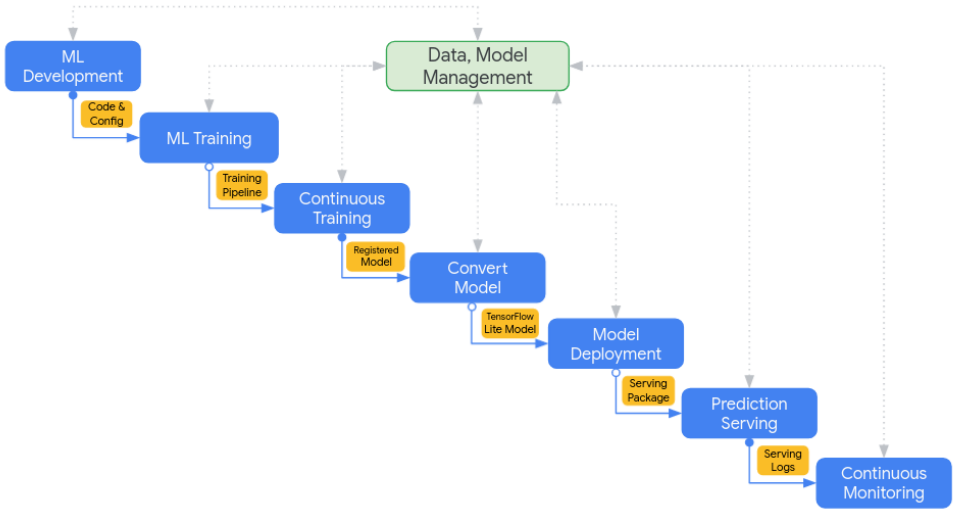
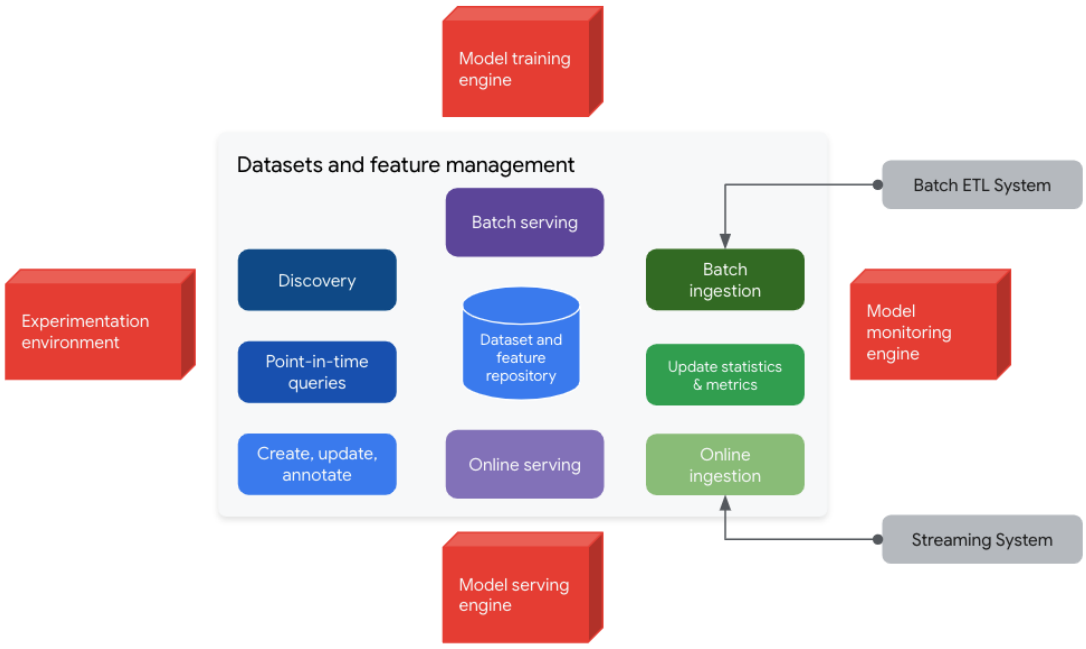
Overview of Data and Model Management



Objective

So far, we have discussed the operations of our system in terms of training, deployment, prediction serving, and continuous monitoring. However, we have, until now, left out the most important aspect of this system: the data and models used within our system! And that is where model management, data management and model governance all come into play. In this section, we will briefly understand these three differen concepts and discuss their relevance.



