标签分布的预设假设。

- 特征生成代码的单元测试。

（2）模型相关测试如下：

- 基于离线数据验证模型性能，确保达到预期精度指标（如准确率）及运行指标（如推理延迟、模型体积）。
- 通过特征重要性分析，量化各特征对模型输出的影响程度。
- 采用少量实时生产数据，对比新模型（或版本）与基线模型 / 现役模型的性能差异，完成模型冒烟测试。
- 系统性评估模型在公平性、偏差控制及包容性方面的表现。

6）持续机器学习训练、评估与部署

随着企业不断拓展 AI/ML 项目，强烈建议遵循持续训练原则，即通过让 ML 模型持续适应动态变化的环境，从而长期获得机器学习项目的投资回报率。

机器学习从业者普遍面临一个现实挑战：由于多种因素，模型性能会随时间推移逐渐衰减。以电影推荐系统为例，用户行为变化会导致输入数据快速更新，这就要求通过持续的模型训练和评估来保持模型性能，甚至实现性能提升。

要高效、安全地实施这一原则，需要构建三大支撑体系：

- 基于业务指标建立的监控机制，在模型性能低于预设阈值时自动触发报警。
- 根据数据变化或性能衰减自动启动的机器学习训练与评估管道。
- 支持新模型版本安全部署的自动化上线流程。

虽然并非所有机器学习场景都需要持续训练机制，但对于动态数据环境下的应用（如推荐系统、需求预测等），该原则能显著提升模型投资回报率。

7）持续监控

“唯一不变的是变化本身。”

——赫拉克利特，古希腊哲学家

机器学习系统需要持续监控的特殊性在于，当模型性能跌破可接受阈值或者开始产生影响用户体验、商业价值及企业声誉的异常预测时，必须及时采取风险控制措施。

这一原则要求对所有机器学习组件（数据、模型、代码、数据处理管道、训练管道等）进行定期主动检测。

在众多监控维度中，最具挑战性的是模型漂移（model drift）现象。随着时间推移，模型预测会逐渐偏离预期轨迹，根源在于训练数据与预测目标之间的映射关系发生了变化。下文将通过具体案例详细解析。

模型漂移主要分为两类：

- 概念漂移：
  - 模型输入与输出之间的映射关系随时间发生改变。
  - 典型特征是输入特征的统计分布未变，但模型性能持续衰减。
  - 典型案例：疫情初期训练的商品需求预测模型，由于突发性卫生用品需求激增，预测准确率大幅下降。
- 数据漂移：
  - 模型输入（训练数据或特征）的统计属性随时间改变。
  - 典型特征是输入特征分布变化与模型性能衰减同步发生。
  - 典型案例：疫情前的通勤时间预测模型，因高峰时段车流量骤减而预测失效。

除模型性能外，还需监控以下运行指标：

- 服务延迟：在线服务场景的核心健康指标。
- 资源利用率：内存、CPU、GPU 等资源的异常占用可能预示系统隐患。
- 预测吞吐量。
- 错误率。

注意 DataOps、ModelOps 和 AIOps 的对比

如今存在多个以 Ops 结尾的术语，其概念在业界尚无统一定义。根据 Gartner 术语库[①]的定义：

DataOps 侧重优化组织内部数据流的协同管理与自动化。

ModelOps 专注于已部署模型的治理与全生命周期管理。

AIOps 融合大数据与机器学习技术，实现 IT 运维流程自动化，这些流程包括事件关联分析、异常检测以及因果关系判定。

① Gartner Glossary - www.gartner.com/en/glossary

1.3　MLOps 标准技术栈

在 MLOps 工程实践与原则之上，构建可扩展的自动化技术体系需要底层技术栈支撑。这套体系需要实现 ML 模型开发、部署、管理、监控的全流程自动化，并确保与业务系统的无缝集成。

本节阐述的 MLOps 标准技术栈，为组织实施机器学习项目提供了基础设施建设的参考框架。我们将从工程视角出发，以技术中立的立场解析各组件及其核心功能。

组织可根据实际需求选择开源或商业技术方案，并制订相应的实施策略。后续章节将深入探讨技术选型与落地方案。

1.3.1　MLOps 体系架构

图 1.8 的架构图改编自 AI 基础设施联盟发布的《2022 年 AI 基础设施生态》报告[①]。原报告提炼了 MLOps 核心能力框架，本架构在此基础上整合了近些年的新兴模块，其中灰色模块表示 MLOps 专属功能。

