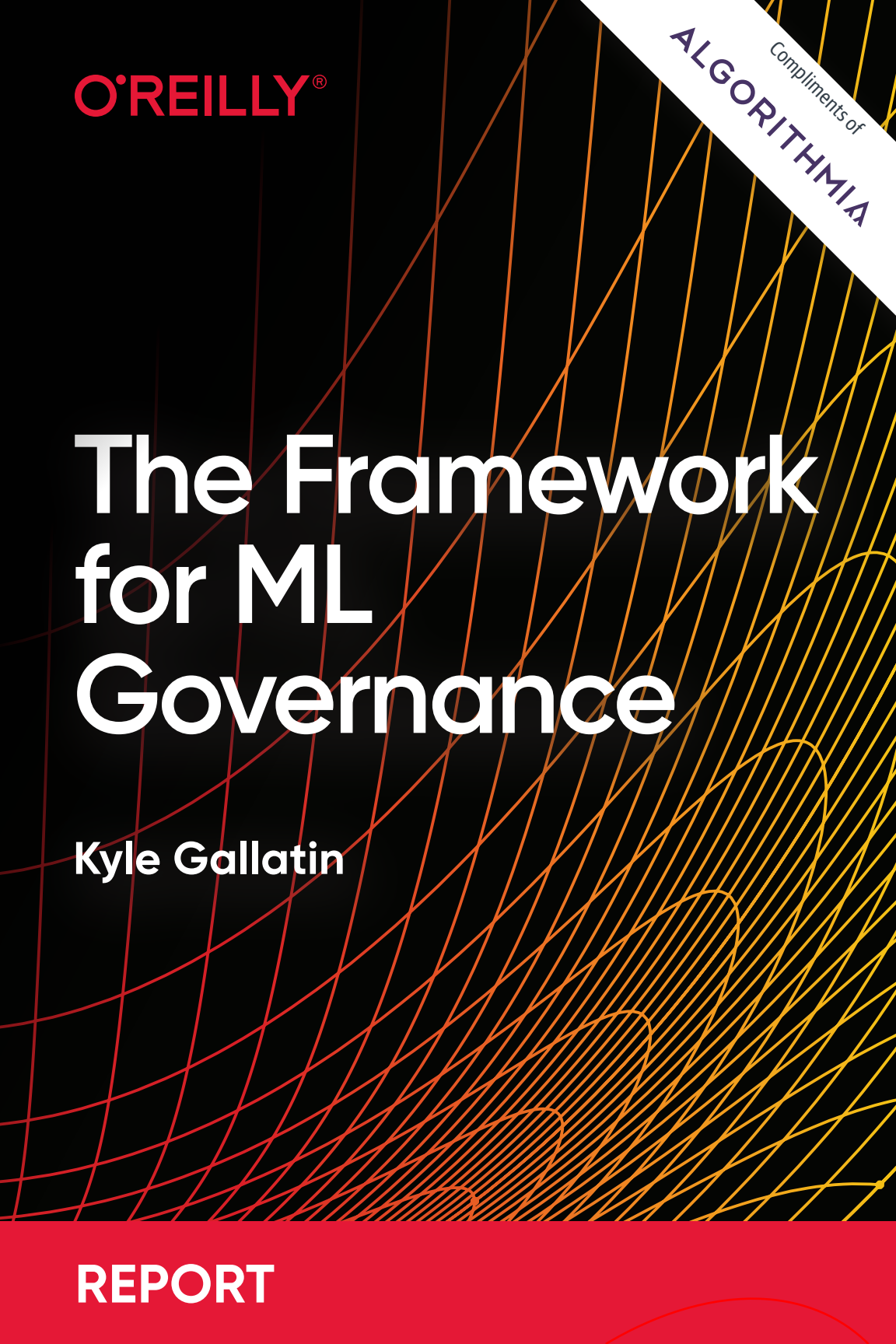
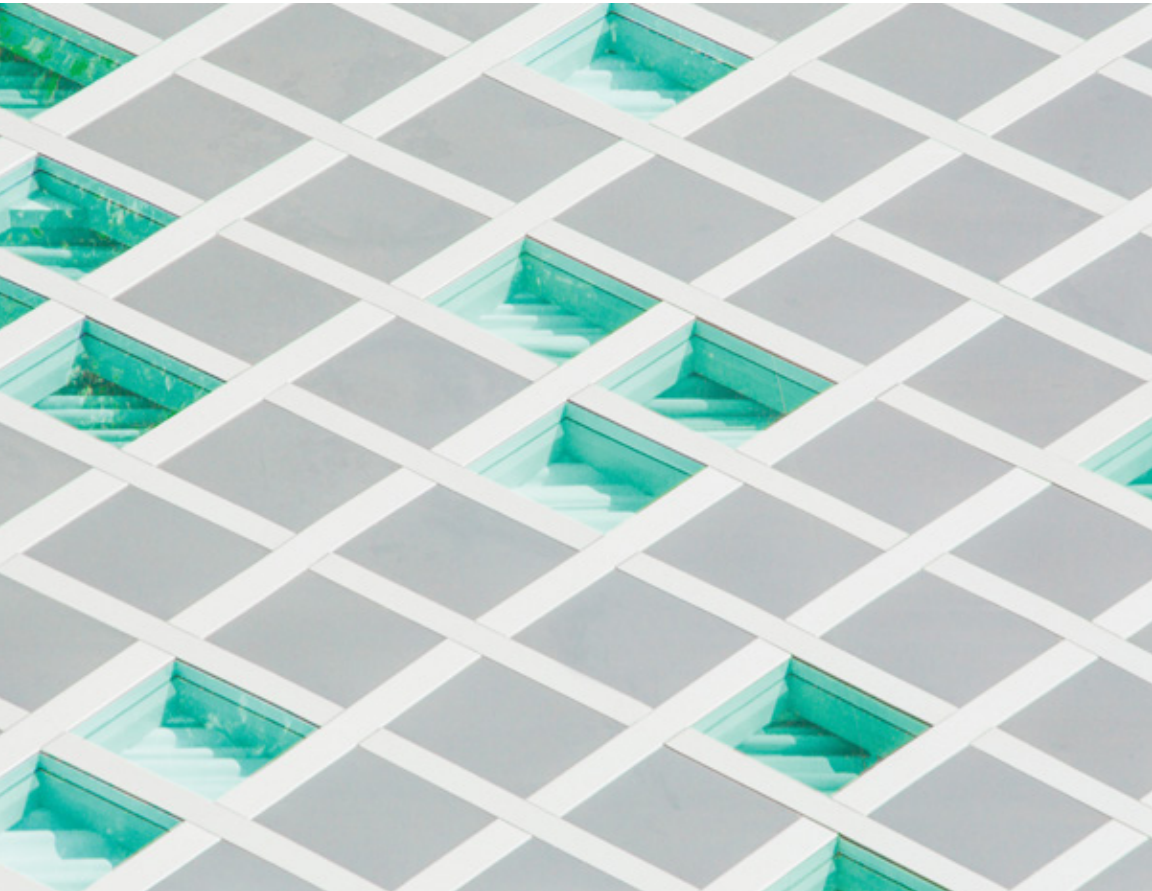

O'REILLY®
Compliments of
ALGORITHMIA
The Framework for ML Governance
Kyle Gallatin
REPORT



The Framework for ML Governance

A Practical Guide for Implementing AI and ML Governance

Kyle Gallatin

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

The Framework for ML Governance
by Kyle Gallatin

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 (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Rebecca Novack
Development Editor: Gary O'Brien
Production Editor: Katherine Tozer
Copyeditor: nSight, Inc.
Proofreader: Piper Editorial Consulting, LLC
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Kate Dullea

August 2021: First Edition

Revision History for the First Edition
2021-08-03: First Release

978-1-098-10045-2

[LSI]

Table of Contents

CHAPTER 1

Delivering Business Value Through ML Governance

The last decade has brought a dramatic boom of machine learning (ML) in both academia and enterprise. Companies raced to build data science departments and bring the promises of artificial intelligence (AI) into their decision making and products.

However, ML remained (and for some, remains) fundamentally misunderstood. Not long after companies began their foray into the realm of ML, they began to experience significant roadblocks to driving value and delivering ML projects. In 2015, Google released the now-famous paper "Hidden Technical Debt in Machine Learning Systems."[1] The paper outlined the common challenges data science groups faced with technical debt, DevOps, and governance of their ML systems.

Organizations hired data scientists in spades and started to generate algorithms. However, there were no existing operational pipelines capable of delivering models to production. This created a bottleneck that began to compound under the growing weight of new algorithms with nowhere to go. AutoML and other ease-of-use frameworks have further commoditized ML to the point that companies can now train hundreds of algorithms with the click of a button. Without a scalable framework to deliver and support models in

1 D. Sculley et al., "Hidden Technical Debt in Machine Learning Systems" (Google, 2015).

production, the exponential explosion of ML models creates more problems than it solves.

Companies were investing in ML, but lack of consideration for the operational challenges needed to scale was significantly inhibiting their ability to deliver. Algorithmia's "2021 Enterprise Trends in Machine Learning" report found that the time required to deploy a model is increasing, with 64% of all organizations taking a month or longer. In fact, the 2020 Gartner AI in Organizations survey showed that only *53%* of ML models successfully make it into production.

