DOMINO

WHITEPAPER

# The Practical Guide to Accelerating the Data Science Lifecycle

## CONTENTS

DOMINO

## Executive Summary

The data science lifecycle (DSLC) is a series of iterative steps to create, deploy and monitor an analytical model. While the need to accelerate and optimize the DSLC has always been important, it became absolutely critical for businesses in the wake of the pandemic, when model predictions rapidly diverged from reality. As one energy company executive put it, "We have our long-term forecasts around the amount of energy that is going to be in demand. COVID came and ate that for lunch."

And while the pandemic was the perfect storm of rapidly changing, never before seen behavior, the need to quickly build, monitor, and update models is always critical for delivering significant business impact. "Our pricing tool has delivered a massive impact on revenue," a chief data officer for a financial services firm told us. 'We estimate we've realized millions of pounds in additional revenue." If, like this executive, you understand how critical models can be to your business, you'll find the ideas and best practices in this white paper helpful for increasing the speed at which your data science teams deliver models for use by the business.

This paper takes a deep dive into the four phases of the DSLC, including insights for improving the sub-steps in each phase. It also includes best-practices checklists to serve as guides for heading off issues proactively and managing any that still arise, despite good planning and strategy. Finally, the paper looks at how adopting MLOps practices and technology unlocks productivity and removes friction from the DSLC.  This helps organizations scale data science so they can manage and deliver a portfolio of thousands of data science projects to become more model-driven businesses.

## What is the Data Science Lifecycle?

At its highest level, the data science lifecycle (DSLC) consists of four phases: Manage, Develop, Deploy, and Monitor. Within each phase there are multiple iterative steps through which a project progresses to ultimately generate  a  production model. The project may loop back through prior steps to get get more data, build new features, try a different algorithm, adjust hyperparameters, and so on. Even after deployment, when model performance decays a model will return to a prior step to be retrained or rebuilt.

## Part of an Ecosystem of Related Processes and Practices

Before diving into the details of the data science lifecycle, it's helpful to understand how it fits into the broader data, data science, and application development ecosystem of processes and practices.

The DSLC is the process followed by data scientists to build models. Similarly there are lifecycles followed by data engineers and application developers. In mature organizations there are tight connections between each of these processes because data feeds data science, which feeds applications. Data is both created and consumed by models and applications.

About 15 years ago, the application development community realized they needed to fix the shortfalls in the traditional application development process. Until then, applications were built in silos, with long build cycles, and delays, resulting in cost overruns, systems that didn't work as intended, and unhappy business stakeholders. DevOps was the result. It focused on building a culture of collaboration, shared responsibility, transparency about interdependencies in the process, quick feedback and problem solving, to accelerate application development. It encompasses the people, processes and technology involved with application development.

Not surprisingly, data management and data science teams also face challenges in getting high quality products efficiently out the door, let alone at scale. As these fields have matured, "Ops" practices have appeared to do the same for data engineering (DataOps) and data science (MLOps). The goals are similar, speed, scale, and quality of the final product delivered to the business. However, the details of each practice differ slightly as they apply to different domains.

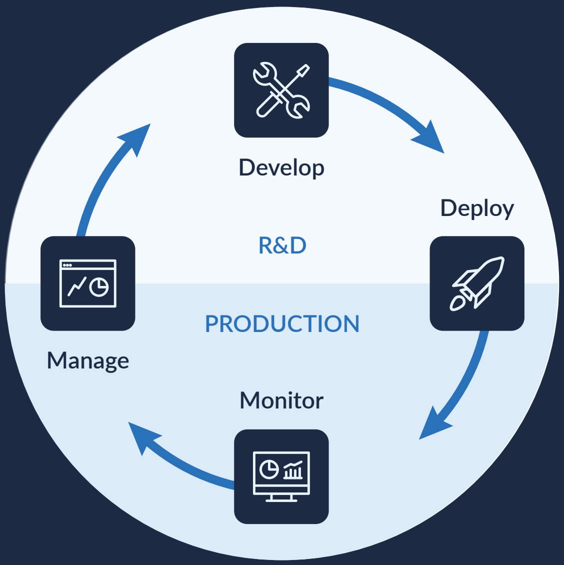
**MLOps** is still an emerging discipline focused on removing friction from the DSLC, improving collaboration across teams, capturing artifacts, and instilling common patterns and practices across the enterprise. When done well it increases productivity and **model velocity** (a measure of how quickly and effectively you can iterate across the entire lifecycle of your models). As a result, data science teams are able to build, deploy and monitor high quality models at a scale previously unattainable. In fact, a **2022 study by BARC** found that the introduction of DataOps and MLOps in an organization makes them 4.2x more likely to be able to deploy models quickly (within days or weeks). More details on key MLOps capabilities that accelerate the DSLC are included at the end of this paper.

**There are many different project frameworks for data science, notably CRISP-DM. Our data science lifecycle builds on decades of experience with data science gathered since the development of CRISP-DM and other frameworks, and is based more on practical realities we've seen working with leading data science organizations across the globe.**

## The Data Science Lifecycle

At its highest level, we break the data science lifecycle into four phases, each containing multiple steps: Manage, Develop, Deploy, and Monitor.

These phases and steps are highly iterative and a project may repeat one or more to improve the model both before deployment and after. Multiple people and teams are involved in the various steps including IT operations, business stakeholders, data science leaders, data scientists, data analysts, developers, data science product managers, ML engineers, and data engineers.



### Manage

Business problems are identified. The problems are scoped, prior work reviewed and potential data sources identified. Working collaboratively with stakeholders, projects are prioritized, including updates to existing models. KPIs and high level ROI estimates are identified.

### Deploy

Models are registered into a central repository and tested to ensure they will perform as expected when in production. All model artifacts are retained and cataloged. They are approved for deployment through model governance processes, and integrated into a system or process for use. Business adoption is assured through change management.

