# **Effective Machine Learning Teams** Best Practices for ML Practitioners

David Tan, Ada Leung, and David Colls

Beijing • Boston • Farnham • Sebastopol • Tokyo



#### **Effective Machine Learning Teams**

by David Tan, Ada Leung, and David Colls

Copyright © 2024 David Tan Rui Guan, Ada Leung Wing Man, and David Colls. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*https://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Nicole Butterfield Development Editor: Melissa Potter Production Editor: Gregory Hyman Copyeditor: Nicole Taché Proofreader: M & R Consultants Corporation Indexer: Judith McConville Interior Designer: David Futato Cover Designer: Karen Montgomery Illustrator: Kate Dullea

February 2024: First Edition

#### **Revision History for the First Edition**

2024-02-29: First Release

See https://oreilly.com/catalog/errata.csp?isbn=9781098144630 for release details.

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *Effective Machine Learning Teams*, the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

The views expressed in this work are those of the authors, and do not represent the publisher's views. While the publisher and the authors have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the authors disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

978-1-098-14463-0 [LSI]

# **Table of Contents**

Preface		ix
1.	Challenges and Better Paths in Delivering ML Solutions	1
	ML: Promises and Disappointments	2
	Continued Optimism in ML	2
	Why ML Projects Fail	2
	Is There a Better Way? How Systems Thinking and Lean Can Help	10
	You Can't "MLOps" Your Problems Away	10
	See the Whole: A Systems Thinking Lens for Effective ML Delivery	11
	The Five Disciplines Required for Effective ML Delivery	13
	Conclusion	32

# Part I. Product and Delivery

2.	Product and Delivery Practices for ML Teams	37
	ML Product Discovery	40
	Discovering Product Opportunities	43
	Canvases to Define Product Opportunities	44
	Techniques for Rapidly Designing, Delivering, and Testing Solutions	51
	Inception: Setting Teams Up for Success	56
	Inception: What Is It and How Do We Do It?	56
	How to Plan and Run an Inception	58
	User Stories: Building Blocks of an MVP	61
	Product Delivery	69
	Cadence of Delivery Activities	70
	Measuring Product and Delivery	74
	Conclusion	79

# Part II. Engineering

3.	Effective Dependency Management: Principles and Tools.	83
	What If Our Code Worked Everywhere, Every Time?	84
	A Better Way: Check Out and Go	85
	Principles for Effective Dependency Management	87
	Tools for Dependency Management	90
	A Crash Course on Docker and batect	96
	What Are Containers?	97
	Reduce the Number of Moving Parts in Docker With batect	101
	Conclusion	107
4.	Effective Dependency Management in Practice.	109
	In Context: ML Development Workflow	109
	Identifying What to Containerize	110
	Hands-on Exercise: Reproducible Development Environments, Aided by	
	Containers	113
	Secure Dependency Management	124
	Remove Unnecessary Dependencies	125
	Automate Checks for Security Vulnerabilities	127
	Conclusion	130
5.	Automated Testing: Move Fast Without Breaking Things	133
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably	<b>133</b> 135
5.		
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably	135
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation	135 136
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It?	135 136 139
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems	135 136 139 142
5.	<ul> <li>Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation</li> <li>If Automated Testing Is So Important, Why Aren't We Doing It?</li> <li>Building Blocks for a Comprehensive Test Strategy for ML Systems</li> <li>The What: Identifying Components For Testing</li> <li>Characteristics of a Good Test and Pitfalls to Avoid</li> <li>The How: Structure of a Test</li> </ul>	135 136 139 142 143 147 152
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests	135 136 139 142 143 147 152 154
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests	135 136 139 142 143 147 152 154 154
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests	135 136 139 142 143 147 152 154 154 154
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests	135 136 139 142 143 147 152 154 154 157 159
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests Post-deployment Tests	135 136 139 142 143 147 152 154 154 157 159 162
5.	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests	135 136 139 142 143 147 152 154 154 157 159
	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests Post-deployment Tests Conclusion Automated Testing: ML Model Tests.	135 136 139 142 143 147 152 154 154 154 157 159 162 165 <b>167</b>
	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests Post-deployment Tests Conclusion Automated Testing: ML Model Tests. Model Tests	135 136 139 142 143 147 152 154 154 157 159 162 165
	Automated Tests: The Foundation for Iterating Quickly and Reliably Starting with Why: Benefits of Test Automation If Automated Testing Is So Important, Why Aren't We Doing It? Building Blocks for a Comprehensive Test Strategy for ML Systems The What: Identifying Components For Testing Characteristics of a Good Test and Pitfalls to Avoid The How: Structure of a Test Software Tests Unit Tests Training Smoke Tests API Tests Post-deployment Tests Conclusion Automated Testing: ML Model Tests.	135 136 139 142 143 147 152 154 154 154 157 159 162 165 <b>167</b>