The reality is that the actual ML model makes up only a small portion of a project. Even if the model is trained in a day, *months* can be wasted in meetings and development across business units trying to create a secure and compliant process for ML models where none currently exists. Even when individual business units succeed in model development and deployment, operational concerns such as security and compliance are often an afterthought. The scattering of incomplete "shadow infrastructure" drains resources and poses potential risk to the organization. A complete machine learning lifecycle encompasses everything from the development of the model to infrastructure and operations (Figure 1-1).

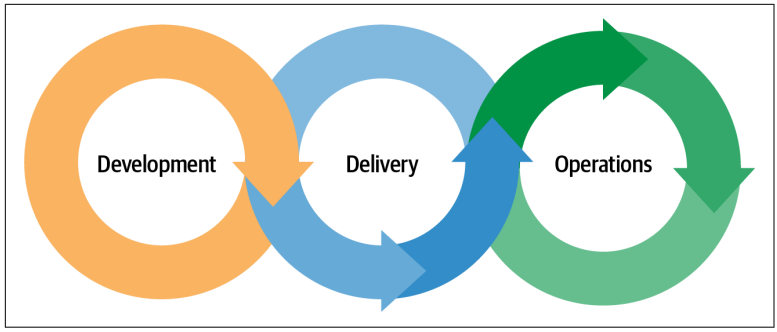


Figure 1-1. The ML lifecycle

The open-source space is meeting the technical needs of the individual data scientist but is failing to deliver the complexity of integrations required to meet the needs of the business. As a result, the number of machine learning operations (MLOps) vendor solutions is exploding. MLOps is now recognized as a critical capability by organizations delivering ML, and organizations' unique governance needs for ML are at the core of what drives the definition of that capability.

The key challenge with all "ops" is that the problem isn't just about technology, but process. Individual ML projects might successfully create "AI," but expend all their budget reinventing the wheel on ops and governance. There is yet to exist a clear set of ML principles and best practices broadly applicable to a number of business verticals. There is a need for a longstanding, comprehensive framework that enables organizations to effectively drive value with ML by successfully implementing ML governance.

In this report we will investigate the state of ML governance, and more importantly *we will define the framework for ML governance*. The purpose of ML is to unlock the value of your data, and the purpose of MLOps is to unlock the value of your ML. The framework for ML governance is not a comprehensive strategy to just deliver ML, it is a comprehensive strategy to deliver *business value* with ML. We have finally reached the point of maturity in ML where the focus isn't the academic or developer but the organization itself.

The Current State of ML Governance

MLOps is the discipline of ML model delivery. It is a set of tools and processes for delivering ML at scale that involves all technical and process-oriented portions of the ML lifecycle. Much like the software development analogue DevOps, MLOps seeks to improve the ML lifecycle by automating reproducible steps and reducing the time it takes to bring models into a production-ready state. It is not just a step in the process but the process itself. MLOps is present all the way from the high-level workflows of your organization, down to the low-level technical implementation of your pipeline from the end of development onward (Figure 1-2).

Beyond the process and technical components is the oversight of that process. ML governance is the management, control, and visibility of your MLOps. It is about both the functional and nonfunctional requirements of your end-to-end ML workflow. For example, MLOps is all about tangible features like continuous integration/continuous delivery (CI/CD), model pipelines, and tooling. ML governance, on the other hand, is about the "abilities": visibility, explainability, auditability, and other more abstract but essential requirements of a successful end-to-end ML lifecycle. These capabilities work together to mitigate risk, reduce delivery time, and provide finer-grained controls for your ML lifecycle. MLOps

without ML governance is like having a TV without a remote control. ML governance is present across the entirety of the ML lifecycle and is wider reaching than MLOps (Figure 1-3).

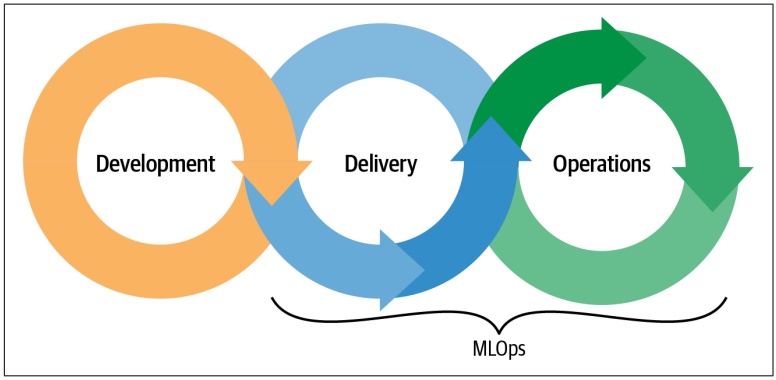


Figure 1-2. MLOps in the ML lifecycle

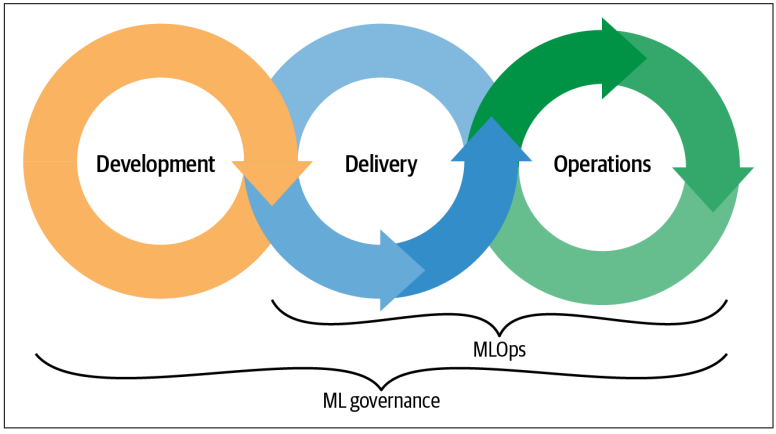


Figure 1-3. MLOps and ML governance in the ML lifecycle

For a lot of companies, the potential value generated by ML is no longer speculative. According to Algorithmia's "2021 Enterprise Trends in Machine Learning" report, organizations are dramatically increasing their investments in ML. However, they're also struggling to scale and get that value out of those investments. Most notably, ML governance is a top issue with 56% of organizations listing it as a concern.

Many organizations are also dealing with significant technical debt, and the lack of a governance strategy is only compounding these

issues. With 67% of organizations required to abide by multiple regulations (ISO, HIPAA, PCI, GDPR, etc.), lapses in governance could potentially incur massive fines or even damage a company's brand.

In order to drive effective machine learning in your organization, it's imperative to have an effective governance strategy. It's becoming increasingly clear that the quality of the models a company produces is not a direct indicator of how successful their ML ventures will be. The companies that struggle to meet their ML governance needs are the ones that will be left behind.

Many companies are still struggling to dig themselves out of the antipatterns described in the "Hidden Technical Debt in Machine Learning Systems" paper. Data scientists do not exist in a vacuum but rather interact with many operational and regulated systems. For companies bound to strict internal or external regulation, governance is likely the single greatest factor that will halt their ability to deliver in a timely manner.

And delivery isn't the only concern. Since machine learning models consume and generate data, they pose a massive risk. Large data privacy fines like the $5 billion penalty imposed on Facebook in 2019 could soon have similar implications for machine learning models which touch that data.[2] While larger organizations likely already have regulations and strategies in place, the difficulty of applying these solutions to new technology is dramatically impacting time to production.

Why Organizations Aren't Seeing Value from ML

The reality is that a number of organizations don't see value from their ML investments, and there are two likely causes:

1. They don't have a well-defined ML use case or data.
2. They lack the MLOps, governance, or infrastructure to move models into a state that drives value within a reasonable timeline.

2 See "FTC Imposes $5 Billion Penalty and Sweeping New Privacy Restrictions on Facebook" (FTC, 2019), https://oreil.ly/NjGSW.

There is a stark difference between building a "successful" ML model and actually using it to drive value. For instance, let's look at an article on Etsy that describes how to choose a metric to evaluate a model. Data scientists often focus on the *offline* metric, which tells us how well the model performs on past data. The *online* metric is often more tangible. Something like "revenue generated" is a more effective way to measure the success of an algorithm than is a purely academic metric like "model accuracy."[3]

The problem is that it can be really tough to get that measurement. After you've trained and deployed a model, you're nowhere near done. You need to monitor and maintain it to ensure continued benefit. Is the model generating valid predictions? Is it rational, or is it growing stale and introducing potential risk? And most importantly, *is it actually moving the bottom line for your organization?*

Good MLOps alleviates all of these potential burdens by providing visibility over your systems and a repeatable path by which you can improve and iterate upon your models. If value-generating ML is the destination, then MLOps is the highway. Machine learning is a continuous process. If it takes six months to deploy a model, then how long does it take to update it? By having MLOps, you provide your organization with a high-speed methodology of not only deploying new models but also updating and monitoring old ones.