### Develop

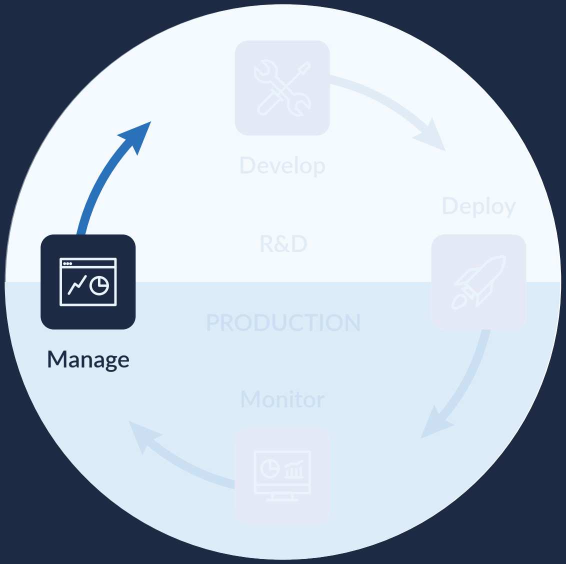
Data is accessed and prepared, then features are created. Multiple models are built, evaluated, and documented. When a final model is selected, it is validated to ensure it solves the business problem and is technically sound.

### Monitor

Models are continually monitored for performance and drift, and action taken when performance degrades. Business value realization is captured and communicated.

## Manage Phase

Some of the most important work in the data science lifecycle happens before a line of code is written. If done well, the Manage phase dramatically de-risks a project by driving alignment across stakeholders. It also ensures that business value is maximized from data science activity by focusing work on the business' priorities. This phase is where the business objective is identified, success criteria laid out, prior artifacts are reviewed, and initial ROI calculations are performed.



### Manage Phase Steps

1. Project ideation and feasibility assessment
2. KPI identification and business value estimation
3. Business adoption planning
4. Model requirements documentation
5. Project prioritization

### Project ideation and feasibility assessment

A business problem is identified that can potentially be improved with data science. Rather than just immediately tackling the project, the team should take time to assess the feasibility of the idea, including researching prior work and scoping the effort.

There are many aspects to investigate before starting a project. They include but are not limited to:

- Has prior work been done on this topic? If so, what was its outcome?
- Who are the business experts and stakeholders for this project?
- What data is available? How will we access it?
- Does the available data have enough signal to be predictive?
- What is the existing business process and will it be feasible to change it to incorporate the output of the model?
- What are the business' timelines for the work to be completed?
- How do we need to deliver the results of the project? Is it a one-off project? An app? A dashboard? An API that will be called in another system?
- Do we have staff with the right skills and right technology to take on this project? Do they have capacity?
- Are data science methods available that can tackle this class of problem?
- How will this work in production? Are there constraints (such as data update frequency or system limitations) that we need to consider?
- What are the risks of the project? How can we mitigate them?
- Are there ethical or fairness considerations that we need to be mindful of?

## Business value estimation and KPI identification

The ability to estimate the potential business impact of a change is one of the best predictors of success for a data science team. However, it's easy to get pulled into trying to exactly predict the ROI of a project. This level of effort isn't necessary at this point in the project. Instead, the goal should be high-level, orders of magnitude math to aid in the prioritization process.

**The KPIs the project is designed to impact** should be identified and agreed upon up front. This will aid in tracking the actual impact of the model over time and ensure that it delivers what is expected. These KPIs may be leading metrics (e.g., the volume of recommendations), lagging metrics (e.g., the number of conversions from recommendations), or usage/health metrics (e.g., how many call center reps are using the recommendations).

A large insurer asked, "If we reduce fraudulent insurance claims by 1%, how much would we save?" Then they asked, "What is a conservative estimate of how much improvement we can expect by the data scientist's efforts?"

They considered all of the project costs: time spent by the data science team, potential data acquisition costs, computing resources, implementation time for IT, and training/adjustment time for stakeholders. Finally, they settled on a rough and conservative estimate based on past experiences.

## Business adoption planning

A model delivers no value until it is used in the business. To avoid wasted work, the change management effort and business process change needed for adoption should be identified, planned for, and incorporated into the overall prioritization of the project. It is critical to work with business stakeholders to understand the magnitude of the change during planning.