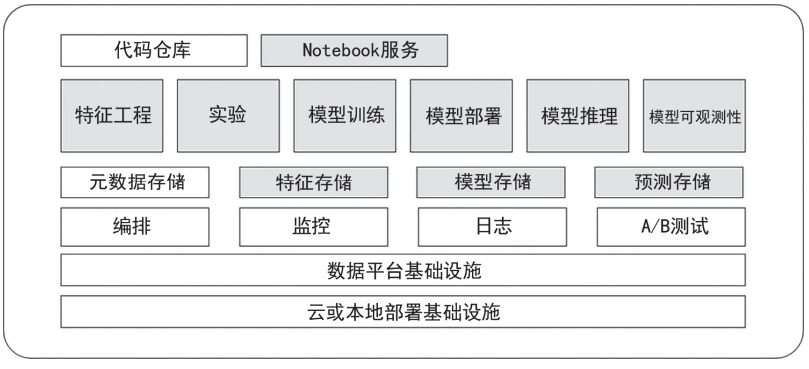

图 1.8　MLOps 体系架构（改编自 AI 基础设施联盟）

① AI Infrastructure Ecosystem of 2022 – https://ai-infrastructure.org/ai-infrastructureecosystem-report-of-2022/

通过观察可以发现，MLOps 技术体系具有显著的复杂性，且高度依赖企业基础架构。其中数据平台作为关键基础设施，直接影响机器学习项目的实施可行性。如果没有完善的数据存储、访问与处理能力，任何大规模机器学习项目都难以落地，实验性小项目除外。

该架构设计旨在支撑完整的机器学习开发生命周期，同时贯彻前文所述的 MLOps 核心原则。

1.3.2 核心组件解析

为深入解析 MLOps 架构中各组件所体现的技术特性，并阐明每个灰盒组件对机器学习生命周期管理及 MLOps 实施原则的支撑作用，本节将系统解析各组件的技术架构与功能定位。通过构建组件间的协同关系图谱，我们将完整呈现从模型开发到部署和运维的全流程技术支撑体系，其中既包含具体工程实践层面的工具链整合，也涵盖方法论层面的最佳实践指导原则。

1. 特征工程

在机器学习项目初期，特征工程占据重要地位。特征工程包含特征筛选、新特征构造、数值缩放与转换等关键步骤，直接影响算法效果。数据科学家往往在此环节投入大量精力。

提升效率的关键在于抽象底层复杂性，使数据科学家能专注特征逻辑设计。理想情况下，数据科学家只需定义特征生成规则，基础设施则负责可靠、高效地实现这些规则。

对于以下三种情况，基础设施要求可适当放宽：

- 机器学习应用处于探索验证阶段。
- 项目数量有限的小型实施方案。
- 特征数据规模较小。

该组件支撑自动化原则，是企业规模化实施机器学习项目的必备能力。

2. 特征仓库

作为集中化的特征管理中心，特征仓库提供特征注册、特征发现等功能，以便了解特

征来源、计算逻辑、质量评估与状态监控，从而实现跨团队的特征共享与复用。

完整特征仓库需同时支持离线分析与在线推理场景，其中在线服务因需满足低延迟要求而更具挑战性。

处于起步阶段的企业，可基于 S3 存储桶搭建简易特征仓库。

该组件对实现 AI 项目规模化运营至关重要。

该组件支撑的核心原则包括：持续版本控制、可复现性、机器学习训练、评估与部署标准化。

3. 交互式开发环境

在项目探索期，数据科学家需进行大量实验性工作，包括数据洞察分析、算法选型与参数调优。交互式开发环境（如 JupyterLab）因其灵活的网页界面，成为主流的实验工具。

集中化部署的交互式环境能提供数据访问、计算资源调度等核心功能，显著提升开发效率。虽然开源工具支持本地化部署，但企业级解决方案还能实现团队协作、版本控制、工具链集成等增值功能。