iv | Table of Contents

	Fitness Functions for ML Models	170
	Model Metrics Tests (Global and Stratified)	172
	Behavioral Tests	177
	Testing Large Language Models: Why and How	179
	Essential Complementary Practices for Model Tests	184
	Error Analysis and Visualization	185
	Learn from Production by Closing the Data Collection Loop	187
	Open-closed Test Design	190
	Exploratory Testing	191
	Means to Improve the Model	192
	Designing to Minimize the Cost of Failures	193
	Monitoring in Production	194
	Bringing It All Together	195
	Next Steps: Applying What You've Learned	198
	Make Incremental Improvements	198
	Demonstrate Value	199
	Conclusion	200
7.	Supercharging Your Code Editor with Simple Techniques	201
	The Benefits (and Surprising Simplicity) of Knowing Our IDE	203
	Why Should We Care About IDEs?	203
	If IDEs Are So Important, Why Haven't I Learned About Them Yet?	205
	The Plan: Getting Productive in Two Stages	206
	Stage 1: Configuring Your IDE	207
	Stage 2: The Star of the Show—Keyboard Shortcuts	214
	You Did It!	226
	Conclusion	230
_		
8.	Refactoring and Technical Debt Management.	231
	Technical Debt: The Sand In Our Gears	232
	Getting to a Healthy Level of Debt Through Tests, Design, and Refactoring	
	Refactoring 101	237
	How to Refactor a Notebook (or a Problematic Codebase)	238
	The Map: Planning Your Journey	239
	The Journey: Hitting the Road	244
	Looking Back at What We've Achieved	255
	Technical Debt Management in the Real World	259
	Technical Debt Management Techniques	260
	A Positive Lens on Debt: Systems Health Ratings	262
	Conclusion: Make Good Easy	266

Table of Contents | v

9.	MLOps and Continuous Delivery for ML (CD4ML).	267
	MLOps: Strengths and Missing Puzzle Pieces	269
	MLOps 101	270
	Smells: Hints That We Missed Something	276
	Continuous Delivery for Machine Learning	280
	Benefits of CD4ML	280
	A Crash Course on Continuous Delivery Principles	281
	Building Blocks of CD4ML: Creating a Production-Ready ML System	284
	How CD4ML Supports ML Governance and Responsible AI	294
	Conclusion	297

### Part III. Teams

10.	Building Blocks of Effective ML Teams	301
	Common Challenges Faced by ML Teams	303
	Effective Team Internals	307
	Trust As the Foundational Building Block	308
	Communication	315
	Diverse Membership	318
	Purposeful, Shared Progress	322
	Internal Tactics to Build Effective Teams	324
	Improving Flow with Engineering Effectiveness	326
	Feedback Loops	327
	Cognitive Load	328
	Flow State	329
	Conclusion	332
11.	Effective ML Organizations.	335
11.	Effective ML Organizations Common Challenges Faced by ML Organizations	<b>335</b> 338
11.	-	
11.	Common Challenges Faced by ML Organizations	338
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams	338 339
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management	338 339 340
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model	338 339 340 341
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model Team Topologies for ML Teams	338 339 340 341 346
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model Team Topologies for ML Teams Organizational Tactics to Build Effective Teams Intentional Leadership Create Structures and Systems for Effective Teams	338 339 340 341 346 358
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model Team Topologies for ML Teams Organizational Tactics to Build Effective Teams Intentional Leadership Create Structures and Systems for Effective Teams Engage Stakeholders and Coordinate Organizational Resources	338 339 340 341 346 358 360
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model Team Topologies for ML Teams Organizational Tactics to Build Effective Teams Intentional Leadership Create Structures and Systems for Effective Teams Engage Stakeholders and Coordinate Organizational Resources Cultivate Psychological Safety	338 339 340 341 346 358 360 361
11.	Common Challenges Faced by ML Organizations Effective Organizations as Teams of Teams The Role of Value-Driven Portfolio Management Team Topologies Model Team Topologies for ML Teams Organizational Tactics to Build Effective Teams Intentional Leadership Create Structures and Systems for Effective Teams Engage Stakeholders and Coordinate Organizational Resources	338 339 340 341 346 358 360 361 362