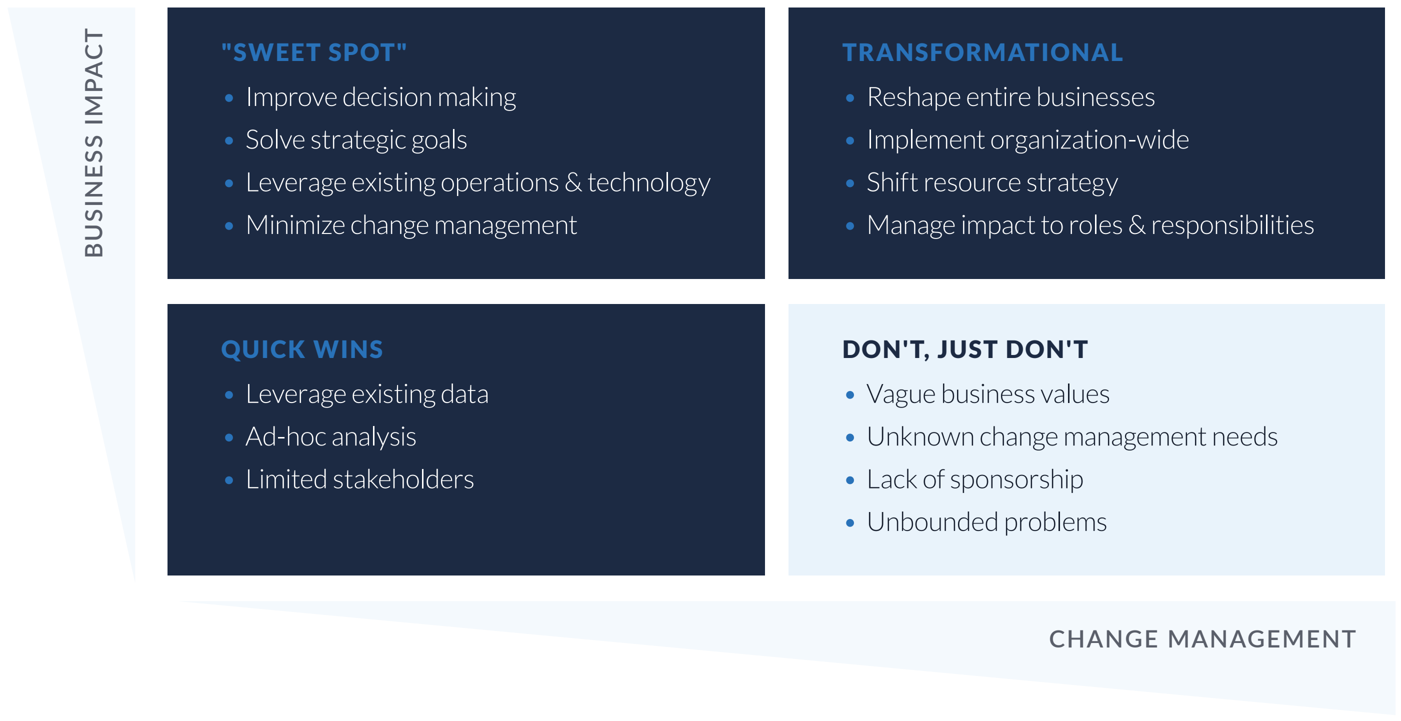
## Model requirements documentation

The project requirements should be captured in a standard Model Requirements Document (MRD). Key components of a good MRD include: problem statement and objective, target KPIs, estimated ROI, target data sources, known risks, relevant stakeholders, prior artifacts, and a stakeholder-centric timeline that includes change management activities to drive business adoption.

## Project prioritization

Data science projects need to be managed and prioritized through a stakeholder-inclusive process. Ultimately data science serves the business and the work underway and in the queue should support the it's strategic priorities. The prioritization process should be agile enough to adapt to changing business conditions. Business stakeholders should be able to see what is in flight and what is in the backlog. That said, some team bandwidth should be for pure research to try out new ideas, explore the art of the possible, and potentially uncover something completely unexpected.

As new projects are identified, they should be prioritized based on a set of agreed-upon criteria. An example of a prioritization matrix is given below. It should be expected that some proportion of project ideas end up never making it onto the project backlog.



MANAGE PHASE

## Best Practices Checklist

- ☐ **Use case first, not data first.** Use case first, not data first. Look for the use cases that lie at the intersection of important business value with large opportunities for improvement, and rich, promising data sets. Ask "is data science the best way to solve this problem?"
- ☐ **Map existing processes.** Documenting the current business process ensures that data science and the business are speaking the same language and specific improvement points are clearly identified.
- ☐ **Practice and master 'order of magnitude' ROI math.** Learn to make high-level back-of-the-envelope estimates based on documented assumptions.
- ☐ **Maintain a repository of past work.** Being able to search and reproduce past work (e.g., data, packages, code, results, discussions) provides an invaluable head start in a project and ensures that your IP is retained regardless of staff turnover.
- ☐ **Maintain a hub of business domain and technical experts.** No one can be an expert in everything. As a team grows, keep track of who has specific skill sets and knowledge.
- ☐ **Create and enforce templates**. Documentation is often viewed as a chore. Make it easy with templates suitable for the majority of cases, and enforce their usage.
- ☐ **Bring in IT and engineering stakeholders early.** Make sure technical stakeholders are part of the team early on so they understand the vision for the project and you understand technical constraints.
- ☐ **Create mockups of different deliverable types.** End users may not know the best way to consume results. Try out different mock-ups showing different approaches to let them identify the most user-friendly way to leverage the results. This may also uncover hidden requirements.

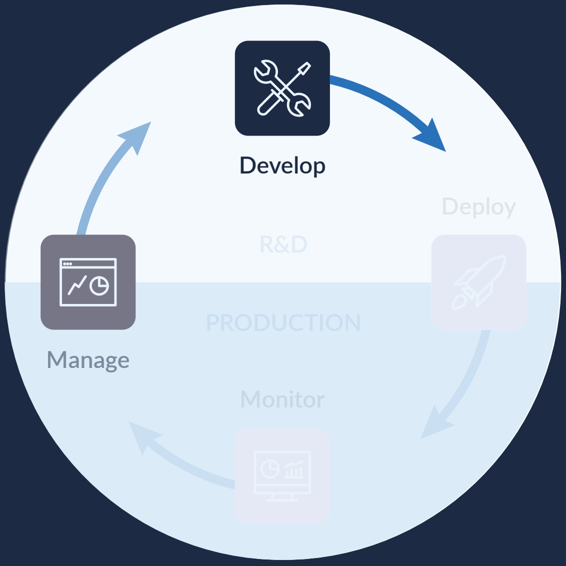
"Reducing hard drive dispatches by just 1% translates into an $8 million gain for the company."

Global computer technology company

DOMINO

## Develop Phase

The majority of data scientist's time is spent in the Develop phase, acquiring data, preparing features, trying algorithms, hyperparameter tuning, validating models, and documenting the project. Many approaches and many data elements should be tried before a final model is selected to move into production.



### Develop Phase Steps

1. Data acquisition and access
2. Data labeling and annotation
3. Feature engineering
4. Algorithm selection
5. Hyperparameter tuning
6. Model validation
7. Documentation

### Data acquisition and access

Making it easy for data scientists to access data sources in a governed manner within an organization is critical for several reasons:

- It dramatically decreases the time spent preparing to build models.
- It enforces data security and governance practices. Without easy, consistent access mechanisms it's easy for one-off workarounds and data replication to occur, increasing risk to the organization.
- The lineage of the data is easier to capture. This is critical not only for governance but also for reproducibility.