该组件的价值在企业扩展机器学习应用规模时尤为显著。

交互式环境支撑的原则包括：自动化、版本控制、实验跟踪与可复现性。通过为数据科学家提供集中式的工作平台，交互式环境有助于推广这些原则，并支持更高效的机器学习开发流程。

4. 模型训练平台

与特征工程组件类似，该平台的核心目标是降低模型训练复杂度。无论特征规模、模型架构或计算需求如何变化，数据科学家都能便捷高效地完成训练任务。

为实现这一目标，平台须具备以下能力：

- 提供高层抽象接口，用简洁代码定义训练流程。
- 灵活调配 CPU/GPU/TPU 等计算资源。
- 支持持续训练机制。
- 确保训练过程的可复现性与一致性。

以下小节将深入探讨上述各项能力的更多具体细节。

1）抽象接口设计

模型训练是一个交互过程，数据科学家通常具备充分的能力和知识，能够选择合适的特征、ML 模型算法以及一组超参数，从而最终针对当前的业务问题生成最优的 ML 模型。

数据科学家需要一种抽象方式，用最少的代码行数表述模型训练过程中需要完成的任务：

- 数据集划分与采样。
- 算法选择与参数配置。
- 模型性能评估与追踪。
- 快速获取计算资源。

这类抽象接口通常以 Python 库形式封装底层框架与企业基础设施（如日志监控系统）。

技术始终处于持续演进之中。通过引入抽象层，我们能够有效应对技术迭代带来的挑战：无论是迁移到依赖库的新版本，还是更换为具备更优功能的新库，甚至是利用新基础设施提供的服务，抽象机制都大大降低了这些技术升级过程的实施难度。这种设计范式既保证了系统架构的灵活性，也为技术栈的平滑过渡提供了可靠保障。

2）计算资源

当企业持续推进机器学习项目规模化时，其应用场景必然趋于复杂化，这往往需要借助海量特征进行模型训练，并采用复杂度不断提升的机器学习算法架构。

若能大规模获取必需的计算资源，既可实现大型复杂 ML 模型的训练，又能将原本耗时数天的训练周期压缩至数小时甚至分钟级，这将显著加速模型的迭代优化进程，最终缩短 ML 模型从开发到投产的周期。

无论是云端部署还是本地部署，昂贵计算资源的启动与管理本质上属于工程实施范畴。理想的解决方案是最大限度降低数据科学家在此环节的学习成本，将计算资源的启停管理、成本核算等核心事务交由专业化的模型训练基础设施统一管理。

3）持续训练

软件开发领域有句经典调侃，即“本地环境运行正常”。同理，基于个人计算机训练的

模型不应直接部署至生产环境，特别是涉及重要业务决策（如信贷审批）的场景。

规范的实践是，所有生产模型必须基于通过了代码评审的训练脚本，通过标准化流程完成训练。

该流程通常集成在持续交付管道中，与版本控制系统深度耦合，确保训练过程可追溯、可审计、可复现。

4）标准化与可复现

随着机器学习应用的不断深化，数据科学家团队规模扩大，人员流动带来的知识传承问题日益凸显。考虑到这些情况，保持训练过程的标准化与可复现性至关重要。

如果训练代码通过严格的代码评审，并以标准化形式存储在版本库中，这不仅能提升团队协作效率，也使后续的模型迭代、问题排查等工作事半功倍。

规范化的开发流程让科学家能快速复现历史模型，专注于新版本优化而非环境调试。

最终，标准化与可复现性将提升团队整体效能，避免重复劳动带来的资源损耗。

模型训练平台作为关键基础设施，对企业机器学习能力的规模化扩展具有战略意义。

5. 实验

实验是数据科学中极具科学性的组成部分，也是 ML 模型训练不可或缺的环节。其核心在于通过探索找到模型架构、特征和调参的最佳组合，从而产出满足业务需求的高性能模型。这个过程需要反复迭代并完整记录所有实验要素，类似于大学化学实验的严谨流程。

该组件的核心价值不在于实验操作本身，而在于为实验相关活动提供全流程支持与追踪能力。

其关键功能如下：

- 提供便捷的实验信息上传通道。
- 采用持久化可靠存储方案保存实验数据。
- 通过 API 或可视化界面实现实验信息的便捷访问。
- 构建直观的对比分析工具，帮助数据科学家快速定位影响模型效果的关键因素。

