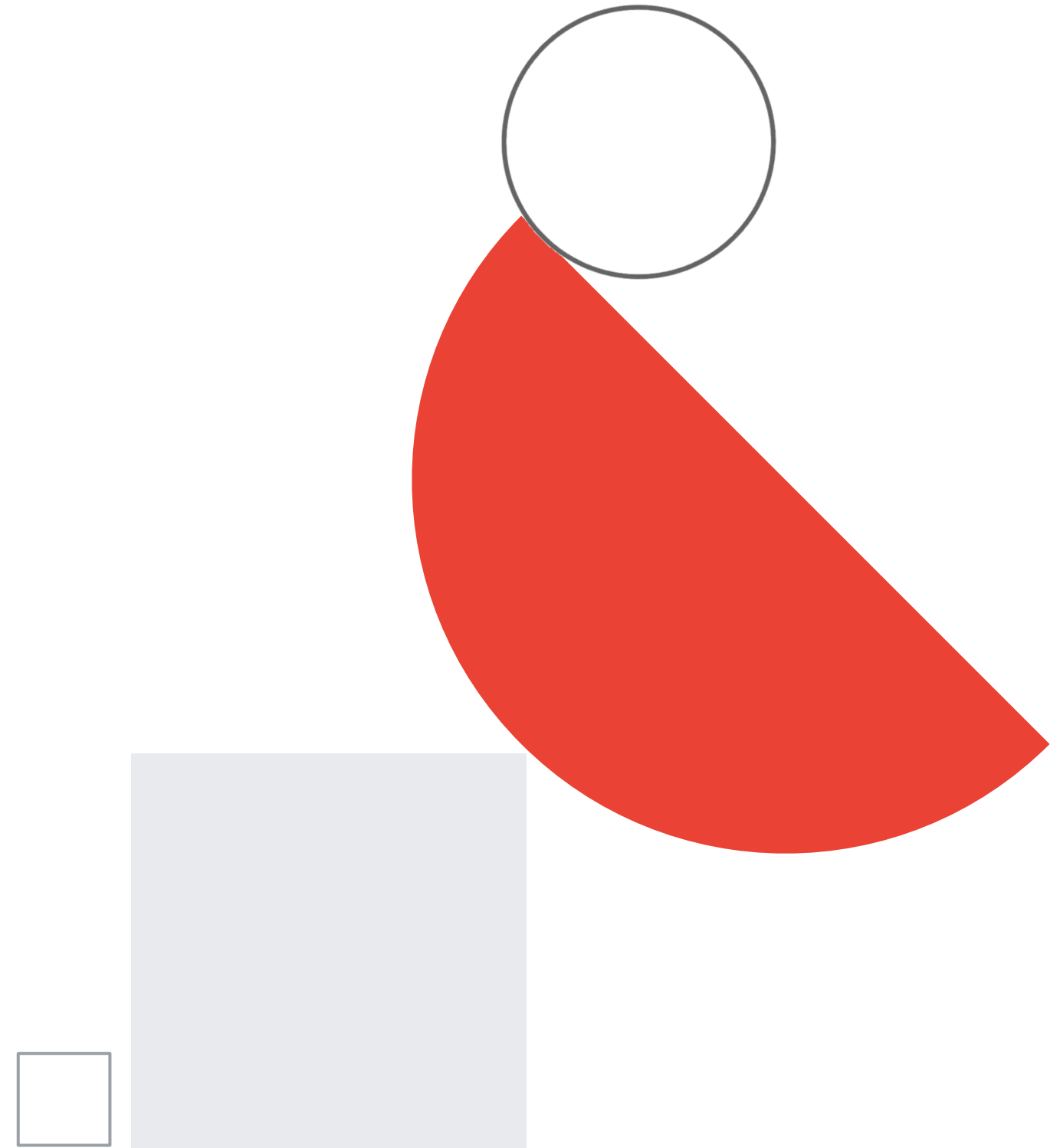


# Best practices for machine learning development



# Best practices

## Data

How it is prepared and stored

## Workbench Notebooks

