Preface

Ever since the first machine learning course I taught at Stanford in 2017, many people have asked me for advice on how to deploy ML models at their organizations. These questions can be generic, such as "What model should I use?" "How often should I retrain my model?" "How can I detect data distribution shifts?" "How do I ensure that the features used during training are consistent with the features used during inference?"

These questions can also be specific, such as "I'm convinced that switching from batch prediction to online prediction will give our model a performance boost, but how do I convince my manager to let me do so?" or "I'm the most senior data scientist at my company and I've recently been tasked with setting up our first machine learning platform; where do I start?"

My short answer to all these questions is always: "It depends." My long answers often involve hours of discussion to understand where the questioner comes from, what they're actually trying to achieve, and the pros and cons of different approaches for their specific use case.

ML systems are both complex and unique. They are complex because they consist of many different components (ML algorithms, data, business logic, evaluation metrics, underlying infrastructure, etc.) and involve many different stakeholders (data scientists, ML engineers, business leaders, users, even society at large). ML systems are unique because they are data dependent, and data varies wildly from one use case to the next.

For example, two companies might be in the same domain (ecommerce) and have the same problem that they want ML to solve (recommender system), but their resulting ML systems can have different model architecture, use different sets of features, be evaluated on different metrics, and bring different returns on investment.

Many blog posts and tutorials on ML production focus on answering one specific question. While the focus helps get the point across, they can create the impression that it's possible to consider each of these questions in isolation. In reality, changes in one component will likely affect other components. Therefore, it's necessary to consider the system as a whole while attempting to make any design decision.

This book takes a holistic approach to ML systems. It takes into account different components of the system and the objectives of different stakeholders involved. The content in this book is illustrated using actual case studies, many of which I've personally worked on, backed by ample references, and reviewed by ML practitioners in both academia and industry. Sections that require in-depth knowledge of a certain topic—e.g., batch processing versus stream processing, infrastructure for storage and compute, and responsible AI—are further reviewed by experts whose work focuses on that one topic. In other words, this book is an attempt to give nuanced answers to the aforementioned questions and more.

When I first wrote the lecture notes that laid the foundation for this book, I thought I wrote them for my students to prepare them for the demands of their future jobs as data scientists and ML engineers. However, I soon realized that I also learned tremendously through the process. The initial drafts I shared with early readers sparked many conversations that tested my assumptions, forced me to consider different perspectives, and introduced me to new problems and new approaches.

I hope that this learning process will continue for me now that the book is in your hand, as you have experiences and perspectives that are unique to you. Please feel free to share with me any feedback you might have for this book, via the MLOps Discord server that I run (where you can also find other readers of this book), Twitter, LinkedIn, or other channels that you can find on my website.

Who This Book Is For

This book is for anyone who wants to leverage ML to solve real-world problems. ML in this book refers to both deep learning and classical algorithms, with a leaning toward ML systems at scale, such as those seen at medium to large enterprises and fast-growing startups. Systems at a smaller scale tend to be less complex and might benefit less from the comprehensive approach laid out in this book.

Because my background is engineering, the language of this book is geared toward engineers, including ML engineers, data scientists, data engineers, ML platform engineers, and engineering managers. You might be able to relate to one of the following scenarios:

- You have been given a business problem and a lot of raw data. You want to engineer this data and choose the right metrics to solve this problem.
- Your initial models perform well in offline experiments and you want to deploy them.
- You have little feedback on how your models are performing after your models are deployed, and you want to figure out a way to quickly detect, debug, and address any issue your models might run into in production.
- The process of developing, evaluating, deploying, and updating models for your team has been mostly manual, slow, and error-prone. You want to automate and improve this process.
- Each ML use case in your organization has been deployed using its own workflow, and you want to lay down the foundation (e.g., model store, feature store, monitoring tools) that can be shared and reused across use cases.
- You're worried that there might be biases in your ML systems and you want to make your systems responsible!

You can also benefit from the book if you belong to one of the following groups:

- Tool developers who want to identify underserved areas in ML production and figure out how to position your tools in the ecosystem.
- Individuals looking for ML-related roles in the industry.
- Technical and business leaders who are considering adopting ML solutions to improve your products and/or business processes. Readers without strong technical backgrounds might benefit the most from Chapters 1, 2, and 11.

What This Book Is Not

This book is not an introduction to ML. There are many books, courses, and resources available for ML theories, and therefore, this book shies away from these concepts to focus on the practical aspects of ML. To be specific, the book assumes that readers have a basic understanding of the following topics:

- *ML models* such as clustering, logistic regression, decision trees, collaborative filtering, and various neural network architectures including feed-forward, recurrent, convolutional, and transformer
- *ML techniques* such as supervised versus unsupervised, gradient descent, objective/loss function, regularization, generalization, and hyperparameter tuning
- *Metrics* such as accuracy, F1, precision, recall, ROC, mean squared error, and log-likelihood

- *Statistical concepts* such as variance, probability, and normal/long-tail distribution
- *Common ML tasks* such as language modeling, anomaly detection, object classification, and machine translation

You don't have to know these topics inside out—for concepts whose exact definitions can take some effort to remember, e.g., F1 score, we include short notes as references—but you should have a rough sense of what they mean going in.

