Model Monitoring Best Practices

Maintaining Data Science at Scale

WHITEPAPER

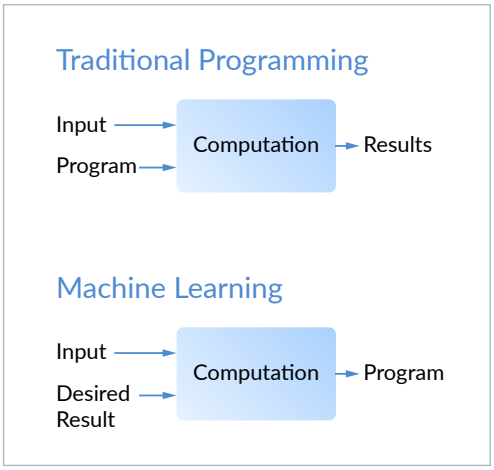
Contents

Executive Summary

A growing number of decisions and critical business processes are automated using models produced with machine learning and other statistical techniques. For a variety of reasons, models can "drift" over time, diminishing their predictive accuracy and leading to unexpected behavior — which in turn, can cause severe financial loss or a degraded customer experience. Too often, companies run models in production without adequately managing the risk of model drift. Or to manage it, they rely on data scientists doing manual and time-consuming work, distracting resources from future research and innovation. This whitepaper describes the common reasons and types of drift, and provides an overview of best practices for mitigating the risk of drift and monitoring to detect drift early.

Introduction

As companies grow and mature their data science organizations, they are deploying more models that automate decisions and critical business processes (e.g., claims processing, fraud detection, and loan approval).



These models — typically developed using machine learning (ML) or other statistical methods — are inherently probabilistic. Unlike in traditional software, where engineers specify explicit logic in code, a model's behavior is learned from data. Once in production, a model's behavior can change if production data diverge from the data used to train the model. Said differently, a model's behavior is determined by the picture of the world it was "trained" against — but real-world data can diverge from the picture it "learned." If you trained a mouse to perfectly navigate a maze, the mouse would not perform as well when placed into a new maze it had not seen before.

Creating additional risk, models often depend on data pipelines and upstream systems that span multiple teams, including data science, software engineering, DevOps, and business. Data engineers, DBAs, analysts, and others can be involved as well. Changes or errors in those upstream systems can change the nature of data flowing into the models, often in silent but significant ways.

The phenomenon of models degrading in performance is called "drift," and it occurs for a variety of reasons, which we explore later. Regardless of the cause, the impact of drift can be severe, causing financial loss, degraded customer experience, and worse.

IT organizations, while highly mature at monitoring the health and performance of traditional software applications, are often ill-equipped to monitor models' health, or entirely unaware of the risk of model drift. Typically, data science leaders carry this burden as ultimately their teams are responsible for the quality of predictions these models are making. Data scientists then spend a significant amount of time analyzing models in production instead of doing new research. Some data scientists, in an effort to reduce manual effort, then develop ad hoc monitoring solutions for each model, leading to a proliferation of disparate, inconsistent, poorly maintained and half-baked monitoring solutions.

Operationalizing model monitoring is key to ensuring models in production are functioning well. Approach monitoring a machine learning model the same way you would think about getting your annual physical check-up or getting periodic oil changes for your car. Model monitoring is an operational task that allows you to check that your models are performing to the best of their abilities.

Models Degrade over Time

Models can degrade for a variety of reasons: changes to your products or policies can affect how your customers behave; adversarial actors can adapt their behavior; data pipelines can break; and sometimes the world simply evolves. The most common reasons fit under the categories of Data Drift and Concept Drift.

Data Drift: Data drift occurs when the patterns in production data that a deployed model predicts on start diverging away from the patterns present in the model's original training data. Data drift results in the model losing predictive power. Data drift can happen for a variety of reasons, including a changing business environment, evolving user behavior and interest, modifications to data from third-party sources, data quality issues, and even issues in upstream data processing pipelines.