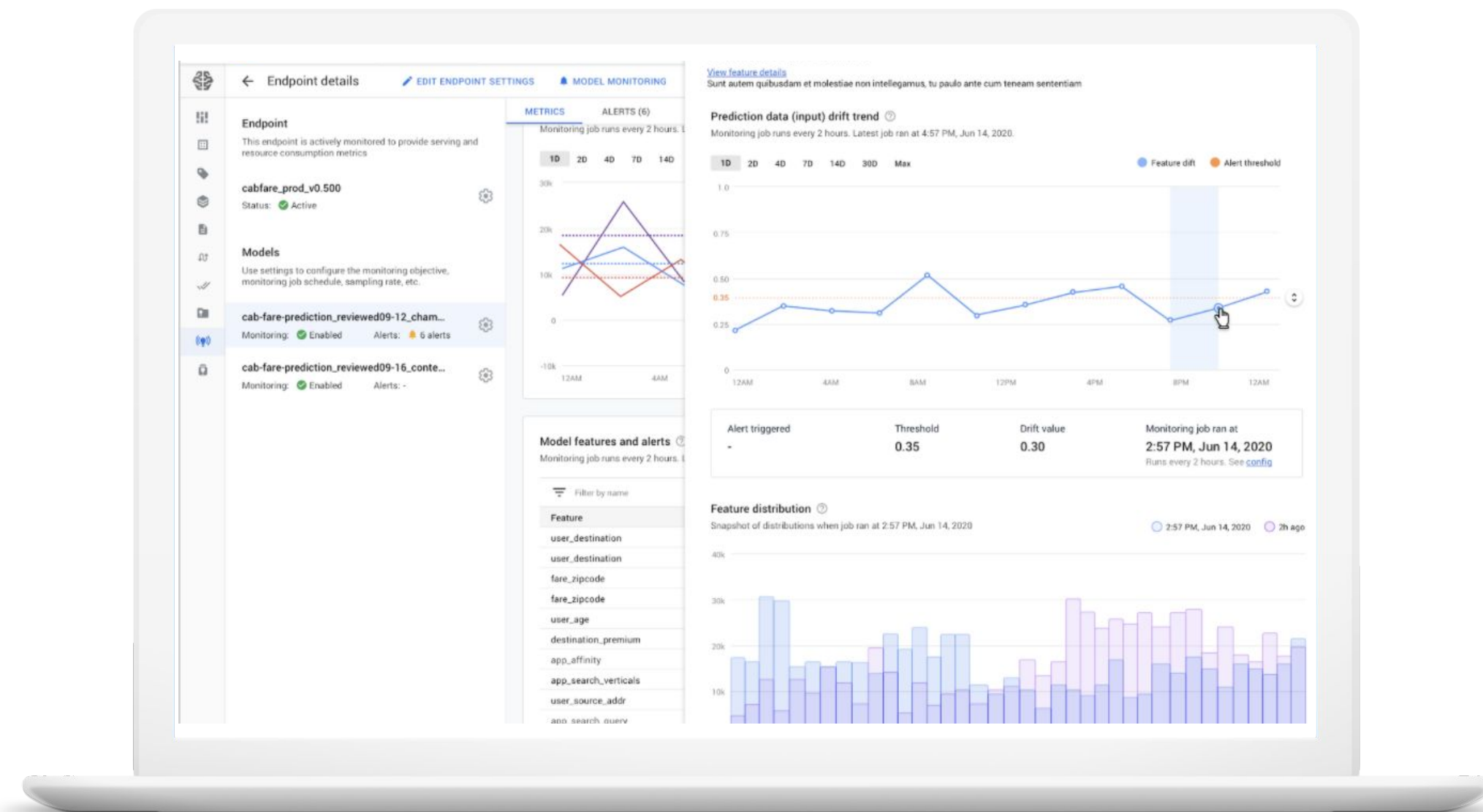
Using Notebooks to evaluate and understand your models

## Model

Tips for training, maximizing predictive accuracy, and feature attributions for insights