We are seeing teams evaluating **alternative datasets** to better understand their customers, such as social data or location data. Organizations that have streamlined the process of vendor selection, data evaluation, procurement, and ingestion eliminate a key bottleneck that can slow down model development. This often requires coordination with procurement, legal, IT, and the business to agree on a process.

### Data labeling and annotation

Data labeling identifies raw data (e.g., images, audio files, text files, videos) to provide the critical information needed to train deep learning image classification, natural language processing, or speech recognition models. For example, labels might include words spoken in an audio clip or whether an image or video contains a specific characteristic (tumor, bridge, bird, etc). Data labeling  is equally critical for machine learning applications machine learning applications - for example when you are trying to predict certain outcomes such as fraud, records with that outcome have to be labeled accordingly.

Data labeling can be challenging, particularly if an organization is trying to do it manually. Quality and consistency of labeling is critical because the model will learn from those labels. Scalability is also critical, and many organizations are turning to technology to assist with this process.

### Feature engineering

Data frequently has to be transformed in some way to deliver analytical value. Those transformed data elements are referred to as features. Features may be created because a specific algorithm requires data with specific characteristics or it may be because you need the data to contain different information or be in a different format than in its raw form to reveal its predictive potential. For example, the distance between two geographic coordinates may be predictive in a way that the two sets of geographic coordinates are not on their own, and vice versa.

When building features it is very important to understand whether the feature can be recreated for use in inference. No matter how powerful a feature is, if you can't recreate it for use by the model in production, it can't be included in the model. Also, make sure to understand whether there are any regulatory constraints in your industry for using certain features.

Feature stores are a popular way to capture the systems and workflows that transform data into features used in model training and inference. Creating features is labor-intensive, so making it reusable and consistently available is critical for scaling data science and increasing model velocity.

### Algorithm selection

Selecting the right algorithm requires balancing factors such as accuracy, explainability, training time, deployability, and performance. Complex algorithms with hundreds of features may be very accurate, but they may take unfeasibly long to train, be difficult to deploy, and have unacceptable delays during inference. Simpler algorithms may be easier to explain, train quickly, and deploy, but may not perform as well. For a given project multiple algorithms (and features) will be tried out before the final model is selected.

In many cases, getting a simpler model into production faster is preferable to working to eke out a bit more lift with a more complex model. For example, one company spent weeks engineering hundreds of features. Then they learned that many of them were either, a) not collected in real-time so couldn't be used in the target use case, or b) not allowed for compliance reasons. They ended up using a simple model with five features and then worked with their IT team to capture other data in real time for the next iteration.

### Hyperparameter tuning

A hyperparameter is a parameter of the model whose value influences the learning process and whose value cannot be estimated from the training data. Examples include branches of a decision tree or the number of clusters in a clustering algorithm.

**Hyperparameters are 'tuned'** one of two ways:

- **Manual**: Hyperparameters are set by experimenting with different values to yield the best model performance.
- **Automated**: The optimal set of hyperparameters is found using an algorithm that runs trials based on user defined ranges or combinations.

Hyperparameter tuning will result in many different iterations of models (experiments). Its important be able to track, search, group, sort and filter experiments, to make it easier to find specific past results and view experiments' performance in granular detail, while putting those experiments in broader context of the ideas they were tested against. This capability also makes it easier to share results with others on the team for feedback and learning.

## Model validation

Validation needs to occur both as each model variation is developed and then again when a final model is selected. As different experiments are done, the candidate models need to be evaluated. It is more than just a code review. They need to be evaluated for data assumptions, code base, model performance, and prediction results to provide confidence that the model can reliably improve business performance. It's also important to look for ethical and bias issues during reviews of each experiment (and throughout development efforts).

Once a final model is selected, further rigorous validation occurs before it is published. Validation should always include peer review by other data scientists on the team. In addition to spotting issues, it also helps with learning and knowledge capture. If you are working on a model where regulatory compliance and risk management are factors, model explainability is also important to make clear what the model is doing and how. It's critical to expand validation efforts beyond just the math to include issues such as whether it will be able to meet latency requirements in production. Ultimately, the goal is to get final sign-off from stakeholders such as the business, the model validation team, IT, and, increasingly, legal or compliance. And of course, if a model is changed or repurposed, it will need to be re-validated.

## Documentation

All aspects of a model version (e.g., code, data sets, models, hyperp-arameters, pipelines, third-party packages, configuration settings) need to be captured and documented in a central knowledge repository. This ensures that work is not duplicated in the future, it can be learned from, reproduced easily and decisions are captured. It also provides comprehensive documentation for regulatory, audit or compliance reviews.

Even if a project yields no material uplift and doesn't get deployed into production, it's critical to document it and preserve it in the central knowledge repository. Too often, data scientists redo work due to a lack of visibility into previous experiments.