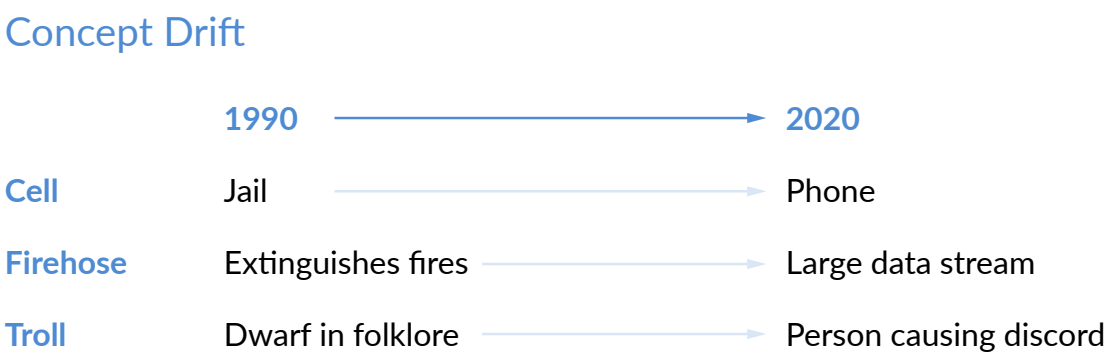
Issues that Lead to Data Drift:

> **Discrete changes in external factors** that affect behavior. For example, if a cellular provider raised a limit on the number of text messages that could be delivered to a device per minute, that might affect security risk models or fraud models that relied on message frequency as an input.
>
> **Natural changes in the state of the world** that were not considered in the training data set, such as seasonal trends affecting consumer behavior, or mechanical wear and tear affecting sensor readings on manufacturing equipment.
>
> **Behavior change due to adversarial adaptation.** If you have a fraud model, the fraudsters may deliberately vary their behavior once they realize the model has "caught on" to their previous patterns.
>
> **Changes in economic or consumer behavior.** A model is optimized based on the variables and parameters at the time that it was created. An economic or consumer behavior model that was created pre-COVID would likely face performance issues now, as the data have drifted.
>
> **System or human errors in the data pipeline** or in the collection of data that occur after the initial model training. Data engineering departments frequently do not communicate all the changes they have made to a pipeline to the data science department. These problems can be attributed to versioning, scheduling, failure to enforce best practices for transformations, lots of copies of copies, and so on. For example, an upstream pipeline might change the units of a variable (such as degrees Fahrenheit to degrees Celsius).

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 as 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. Similarly, news themes of interest last week will change for this week. People's interest in product categories have also changed during the COVID pandemic, leading to stocking failures as many retail prediction models continued to predict based on training data for pre-COVID consumer interest.

As an illustrative example of concept drift, consider how a sentiment model created thirty years ago may falsely classify the sentiment of various words and phrases, as the way we speak, words that we use, and slang that we develop constantly change over time:



For models to work in such situations, it is necessary to monitor the model's performance with respect to expectations about its results as they are today — captured through recent ground truth data — and not on past expectations from the model's old training datasets.

Approaches to Identifying Model Degradation

Given the potential of significant negative impact from model degradation (e.g., consider the speed at which a bank could lose money due to a degraded loan default model), it is critical to detect model drift as soon as possible. Here are some recommendations for detecting model degradation before outdated models can cause serious impact to your business.

Checking Model Predictions: One common methodology for detecting model degradation involves checking if the model's predictions are no longer valid or accurate. For example, a large insurance company identified that it was spotting far fewer fraudulent claims than it had in the past. The anomalous results led the company to review the accuracy more closely, once actual results (i.e., ground truth data) were available.

On the output side, many models produce some sort of score or set of scores, which often represent a probability estimate. If the score distribution produced by the model changes unexpectedly, that implies model degradation. This could be caused by a change to the model inputs that reflect some change in the outside world, or a change in the systems that the model depends on (e.g., for feature extraction). In either case, observing the change in score distributions is the first step in identifying the problem.

You can also run logical checks on the quality of predictions to assess a model's health. For example, a model that predicts credit scores should result in virtually all predictions in a range between 300 and 850. And if a retailer is predicting store visits, that number should not be negative.

> One of the most effective approaches to detecting model degradation is monitoring the input data presented to a model.

It is also important to assess if you can use ground truth data when assessing a model. As nice as real-time ground truth feedback may be, some use cases involve months or years of lag time to know the ground truth of a prediction, if they're available at all. For instance, you may not know whether customers who are predicted to default on a 30-year mortgage are actually defaulting for many years. But you can use the distribution analysis and other techniques outlined above, if using ground truth data is not feasible.