## Tensorboard

Using Vertex AI TensorBoard to visualize experiments

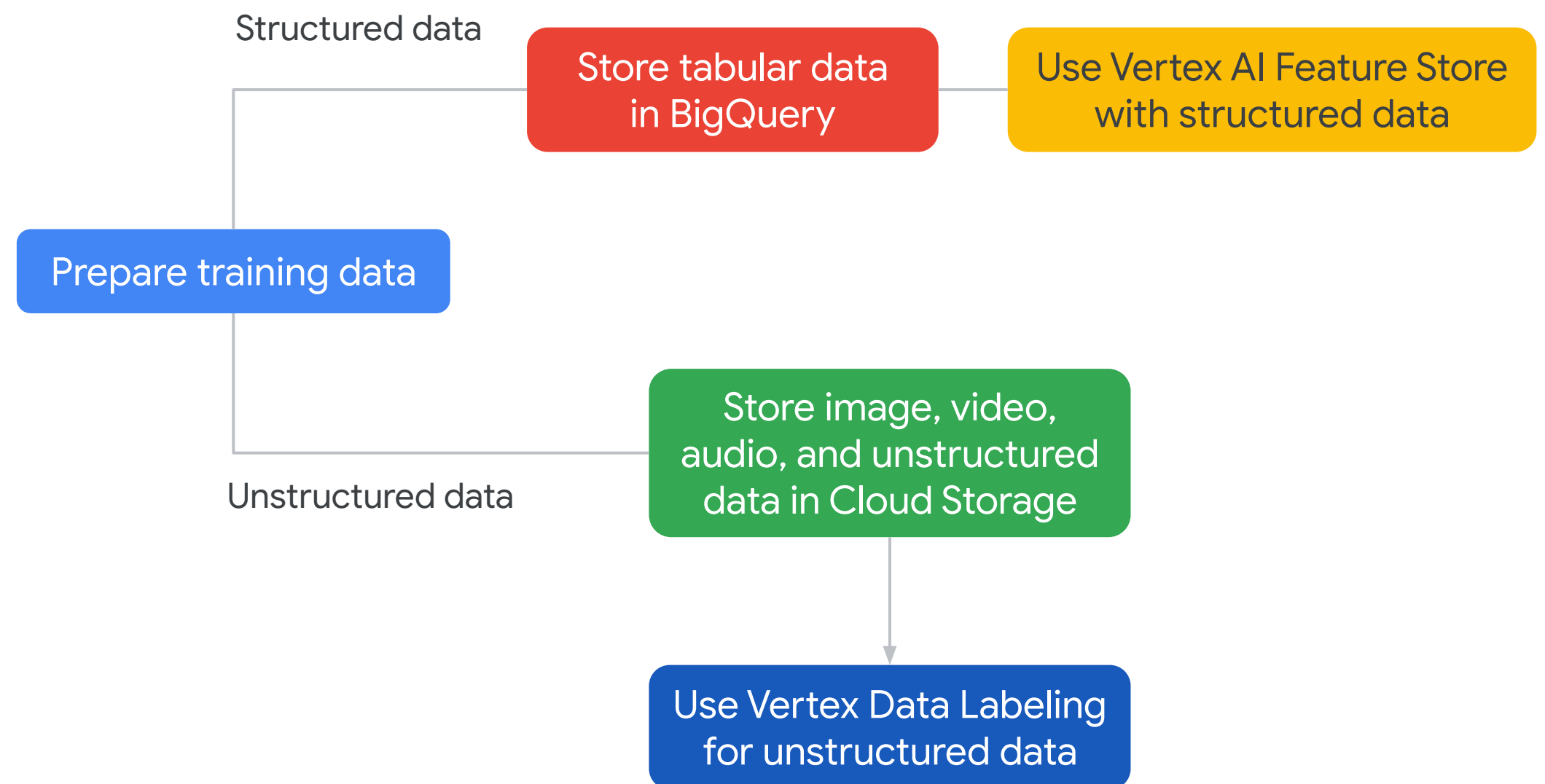


# Best practices for preparing and storing data

Data

How it is prepared and stored

Avoid storing data in block storage



# Vertex AI Feature Store

## Feature Store

Use Feature Store with  
structured data

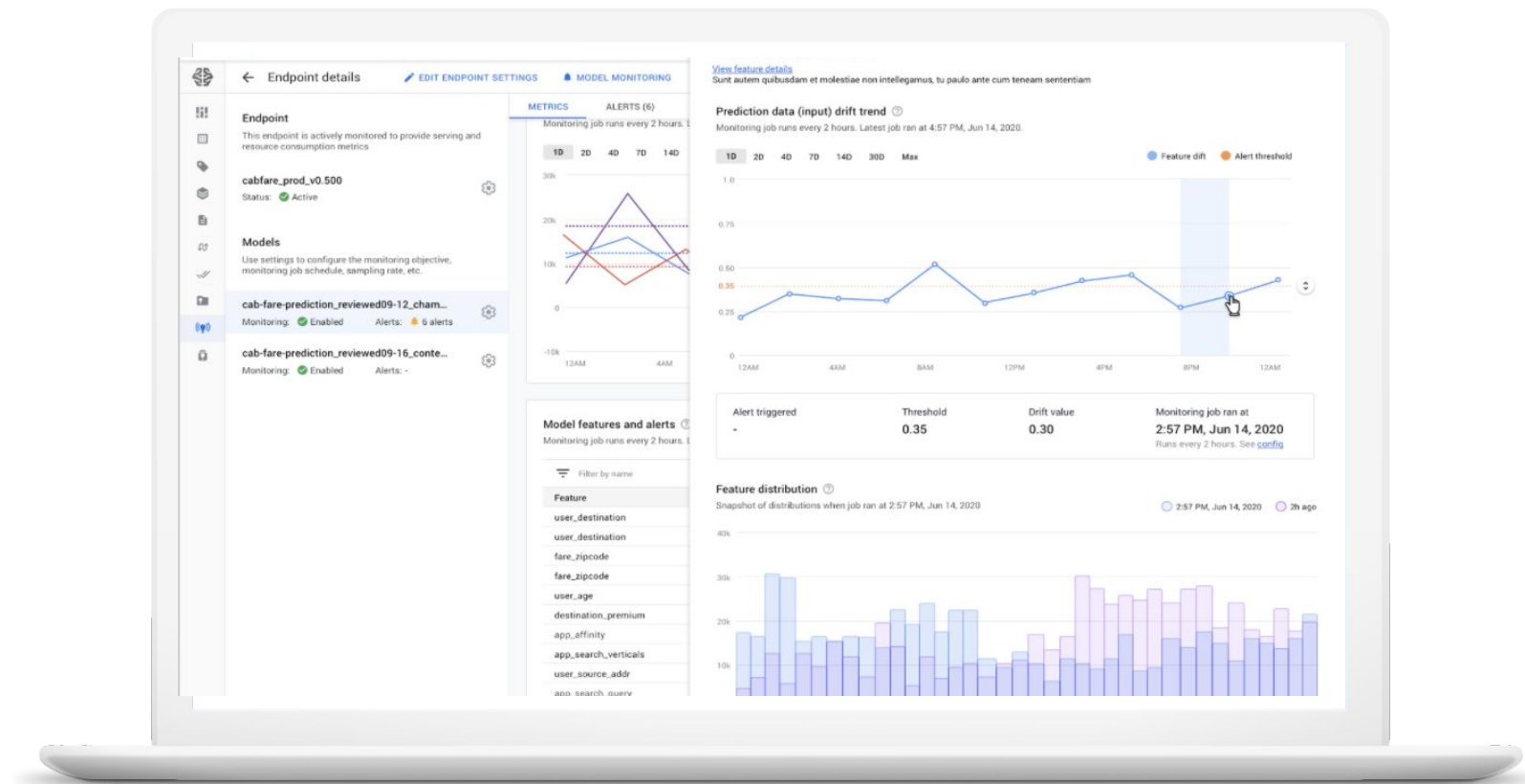
Follow these steps:

1. [Search Vertex AI Feature Store](#)
  - a. Search to see if a feature already exists.
  - b. Fetch those features for your training labels using [Vertex AI Feature Store's batch serving capability](#).
2. [Create a new feature](#)
  - a. Create a new feature using your Cloud Storage bucket or BigQuery location. OR
  - b. Fetch raw data from your data lake and write your scripts to perform feature processing.
  - c. Join the feature values and the new feature values. Merging those feature values produces the training data set.