Governance and security should be an inherent and essential part of this process. More than once I've seen an ML project get delayed by months because it failed to consider governance and security at inception. Machine learning may be new, but it's not immune to the same governance standards for software and data within your organization. Even if the technical implementation of your production ML models is near perfect, lack of proper governance can stop a project in its tracks—while lack of adequate security can pose a massive risk.

While the basic operational principles are the same as that of software, ML is completely new in the enterprise and presents a novel set of challenges for the governance of software, data, and the ML model itself. An even bigger problem is that most companies only recently started thinking about this, and the companies that haven't

3 See "How to Pick a Metric as the North Star for Algorithms to Optimize Business KPI: A Causal Inference Approach" (Etsy, 2020).

are poised to be the ones left behind as carefully designed model pipelines and strategy open the door for the first *real* explosion of ML-driven value in the enterprise.

What's Needed to Derive Value from ML

Production software and data workflows have well-defined standards and requirements for everything from API definitions to source-code management (SCM). ML in production needs to meet the same standards, but there is yet to exist a one-to-one mapping of these workflows to the ML workflow. In order to deliver value with ML, these same standards need to be met and applied within the context of the ML lifecycle.

Machine learning is novel technology, but it still lives within the context of both software and data. ML needs an analogue from the well-defined software engineering frameworks that let engineering teams deliver value instead of endlessly spinning their wheels, and *value* happens at the business level. Most companies know (to a degree) how to ship software. Now we need to adapt that pattern to the machine learning lifecycle and next generation of software.

MLOps and governance go beyond the ML model itself to the ecosystem of machine learning technology in modern business. Delivering value with ML is about looking at the true, expanded lifecycle of algorithms in the enterprise. Successful machine learning is often dependent on a successful implementation of this lifecycle.

Like software, the ML lifecycle can be abstracted into high-level stages.[4] We can consider them *development*, *delivery*, and *operations*. Each of these stages involves different stakeholders and critical components. However, *all* of these stages are subject to MLOps and ML governance (Figure 1-4).

4 Based on Algorithmia's MLOps management guide.

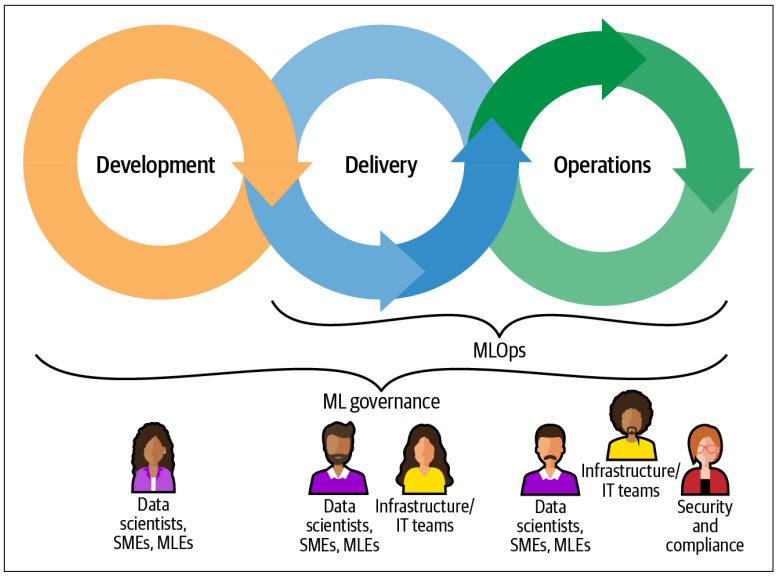


Figure 1-4. The delivery and operation stages of the ML lifecycle are more complex than development

Development

This stage involves a few different parties—usually the business subject matter experts (SMEs), data scientists, machine learning engineers (MLEs), and various other roles. It is very experimental and iterative. At this point in the process, data scientists require significant flexibility to run the number of experiments required to generate a successful model. Functionally, this is very similar to software development but with more focus on data and artifacts than code.

Delivery

At this point, a "successful" model has been trained and it must be deployed to production. This means very different things for different ML teams. The model might be deployed for real-time inference or batch inference, built into an application, or even used in an executive report. This stage often involves an interface between data science, ML engineering, and infrastructure/IT teams.

Delivering a model should be a repeatable, secure, and automated process. In order to enable your models to generate value, you need to enable your ML teams to effectively deliver. The "bridge" between

experimental data science teams and software/infrastructure teams is fundamental in creating reproducible delivery and allows everyone to focus on what they do best. This stage is characterized by technical components such as model versioning, API development, and source code management.

Operations

A common misconception is that scale = production, but this is not the case. Simply deploying a model on a large-scale framework like Kubernetes doesn't mean that it's in a production state. "Production" requires a reproducible pipeline, checks and balances, security, tests, and other common DevOps concepts. Operations outline the baseline characteristics of a true production system.

Operations are an infrastructure-level concern. At this point, we are integrating with multiple systems, dependent on monitoring/alerting, and there is a customer-facing component to the ML application. Moving into production also requires you to meet the security and compliance requirements of your organization and meet other business needs beyond the pure functionality of your application.

At this stage, you need to address business concerns like financial visibility (chargebacks, showbacks), regulatory concerns, user authentication, role-based access control (RBAC), and a litany of other ML-agnostic features.

Not only do these components need to be built, but they also need to be supported and maintained over time. To have successful operations, you need to have repeatable and sustainable processes that meet the internal and external requirements of your organization.

ML Governance and the ML Lifecycle

MLOps and ML governance are not "stages" in the ML lifecycle themselves but span its entirety, enabling businesses to derive value from ML. Components of a true production system such as monitoring, observability, and security wouldn't be possible without ML governance, and its presence at every stage in the lifecycle enables teams to rapidly iterate effectively and drive value with ML.

The specific implementation of ML governance may vary based on what stage in the process it's being applied, but MLOps and ML governance are the glue that holds the otherwise disparate lifecycle

components together, resulting in a much smoother and quicker iterative process.

In order to deliver value with ML, you need an ML lifecycle. In order to have an ML lifecycle, you need to implement effective MLOps and ML governance.

A Consistent Framework for ML Governance

It's been established that companies are struggling to set up effective governance for their models, which is inhibiting their ability to deliver ML. But what makes ML governance so difficult? It's also been made clear that ML governance is similar to existing governance for both data and software infrastructure. If this is the case, then what makes ML governance especially challenging? The reasoning is multifaceted:

ML is new to business leaders.
: Corporate executives only recently came to understand ML and started to see it not as something entirely new but as a specific implementation of technology that already existed.

Data science is an academic discipline.
: Data science evolved from inherently academic practices. Data scientists were expected to enact ops and governance without the skills or knowledge to do so.

Governance is ad hoc.
: Poor governance solutions often arise from unprepared data scientists. Many teams developed antipatterns and technical debt that's difficult to dig themselves out of.

ML differs from software.
: ML, for all its similarities, does differ from software in some key areas. It is experimental, with its own set of requirements, and often difficult to explain—making direct governance analogues difficult to apply.

There is no set of universal best practices for ML governance.
: Due to these differences, there is no generally accepted best practice for ML governance. Most organizations are left to figure it out.

Any combination of these factors can easily stop an organization in its tracks, leaving data scientists isolated from the engineering and business-level support they need to implement a robust solution to the problem. But to implement ML governance, you first need to really understand what it is and build a culture around it.

ML governance is present across each stage of the lifecycle: development, delivery, and operations. Each is critical to successfully driving value with ML.

The Need for Governance in Development

Development is the experimental cycle in which data scientists compare different models, features, and parameters in an effort to find the most performant algorithm. In this stage of the lifecycle, ML governance is primarily about managing experiments, infrastructure, and the transition from experimentation to production. Implementing operational foundations in the development stage enables data scientists to easily evaluate experiments while providing a lightweight structure to their work. This gives data scientists the freedom they need to experiment without too many guardrails, while at the same time smoothing the transition from development to production during the delivery stage.

From a governance perspective, development is often fairly straightforward. Since nothing is actually in production or serving real users, there are significantly fewer procedural hurdles. If there are difficulties, they lie in the experimental nature of data science. It is the job of a data scientist to optimize machine learning models, *not* keep their code in a production-ready state.

That being said, adopting MLOps early on has the benefit of yielding solid governance. Tracking, versioning, and logging experiments increases visibility and auditability early on in the process. Even if the models themselves aren't in production, the ML development phase still uses potentially sensitive and regulated data. This means even ML development isn't immune from possible audits or the risk of a compliance breach.

ML development also has a huge effect on the pipeline to production. The handoff between data scientists and IT professionals is one of the more difficult parts of the ML lifecycle. MLOps in this part of the process will have significant implications on how effective that handoff is, and how quickly the model can make it to production. In

addition to being the first step of the ML lifecycle, development is also the first step of an ML project. This means it has larger implications involving stakeholders and other engineering professionals required to make it successful.

Again, the governance at this experimental stage is inherently less intensive than that of the operations stage. Still, effective governance unlocks value from their workflows and feeds into a sustainable governance strategy. Implementing just the baseline of the components above can form the bedrock for the more intensive governance concerns as the model moves closer to a production state.