While this book mentions current tools to illustrate certain concepts and solutions, it's not a tutorial book. Technologies evolve over time. Tools go in and out of style quickly, but fundamental approaches to problem solving should last a bit longer. This book provides a framework for you to evaluate the tool that works best for your use cases. When there's a tool you want to use, it's usually straightforward to find tutorials for it online. As a result, this book has few code snippets and instead focuses on providing a lot of discussion around trade-offs, pros and cons, and concrete examples.

Navigating This Book

The chapters in this book are organized to reflect the problems data scientists might encounter as they progress through the lifecycle of an ML project. The first two chapters lay down the groundwork to set an ML project up for success, starting from the most basic question: does your project need ML? It also covers choosing the objectives for your project and how to frame your problem in a way that makes for simpler solutions. If you're already familiar with these considerations and impatient to get to the technical solutions, feel free to skip the first two chapters.

Chapters 4 to 6 cover the pre-deployment phase of an ML project: from creating the training data and engineering features to developing and evaluating your models in a development environment. This is the phase where expertise in both ML and the problem domain are especially needed.

Chapters 7 to 9 cover the deployment and post-deployment phase of an ML project. We'll learn through a story many readers might be able to relate to that having a model deployed isn't the end of the deployment process. The deployed model will need to be monitored and continually updated to changing environments and business requirements.

Chapters 3 and 10 focus on the infrastructure needed to enable stakeholders from different backgrounds to work together to deliver successful ML systems. Chapter 3 focuses on data systems, whereas Chapter 10 focuses on compute infrastructure and ML platforms. I debated for a long time on how deep to go into data systems and where to introduce it in the book. Data systems, including databases, data formats, data movements, and data processing engines, tend to be sparsely covered in ML coursework, and therefore many data scientists might think of them as low level or irrelevant. After consulting with many of my colleagues, I decided that because ML systems depend on data, covering the basics of data systems early will help us get on the same page to discuss data matters in the rest of the book.

While we cover many technical aspects of an ML system in this book, ML systems are built by people, for people, and can have outsized impact on the life of many. It'd be remiss to write a book on ML production without a chapter on the human side of it, which is the focus of Chapter 11, the last chapter.

Note that "data scientist" is a role that has evolved a lot in the last few years, and there have been many discussions to determine what this role should entail—we'll go into some of these discussions in Chapter 10. In this book, we use "data scientist" as an umbrella term to include anyone who works developing and deploying ML models, including people whose job titles might be ML engineers, data engineers, data analysts, etc.

GitHub Repository and Community

This book is accompanied by a GitHub repository that contains:

- A review of basic ML concepts
- A list of references used in this book and other advanced, updated resources
- Code snippets used in this book
- A list of tools you can use for certain problems you might encounter in your workflows

I also run a Discord server on MLOps where you're encouraged to discuss and ask questions about the book.

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.

This element signifies a general note.

This element indicates a warning or caution.

Using Code Examples

As mentioned, supplemental material (code examples, exercises, etc.) is available for download at *https://oreil.ly/designing-machine-learning-systems-code*.

If you have a technical question or a problem using the code examples, please send email to *bookquestions@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: "*Designing Machine Learning Systems* by Chip Huyen (O'Reilly). Copyright 2022 Huyen Thi Khanh Nguyen, 978-1-098-10796-3."

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
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

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/designing-machine-learning-systems*.

Email *bookquestions@oreilly.com* to comment or ask technical questions about this book.

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

This book took two years to write, and many more years beforehand to prepare. Looking back, I'm equally amazed and grateful for the enormous amount of help I received in writing this book. I tried my best to include the names of everyone who has helped me here, but due to the inherent faultiness of human memory, I undoubtedly neglected to mention many. If I forgot to include your name, please know that it wasn't because I don't appreciate your contribution and please kindly remind me so that I can rectify as soon as possible!

First and foremost, I'd like to thank the course staff who helped me develop the course and materials this book was based on: Michael Cooper, Xi Yin, Chloe He, Kinbert Chou, Megan Leszczynski, Karan Goel, and Michele Catasta. I'd like to thank my professors, Christopher Ré and Mehran Sahami, without whom the course wouldn't exist in the first place.

I'd like to thank a long list of reviewers who not only gave encouragement but also improved the book by many orders of magnitude: Eugene Yan, Josh Wills, Han-chung Lee, Thomas Dietterich, Irene Tematelewo, Goku Mohandas, Jacopo Tagliabue, Andrey Kurenkov, Zach Nussbaum, Jay Chia, Laurens Geffert, Brian Spiering, Erin Ledell, Rosanne Liu, Chin Ling, Shreya Shankar, and Sara Hooker.

I'd like to thank all the readers who read the early release version of the book and gave me ideas on how to improve the book, including Charles Frye, Xintong Yu, Jordan Zhang, Jonathon Belotti, and Cynthia Yu.

Of course, the book wouldn't have been possible with the team at O'Reilly, especially my development editor, Jill Leonard, and my production editors, Kristen Brown, Sharon Tripp, and Gregory Hyman. I'd like to thank Laurence Moroney, Hannes Hapke, and Rebecca Novack, who helped me get this book from an idea to a proposal.

This book, after all, is an accumulation of invaluable lessons I learned throughout my career to date. I owe these lessons to my extremely competent and patient coworkers and former coworkers at Claypot AI, Primer AI, Netflix, NVIDIA, and Snorkel AI. Every person I've worked with has taught me something new about bringing ML into the world.

A special thanks to my cofounder Zhenzhong Xu for putting out the fires at our startup and allowing me to spend time on this book. Thank you, Luke, for always being so supportive of everything that I want to do, no matter how ambitious it is.