# Best practices for training a model

## Model

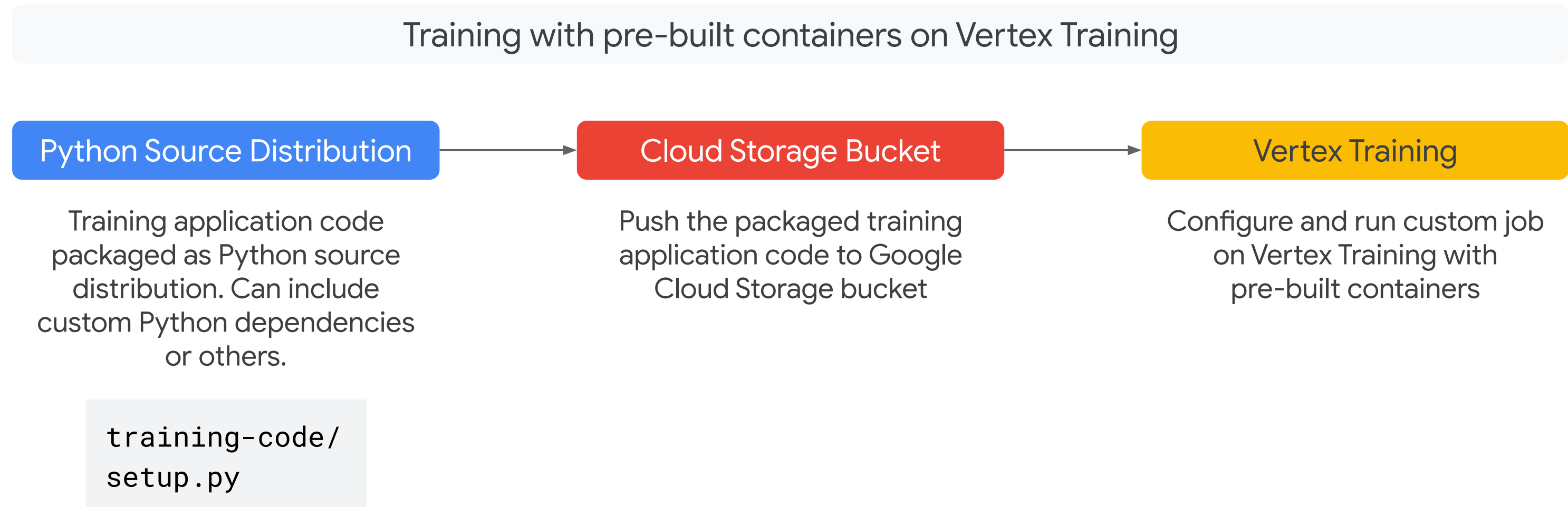
Tips for training, maximizing predictive accuracy, and feature attributions for insights



For small datasets, train a model within the [Notebooks instance](#).

For large datasets, distributed training, or scheduled training, use the [Vertex training service](#).

# Training with pre-built containers on Vertex AI



# Best practices for Explainable AI

## Model

Tips for training, maximizing predictive accuracy, and feature attributions for insights



Offers feature attributions to provide insights into why models generate predictions.

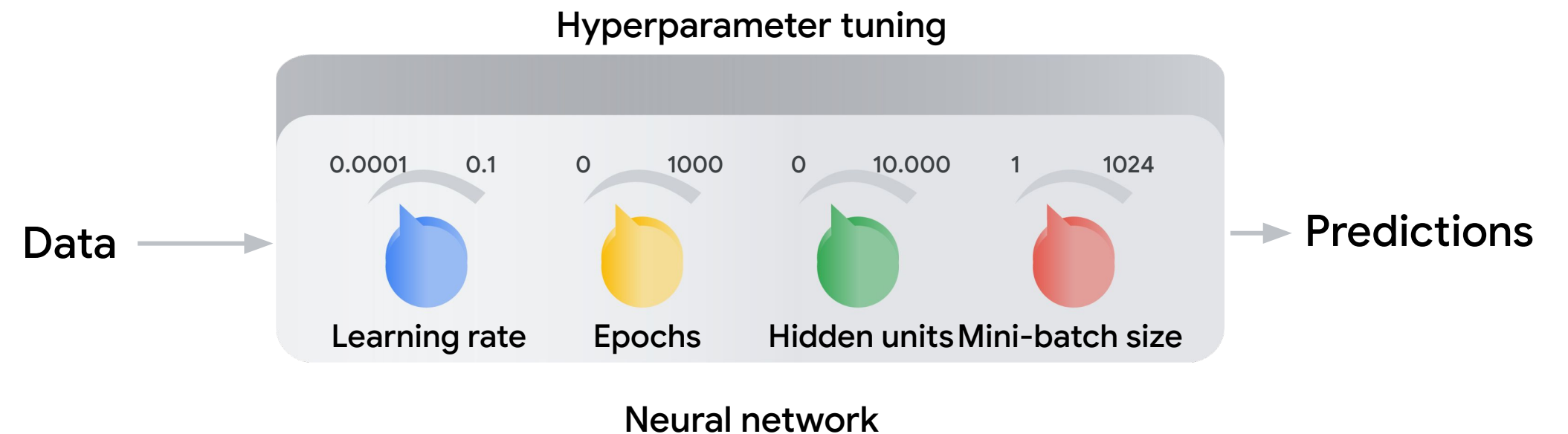
Details the importance of each feature that a model uses as input to make a prediction.