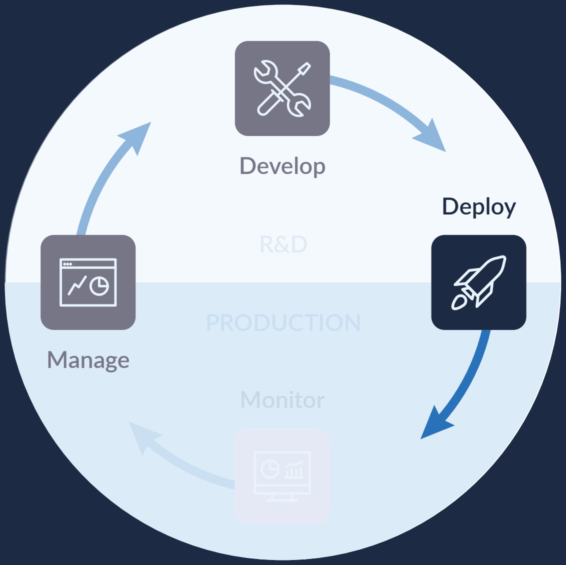
DEVELOP PHASE

## Best Practices Checklist

- ☐ **Check-in with stakeholders about the data.** Stakeholders often have really good knowledge about what features will matter. They can also potentially explain data irregularities.
- ☐ **Make data used in the model will be available in production.** Its critical to check that the data used to train the model will also be accessible in production, either because it is already there or because you will be able to get it ingested into your production environment as part of the project.
- ☐ **Publicize available data and features.** Make sure the effort invested in data acquisition and feature creation pays off by publicizing data sources and features to data science teams.
- ☐ **Demonstrate incremental value.** Share insights early and often. For example, one leading tech company has its data scientists share an insight every 3-4 days. If they can't publish a short post on incremental findings in business-friendly language, then chances are they are down a rabbit hole. The insight can be as simple as "this experimental path didn't pan out."
- ☐ **Track KPIs consistently over time.** Leading teams ensure that the relevant KPI is never far from their experimental flows. One hedge fund tracks the overall performance of its simulated investment portfolio across hundreds of experiments and then shows this to its Investment Committee as proof of data science progress.
- ☐ **Capture the full project record.** Reproducibility is very difficult if only some aspects of a model are captured. Data scientists, auditors, validators, and regulators will all benefit from comprehensive documentation.
- ☐ **Use automated validation checks to support human inspection.** There are often repeated steps in a validation process that can be automated. That may be a set of summary statistics and charts, a portfolio backtest, or any other step that could be turned into an automated diagnostic. This lets human validators focus on the critical gray areas.
- ☐ **Maintain a record of discussions in context.** Don't use email or Slack for capturing discussions and decisions. Capture them in a central location associated with the project.
- ☐ **Where ever possible, require explainability.** Data scientists should be able to understand and clearly explain the their modeling strategy before implementation. Besides limiting unanticipated consequences of deploying a model, being able to offer interpretation as to why the results of a model are what they are and what input variables are the most significant is essential for successful model adoption.
- ☐ **Where appropriate, check for bias and other ethical considerations.** Not every application of data science has bias or ethical considerations. But for those that do, its important to pay attention as your company's reputation and bottom line as well as people's well being is on the line. It's easy for **existing harmful bias in your data**, that you may not even know exists, to carry through to models built on that data.
- ☐ **Make appropriate compute easily available.** Significant productivity gains and model accuracy improvements are generated when data scientists are able to easily access the right compute for the job. This includes GPUs and distributed compute frameworks such as Spark, Ray and Dask.
- ☐ **Provide choice of languages, IDEs and frameworks.** Data scientists bring different skillsets to the table and different projects require different approaches. Making sure your team has access to a wide variety of governed tools will improve their productivity and ability to creatively solve business problems.

## Deploy Phase

In the Deploy phase, models are put into use by the business. This so called 'last mile' is frequently the most difficult to accomplish, let alone quickly. As a result, finding ways to accelerate, standardize and remove friction from this phase of the lifecycle is critical to achieving high model velocity. In this phase, models are registered into a central repository and tested to ensure they will perform as expected in production. All model artifacts are retained and cataloged. Models are approved for deployment through model governance processes. Finally, they are integrated into a system or process for use. Business adoption is assured through change management.



### Deploy Phase Steps

1. Publishing
2. Pre-deployment testing
3. Approval for deployment
4. Operationalizing
5. Business adoption

DOMINO

Not every model will need every step depending on the industry and use case. This is because deploying into production can mean many things. It can be as simple as publishing the results as reports or dashboards. It may be incorporating the modeling pipeline into batch processes (e.g., new direct marketing campaigns), or deploying models into production systems for automated decision making (e.g., online credit decisions or product recommendations). Each requires a different level of testing, approvals and integration.

### Publishing

The model is packaged up and put into a central model registry where it is staged for easy access. The registry in effect bridges experimentation and operationalization of a model. It ensures that everyone in the process knows what version of the model is to be used and all the necessary components for deployment are in one place. The status of the model (e.g., staged, deployed, retired) can be tracked easily. The registry can serve as a point of connection to broader CI/CD processes where as updates are made and published to the registry they are then deployed as part of an overall automated training pipeline.

## Pre-deployment testing

The model is tested in a non-production environment to make sure it performs as expected. This should include testing that the input data aligns with the data used to develop the model, and that the integration assumptions are accurate. The infrastructure the model will run on should also be tested to ensure it can handle anticipated loads within the latency requirements set by the business. The output of the model should be tested in the selected endpoint (e.g., existing application, decisioning workflow, app) with business users to ensure it is delivering as expected.

## Approval for deployment

Once all testing is complete and documentation is complete, the model needs to be approved for deployment/use. In regulated and/or mature organizations this is typically done through a formal model governance process.

## Operationalizing

The model is placed into use on production infrastructure and made available to the business for use in day-to-day operations. For an app or dashboard, the process to put the model in use can be very streamlined and requires little to no engagement with broader DevOps processes. However, when the model is part of broader systems and processes, it will be operationalized as part of standard IT processes such as CI/CD.

Depending on the situation, a progressive delivery approach may be appropriate where a new model is tested by gradually introducing traffic to it alongside the existing model. That way performance can be monitored before it is released to take over all traffic from the prior model.

## Business adoption

For models to deliver value, they need to be adopted by end users. This will probably require changing how they do their job, at least slightly. It might also require learning new skills. Making the change will require users to trust the results, which might be difficult if they don't understand data science or trust the underlying data.

To ensure business adoption, it's critical to have relationships in the business, bring them along on the journey to build the model, and agree on how a model will fit into the business process early on. This is where where the Data Science Product Manager role is critically important as they will build and nurture these relationships. Providing mock-ups and getting feedback early will help create ownership and excitement (as described in the Manage section previously).

Successful adoption will require planning for change management and resourcing it appropriately. As with all change projects, building relationships and getting buy-in early, explaining the why, finding a champion, showing the positive impact, taking feedback throughout the project, providing training, and quickly responding to any issues or concerns are key components to the change management process. For example, a leading insurer has a dedicated role that helps train business units on how a model works and why they did it. They also act as a single point of feedback for future iterations on the model. This has dramatically increased adoption across non-technical stakeholders, such as claims adjusters.