Checking (Input) Data Drift: One of the most effective approaches to detecting model degradation is monitoring the input data presented to a model to see if they have changed; this addresses both data drift and data pipeline issues. There are several effective approaches to checking input data, including:

> **Reviewing descriptive statistics**, data types (e.g., strings, integers, etc.), data ranges, and data sparsity/ missing data, and then comparing them to the original training data. Drastic changes in input data distributions may highlight serious model degradation.
>
> With either an automated or manual process, you can compare your model input distributions with statistical tests. Basic statistics, like median, mean, standard deviation, and max/min values, as well as more advanced statistical tests, like chi-square, can highlight model deterioration. For example, if the model input variables are normally distributed, you would expect the mean values to be within the standard error of the mean interval.
>
> **Defining metrics** to track the difference between data used to train the model versus data that are being presented to the model to score. If the difference crosses a threshold or is drifting significantly, that is a strong indicator of model drift and degradation.
>
> Depending on the model configuration, you will allow certain input features to be null or not. If features you generally expect to not be null suddenly become null, that could indicate a broken data pipeline or a significant change in consumer behavior, both of which would be cause for further investigation.

Note that some features are more important (predictive) than others. Important features that have drifted a small amount may be cause for retraining, while features with low/no predictive power can drift a lot with negligible impact on the overall model. It is important to identify the most important features to a model when you set up your monitoring so you are tracking them closely.

Checking Concept Drift: Similarly to data drift distribution analysis, you can analyze concept drift using the same methods. Typically, you would compare the distribution of the labels of your training set versus those of your production data in real-time. You can also check that a) the input values fall within an allowed set or range, and b) that the frequencies of each respective value within the set align with what you have seen in the past. For example, for a model input of "marital status" you would check that the inputs fell within expected values such as "single", "married", "divorced", and so on.

Monitoring label distribution can also be used to trigger an event in order to retrain the model. When distributions change naturally, the label distribution is bound to change. Having that information early can help you be more in control when you retrain your model or debug various problems, and use fewer resources as well.

Organizational Best Practices for Monitoring Models

Approaches to model monitoring vary widely across companies - often even within a single company's data science department! Commonly, a single data scientist or ML Engineer on the team is given responsibility for monitoring a model. Alternatively, some departments rotate responsibility amongst data scientists on a quarterly basis. Other companies require a monitoring plan to be in place before an advanced analytics project is considered completed. Setting up the proper monitoring can add weeks to a project's lifecycle.

Other companies require that the IT department be responsible for production model monitoring. This puts both a time and education burden on already overburdened IT departments. Traditional IT tools and experiences apply to monitoring infrastructure, uptime, and latency, and do not translate well to model monitoring, which depends more on statistical methods.

Some companies build tools to address ongoing monitoring. For example, one large agricultural company has created Shiny dashboards and pipelines specifically for ongoing production model monitoring. While this approach may be fine for monitoring a few models, complexity grows significantly as more models are built using different languages and environments, and the number of models to be monitored increases.

Five Best Practices for Monitoring Models in Production:

1. Acknowledge that model monitoring takes different tools and skill sets 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 skillset of IT/Ops. For example, Kolmogorov-Smirnov, chi-square, and other advanced statistical tests are frequently utilized in model monitoring to assess data samples and distributions, and identify early signs of model health degradation.
2. Actively monitor ALL models in production from a central place. A centralized approach breaks down silos, and enhances consistency and responsiveness. You will 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 models.
3. Establish a consistent set of metrics for assessing the health of models. Make sure the definition of when to retrain or rebuild a model is agreed upon. Building consistent methodologies, and learning from what's working and what's not working for your team will enhance both management and organizational improvement.
4. 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.
5. Automate as much as possible. As you scale models in production, you will either need more ML Engineers or find ways to automate. Automating model monitoring brings numerous benefits, including consistent intelligence, reliable notifications, and more time for data scientists to focus on value-adding model development work instead of monitoring. A large financial company set up automated monitoring for data quality-related checks, which assess tolerance levels and valid ranges for the input data. Whatever your approach, developing an early warning system to alert around deteriorating model health is critical as more businesses rely on data science models for their essential operations and profit.