vi | Table of Contents

Build the Culture We Wish We Had	363
Encourage Teams to Play at Work	364
Conclusion	366
Epilogue: Dana's Journey	367
Index	369

# Preface

It was 9:25 p.m. and the soft glow of Dana's computer screen glared into her bleary eyes as she logged on to continue fixing an error, red pipelines and countless open tabs filling her screen. She had eaten dinner and finished her everyday chores, but her mind wasn't really there—it was in a few places, in fact.

It had been an intense day, scattered between long training runs and back-and-forth messages with the support team on customer queries about why the model denied their loan applications. She was in and out of the depths of debugging why the model's performance just wouldn't improve, despite various tweaks to the data and model architecture. The occasional stack traces didn't help, either.

She was tired, and the tangled heap of uncommitted code changes sitting on her local machine added to the latent cognitive load that was bubbling over in her head. But she had to keep going—her team had already missed the initial release date by four months and the executives' impatience was showing. What made things worse was a fear that her job might be on the line. One in ten employees in her company—several of whom she knew—were laid off in the latest round of cost-cutting measures.

Everyone on her team was well-meaning and capable, but they were getting bogged down every day in a quagmire of tedious testing and anxiety-laden production deployments, stepping through illegible and brittle code. After a few months of toil, they were all worn down. She and her team were doing their level best, but sometimes it felt like they were building a house without a foundation—things kept falling apart. She didn't know it yet, but they weren't set up for success.

Many individuals begin their machine learning (ML) journey with great momentum and gain confidence quickly, thanks to the growing ecosystem of tools, techniques, tutorials, and community of ML practitioners. However, when we graduate beyond the controlled environment of tutorial notebooks and Kaggle competitions into the space of real-world problems, messy data, interconnected systems, and people with varied objectives, many of us inevitably struggle to realize the potential of ML in practice.

When we peel back the glamorous claims of data science being the sexiest job, we often see ML practitioners mired in burdensome manual work, complex and brittle

codebases, and frustration from Sisyphean ML experiments that never see the light of day in production.

In 2019, it was reported that 87% of data science projects never make it to production (*https://oreil.ly/xy9Xi*). According to Algorithmia's 2021 Enterprise AI/ML Trends (*https://oreil.ly/HP6Qh*), even among companies that have successfully deployed ML models in production, 64% of survey respondents say it takes more than a month to deploy a new model, an increase from 56% in 2020. Algorithmia also found that 38% of organizations surveyed are spending more than 50% of their data scientists' time on model deployment.

These barriers impede—or, in some cases, even prevent—ML practitioners from applying their expertise in ML to deliver on the value and promise of AI for customers and businesses. But the good news is it doesn't have to be this way. In the past few years, we have had the privilege to work on various data and ML projects, and to collaborate with ML practitioners from multiple industries. While there are barriers and pains, as we have outlined above, there are also better paths, practices, and systems of work that allow ML practitioners to reliably deliver ML-enabled products into the hands of customers.

That's what this book is all about. We'll draw from our experience to distill a set of enduring principles and practices that consistently help us to effectively deliver ML solutions in the real world. These practices work because they're based on taking a holistic approach to building ML systems. They go beyond just ML to create essential feedback loops in various subsystems (e.g., product, engineering, data, delivery processes) and enable teams to fail quickly and safely, experiment rapidly, and deliver reliably.