The Need for Governance in Delivery and Operations

The delivery and operations stages of the ML lifecycle rely heavily on effective MLOps—the set of best practices for the technical and process-oriented portions of an ML pipeline delivering ML to production. MLOps is the hard part of delivering ML to production, involving all the complexities of any production software system. Once a data scientist has identified an effective model, that model needs a robust, automated pipeline that can consistently deliver it to a scalable production state. In addition, MLOps needs to provide the automation to *repeat* this process as models in production get stale and new data and models become available.

ML governance is the management, control, and visibility of your MLOps. Governance works to democratize ML in your organization, driving control and visibility beyond the developer to the level of the businessperson. Without proper visibility, it's difficult for anyone other than the data scientists themselves to understand the inner workings of their models. This has the dual detriment of making it difficult to evaluate both model success and potential risk.

"Model risk" often arises as a result of bias in the model. In the financial sector, models with poor performance have a greater risk of losing the company money.[5] Models making decisions on biased data can adversely affect both individuals who rely on those models as well as the face of the company itself. When Apple's algorithms were giving different credit ratings to people based on gender, they

5 See the Algorithmia blog post, "What You Need to Know About Model Risk Management".

not only received significantly bad press but underwent an investigation.[6]

This risk is also prevalent in model performance. Models that undergo "drift" lose predictive power over time, and identifying this deterioration is one of the primary use cases for effective governance. Consider a credit-scoring model that helps a bank decide whether or not to approve a loan for an individual. Loss of predictive power in the model could result in the model denying viable applicants or granting loans to applicants without the ability to pay them back. Among other concerns, this represents significant potential risk for both parties. Through enhanced visibility, ML governance keeps tabs on models to identify and mitigate these problems before they have adverse effects.[7]

A lack of ML governance will not block you from deploying a single model at the delivery stage, but it will completely block you from quickly and effectively deploying many of them into a true production system. The greater the number of models you have, the greater the need for fine-grained controls to manage them. Your algorithms should demonstrate transparent metrics so that decision makers can use them to inform strategy. Monitoring a model for statistical, financial, and computational performance sounds like very different things, but they're all central to a robust ML governance strategy.

Prioritizing ML visibility will enable the business to better leverage that model for risk-free decisions. If you don't have a model catalog, your developers or users will have a difficult time discovering and leveraging your models in the first place. Even if all of the technical implementations of your deployed model are sound, it still needs to be visible and discoverable within your organization to provide value.

6 See "Apple's 'Sexist' Credit Card Investigated by US Regulator" (BBC News, 2019).

7 See the Algorithmia blog post, "What Is Model Governance?".

CHAPTER 2

Governing ML During the Development Stage

Data scientists are, first and foremost, scientists. Much like biology researchers in a lab, they conduct iterative, potentially long-running experiments to solve a problem. Researchers are almost always required to keep an auditable trail of their experiments, and the same applies to the data scientist.

ML development is the iterative cycle data scientist(s) undergo to optimize model performance for a given problem. It involves data preparation, model training, and evaluation. It's worth noting that this process is a bit different from software development. In software engineering, code is the primary concern. But in ML development, *data* is the primary determinant of success.

Code in an ML application is all about servicing the data rather than the application itself. Most ML microservices have a minimal amount of code but interface with a large number of data sources and services. At the high level, software engineering is *code-driven* while ML is *artifact-driven* (an "artifact" in this case being the ML model binary and/or data powering the model). This shift in priorities gives ML applications a different set of concerns than standard software applications.[1]

1 See the Algorithmia blog post, "Five Myths of MLOps".

We mentioned earlier that governance at the development stage is not as difficult as it is at the delivery or operations stage. Since development is preproduction, organizations can generate significant value just by implementing a consistent framework for ML experiments, which benefits data science teams all the way through to production. First, let's define MLOps and ML governance as they pertain to the development stage of the ML lifecycle (Figure 2-1).

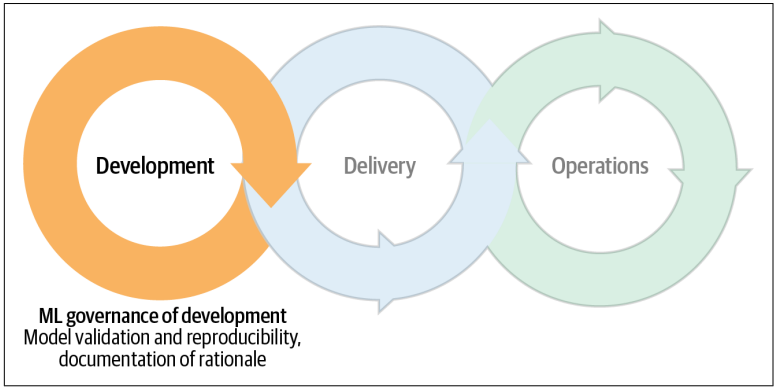


Figure 2-1. ML governance during development

MLOps of Development

As previously stated, MLOps is a set of tools and processes for delivering ML at scale by leveraging your existing software development lifecycle (SDLC). For development, this means a focus on experiment management: model packaging, management of the training cycle, and the interface between the data science and infrastructure required for scalable training. This is a prerequisite for creating a pipeline that delivers ML artifacts to production. Your MLOps tools should help data scientists iterate quickly, compare experiments, and generate a standardized output for consumption by downstream ML teams.

ML Governance of Development

Only by fostering complete control and visibility over their models in production can organizations effectively manage their ML. They can also satisfy business requirements by generating compliant models and reduce the chances of risk and technical debt. Organizations need to create capabilities for reproducible experiments that

enable the parties of an AI project to effectively validate results. They also need to document the technical approach, business logic, and steps followed to better inform future businesspersons, data scientists, and developers.

While these concepts are functionally interdependent, here we discuss the framework for governance in particular, and the components of governance with respect to development.

Model Validation and Reproducibility

The simplest analogy for this part of the process lies in the physical sciences, where it's a hard requirement to keep a running lab notebook detailing experiments in depth. All of the steps necessary to reproduce any experiment should be logged in depth so anyone can replicate it.

In science, reproducibility is the ability to arrive at the same result twice. ML reproducibility is the combination of tools and information that allows data scientists to successfully re-create experiments. Essentially, it is a recipe for re-creating any model. Then using this metadata, we can more easily validate the model(s). Model validation is the process of ensuring your model is performant, statistically sound, delivers statistically significant benefits, and meets the definition of "success" put forward by your AI project.

The capability to reproduce and validate not only benefits the individual data scientist but also is a catalyst for solid governance. A log of experiment metadata keeps an AI project auditable and explainable. An *auditable* project is one in which an auditor or other regulatory party can easily identify and find the information necessary to validate the checks and balances. An *explainable* model is one in which the relationship between the statistical performance, features, parameters, and model selection is well understood.

At its core, the practices of fostering model validation and reproducibility are about *not being messy* during a part of the process that easily lends itself to poor governance. Organized data scientists will often group models by project. Each attempt to train a model for that project is called a "run," with all of the runs for that project being rolled up into an "experiment." Putting forth a simple metadata framework centered on the concept of an experiment yields increased visibility and auditability for any AI project.

Subcomponents of model reproducibility

When we talk about the "subcomponents" of model reproducibility, we are talking about technical information. Just as a biologist in a lab would extensively document the steps to conduct an experiment, a data scientist should be logging the same information. The following are commonly required metadata to reproduce an experiment, or a run of an experiment:

- Type of algorithm (random forest, neural net)
- Features and transformations
- Data snapshot (reference to input and output data at a particular point in time)
- Model tuning parameters
- Performance metrics
- Verifiable code from source control management
- Training environment (Docker image, package requirements)

Subcomponents of model validation

Validating models for a project is a multistep process and often requires more than one metric. In the process of validating a model, we should strive for the following auditable and explainable characteristics:

- The model achieves acceptable statistical performance for a sensible offline metric (such as accuracy).
- The model achieves a statistically significant improvement when compared to a control on some online metric or key performance indicator (KPI) (clicks, conversions, purchases). This is usually determined through an A/B test.
- The model is statistically sound. There is no data leakage and the supervised ML problem was framed correctly.
- The model can be reproduced from existing metadata.
- The performance of the model can be successfully explained based on the available features.

Setup within your organization

Adoption by data scientists is critical in enabling model reproducibility in your organization. Different data scientists have different preferred workflows, and different data science teams operate in almost completely different spaces. Even if you provide a central tool to keep track of the wide variety of experiments, you have no guarantee anyone will even use it.

The highly iterative nature of data science also means that your approach will need to be designed for flexibility and speed. It's not uncommon for data scientists to suddenly pivot their approach in the middle of a project or generate and then trash large chunks of code based on experiment results. The goal is to achieve this level of flexibility without compromising the quality of your delivery pipeline and production model.

Of course, like any initiative, you will need an inherent understanding of your users (the data scientists), their requirements (in addition to governance requirements), and well-defined responsibilities. If you only have one data science team, it's possible that responsibility for driving this initiative could live within that team itself. However, if you need to unify many teams across an organization, then this is best managed centrally as part of your ML governance strategy. Without well-defined requirements and responsibilities, it will be difficult to successfully enact this framework at scale.