Model Monitoring with Domino

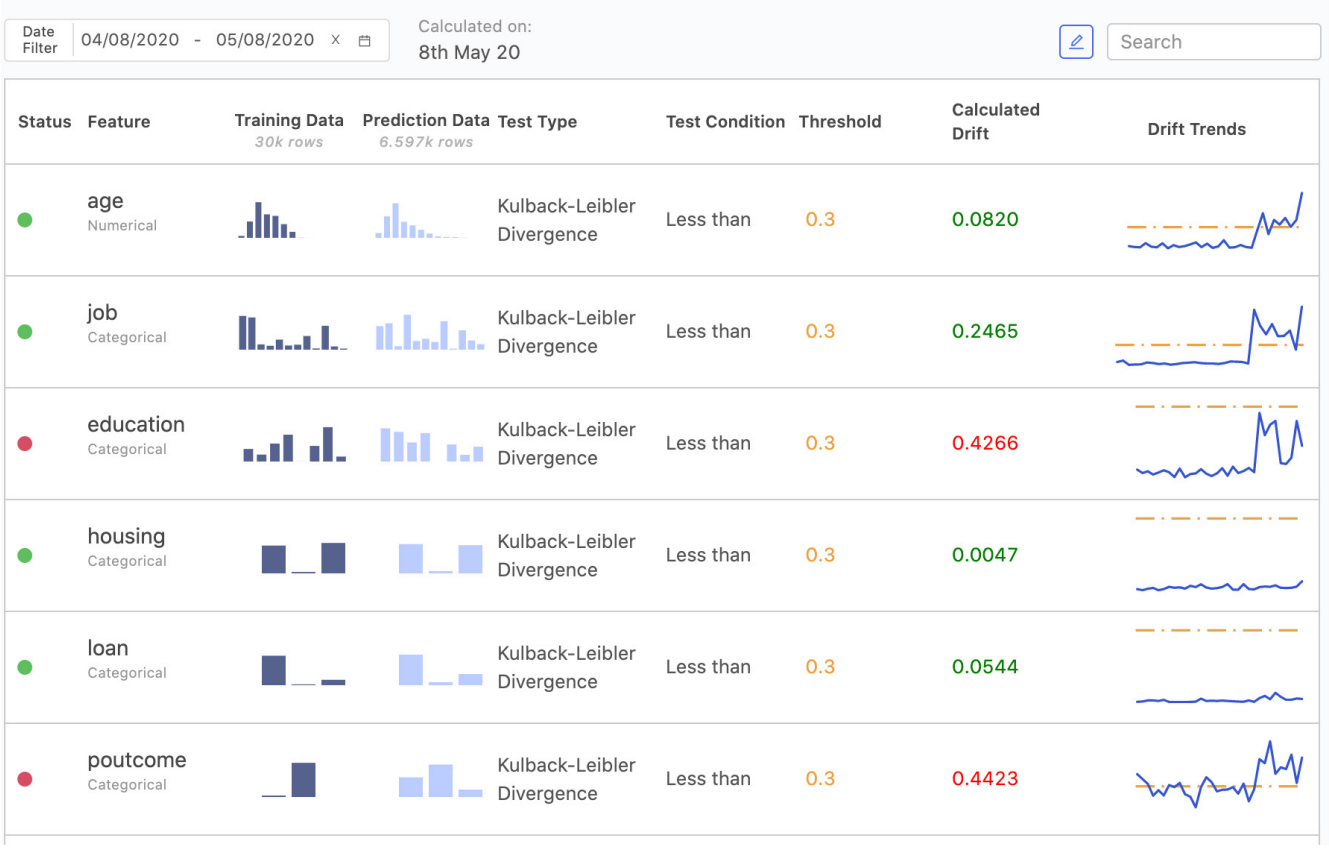
Given the criticality of some data science models to your business, you need to have some sort of production monitoring system in place. Without observability, maintaining an end-to-end model lifecycle is an impossibility. You do not want to be caught unaware of critical production models deteriorating beyond acceptable levels.

Rather than dealing with the expense of developing your own monitoring tools, or assigning data scientists as ML Engineers to manually monitor all your production models when they could be working on other value-added projects, Domino Model Monitor (DMM) can solve these problems.