### Who This Book Is For

Whether you think you can, or you think you can't-you're right.

—Henry Ford

Whether you're a ML practitioner in academia, an enterprise, a start-up, a scale-up, or consulting, the principles and practices in this book can help you and your team become more effective in delivering ML solutions. In line with the cross-functional nature of ML delivery techniques that we detail in this book, we address the concerns and aspirations of multiple roles in teams doing ML:

#### Data scientists and ML engineers

The job scope of a data scientist has evolved over the past few years. Instead of purely focusing on modeling techniques and data analysis, we're seeing expectations (implicit or explicit) that one needs to possess the capabilities of a full-stack data scientist (*https://oreil.ly/jV7EP*): data wrangling, ML engineering, MLOps, and business case formulation, among others. This book elaborates on

the capabilities necessary for data scientists and ML engineers to design and deliver ML solutions in the real world.

In the past, we've presented the principles, practices, and hands-on exercises in this book to data scientists, ML engineers, PhD students, software engineers, quality analysts, and product managers, and we've consistently received positive feedback. The ML practitioners we've worked with in the industry have said that they benefited from improvement in feedback cycles, flow, and reliability that comes from practices such as automated testing and refactoring. Our takeaway is that *there is a desire* from the ML community to learn these skills and practices, and this is our attempt to scale the sharing of this knowledge.

Software engineers, infrastructure and platform engineers, architects

When we run workshops on the topics we cover in this book, we often come across software engineers, infrastructure and platform engineers, and architects working in the ML space. While capabilities from the software world (e.g., infrastructure-as-code, deployment automation, automated testing) are necessary in designing and delivering ML solutions in the real world, they are also insufficient. To build reliable ML solutions, we need to widen the software lens and look at other principles and practices—such as ML model tests, dual-track delivery, continuous discovery, and ML governance—to handle challenges that are unique to ML.

#### Product managers, delivery managers, engineering managers

We set ourselves up for failure if we think that we need only data scientists and ML engineers to build an ML product. In contrast, our experience tells us that teams are most effective when they are cross-functional and equipped with the necessary ML, data, engineering, product, and delivery capabilities.

In this book, we elaborate on how you can apply Lean delivery practices and systems thinking to create structures that help teams to focus on the voice of the customer, shorten feedback loops, experiment rapidly and reliably, and iterate toward building the right thing. As W. Edwards Deming (*https://oreil.ly/eUxHc*) once said, "A bad system will beat a good person every time." So, we share principles and practices that will help teams create structures that optimize information flow, reduce waste (e.g., handoffs, dependencies), and improve value.

If we've done our job right, this book will invite you to look closely at how things have "always been done" in ML and in your teams, to reflect on how well they are working for you, and to consider better alternatives. Read this book with an open mind, and—for the engineering-focused chapters—with an open code editor. As Peter M. Senge (*https://oreil.ly/9HEwI*) said in his book *The Fifth Discipline* (Doubleday), "Taking in information is only distantly related to real learning. It would be nonsensical to say, 'I just read a great book about bicycle riding—I've now learned that." We encourage

you to try out the practices in your teams, and we hope you'll experience firsthand the value that they bring in real-world projects.

Approach this book with a continuous improvement mindset, not a perfectionist mindset. There is no perfect project where everything works perfectly without challenges. There will always be complexity and challenges (and we know a healthy amount of challenge is essential for growth), but the practices in this book will help you minimize *accidental* complexity so that you can focus on the *essential* complexity of your ML solutions and on delivering value responsibly.

### How This Book Is Organized

Chapter 1, "Challenges and Better Paths in Delivering ML Solutions", is a distillation of the entire book. We explore high-level and low-level reasons for why and how ML projects fail. We then lay out a more reliable path for delivering value in ML solutions by adopting Lean delivery practices across five key disciplines: product, delivery, machine learning, engineering, and data.