Documentation of Rationale

Much like in software, documentation of any ML algorithm gives it longevity and reduces the chances of technical debt. Institutions that have been using machine learning for a long time often accrue surprising debt in their models themselves. Financial institutions might have tens of models that have been in production for a decade, with the employees who created them long gone. As a result, current data scientists have to invest significant time and effort reverse-engineering them or making potentially false assumptions.

Extensive documentation may also be a hard requirement for models deployed in highly regulated spaces. Your software may require approval from institutions such as the FDA or SEC. You may need in-depth, statistically justified explanations for your ML approach at the low level. There are often a number of acceptance criteria for

such documents and the information within them, making them nontrivial to assemble without strategy and domain knowledge.

Documentation in ML can even alleviate risk, but the key is having *good* documentation. Effective documentation takes reproducibility one step further by fostering longevity and reusability through visibility and explainability. Documentation for a model builds on the information captured in "Subcomponents of model reproducibility" on page 18 by adding more context, and specifically more *business* context. It is not a long-running list of experiments or model runs but a story about how the best experiment run was created and got where it is today.

Subcomponents of model documentation

Effective ML documentation outlines the following information in a clear and concise manner, using diagrams and images where applicable:

- Business context
- Business justification for the algorithm
- High-level explanation of the algorithm itself
- Model parameters for tuning
- Feature choices and definitions (inputs and outputs)
- Any customizations to the algorithm(s)
- Instructions for reproducing the model
- Examples for training the algorithms
- Examples for making predictions from the algorithm

Documentation should be clearly written and easily digestible. It's highly likely that the people reading your documentation years from now will not have the context you or your team have. Much like good writing and presentations, you need to know your audience and refrain from making assumptions about their knowledge. O'Reilly provides eight high-level rules for writing good software documentation.[2]

2 See the O'Reilly article, "The Eight Rules of Good Documentation".

Setup within your organization

Documentation is almost always a hard requirement for delivering a technology project. Even pure Agile methodologies that lessen the focus on documentation in favor of results can still deliver effective, streamlined content for the developer. It's highly likely your organization has SDLC or compliance requirements that merit some level of documentation, so it's presumably already being generated.

The difference is in the ML-specific components of the documentation. If ML projects get treated purely as software, then the documentation is prone to leaving out critical information needed to re-create an experiment. The data-driven, statistical considerations of ML merit in-depth explanation. It will be important to build the ML requirements above into the existing compliance requirements for your organization. This way, you ensure AI projects aren't making it out the door without a sufficient explanation of the components at play.

At its core, documentation is about good writing and communication. Understand the audience and provide critical information without being verbose. No one *likes* to read documentation. It should get straight to the point, provide gratuitous diagrams over text, and always start from the basics.

CHAPTER 3

Governing ML During the Delivery and Operations Stages

Governing the delivery and operations stages of the lifecycle is the most difficult part of a comprehensive ML governance strategy (Figure 3-1). During development, there was an expected margin of error and a low risk of failure. It doesn't matter if your data scientists make statistical mistakes along the way, as long as the mistakes are ironed out. Production does not grant the same lenience. Inadequate governance can easily lead to lapses in revenue, wasted resources, and even potentially irreparable damages to the company.

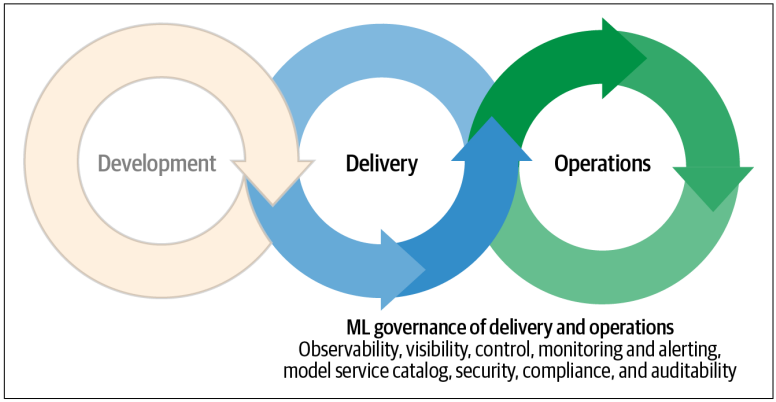


Figure 3-1. ML governance of delivery and operations

MLOps is the sum of production considerations for a machine learning model or pipeline. Again, it is a set of best practices for the technical and process-oriented portions of an ML pipeline delivering ML to production. During the delivery stage, data scientist responsibilities are phasing out while software, infrastructure, IT, and security responsibilities phase in. This stage has a *massive* bearing on how fast models get to production, and the value they can generate once they reach the final stage of operations. Operations includes at a minimum:[1]

- Applications (software and business-facing)
- Application integrations
- Business intelligence (BI) dashboards and visualization
- Model management
- CI/CD (DevOps and MLOps)
- Infrastructure

The operational considerations here are the business-critical components of your ML application. The technical pieces of MLOps enable a business to have machine learning models in production. However, there is a big difference between *having* MLOps and *managing* MLOps. In Chapter 1, we defined ML governance as MLOps management, encompassing the functional *and* nonfunctional requirements of your production machine learning platform. Governance sits above development, delivery, and operations as a complete control plane for your models in production.

This control plane should provide enterprise-level management and visibility over your entire machine learning lifecycle. Monitoring a model for statistical, financial, and computational performance may sound like very different things, but they're all central to a robust ML governance strategy. A governance framework can *explain* performance by logging model predictions, *monitor* performance, *catalog* existing models, remain *compliant* and *auditable* within your organization, and *secure* these models against internal or external threats.

1 Based on Algorithmia's MLOps management guide.

MLOps is a platform-level concern, but ML governance is a business-level concern. It is the difference between just having metrics and making them visible and widely consumable in a dashboard. The ability to deploy a model and the ability to manage it are two fundamentally different things. Organizations that implement successful governance can mitigate risk and deploy models significantly faster. But on top of that, they're also the only organizations set up to *improve* faster. The first step in solving a problem is identifying it, and in the "black box" world of machine learning that capability can make or break the next generation of enterprise.

For a lot of companies up until now, MLOps has probably felt like an uphill battle. The ML space can be dauntingly complex, and poor understanding of the space has left AI projects fighting against delivery and operations—each a more difficult and impassable wall. The purpose of ML governance is to simultaneously break down those walls and provide a repeatable path for everyone.

Here, we propose the individual components, subcomponents, and subsequent implementation of a robust framework for ML governance at the delivery and operations stages of the ML lifecycle. Each of these components plays a critical role in enabling the success of AI in your business and thus ensuring the following success of your organization as an AI company.

Observability, Visibility, and Control

This component is the "who, what, when, where, and why" of your machine learning deployments. Organizations need to be able to understand their model performance, metrics, infrastructure, and access controls. Governance should enable visibility into model access, model and data changes, model interdependencies, and what data is used by which model.

Without visibility, it is significantly more difficult to identify problems, generate reports, showcase ML value, keep software and data compliant, and iterate on models in production. To improve your ML, you first have to understand what it's even doing. If there's no visibility, then data scientists, developers, and other project stakeholders will constantly be scrambling for information, and turnaround time on issues will be much higher. A growing requirement for ML deployment tooling is first-class integration with dashboards

and visualization software. Otherwise, the information necessary for key business and project decisions is not democratized.

If you have good observability and monitoring, you typically will also have:

- Faster turnaround time on production issues
- Faster onboarding times for new members
- Decreased time to delivery, and therefore increased value-driven ML
- Decreased compliance and security risk
- Increased impact visibility within your organization

Users with the necessary access controls should be able to discover and access the same high-level information about your ML deployments. Instead of going to specialized team members (developers, security, etc.), nontechnical team members can access information unaided. Depending on your organization, "good" observability is typically an amalgam of some, or most, of the following:

Model logging, metrics, and auditing
: Good logging is software 101. It's impossible to overstate the value of good logging for mitigating issues, understanding model inputs and outputs, and rapidly solving problems. With ML deployments, there is also more to log. It's likely your ML applications ingest and generate significant amounts of data. In addition to generic software application logging, you may also want to collect information about the features and predictions for requests to your model.

  Shifts in the statistical distribution of data or predictions could imply model drift—a condition in which a model's performance is reduced due to changes in the underlying data. Model drift is complex and could be the result of a shift in features (data drift) or even the model target itself (concept drift). Without the logs to analyze either, it will be impossible to diagnose and remediate.

  This information can then be rolled up into digestible metrics and dashboards for consumption by data scientists, software engineers, and managers alike. Logs are informative to technical team members, but metrics and dashboards service a larger

audience and significantly reduce the pressure on individual engineers to generate ad hoc metrics for presentations or papers.

You can also use these insights to perform routine model audits by ensuring that your performance remains up-to-date and within the guidelines of any business SLAs. Logging gives you the ability to check that data ingested by the application is being used in accordance with internal and external guidelines, and only the appropriate users are accessing the model.