Domino Model Monitor provides a single pane of glass with automated alerting across all your production models. With this model health dashboard, it is reasonable for IT to assume much of the responsibility for monitoring data science models. DMM can send alerts to the IT team, data scientists, as well as other interested parties, so model degradation identification becomes integrated into your workflows.

DMM detects and tracks data drift in the model's input features and output predictions. Users can select among a variety of statistical checks to best suit your monitoring needs. If you have ground truth data for the model, DMM can ingest it to calculate and track the model's prediction quality using standard measures such as accuracy, precision, and more.

DMM checks if the characteristics of the predictions versus the target data that were used to train the model are significantly different. DMM also tracks the difference between data that were used to train the models versus the data that are being presented to the models to score.



By setting up scheduled checks, using APIs to ingest data, and configuring alert notification recipients in DMM, you can continuously monitor hundreds of models in a standardized way across your organization. APIs enable integration into existing business processes, as well as a programmatic option for auto retraining. DMM enables both your IT department and data scientists to be more productive and proactive around model monitoring, without requiring excessive data scientist time.

Best Practices for Addressing Model Drift

Once you have detected that a model in production is experiencing model drift, you need to estimate the impact, analyze the root cause, and then take an appropriate corrective action. There are a number of best practices around addressing model drift:

1. **Estimate the impact:** It is important to quickly estimate the impact of model drift to determine how much to prioritize the analysis and how many resources to spend on model repair. If one of your business-critical models is experiencing data drift in features of high importance or if important model quality metrics show significant drop, this would get top priority. However, if the data drift is for less important features or only happened for a short time period, then it may be acceptable to leave the model in production and watch for similar occurrences. This is where having robust and high signal-to-noise ratio alerting is important.
2. **Analyze the root cause:** To understand the causes better, you need to drill down and do a time based analysis. If your checks are run weekly, see how the drift numbers evolved during each of the days. You can look at drift not just for that period, but for time periods before and after as well.

   It is important to determine if there is gradual or abrupt drift. For gradual drift:

   - Determine if drift is consistently rising and how fast.
   - Assess if the same features or metrics drifted in the preceding time periods.
   - If model quality metrics have drifted, check if that was preceded by data drift for important features.

   For abrupt drift:

   - Check if data for a particular feature are suddenly having a lot of nulls, values stuck at a constant level, or label encodings with missing codes. All of these may indicate a breakage in your data pipeline instead of the model having the issue.
   - Visually compare the histogram of the prediction data during the failing period against that of the training data for a feature which has significantly drifted.
   - Determine if there was a shift in specific classes or distribution bins.
   - For failed model quality metrics, investigate if all classes of prediction variables are showing poor results or if there is one particular class which is seeing significant degradation of metrics.

   You can also run alternative divergence tests to confirm it is not only one particular test raising an alarm. All of the above methods help you to identify what kind of data needs to be added to the training dataset to cover the missing patterns.

3. **Resolve the issue:** The first approach here would be to use the same algorithm, architecture and hyper-parameters as the original model, but retrain the model on a new training dataset with more recent or relevant samples added to it. This may allow you to circumvent lengthy model validation processes, and get models back into production quickly, since the underlying data and assumptions remain the same.

   You can evaluate the performance of this new model on new evaluation data that reflect the changed data patterns and/or prediction expectations. If retraining with a richer training dataset does not resolve the issue, you may need to build a new model from scratch using a different model architecture or algorithm.

Conclusion

As your company moves more machine learning systems into production, it is necessary to update your model monitoring practices to remain vigilant about model health and your business' success in a consistent, efficient manner. Using the best practices described in this paper, including Domino Model Monitor, will help you set up your organization for best-in-class monitoring and retraining results, without requiring excessive model "babysitting" time from your valuable data scientists. Your business will appreciate these efforts as there will be much less model degradation, and you will be able to intelligently retrain models to optimize business success.

About Domino Data Lab

Domino Data Lab empowers data science teams with the leading, open data science platform that enables enterprises to manage and scale data science with discipline and maturity. Model-driven companies including Allstate, Dell Technologies, and Bayer use Domino as a data science system of record to accelerate breakthrough research, increase collaboration, and rapidly deliver and monitor high-impact models. Founded in 2013 and based in San Francisco, Domino is backed by Sequoia Capital, Bloomberg Beta, Dell Technologies Capital, and other leading investors. For more information, visit dominodatalab.com.


DOMINO