DEPLOY PHASE

## Best Practices Checklist

☐ **Avoid recoding models whenever possible.** Recoding models to put them in a final state for consumption adds time, complexity, and the potential for error. Instead build them on a platform that allows them to be easily packaged for deployment regardless of what language they were built in or where they are going to be used.

☐ **Enforce a promote-to-production workflow.** A predefined workflow reduces both bottlenecks during production and risks. It should include determining how much human inspection of early results is necessary for staging, as automated testing may not be sufficient. It should also include any governance approvals.

☐ **Flag upstream and downstream dependencies**. A model is most risky when it finally makes it to production. Ensure that you know the upstream dependencies (e.g., what training data was used, what transformations were done with what tools,what modeling packages were used). Also make sure you know the downstream dependencies (e.g.,the nightly batch model is stacked on another model).

☐ **Invest in change management capabilities.** Change is hard. Plan for more time and effort than you think you will need, particularly for early projects and new areas of the business.

☐ **Use a model registry.** Capturing metadata about a model both creates a central governed location for everyone to access information about a model (version, status, code, environment variables, etc.) and provides ways to automate certain tasks. Depending on your industry or use case, a registry can be an integral part of model risk management processes.

☐ **Leverage a feature store.** Feature stores eliminate the need to recreate data transformations as a model is moved into production - which can at a minimum slow model deployment and at worse prevent it. A feature store also makes sure features are consistently available for inference.

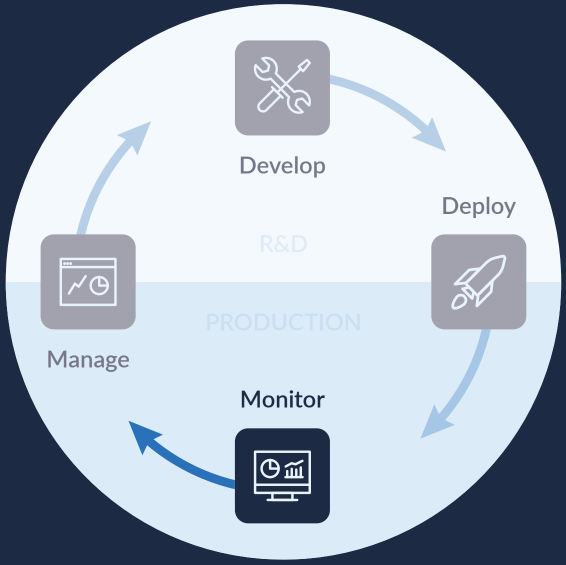
☐ **Capture lineage.** Make sure you capture full **lineage** for all deployed models (and ideally any that never make it deployment for future reference) for audit and reproducibility purposes. This will save a lot of time when you have to explain what the model does and what was used to create it. It will also help when a model needs retraining as you know everything about the model in its current state.

☐ **Create "Model cards".** Creating a simple overview of a model that can be easily understood by non-data scientists can go a long way towards transparency and understanding. They can capture information such as what it is, what data it uses, what it does, its purpose, and whether it has limitations or weaknesses as well as its strengths.

## Monitor Phase

When models are "live" they finally provide business impact. But that impact can degrade over time for a variety of reasons: changes to products or policies can affect how your customers behave; adversarial actors can adapt their behavior; data pipelines can break; your systems can change, and sometimes the world simply evolves.

In the Monitor phase, models are tracked for changing performance - statistical, semantic and operational - to quickly react to degrading performance. This phase is also where business impact is captured and communicated to stakeholders.



### Monitor Phase Steps

1. Performance monitoring
2. Alert generation
3. Capture business impact
4. Communicate impact

### Performance monitoring

All models have errors and these rates of errors will shift over time. As a result, you need monitoring to detect and measure the shifts in these errors over time, so you know when to retrain a model. Understanding model performance data is not trivial. It requires it requires both statistical expertise and business understanding, which requires the involvement of data scientists and sometimes the business. Models also need monitoring for operational performance, which requires the involvement of IT and/or data engineering.

When **model performance degrades** it's important to catch it quickly. Optimally, monitoring will be automated, continual, and model-specific KPIs. Critical areas to monitor include:

- **Data Drift:** When the patterns in production data start diverging away from the patterns present in the model's original training data, a model can lose predictive power. Data drift can happen for various reasons including a changing business environment, evolving customer behavior and interests, modifications to data from third-party sources, data quality issues, and even issues in upstream data processing pipelines.

  Catching these changes quickly is critical. For example, a financial firm issuing credit cards expected nulls in their probability of default model if an applicant was not present in the database (typically a sign of elevated risk). The credit agency changed to reporting "no hits" as a "999" code, which was interpreted by the model as a high credit score, leading to a surge in approvals and millions in loan losses.
- **Prediction drift:** Prediction drift is a change in the distribution of the outputs of the model. For example, suddenly a model starts rejecting an unusually large number of people for loans. Usually, you get this when there is a long lag in getting actual outcomes and so monitoring them is too late (e.g., you don't see whether someone defaulted until months later).
- **Concept drift:** Concept drift occurs when the expectations of what constitutes a correct prediction change over time even though the distribution of the input data has not changed. For example, loan applicants who were considered attractive prospects last year (when the training dataset was created) may no longer be considered attractive because of changes in a bank's strategy or outlook on future macroeconomic conditions.
- **Performance:** Latency and uptime should be tracked to ensure the model is meeting agreed upon service levels.

Ideally monitoring for all models is captured in a central "pane of glass" that gives all interested parties a holistic view of how all models in the portfolio are performing, regardless of how it was built and where it is deployed.