Dataset Management

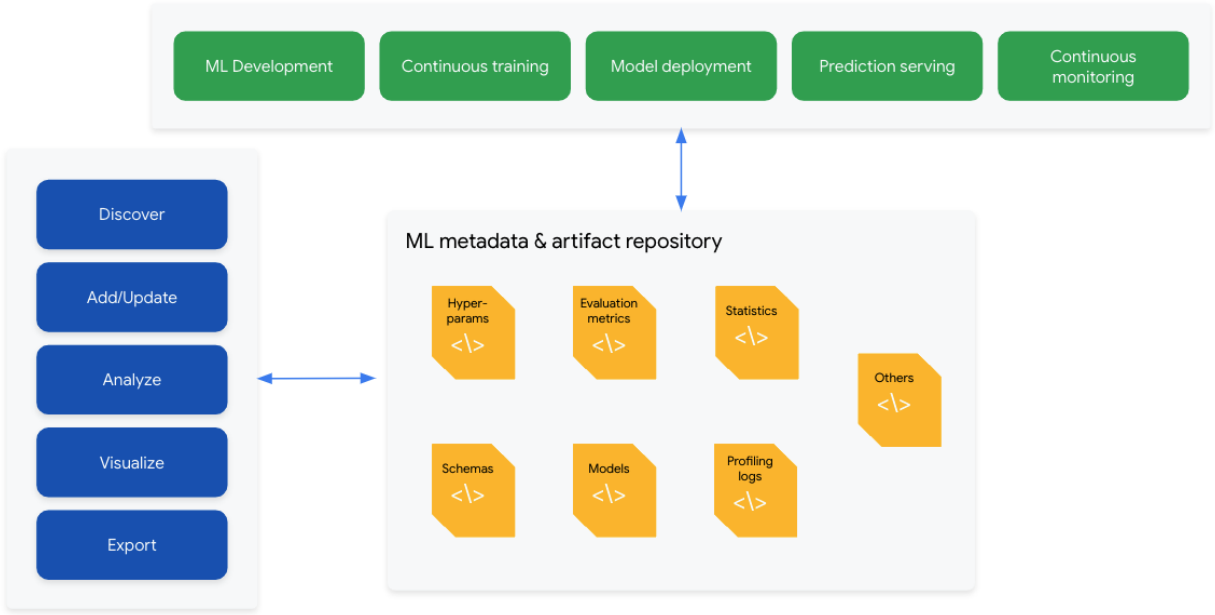
In dataset management, features describe fundamental entities related to our application, such as products, customers, and locations. While individual features can be used in a large number of datasets, an individual dataset is typically used for a specific model or application. For example, a recommendation system might require information about both customers and products, while a propensity model might only require information about a customer. To produce the datasets used for training, some transformations or operations will typically be performed on the features included within the company's databases. These sets of operations might be described in scripts, which will need to be saved and monitored for future use in some kind of repository. These scripts might have to be versioned depending on whether they are for training, testing, or deployment. We will also need to know how the data was split, along with any other information about the data in order to ensure we can reproduce the results in the future. All of these activities fall under the purview of dataset management.