Cost visibility
: Clearly surfacing insights on the cost and associated resource utilization of your platform can be difficult to achieve. Most organizations rely on a myriad of tools for their ML workflows, and calculating the cost of a specific model in a large Kubernetes cluster isn't always straightforward. Cost visibility can also be a spontaneous priority, when sudden requirements from executives require team members to generate ad hoc metrics or reports.

  In addition, if you're delivering ML, then you may need to generate chargeback and showback reports for the customers consuming your models. Different teams will need to be charged for their specific model and resource usage. Cost is likely to fluctuate based on a number of different factors, so building visibility during platform development is an important way to proactively mitigate issues down the road. Automated generation of chargeback or even just showback reports for model subscribers can help individual departments remain on top of their finances and make cost visibility a feature rather than a concern.

Usage reports
: Usage reports give you holistic insight into the impact of your platform and the individual models served on that platform. The ability to break down usage data to a more granular level gives you comprehensive visibility into the success and adoption of different models in your organization. This also helps during the audit process, helping to make sure models are being accessed by only authorized users.

Endpoint management
: Endpoints are the accessible APIs for models within your organization. Designated users of a platform should be able to cre-

ate, alter, or delete the available endpoints for models in accordance with project requirements.

Versioning
: Like software, models change and grow over time. Models should have versioned releases to ensure the principle of immutability—that is to say, once something has been released, it should not change. The model *version* becomes the immutable object, so that it is always possible to roll back or access previous versions of trained models. No data is lost, previous versions aren't altered, and the metadata tied to a specific version is visible to the members of your organization.

Monitoring and Alerting

While explainability involves collecting and surfacing different types of information, monitoring and alerting are about the automated ways you keep track of those metrics. Sudden shifts in data distribution or platform uptime could spell trouble for consumers of your model endpoints. Actively monitoring these issues and alerting team members where applicable reduces the cycle time for troubleshooting sudden issues:

Platform and infrastructure integration
: A prerequisite for having comprehensive monitoring is having systems that easily integrate with one another. Integration itself is such an important feature, it's often the first thing IT professionals look for in evaluating third-party software. If some enterprise tool solves 80% of your requirements for a given use case, but it also allows you to easily build the remaining 20%, then you have the opportunity to dramatically accelerate your workflows without losing the ability to develop custom tooling. Your ML solutions should provide easy integration with common dashboard and monitoring tools, such as Grafana, Kibana, Datadog, New Relic, or Splunk. Integration as a feature allows your engineers and data scientists to focus on ML-specific tooling without reinventing the wheel on established, difficult areas like security and authentication.

Uptime SLA
: Uptime SLA is a common metric denoting the time an application spends active and available to serve requests. Uptime is a critical metric for the stability and availability of a platform, and

monitoring the metric over time provides insight into whether your models and infrastructure are meeting the needs of your consumer. If you have an uptime SLA, you quite literally *have to* monitor this metric to successfully deliver models to your customers.

Alerting (model logging, metrics, and auditing)
: Earlier, we referenced all of the logs and metrics generated from our ML deployments. While surfacing these in dashboards is great, without further automated processes it solves very little. Alerting users about issues or errors in software is extremely common but alerting for the data-specific issues is a much more novel concern. Large shifts in data distribution can lead to model drift or other downstream ML issues that should automatically trigger notifications to experts immediately. Uber has implemented a novel data quality monitoring platform that implements more intelligent alerting.[2] Many of the best practices here could help inform future ML monitoring strategies.

Model Service Catalog

Like data catalogs, model catalogs are growing in popularity. Having a marketplace for your services is a key feature driving discoverability, and thus adoption. Model catalogs have quickly moved from a "nice-to-have" feature to a requirement for organizations generating numerous models that may be reusable across a wide scope of the business. Model catalogs dramatically increase the usability and reusability of your ML, which in turn drives value and reduces duplicative effort by similar groups in your organization.

A model catalog is not just a magazine for perusal; it should feature comprehensive integration with internal services and logical grouping of various services. Good model catalogs should contain some form of documentation, including sample requests for users looking to get started:

Model catalog, storage, and versioning
: A model catalog is actually an MLOps-centric combination of multiple software and governance components. The catalog itself should have good UX, needs to connect to blob stores or

2 See Uber's blog post, "Monitoring Data Quality at Scale with Statistical Modeling".

other locations where models are commonly stored, and needs to keep a running log of relevant metadata for a model like the latest version, inputs, and outputs.

SCM integration (GitHub, GitLab, Bitbucket, etc.)
: In addition to the model artifacts stored in places like S3 or GCS, you also need access to the code that powers model predictions stored in GitHub or some other SCM.

Custom library catalog and labels
: Model catalogs don't generalize very well beyond a specific organization. Since each individual organization's use cases are so specific, each organization will also have their own labels and categories for models that fit their purposes. The library of models and their labels will be customized to specific business use cases to enable better discovery and usability of models among users.

Usage metrics
: In a way, your catalog is a product. It is the entry point for existing and potential users, where they can "purchase goods" created by your data scientists and ML engineers. This is the primary place to collect the usage metrics we outlined in "Observability, Visibility, and Control" on page 25. A well-engineered model catalog should have good UX that enables users to find the models they need and should help you gain insight into the success of individual models, model categories, labels, and other subdivisions for decision-making purposes.

Security

For data scientists, security is often an afterthought—one of those "last step" concerns that can be dealt with at any point in time. For your security and IT professionals, this mindset is their nightmare. Security is a *huge* concern with a wide range of implementations and implications. Without good security on your ML, you put sensitive data, IP, user information, and even wider infrastructure at risk. In ML, where data touches every part of the application, sensitive and regulated information might be ubiquitous. The expectation is that this data is protected by layers of competent engineering and sensible access controls.

Unless your data scientists come from extensive infosec (information security) backgrounds, it's highly likely you will be relying on third-party providers for all or most security features of your ML infrastructure:

Data and information security
: Data security is the primary concern of most ML applications. Algorithms trained on sensitive data often read from and write to highly regulated sources. With your ML positioned as a central point of failure, data security is even more critical. If the model is making predictions on regulated data, data may need to be encrypted at rest and in flight. Realistically, this should mirror the same level of security and control this data has in other applications or the database.

Compliance with IT standards
: Exposing model endpoints comes with the potential risk of users abusing models. By exposing an HTTP endpoint, you effectively make it so that anyone with the right knowledge can request your model, get access to predictive insights, and drive up your compute costs. Even if the model is only available within the company's virtual private cloud (VPC), you need to ensure that it has the proper amount of access controls and isn't just widely accessible by the organization, especially if it has the ability to return sensitive information.

  Networking is often complex to engineers without experience in the space, involving multiple DNS, proxies, and load balancers for traffic. All of these features are necessary to serve your models at scale, and therefore need to be developed in a secure and compliant manner. It is unlikely that a data scientist alone will have the domain knowledge necessary to implement these features. Instead, most companies opt for provided or somewhat-managed solutions from cloud or enterprise vendors.

Authentication, SSO, and RBAC
: This is a big one, and often overlooked by data scientists. Similar to network security, your ML models *must* have the ability to integrate with existing authentication solutions, like SSO (single sign-on), or provide some form of token-based authentication. This way, only authorized users can make requests with the model *and* you can identify users. Role-based access control (RBAC) is important for further controlling the permissions of

your platform. There will be those with admin-level permissions, user-level permissions, or other subsets based on the project, model, data, and business unit. Without the ability to fine-tune authentication to the role level, your models aren't secure.

For instance, a user would have the ability to call or get insights on a model. However, a user would *not* be able to create new endpoints or generate models of their own. Role-based division of responsibilities is important to keep your platform secure. Such security may also come in the form of an admin UI, or privileged access to certain resources (like usage logs). If the data from requests is also being logged and that data is controlled, then it's important the access to those logs is also controlled.

Endpoint and API management
: Endpoints are the entry points into your infrastructure, which always makes them potential security risks. They need to be secure (potentially protected by OAuth2 or other tokens) and managed in a secure manner (authorized users can create, edit, or delete them). If they provide access into a secure network, like a VPC, then it's even more important to control the availability of these endpoints with strict regulatory controls and best practices in software engineering.

Infrastructure security
: Securing infrastructure as a whole is a broad topic. Developers have access to a number of resources for developing and deploying their ML models. This includes UIs for building, Secure Shell (SSH) access to various machines, permissions to update clusters and infrastructure, and more. Access to any one of these resources gives an individual a lot of power over the ML models in production and opens up the opportunity for lots of damage. Your infrastructure should have similar RBAC as your network and other tooling for alerting or editing models.

Key and secret management
: A common mistake many data scientists make is to try deploying models and building the authentication mechanisms themselves. While this is perfectly acceptable, too often the solutions implemented aren't scalable and end up involving shared passwords, secrets, or at worst passwords stored in plain text. Your

end-to-end ML platform should integrate with a solution for creating, storing, and managing all of the secrets and keys associated with your machine learning deployments.

Systems audit (network and infrastructure security)
: For a lot of companies, it's not so much how you build something but *where* you build it. Larger organizations have a lengthy process for evaluating new technology and ensuring its security. The infrastructure, tools, and platform on which you deploy your models will have to undergo a potentially lengthy security audit. Making any assumptions about the approval and use of technology within your organization without the consultation of IT and compliance is a big project risk.