In the remaining chapters, we describe practices of effective ML teams and ML practitioners. In Part I, "Product and Delivery", we elaborate on practices in other subsystems that are necessary for delivering ML solutions, such as product thinking and Lean delivery. In Part I, "Product and Delivery", we cover practices that help ML practitioners when implementing and delivering solutions (e.g., automated testing, refactoring, using the code editor effectively, continuous delivery, and MLOps). In Part I, "Product and Delivery", we explore the dynamics that impact the effectiveness of ML teams, such as trust, shared progress, diversity, and also engineering effectiveness techniques that help you build high-performing teams. We also address common challenges that organizations face when scaling ML practices beyond one or two teams, and share techniques on team topologies, interaction modes, and leadership to help teams overcome these scaling challenges.

### Part I: Product and Delivery

Chapter 2, "Product and Delivery Practices for ML Teams"

We discuss product discovery techniques that help us identify opportunities, test market and technology hypotheses rapidly, and converge on viable solutions. By starting with the most valuable problems and feasible solutions, we set ourselves up for success during delivery. We also go through delivery practices that help us shape, size, and sequence work to create a steady stream of value. We address the unique challenges resulting from the experimental and high-uncertainty nature of certain ML problems, and discuss techniques such as the dual-track delivery model that help us learn more quickly in shorter cycles. Finally, we cover techniques for measuring critical aspects of ML projects and share techniques for identifying and managing project risks.

### Part II: Engineering

#### Chapters 3 and 4: Effective dependency management

Here, we describe principles and practices—along with a hands-on example that you can code along with—for creating consistent, reproducible, secure, and production-like runtime environments for running your code. When we hit the ground running and start delivering solutions, you'll see how the practices in this chapter will enable you and your teammates to "check out and go" and create consistent environments effortlessly, instead of getting trapped in dependency hell.

#### Chapters 5 and 6: Automated testing for ML systems

These chapters provide you with a rubric for testing components of your ML solution—be they software tests, model tests, or data tests. We demonstrate how automated tests help us shorten our feedback cycles and reduce the tedious effort of manual testing, or worse, fixing production defects that slipped through the cracks of manual testing. We describe the limits of the software testing paradigm on ML models, and how ML fitness functions and behavioral tests can help us scale the automated testing of ML models. We also cover techniques for comprehensively testing large language models (LLMs) and LLM applications.

#### Chapter 7, "Supercharging Your Code Editor with Simple Techniques"

We'll show you how to configure your code editor (PyCharm or VS Code) to help you code more effectively. After we've configured our IDE in a few steps, we'll go through a series of keyboard shortcuts that can help you to automate refactoring, automatically detect and fix issues, and navigate your codebase without getting lost in the weeds, among other things.

#### Chapter 8, "Refactoring and Technical Debt Management"

In this chapter, we draw from the wisdom of software design to help us design readable, testable, maintainable, and evolvable code. In the spirit of "learning by doing," you'll see how we can take a problematic, messy, and brittle notebook and apply refactoring techniques to iteratively improve our codebase to a modular, tested, and readable state. You'll also learn techniques that can help you and your team make technical debt visible and take actions to keep it at a healthy level.

#### Chapter 9, "MLOps and Continuous Delivery for ML (CD4ML)"

We'll articulate an expansive view of what MLOps and CI/CD (continuous integration and continuous delivery) really entails. Spoiler alert: It's more than automating model deployments and defining CI pipelines. We lay out a blueprint for the unique shape of CI/CD for ML projects and walk through how you can set up each component in this blueprint to create reliable ML solutions and free up your teammates from repetitive and undifferentiated labor so that they can focus on other higher-value problems. We'll also look at how CD4ML serves as a

risk-control mechanism to help teams uphold standards for ML governance and Responsible AI.

### Part III: Teams

Chapter 10, "Building Blocks of Effective ML Teams"

In this chapter, we go beyond the mechanics to understand the interpersonal factors that enable good practices in effective teams. We'll describe principles and practices that help create a safe, human-centric, and growth-oriented team. We'll examine topics like trust, communication, shared goals, purposeful progress, and diversity in teams. We'll share some antipatterns to watch for and some tactics that you can use to nurture a culture of collaboration, effective delivery, and learning.