该组件主要遵循的核心原则是可复现性。

6. 模型仓库

完成训练的 ML 模型需纳入中央仓库统一管理。这个核心存储库将支撑模型的全生命周期管理，涵盖测试环境验证、预发布调试直至生产环境部署等环节。作为企业所有 ML 模型的唯一可信数据源，中央仓库不仅实现了版本控制和协作共享，更使组织能够系统化追踪模型表现，科学决策模型的部署策略。

当企业的模型资产规模突破临界点（通常是 20）后，集中式仓库在解决版本混乱、管理低效等问题上的价值将凸显。

其核心优势体现在：

- 全生命周期管理体系。
- 完备的溯源与复现能力。
- 模型治理：使组织能够制订在模型整个生命周期中对其进行管理的政策和流程，促进合规性，并降低与模型性能或行为相关的风险。
- 安全性：为模型提供安全的存储和访问控制，保护模型免受未经授权的访问或篡改。

如图 1.9 所示，模型仓库在机器学习开发流程与生产环境之间架起关键桥梁，确保模型资产的安全流转。

图 1.9　模型仓库

1）全生命周期管理

正式完成训练的候选部署模型，必须纳入仓库的标准化管理体系中。这个中央枢纽不仅提供模型资产的可视化检索，更赋予企业审计追踪、权限管控等核心管理能力。

不同企业的管理复杂度存在显著差异。部分企业可能只需简单的生命周期管理，而强监管行业的企业往往需要构建包含数十个审批节点的复杂流程。通过定制符合企业特性的生命周期管理方案，模型仓库可成为确保合规运营的关键基础设施。

2）溯源与复现

除了存储模型文件和相关元数据，模型仓库还能完整记录部署日志，包括操作人员、时间戳、变更说明等关键信息。对于需要严格审批流程的金融、医疗等行业，这种溯源能力在满足监管要求方面具有不可替代的价值。

在模型入库环节，系统会自动捕获训练数据集指纹、超参数配置等关键信息。当特定场景需要模型复现时，所有关联数据均可即时调取，杜绝“黑箱”操作。

该组件严格遵循版本控制与可复现性原则，为 ML 模型的工业化应用奠定了基础。

7. 模型部署

在传统软件开发领域，每日数次的自动化部署已成为行业常态。

机器学习领域同样遵循这一趋势，模型部署工作流致力于实现从仓库到生产环境的无缝衔接，支持快速迭代与敏捷回滚。

标准部署流程包含两个关键阶段：首先从仓库提取模型文件并完成服务化封装，同时更新模型状态标记；随后将封装后的模型移交至服务系统，触发服务更新，使新模型开始处理预测请求。

图 1.10 清晰展示了该流程中各组件间的协作关系。

图 1.10　模型部署

由于不同行业的技术架构和合规要求存在差异，具体实施方案可能千差万别。但提升部署效率的核心要义始终是明确的，即构建高度自动化、可配置的部署管道。

该组件深度践行自动化与持续部署原则，有力支撑企业的快速迭代需求。

8. 模型推理

模型推理是机器学习价值兑现的核心环节，承担着将训练模型转化为实际预测能力的重任。根据应用场景差异，预测服务可分为离线批量预测与线上实时预测两种模式。

注意 批量预测和线上实时预测

当讨论模型推理时，必须明确其运作模式。

批量预测通过定期或按需的批处理模式生成，可一次性产生数以万计甚至百万量级的预测结果。这种方式尤其适用于批量数据预测的场景，例如基于历史数据建模或需要聚合时间段内输入数据的场景。以电商行业为例，企业可能根据用户的过往购买记录批量生成商品推荐预测，这些预测结果将被用于制订营销策略。生成的预测数据通常存储于 SQL 表或 S3 存储桶等系统中，后续可能迁移至内存数据库或分布式存储引擎等高性能在线系统，以支撑实时服务需求。Netflix 电影推荐系统等经典案例均采用此类模式。由于批预测的延时特性，它也被称作异步预测。