**"We have our long-term forecasts around the amount of energy that is going to be in demand. COVID came and ate that for lunch."**

Director of Analytics, large power company

“ Our pricing tool has delivered a massive impact on revenue. We estimate we’ve realized millions of pounds in additional revenue. ”

Chief Data Officer, financial services

### Alert Generation

Monitoring activities should include proactive alerting that notifies data scientists and other stakeholders (such as the business and IT) when a model no longer meets performance thresholds. Organizations may want to set up automated workflows that trigger retraining when an alert occurs. Alternatively a manual review can occur to determine what action is appropriate to take.

### Capture business impact

Tracking KPIs and using techniques such as hold out groups, capture the **value of the improvements** the model has made in business processes. Monetary business value accrues from the model impacting factors such as staffing costs (impacted by productivity increases or time savings), operational costs, and revenue. Business value may also accrue from intangible factors such as employee satisfaction, brand enhancement, better customer experience, and skill acquisition. These are harder to estimate but it is possible, especially if standards are set for how to estimate the savings. Business impact of all these factors should be captured on an ongoing basis.

Another aspect of capturing business impact is to look at the relative costs/gains for various levels and types of drift. It's possible a small level of drift has a high business impact. It may be large drift has no business impact. Understanding the impact of changes in model performance can help determine when a model needs to be updated or replaced.

Capture the costs of projects as well, so as to create a true ROI measure. Costs include factors such as staff time, data acquisition costs, infrastructure, storage, and licenses. Some are easier than others to measure, so like with the intangible benefits, estimates may be needed.

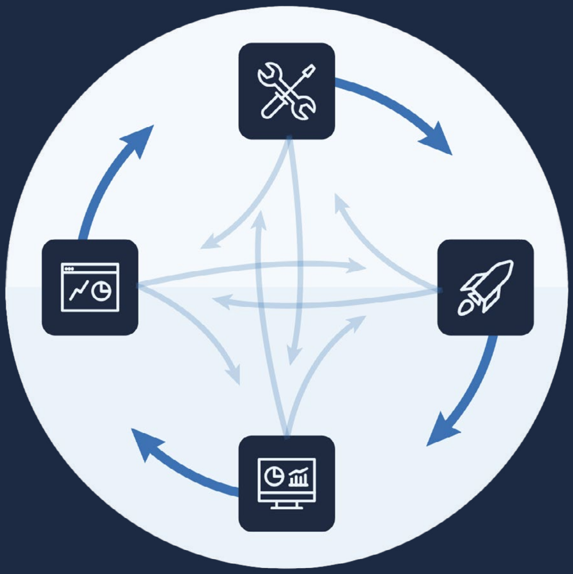
### Communicate impact

Share the impact of data science initiatives with stakeholders regularly. This will not only support further investment, it can provide guidance in terms of expected returns for future prioritization of proposed projects. Sharing the impact of data science programs can also generate positive press for the company and improve your ability to attract and retain data science talent by showing how your data science program is delivering high value with interesting projects.

MONITORING PHASE

## Best Practices Checklist

- ☐ **Acknowledge that model monitoring takes different tools and skillsets than traditional software applications.** Some metrics are the same and are governed by SLAs, but there are more data, methodologies, and statistics with model monitoring that fall outside the skills of IT/Ops.
- ☐ **Centrally monitor all models in production.** A centralized approach breaks down silos and enhances consistency and responsiveness. You will also get a better sense of what models are in use, by whom, for what, etc. And, you will be in a better position to prioritize resources and retraining across your model portfolio.
- ☐ **Establish a consistent set of metrics for assessing model health.** With standard definitions, everyone will be able to quickly and consistently assess the health of a model.
- ☐ **Assess trends in model health.** Model health is more than just a snapshot in time. You need to see how model health is changing over time to find inflection points, pinpoint big changes, see improvements, and so on.
- ☐ **Automate monitoring as much as possible.** Automation brings numerous benefits including consistent intelligence, reliable notifications, and more time for data scientists and ML Engineers to focus on value-adding activities.
- ☐ **Set up a monitoring plan with notification thresholds and agreed upon actions.** Proactively codify knowledge of errant behavior into your monitoring activity and have specific actions that will be taken when those boundaries are crossed. As part of the plan, make sure the definition of when to retrain or rebuild a model is agreed upon by all stakeholders.
- ☐ **Set up policies for how a model can be used.** Acceptable use policies head off risky behavior, such as a new department using a model in a way it wasn't intended. For example, a model that was built to approve home mortgage loans specifically in California should not be used in Arkansas.
- ☐ **Hold out a control group in production.** By comparing the results of the control group to those used by the model, the incremental lift from the model can be clearly demonstrated. For example, an organization established a global holdout group from all of its customer segmentation and price elasticity models. After a year, they compared the average revenue from the holdout group to the customers whose experiences were guided by the predictive models. The overall lift was more than $1 billion, which gave them the credibility to dramatically expand the team and push models into more steps of the customer journey.



## The Iterative Nature of the Data Science Lifecycle

When model performance decays it is critical to quickly assess whether the model needs to be retrained (updating the parameters), rebuilt (new algorithms, new features), or replaced (retired and a new model put in its place).

At this point, the model will return to some earlier step in the DSLC. It may happen automatically through retraining pipelines that are triggered at certain thresholds. It may be manual where the data scientist looks at the model performance, does some investigation, and then makes a decision about what to do next. They may want to create additional features or acquire additional data. The model may just need to be retrained on newer data. New models may be evaluated and the existing model retired.

If models are not put through an automated retraining process, then the work to improve the model should be prioritized against other data science work in the backlog (as described in the Manage phase of the DSLC) to maximize business impact from available resources.

Returning to earlier phases of the DSLC doesn't just happen when a model needs retraining. At any point in the lifecycle a project may return to a prior step before moving forward again. More data may be needed. A different set of algorithms tried. A model may fail on testing and need to be reworked.

Due to the highly iterative nature of the DSLC, it's critical to focus on achieving high model velocity. Moving through the various steps at high velocity means more models get into production and kept in peak form, leading to higher business impact.