Chapter 11, "Effective ML Organizations"

This chapter introduces various shapes for ML teams and addresses the common challenges that organizations face when scaling their ML practice to multiple teams. We draw from and adapt strategies discussed in *Team Topologies* (IT Revolution Press) and outline unique structures, principles, and practices that help teams find a balance between flow of work and concentrated expertise, collaboration, and autonomy. We evaluate the benefits and limits of these structures and offer guidance for their evolution to meet the organization's needs. We conclude by discussing the importance of intentional leadership in shaping agile, responsive ML organizations.

# **Additional Thoughts**

We'd like to touch on four things before we wrap up the Preface.

First, we want to acknowledge that ML is more than just supervised learning. We can also solve data-intensive (and even data-poor) problems using other optimization techniques (e.g., reinforcement learning [https://oreil.ly/7PjY6], operations research [https://oreil.ly/ZezrC], simulation [https://oreil.ly/UVhfB]). In addition, ML is not a silver bullet and some problems can be solved without ML. Even though we've chosen a supervised learning problem (loan default prediction) as an anchoring example in the code samples throughout the book, the principles and practices are useful beyond supervised learning. For example, the chapters on automated testing, dependency management, and code editor productivity are useful even in reinforcement learning. The product and delivery practices outlined in Chapter 2 are useful for exploratory and delivery phases of any product or problem space.

Second, as Generative AI and LLMs entered the public consciousness and product roadmaps of many organizations, we and our colleagues have had the opportunity to work with organizations to ideate, shape, and deliver products that leverage Generative AI. While LLMs have led to a paradigm shift in how we steer or constrain models toward their desired functionality, the fundamentals of Lean product delivery and engineering haven't changed. In fact, the fundamental tools and techniques in this book have helped us to test assumptions early, iterate quickly, and deliver reliably—thereby maintaining agility and reliability even when dealing with the complexities inherent in Generative AI and LLMs.

Third, on the role of culture: ML effectiveness and the practices in this book are not—and cannot be—a solo effort. That's why we've titled the book Effective Machine Learning *Teams*. You can't be the only person writing tests, for instance. In organizations that we've worked with, individuals become most effective when there is a cultural alignment (within the team, department, and even organization) on these Lean and Agile practices. This doesn't mean that you need to boil the ocean with the entire organization; it's just not enough to go it alone. As Steve Jobs once said, "Great things in business are never done by one person. They're done by a team of people."

Finally, this book is not about productivity (how to ship as many features, stories, or code as possible), nor is it about efficiency (how to ship features, stories, or code at the fastest possible rate). Rather, it's about effectiveness—how to build the right product rapidly, reliably, and responsibly. This book is about finding balance through movement and moving in effective ways.

The principles and practices in this book have consistently helped us to successfully deliver ML solutions, and we are confident that they will do the same for you.

# **Conventions Used in This Book**

The following typographical conventions are used in this book:

Italic

Indicates new terms, URLs, email addresses, filenames, and file extensions.

#### Constant width

Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

#### Constant width bold

Used to call attention to snippets of interest in code blocks.



This element signifies a general note.



This element indicates a warning or caution.

# **Using Code Examples**

Supplemental material (code examples, exercises, etc.) is available for download at:

- https://github.com/davified/loan-default-prediction
- https://github.com/davified/ide-productivity
- https://github.com/davified/refactoring-exercise

If you have a technical question or a problem using the code examples, please send email to *support@oreilly.com*.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but generally do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Effective Machine Learn-ing Teams* by David Tan, Ada Leung, and David Colls (O'Reilly). Copyright 2024 David Tan Rui Guan, Ada Leung Wing Man, and David Colls, 978-1-098-14463-0."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

# O'Reilly Online Learning



For more than 40 years, O'Reilly Media has provided technology and business training, knowledge, and insight to help companies succeed.

Our unique network of experts and innovators share their knowledge and expertise through books, articles, and our online learning platform. O'Reilly's online learning

platform gives you on-demand access to live training courses, in-depth learning paths, interactive coding environments, and a vast collection of text and video from O'Reilly and 200+ other publishers. For more information, visit *https://oreilly.com*.