Supports custom-trained models based on tabular and image data.

# Hyperparameter tuning with Vertex Training

## Model

Maximize your model's predictive accuracy with hyperparameter tuning



The hyperparameters are knobs that act as the network-human interface.

Maximize a model's predictive accuracy. [Vertex Training](#) provides an automated model enhancer to test different hyperparameter configurations when training your model.

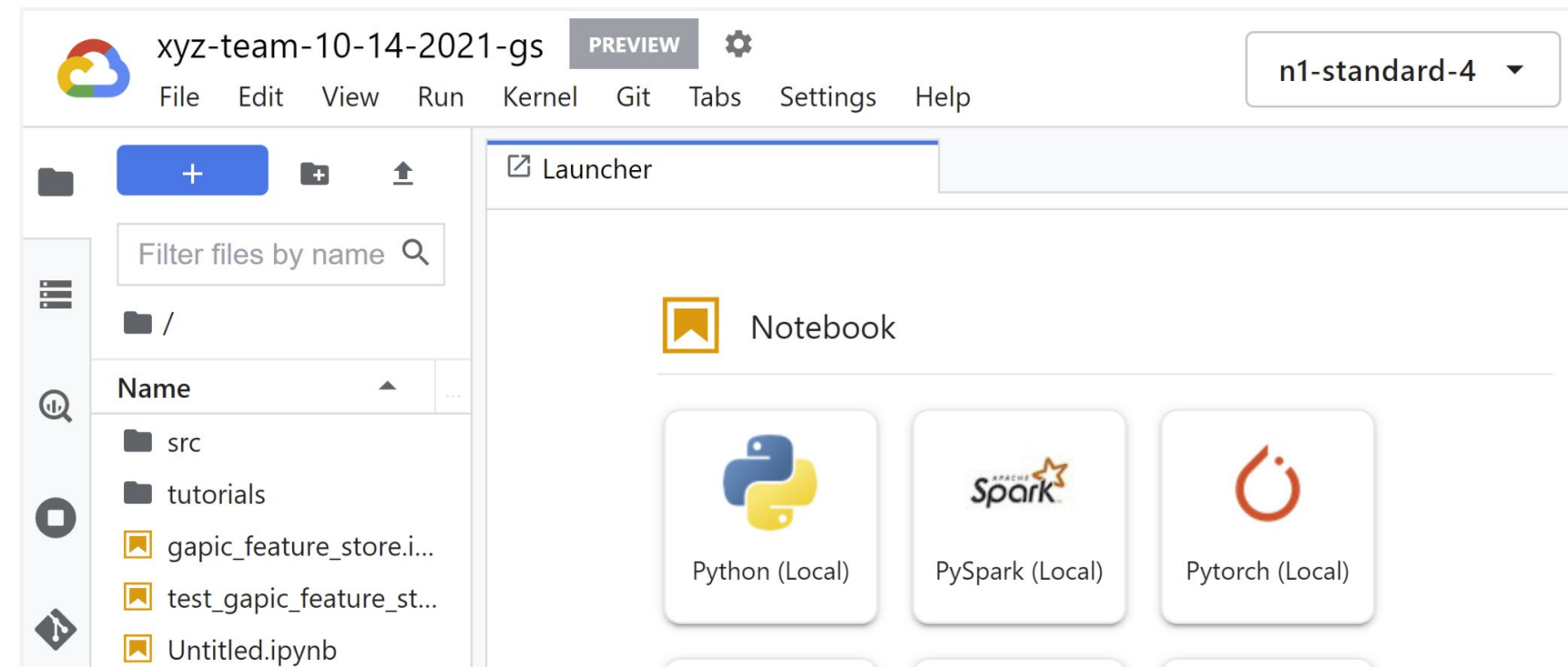
No need to manually adjust hyperparameters over the course of numerous training runs to arrive at the optimal values.