线上实时预测则直接响应实时请求，单次请求通常会产生一至数千个预测结果。其核心特征在于即时性——预测结果会同步返回请求方，因此这类预测也被称为同步预测。

模型推理组件专注于实时在线预测场景，通过封装模型为标准化服务接口（支持 HTTP/HTTPS、REST/gRPC 等协议），为业务系统提供低延迟的预测能力。

该组件本质上架起了 ML 模型与微服务架构的桥梁，其核心能力指标包括：

- 低延迟、可扩展、高可靠。
- 跨框架兼容性支持。
- 异构计算资源适配，支持 CPU、GPU、TPU 和其他 AI 加速设备。
- 影子部署等高级功能。
- 集成 A/B 测试。
- 特征实时获取能力。
- 完整的预测日志体系（特征、模型标识符/版本、预测结果等）。

当企业开始将机器学习集成到处理在线客户请求的在线产品（如推荐、搜索与排名、欺诈检测、语言翻译、自动完成等功能）中时，模型推理组件是 MLOps 体系中最关键的组件之一。

该组件所秉持的原则是自动化与持续部署。

9. 预测存储

相较于其他成熟组件，预测储存是近年兴起但尚未获得足够重视的基础设施。这个中央存储库专门收录实时预测产生的细日志，包括输入特征、模型版本、预测结果及系统运行指标等关键信息。

对数据科学家而言，这些数据具备多重价值：

- 诊断生产环境中的模型异常。
- 评估影子模型的线上表现。
- 积累增量训练所需数据。

数据科学家需要能够以简便、高效且可扩展的方式访问、分析和处理这些预测日志。

该组件所支持的原则是可复现性和持续监控。

注意

Josh Tobin 是一家名为 Gantry 的初创公司的联合创始人，他倡导设立一个名为“评估存储库”的组件。在他题为《机器学习基础设施堆栈中缺失的一环》[①]的演讲中，“评估存储库”被定义为“一个集中存储和查询在线及离线基准事实以及近似模型质量指标的地方”。

除了具备上述预测存储的功能外，评估存储库还会存储训练阶段的预测结果以及评估阶段的 ML 模型指标，让机器学习从业者能够更自信地部署 ML 模型，并更快发现生产中的漏洞。

10. 机器学习可观测性

机器学习可观测性在 ML 模型部署至生产环境并整合至企业在线产品后，承担着关键的风险防控职能。

其核心价值不仅在于监测模型对预设业务指标的负面影响，更重要的是为团队提供持续优化的能力。通过分析模型性能衰减、快速定位生产环境问题的根本原因，避免模型上

① Josh Tobin, “A Missing Link in the ML Infrastructure Stack”, http://josh-tobin.com/assets/pdf/missing_ link_in_mlops_infra_031121.pdf

线后陷入“黑箱”运行状态。

作为 MLOps 的核心组件，该体系包含监控、可观测性与可解释性三大支柱，共同保障机器学习系统的可靠运行与高效运维。

1）监控

监控系统通过持续追踪模型准确率、数据漂移、预测失败率等关键指标，构建“故障定位－时间标定”的双重预警机制。

2）可观测性

可观测性旨在为ML模型的行为和性能提供更多背景信息或洞察，使团队在问题出现时能够快速识别并调试。例如，如果模型性能开始下降，机器学习团队应该能够轻松且快速地确定根本原因，比如生产特征数据的变化、导致特征数据陈旧的特征数据管道故障、近期模型部署引入的漏洞，或者底层基础设施或环境的问题。

3）可解释性

可解释性旨在帮助人们理解模型为何做出特定预测，或哪些因素对这些预测有重大影响。这些信息对于业务团队或非数据科学团队而言非常有用，能让他们对 ML 模型的性能建立信心，直观了解这些模型的运行方式，还有助于在 ML 模型的验证阶段或在投入使用后，发现潜在问题。

为满足上述三个领域的需求，该组件需要提供的关键要素有：

- 设置对各种与机器学习相关指标（如模型漂移和特征漂移、模型性能指标）的监测，并设置阈值，以便在触发时提醒值班的机器学习工程师或数据科学家进行调查。
- 可视化各种模型性能指标，了解发生了什么情况，并轻松分析这些指标，以便数据科学家能够快速确定问题所在。
- 查看哪些特征在影响预测结果方面发挥重要作用。
- 分析每个 ML 模型的性能指标。