# How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc. 1005 Gravenstein Highway North Sebastopol, CA 95472 800-889-8969 (in the United States or Canada) 707-827-7019 (international or local) 707-829-0104 (fax) support@oreilly.com https://www.oreilly.com/about/contact.html

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at *https://oreil.ly/effective-ml-teams*.

For news and information about our books and courses, visit https://oreilly.com.

Find us on LinkedIn: https://linkedin.com/company/oreilly-media

Follow us on Twitter: https://twitter.com/oreillymedia

Watch us on YouTube: https://youtube.com/oreillymedia

# Acknowledgments

When we started writing this book, we set out to share a collection of point practices that have helped us in building ML systems. But we ended up with a comprehensive guide that we firmly believe will elevate the common denominator of ML teams and transform how teams shape and deliver ML products. This book would not be possible without many pockets of people who—by their example, word, and actions —have influenced and shaped our approach.

We'd like to thank the wonderful folks at O'Reilly who helped to make this book a reality: Nicole Butterfield, Melissa Potter, Gregory Hyman, Kristen Brown, Nicole Taché, and other editors, designers, and staff working behind the scenes to continually refine this book from its conception to production. A massive thanks to our technical reviewers who took the time and effort to pore through more than 300 pages of content and provide thoughtful and candid feedback: Hannes Hapke, Harmeet Kaur Sokhi, Mat Kelcey, and Vishwesh Ravi Shrimali.

### **From David Tan**

Thank you Nhung for being so patient and supportive through the late nights that I spent on this book. I would not have finished this book without your support.

I see something, Jacob and Jonas—a tree! Stay curious always.

Special mention to Jeffrey Lau—your mentoring and duck noodles haven't gone to waste.

Thank you to colleagues at Thoughtworks past and present who have taught me so much about the beauty of asking questions and showing me that it's OK to tread new paths. I tried to name you all, but the list will get too long. You know who you are—a big thank you for being candid, kind, and just plain good at what you do. Special thanks to Sue Visic, Dave Colls, and Peter Barnes for your encouragement and support in writing this book.

Neal Ford: When I reached out to ask some logistical questions about writing a book, you went above and beyond to share your writing process, how to test ideas, and introduced me to Stephen King's and Annie Dillard's ideas on writing. You didn't have to but you did. Thank you for being a multiplier.

It almost goes without saying, but a massive thanks to my coconspirators Ada and Dave. You've elevated the quality and breadth of this book beyond what I could've imagined, and I'm excited to see this guidebook help ML teams and practitioners through our collective experience.

### From Ada Leung

I'd like to thank my partner, friends, and family. You know who you are. Your endless encouragement and admiration that I actually coauthored a book (Yeah, I know right?!) reminds me of how cool it is to belong among incredibly smart and impressive technologists.

I'd like to thank my Thoughtworks colleagues I've met along the way, have been inspired by from afar, and have been fortunate enough to be mentored by—your passion and generosity toward knowledge sharing has set the bar high for what good looks like. There isn't a more fitting word to describe this community than the philosophy of *Ubuntu*: I am because we are.

Finally, to my coauthors David and Dave: thank you for your unwavering support throughout this journey. From sharing our ideas and discovering the breadth and overlap of our collective knowledge, I'm reminded of how much I value teamwork and camaraderie. It's been a real joy and privilege.

### **From David Colls**

I'd like to thank my family for putting up with a husband and father writing, reviewing, and researching content for a few months straight, on weekends, during movie nights, and on the basketball sidelines.

I'd like to thank the many Thoughtworks colleagues around the world who have gone before us in writing books and creating transformative perspectives on technology, stoking our determination to do the same, and showing us what good looks like. Closer to home, I'd like to thank all the Australian Thoughtworkers I've worked alongside over more than a decade for broadening my perspectives and enriching me professionally and as a human being.

I'd especially like to thank all those members of the Thoughtworks Australia Data & AI practice who I've had the privilege to work with as we built something new together—there's a little bit of each of you in this book. I'd also like to thank our clients for the trust they place in us to develop new approaches to their biggest opportunities and challenges.

Finally, I'd like to thank my coauthors David and Ada for their expertise and insight, feedback on my ideas, and structured approach to distilling and sharing our knowledge in this book. It has been a pleasure working with you.