  The largest concern usually comes down to the location of the deployment in regard to the network. Deploying ML within your company's VPC is considered a far lower risk than an external deployment. You will need to show that the application, network, and location of the data do not put your ML deployments at risk.

  In addition to the VPC itself, you may also have proxies, internal domains, and firewalls. Just because an application is "internal" does not mean it isn't still subject to certain concerns. Simple things like HTTPS may be a requirement for your model endpoints but beyond the level of familiarity for your data scientists. When building out a machine learning strategy in your organization, make sure to sync with IT and ops professionals to create a comprehensive outline of these requirements so they are well understood from the beginning.

Compliance and Auditability

This section starts to take us even further away from the expertise of your data scientists. Engineering for compliance and auditability is a highly specialized skill that takes years of experience. Many companies work with domain-specific data or regulations that heavily alter their SDLC. As a result, the requirements and subsequent audits of this software are highly subjective to the domain and organization. Both financial and healthcare organizations are heavily regulated but in completely different ways.

That being said, compliance and auditability for ML can still be addressed from the high level. If you think about compliance and auditability from the beginning, you should have far less trouble getting ML off the ground and into a usable state. AI projects that involve compliance and security professionals early are more likely to deliver ML on time, as opposed to misscoping the necessities outright and allowing the project to slowly creep far beyond budget. Remember that audits can even extend as far back as development, where our "lab researchers" need to be keeping robust experiment logs if they're working with sensitive information.

If compliance and auditability are both must-haves for your highly regulated domain, then the ML governance strategy, tooling, and platform on which you train and deploy your models all need to have a high level of visibility and automation in order to meet your requirements. Otherwise, audits will take as much time as the product itself.

Model logging, metrics, and auditing
: The importance of logging and metrics in regard to compliance can't be overstated. Compliance is driven by reporting, for which you need copious records corroborating your model's adherence to your company's policy. Without good logging, it will be next to impossible for auditors to validate that your model is secure and compliant.

Compliance attestation (SOC 2, HIPAA, etc.)
: Compliance attestation is an engagement in which an organization is evaluated for the quality of its adherence to a given law or regulation.[3] Preparation for such attestation and audit can be extremely time consuming and difficult, as it involves gathering and generating tons of information from different parts of your infrastructure. If you operate in a highly regulated industry, you should be thinking about the compliance of your ML deployments *as early as possible*. Otherwise, compliance failures could potentially delay your projects or waste resources—and making it to production without satiating these concerns is a massive PR and financial risk.

3 As explained in the Roosa CPA article, "Compliance Attestation".

Authentication
: For the purposes of an audit, you will almost definitely have to outline which users have access to your ML application, how they have access, and the level of their permissions. Mistakes like sharing passwords or accounts and universal admin permissions are huge security concerns and would be raised in an audit. Your platform should easily allow for the discovery of users, roles, permissions, and access logs. Since users in different departments may have different compliance requirements and data access rights, it's critical to understand how each person is accessing your ML models to ensure sensitive data is not being leaked.

  Authentication is *critical* and virtually impossible to build effectively by yourself. Having subpar authentication implemented by a data scientist with minimal authentication experience could potentially be less safe than no authentication at all. SSO integration, managing usernames and passwords, and role-based access controls are all specialized fields of computer science that require specialized software.

Setup Within Your Organization

As you can imagine, setting up an ML governance framework for the delivery and operations stages of the ML lifecycle is *significantly* more involved than it is for the development stage. ML development is relatively self-contained, serving mostly the needs of the data scientists. The governance components of ML operations have the potential to service your entire organization and need to integrate with an unknown number of different systems.

Many of these components are also highly specialized and technical. Unlike tracking and documenting models in development, things like security require dedicated teams of trained specialists that you may or may not have access to in your organization. In reality, a concrete implementation of this framework will be a combination of in-house, open-source, and enterprise solutions. Keep the following in mind:

1. *Break down work and set sensible goals.*
   ML governance is massive, and you won't be able to implement it in just one business quarter—or even a few. Take the categories above and come up with a step-by-step plan for building

out those features and requirements. Breaking down work into small, achievable pieces is a common tenet of Agile and effective in delivering more concrete work. High-level, abstract actions are less likely to move forward, often stagnating without a clear direction.

2. Buy where you can, build where it makes sense.
A big mistake data science teams make is to try to do everything themselves. If you have the support of a dedicated internal team, you're more likely to come out with a positive return on investment (ROI) and *significantly* less risk without overburdening your data science team. Organizations that silo their data science teams set them up for failure, instead of helping them bring out their value.

3. Locate domain experts within your organization.
While it's tough to guarantee support from other groups if your objectives aren't aligned 100%, you can usually rely on them for expertise and advice on the various requirements you'll need to meet both internally and externally. When you evaluate technology for areas outside your expertise, you'll need to make sure experts in your organization are involved in the conversation. The last thing you want to do is purchase a technology that ends up not meeting security requirements that you weren't even aware of.

4. Focus on integration as a primary feature of your governance technology.
There is a lot of great ML tooling out there but less tooling that focuses on governance and very little that focuses on integration as a first-order concern. Integration doesn't always sound important, but it's critical that anything you bring into your organization can fit with existing infrastructure. If it doesn't integrate, you also can't build and grow with it—and you become limited and restricted by the features of that platform. IT and ops professionals often look for integration as a primary feature to avoid becoming dependent or limited on a specific technology.

CHAPTER 4
Putting It All Together

We've established that companies are struggling to get value out of their AI and that a comprehensive ML governance implementation is how to achieve that value. While MLOps has become ubiquitous (even if only in name), taking it one step further to manage your MLOps with ML governance is the next stage of maturity for enterprise ML.

ML governance isn't nearly as much of a buzzword as "AI" or "MLOps," but it is by far the most important component for delivering value with ML. It is the final step to making ML a standard part of any organization. The first companies to implement standardized ML governance will have a once-in-a-lifetime opportunity to dominate ML in their business vertical.

Getting Value from Your ML with ML Governance

MLOps is the set of best practices and tools that allow you to deliver ML at scale. ML governance is how you manage those practices and tools, democratizing ML across your organization through nonfunctional requirements like observability, auditability, and security. As ML companies mature, neither of these are "features" or "nice-to-haves"—they are hard requirements that are critical to an ML strategy.

The value of a comprehensive governance implementation is inherent. Through effective management and controls, you unlock better,

faster, and more secure ML. While some governance features may sound vague or high level, they're components of software and ML alike. Governance drives:

- More *accurate* ML through observability and monitoring
- Shorter *time to deployment* through well-managed pipelines
- *Secure* ML through controlled processes and security tooling
- Less *incurred risk* through explainability and visibility
- Rapid improvement cycles

The best part about ML governance isn't that you'll deliver more value with ML. It's that you'll *continue* to deliver more value with ML at every step. The most essential part of governance is that it sets you up for long-standing, continuous improvement. Like Agile, governance is a framework for surfacing important information that enables iteration and improvement. Companies that implement governance won't win the AI race because it helped them deploy a model. They'll win because "governing" ML enables cyclic and organic growth and is essential in maintaining a competitive advantage in an era of rapid digital transformation.

ML governance is a framework for delivering value in many ways, but it's primarily a framework for growth. While there's no underestimating the importance of the ML models themselves, it's the framework that surrounds them that enables actual value. Without governance, no matter how much an individual model improves, the ML of your organization will not.

How to Set Up an ML Governance Program

ML governance is not an individual responsibility. Part of the reason ML ventures have been so challenging is that data scientists have been somewhat siloed in their efforts. This is not their fault, and data scientists have always continued to do what they do best—train effective models. However, organizations need to recognize ML projects as the massively cross-functional initiatives that they are.

Algorithmia's "2021 Enterprise Trends in Machine Learning" report found that a wide range of business units and roles are involved in setting ML priorities for organizations—from IT infrastructure and operations to data science to DevOps to product teams. To succeed

with ML, organizations must consider all the decision makers who will be involved, regardless of department or role. If your ML model will touch almost every unit of your organization, it makes sense that you would want each of those units represented for alignment and support. Most notably, at 57% of organizations, IT infrastructure operations leaders are responsible for setting the priorities for AI/ML, whereas a CTO or similar head of innovation set the priorities for AI/ML at 39% of organizations. If you want to develop a successful ML strategy in your organization, then it will require alignment across a number of business units and domain experts in your organization. Successful AI absolutely cannot be driven by data science teams alone.

It might come as a shock that the head of data science sets the priorities for AI/ML at only 31% of organizations—but actually, this exemplifies just how important the operations side of a project is. Any organization will likely have very particular guidelines for software that will also affect ML.

To make things harder, a lack of domain knowledge can make it difficult to communicate across business units. IT and infrastructure may not understand AI, and few data scientists have extensive experience delivering compliant software. When these groups come together to build an ML governance strategy, it can be difficult to bridge the gap and find common ground for alignment. Porting existing software policies to ML deployments may not be a one-to-one mapping and will require experience in both areas.