## MLOps Capabilities that Accelerate the Data Science Lifecycle

MLOps practices, enabled by a platform such as Domino's Enterprise MLOps Platform, make it easier for data scientists and others involved with the data science lifecycle to do their work, improve productivity, reduce risk, and shorten the time it takes to move models into production. The following MLOps capabilities that combine people, process and MLOps platform technology accelerate the entire DSLC, increasing model velocity.

### Self service tools and infrastructure

Nothing will bring data scientist productivity to a halt faster than having to wait for IT to provision a workspace, having to take on DevOps challenges themselves, or sit around waiting for a job to process on insufficient infrastructure.

Today's large, complex datasets require easy access to distributed software frameworks (e.g., Spark, Ray, or Dask), compute clusters, and GPUs, to have the processing power needed to solve complex problems and quickly test ideas. This is particularly true for organizations wanting to do deep learning, computer vision, or text analytics.

Rather than having to wait or struggle with provisioning, data scientists should be able to quickly provision a workspace through a MLOps platform that has the needed compute and memory, including CPUs, GPUs, and distributed compute frameworks, that autoscale to their needs. The platform should also be flexible enough to provide a wide choice of governed data sources, languages, packages, IDEs, and other tools for data scientists to use in their work.

A flexible and open MLOps platform is especially important given how fast the data science tools ecosystem is growing and evolving. It's critical to quickly test and adopt new innovations to both enable data scientists to undertake a wide range of projects effectively, but also to accommodate a wide range of skillsets. This will enable better hiring and retention, but also allow collaboration between teams that prefer different tools.

A MLOps platform with these capabilities also dramatically reduces the **IT support burden, and improves cost control, governance, and security.** These are key issues for IT organizations, and they are very difficult to manage when data scientists are working in bespoke environments that they tinker with themselves. Additionally, a MLOps platform shouldn't lock the organization into being on premise or in a specific cloud to maximize flexibility, cost control, and ability to easily comply with data locality requirements.

### Standard patterns and practices

Data scientists often think of their work as bespoke and highly specialized. While their skill set may be, there are often many intermediate and final artifacts they create that can and should be documented and reused such as software packages, modeling datasets, and features. Moreover, on their own few data scientists document their development process, much less abstract and modularize the process.

By adopting and enforcing a set of common patterns and practices across the DSLC you reduce variability across projects, reduce onboarding time, improve governance, and increase reusability of work. Less reinventing of the wheel occurs. Ultimately this all leads to higher model velocity, increased productivity, and more models moving into production.

**“ What I love most about Domino is its simplicity. We are able to get our data scientists up and running on very advanced hardware tiers and compute environments without them knowing how this is working. ”**

Soren Pederson, Data Scientist, Topdanmark

“ The less time and effort we have to spend relearning or recreating results, the more time we can spend creating additional value, and the more projects we can support. ”

Chief Data and Analytics Officer, large insurer

## Project management

When you only have a couple of projects in flight, status tracking isn’t complicated. But as it scales and silos form, it's impossible to track all the work in flight, eliminate blockers, reprioritize work, and keep stakeholders informed of status without a central means of managing the portfolio.

A central project status tracker or dashboard that includes where it is in the lifecycle, status (e.g., green/progressing, red/blocked, gray/done), and discussion capabilities provides a self-service way for data science leaders, IT, business stakeholders, and data scientists to stay up to date on projects. This frees up management time, because data science leaders aren’t having to chase status for reporting. It focuses attention on issues that need resolving so the team can quickly take action. Data scientists don’t have to manually report status in a variety of trackers. The business stays engaged and informed on the project. The net results is that projects move faster, communication and buy-in are assured, and productivity improves.

## Central repository

As stated previously, documentation is never a favorite activity. But neither is undertaking an archaeology project to recreate a model that takes months and many resources. A central repository as part of your MLOps platform that automatically captures all model artifacts and documentation, ensures everything is comprehensively captured in a single location for every project. It simplifies searching prior work inorder to pick up where someone else left off, reproduce work, and accelerate onboarding.

Exact reproducibility is possible when you capture all artifacts (exact packages, environment, code, data, IDEs, libraries, etc.). It is the foundation for updating a model, complying with audit requirements, and efficiently validating models. In heavily regulated industries such as pharmaceuticals, banking/finance, and insurance, this capability is non negotiable.

### Collaboration

Data scientists are most productive and innovative when they are able to seamlessly collaborate with each other. Too often silos - either technological or organizational - get in the way. When different technology (infrastructure, packages, libraries, etc.) is used across teams and departments, too often it becomes a situation of "well it worked on my computer…". It can also create the dangerous situation where a model delivers different results in different environments. This situation makes it very difficult or even impossible for data scientists to collaborate on projects and pick up where someone else left off.

Having a MLOps platform that abstracts away infrastructure friction and provides a central place to capture discussions, makes it easy to share work or work collaboratively as a team on a project, fosters experimentations, and compound knowledge. It also makes it easy for peer reviews to occur on work that is in flight - another way to foster innovation and eliminate siloed efforts that can go off track.

> **" Domino is really an enabler to share knowledge, but also to train and teach people how to leverage the best data techniques, models and making sure that they do it in a secure environment and by following best practices. "**
>
> Antoine Ly, Chief Data Science Officer, SCOR

## About Domino

Domino powers model-driven businesses with its leading Enterprise MLOps platform that accelerates the data science lifecycle while increasing collaboration and governance. A good place to start learning about what Domino can do for your organization is the Forrester Research study, **The Total Economic Impact of the Domino Enterprise MLOps Platform**. Forrester found that Domino provides a 542 percent return on investment over three years, and pays for itself in under six months.

DOMINO

## Increase Your Model Velocity

Start a free trial of Domino to see how our Enterprise MLOps Platform can accelerate the data science lifecycle in your organization.

To start your trial, visit dominodatalab.com/trial
To learn more about Domino, visit dominodatalab.com

DOMINO