Model Management

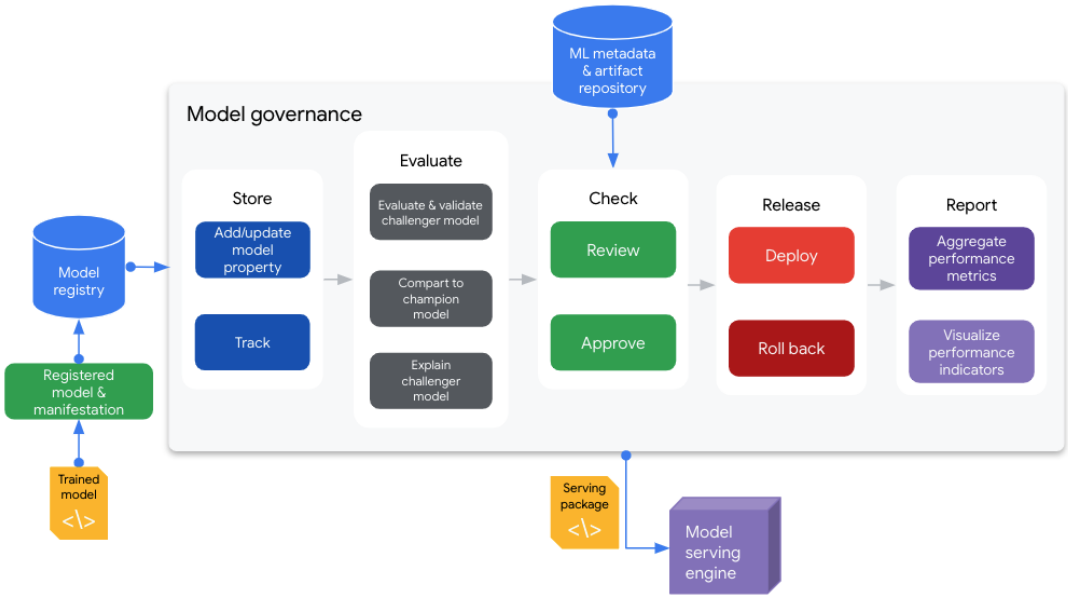
Keeping track of models becomes increasingly difficult when they are all simultaneously being constructed, tested, and deployed. To prevent models from being misplaced or the wrong models tested or deployed, it is important to have a strong model management system in place. Model management involves tracking metadata related to the model, such as what training data and hyperparameters were used to train the model, as well as model governance. These aspects help to ensure that (1) data being used to train a model is accurate, unbiased, and does not violate licensing or privacy requirements, (2) models are evaluated using metrics, such as those related to quality control or fairness indicators, (3) model outcomes are interpretable and

explainable (necessary for GDPR), and (4) that issues during model training or deployment are reproducible, traceable, and can be debugged.

For **model metadata and artifact tracking**, artifacts related to the machine learning model are captured and stored in a repository. Metadata that might be captured might include pipeline run ID, start and end time of training, run status, and environment configuration. Artifacts that might be captured include evaluation metrics, hyperparameters, and schemas. Note that model metadata refers to aspects of the model runtime, whereas artifacts describe aspects of the model itself. Keeping track of artifacts and model metadata is helpful for data scientists and ML engineers to ensure reproducibility and traceability since the parameters used during experimentation as well as the configuration environment are known. These aspects also make searching through large numbers of models for a specific experiment easier.



Model governance dictates what happens to a model during its development. This entails registering (i.e., instantiating within the model registry), reviewing, validating, and approving a model for deployment. The process of model governance is highly context-dependent, and will vary between application, organization, and regulatory environment. Some implementations may require more checks and balances than others, and the same implementation may require changes to the governance model over the course of its lifetime due to changes in the regulatory environment as well as other factors. The process of model governance can itself be manual, semi-automated, or fully automated.



Automated model governance typically relies on information from the model metadata repository, as well as the model registry. This allows the governance model to perform various tasks. The first task is to be able to add and update model properties, as well as to track model versions (such as the version used in training compared to that used for deployment). This makes it easy to keep track of models for future reference, for traceability and repeatability purposes. A new “challenger” model can also be incorporated into the model governance system, which can then compare it to the available model and test it across various evaluation metrics, as well as business key performance indicators to discern whether deploying the new model would yield superior results compared to the existing model.

The governance model can then also check the model for non-functional requirements, such as fairness indicators, privacy concerns, and regulatory compliance (e.g., ensuring no code was used that is distributed under a software license containing a share-alike attribution policy). The governance model can also control how deployment is implemented (e.g., whether a blue-green or canary deployment is used for a particular model) and how traffic is split between the deployments (if a deployment requiring multi-way traffic is designated). Finally, the governance model details how the continuous monitoring protocol works, including how and which metrics are aggregated and reported.

Additional Resources

Challenges with Model Management
Stanford Course on Data Management