Setting up an ML governance program involves education, coordination, and effective communication. Every group expected to contribute must be incentivized (and remain incentivized) to create a robust governance framework. An organization needs to foster a governance-forward culture with clear roles, responsibilities, and value-driven goals.

Involving business leaders such as the CTO/CIO can provide AI projects with the necessary sponsorship to bridge these gaps and generate a culture necessary for the success of your governance strategy. Bringing in other essential groups like IT is crucial in removing governance roadblocks before they happen. Something holistic like ML governance cannot feasibly be created in a vacuum without the input of multiple business units.

Aside from the obvious data science team, any successful AI project will need key stakeholders involved from across the business.

Infrastructure and Operations

These individuals should be responsible for helping integrate and scale new technology in your organization. They will also be well versed in the governance concerns of software and the requirements for your AI application. Since you will need to meet all of the infrastructure and operations (I&O) requirements (which could be extensive), it's imperative to involve them as early as possible in the planning process. Meeting security or other governance requirements could have a significant impact on the scope of a project and the time it takes to deliver.

I&O is far less concerned with the performance of your model than they are about the technical implications and risk of introducing it into their production systems. I&O isn't as concerned with the statistical performance of your model as they are with the operational and compliance implications of introducing it into production infrastructure. Having a time investment and relationship from these individuals is absolutely critical in ironing out the details and requirements they'll inevitably need to impose on your AI solutions.

If you wait until development is almost done before finding out that your AI deployment will require role-based access controls and SSO integration, your project is likely to experience massive delays and incur unexpectedly high cost or risk. Involving these individuals during the planning phase of a project makes sure that groups are aligned and the effort can be estimated accordingly. Making them part of the process also helps foster the necessary relationships and ML governance culture required to keep individuals responsible, interested, and involved.

CTO and CIO

Although they are unlikely to be involved in the day-to-day, it's obviously important to have executive sponsorship to ensure continued dedication and support for a project. It should come as no surprise that CTOs are involved with ML decision making at more than a third of organizations. The involvement of C-level executives almost always increases the chance of success for a project.

While it's unlikely that someone at this level will be naive to the potential value proposition of AI, it might not be clear how important governance is in enabling that value. Getting C-level sponsorship for ML governance initiatives will come down to how effectively you can communicate the value of that governance. Common metrics for ML "success" include time saved or money saved. ML governance is a strategy to dramatically increase the ROI on ML initiatives through fine-grained controls and repeatability, and that is a strong motivator for value-driven businesses.

Head of Data Science

The leader of data science at a company is a must-have for setting the AI vision and keeping delivery on track. The responsibilities of this role may vary widely, but as the chief representative for data science across the organization it's important they create and manage the unified vision for AI delivery. Lack of unification will often lead to duplication of effort. Without a centralized messaging and approach, organizations can end up with multiple, incomplete ML workflows instead of extensible, integration-first platforms that are potentially reusable across multiple use cases within your company. The head of data science should shoulder this responsibility and play a key role in fostering an ML governance culture to achieve success as an ML-driven organization.

Other Business Team Members

The persona of other stakeholders in an AI project can vary widely from company to company. This usually includes SMEs, product teams, marketing experts, and others. AI initiatives require nontechnical, business-oriented representation to stay focused on the primary goal of delivering value for the company and contextualizing the purpose of AI in the first place. Data science teams need to align with the business on key success metrics for the project to ensure they are not delivering AI for AI's sake but to achieve a business-level goal.

How to Action on This Framework

The implementation of a framework for ML governance will depend on the state and maturity of your organization. Some organizations have robust, efficient pipelines but struggle to manage them, while others are still working to better automate the deployment stage of the lifecycle. Regardless of where your company is, here are five strategies for being successful in creating, driving, and delivering a holistic ML governance strategy:

Get C-level sponsorship.
: The fact is that C-level backing will always increase the chances of success for a project. If directors and above aren't convinced that they need ML governance (or ML), then it will be difficult to align this initiative to your team goals and get the investment you need from other key groups, like IT and ops. If your company wants to be an ML company, then key decision makers need to understand the long-term value and necessity of a comprehensive governance strategy for ML. The ability to bring ML governance into your organizational goals and all the way down to your objective key results (OKRs) will focus teams across your organization and guarantee commitment.

Bring in domain experts early.
: IT, ops, and platform professionals are not waiting around for the next project. Most likely, they have a long backlog themselves and will need ample notification to assist on any initiatives. If you don't bring domain experts in early, not only are you at risk of not having their time—but you may also completely miss key requirements for your ML deployments. If your company has specific security requirements for software, then you *need* to be aware of that ahead of time to estimate a project more accurately. Lack of effective notice and communication for key enabling teams is a common and avoidable cause of ML project delay.

Focus on transparency and communication.
: It sounds obvious, but miscommunication can lead to large lapses in initiatives. Make sure action items have clear due dates and owners. This is part project management and part effective communication. You can discuss ML governance all you want, but until ownership for each action is clearly defined, progress will likely remain slow. Implementing a governance framework

is about as cross-functional as it gets, and coordinating work across a number of teams requires excellent project management and communication skills.

Don't reinvent the wheel.
: Organizations will vary widely in their technical literacy and capacity. So, buy where you can and build where it makes sense. A common mistake data science groups make is doing everything themselves—from security to infrastructure. *This is a surefire way to fail.* Depending on where a data science team sits within an organization, they may not even have the direct support of software engineers or I&O professionals. If an enterprise solution solves 80% of your requirements and enables you to build the remaining 20%, then you want to heavily consider purchasing it. Top players in the ML governance space like Algorithmia are integration-first platforms. They take care of the boring, difficult pieces like security and monitoring so data science teams can focus on what they need to focus on—data science.

Remember the core tenets of ML governance.
: ML governance is MLOps management, a control plane for your MLOps. The largest features it delivers are control and visibility, which in turn generate growth. In some cases, governance is a bit abstract, and it isn't always as clear-cut as implementing a tangible feature like security. However, if you're driving initiatives that improve the data science experience and allow you to deliver quickly, then you're helping foster governance.

Conclusion and Next Steps

The framework for ML governance is a framework for managing your MLOps and getting value out of your ML. The next few years will see a stark contrast in companies that treat ML like a requirement, and those who treat it like a "nice-to-have." In Gartner's "Top 10 Data and Analytics Trends for 2021", the number one trend will be smarter, more responsible, and scalable AI. MLOps has dominated the ML space over the last few years, and ML governance is beginning to build on that momentum and drive companies into the next phase of enterprise ML adoption.

MLOps may be a buzzword, but that's because of the undeniable value of MLOps to drive real business impact. ML governance, the ability to control and manage MLOps, is the next big phase for companies as "ML" settles in as a permanent fixture of the businesses of 2021 and beyond. The advantage ML governance will give companies in the coming years cannot be overstated, and those who treat it as a "data science" concern rather than an enterprise concern will be left behind.

We need to start treating ML more like software and foster a culture of ML governance that allows for rapid, risk-free model development and deployment. ML is all about iterative cycles of improvement, and ML governance is about *dramatically* shortening that cycle so that your company can drive value faster than ever before.

We've outlined the components of a comprehensive governance strategy, but we also know that change at large companies doesn't happen overnight. Building and implementing an ML governance strategy requires a lot of work and, more than anything else, cross-functional coordination. Here are four ways you can get started today:

Start the conversation.
: You won't be able to implement ML governance if no one knows about it. Even if it isn't a sanctioned project, folks within your organization should be aware of the industry's move to governance and what they need to do to stay relevant. Start conversations with key stakeholders now, so that when the time comes, they're aware and ready to action on it.

Research end-to-end MLOps platforms.
: Whether or not you have a large, capable engineering organization, the fastest way to develop the technical parts of ML governance is to buy them. The enterprise space for MLOps has exploded, but there are fewer integration-first platforms focusing on first-class governance.

Gather statistics.
: Whenever you want to make changes within your organization, you'll have to make a case for it. Start gathering important metrics from recent reports so you can accurately showcase the value of ML governance to key decision makers in your organization.

Implement ML governance in your everyday work.

ML governance has a lot of technical components, but it also has a lot of components that just come from good software development culture. If you're involved in ML projects on a daily basis, do what you can to encourage the implementation of different ML governance framework components. Making these a standard part of ML projects at your organization is the first step toward a robust and complete framework.

For your organization, ML governance may happen gradually or it may seem to happen overnight. But if you want to generate value with ML, it *needs* to happen. Otherwise, you won't really be doing ML. The framework for ML governance is essentially the framework for valuable, reliable, and secure ML.

About the Author

Kyle Gallatin is a machine learning infrastructure engineer at Etsy. He previously held positions as a data scientist and machine learning engineer at the pharmaceutical company Pfizer. There, he cultivated an interest in "MLOps," and led the design and development of an end-to-end MLOps pipeline. In his current role at Etsy, Kyle is redesigning existing ML systems with a focus on ML model training, real-time model serving, MLOps processes, and model governance. Kyle spends his free time teaching and volunteering within the ML space. He also writes articles for technical publications on ML engineering, MLOps, and infrastructure on Medium.