综上所述，借助机器学习的可观测性，数据科学家能够轻松、快速地洞察模型性能问题，理解ML模型如何做出预测，并能够回应非数据科学团队针对 ML 模型行为提出的疑问。

注意

该组件高度依赖预测存储的可用性，以便对预测日志数据进行访问和计算各种汇总信息。

该组件所支持的原则是自动化原则与持续监控原则。

11. MLOps 支柱体系

图 1.8 展示了众多组件，其详细信息在上一节已有描述。我们不妨放宽视角，从逻辑上将这些组件归为更宽泛的类别（笔者称之为“支柱”），这样有助于对 MLOps 体系有一个整体概览。这些支柱抓住了 MLOps 体系的关键方面，便于向其他团队分享和阐释，也有助于在企业将 MLOps 投入生产的过程中，跟踪其进展和成熟度。每个支柱都有足够的范畴和影响力，且边界划分合理。因此，很容易理解为何每个支柱都需要一个团队来开发和支持。

如图 1.11 所示，笔者倡导的四个支柱分别是特征工程、模型训练和管理、模型推理以及机器学习可观测性。

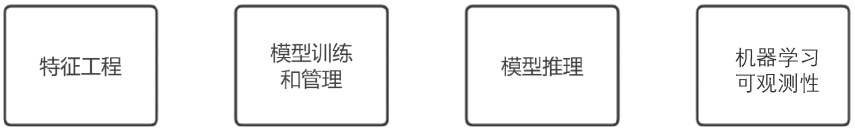

图 1.11　MLOps 四大支柱

1）特征工程

这一支柱负责提供所有必要的基础设施，以支持与特征生成、特征管理和特征存储相关的活动及流程。

2）模型训练和管理

这一支柱负责提供所有必要的基础设施，以支持与模型训练、模型存储和模型生命周期管理相关的活动与流程。

3）模型推理

这一支柱负责提供所有必要的基础设施，以支持与模型推理和预测存储相关的活动及流程。

4）机器学习可观测性

这一支柱负责提供所有必要的基础设施，以支持与机器学习监控、可观测性及可解释性相关的活动和流程。

> **注意** 机器学习治理
>
> 机器学习治理是一套涵盖策略、流程与管控措施的综合体系，用于规范 ML 模型的全生命周期管理，重点解决伦理合规审查、风险管控等核心问题。MLOps 的核心关注点在于构建加速机器学习研发的基础设施，因此本章暂不展开讨论与机器学习治理相关的议题。

1.4 小结

本章系统阐释了 MLOps 诞生的技术动因，以及机器学习项目产业化落地面临的独特挑战。通过剖析机器学习项目区别于传统软件工程的本质特征，尤其是其强实验性、持续迭代性等属性，为理解 MLOps 奠定了基础。

进一步地，本章将 MLOps 定位为一门融合三大工程领域精髓的交叉学科：汲取 DevOps 的协同交付理念、数据工程的基础架构能力，以及机器学习的技术方法论。

为了深入解析 MLOps 体系，本章采用分层解构方法论，从顶层范式设计、中观工程原则到底层实施准则，逐层揭示其技术内涵。需要特别强调的是，成功推行 MLOps 的首要前提是思维范式的转变，即将机器学习要素（如模型、数据集）置于项目研发的核心地位。通过标准化践行 MLOps 原则，企业可实现机器学习项目的规模化部署，最大化技术投入的商业回报。

在阐释工程原则后，本章从系统工程视角剖析了 MLOps 技术栈的组成要素。该标准化技术蓝图旨在为企业提供可扩展的自动化框架，覆盖 ML 模型开发、部署、运维与监控的全流程。

最后，本章提出基于功能支柱的技术组件分类框架，帮助从业者清晰把握 MLOps 各领域的技术边界与协作关系。

当企业完成 MLOps 知识体系构建后，如何将其有效整合至现有技术生态？应采取哪些实施策略以提升成功率？可能遭遇哪些典型挑战？第 2 章将针对这些关键问题展开深度探讨，为正在实施或计划部署 MLOps 的企业提供战略级实施指南。