# Best practices for using Workbench Notebooks

## Workbench Notebooks

Use Notebooks to evaluate and understand your models.

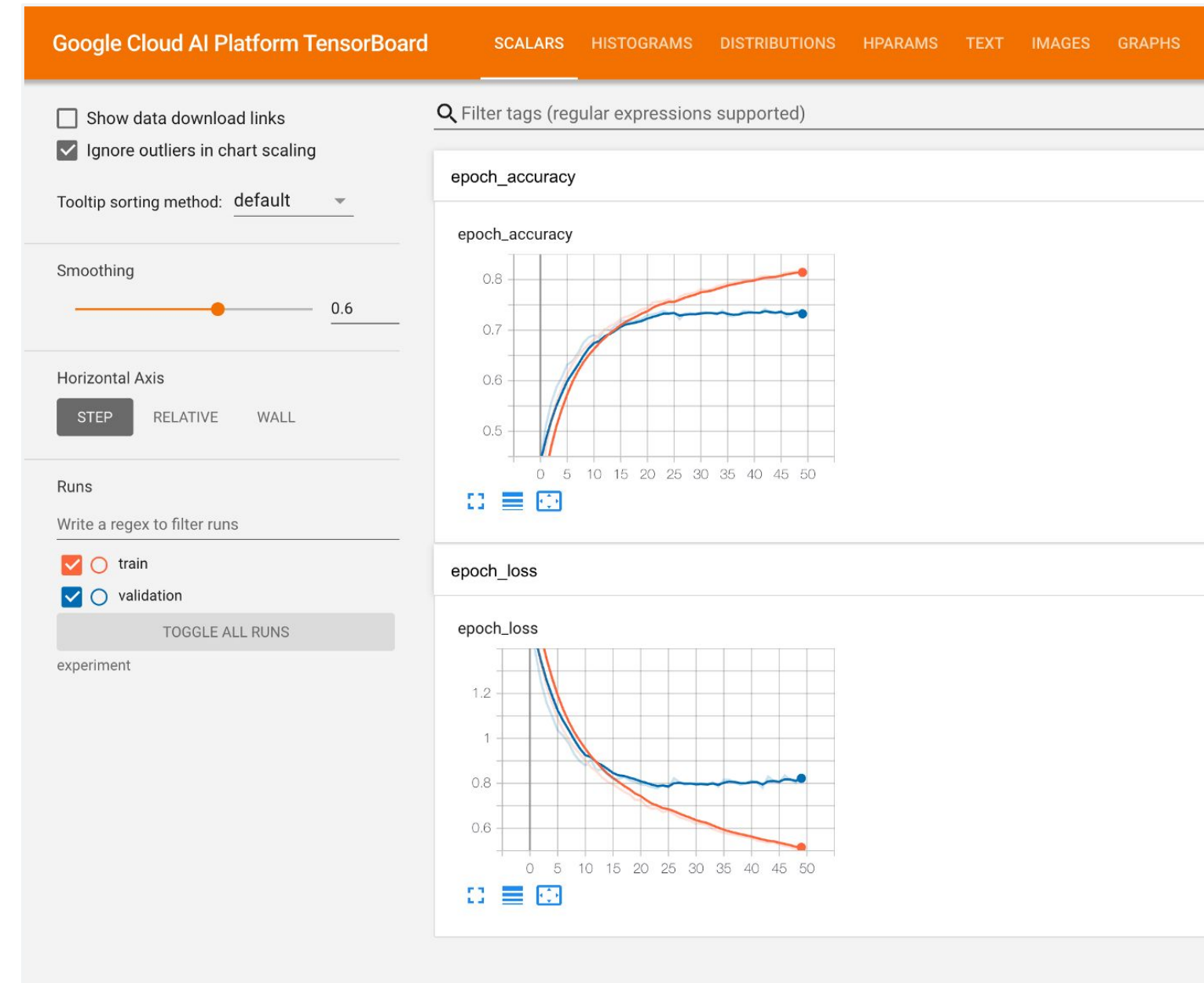


Use [Notebooks](#) to evaluate and understand your models. In addition to built-in common libraries like scikit-learn, Notebooks offers [What-if Tool \(WIT\)](#) and [Language Interpretability Tool \(LIT\)](#).

# Best practices for using Vertex AI TensorBoard

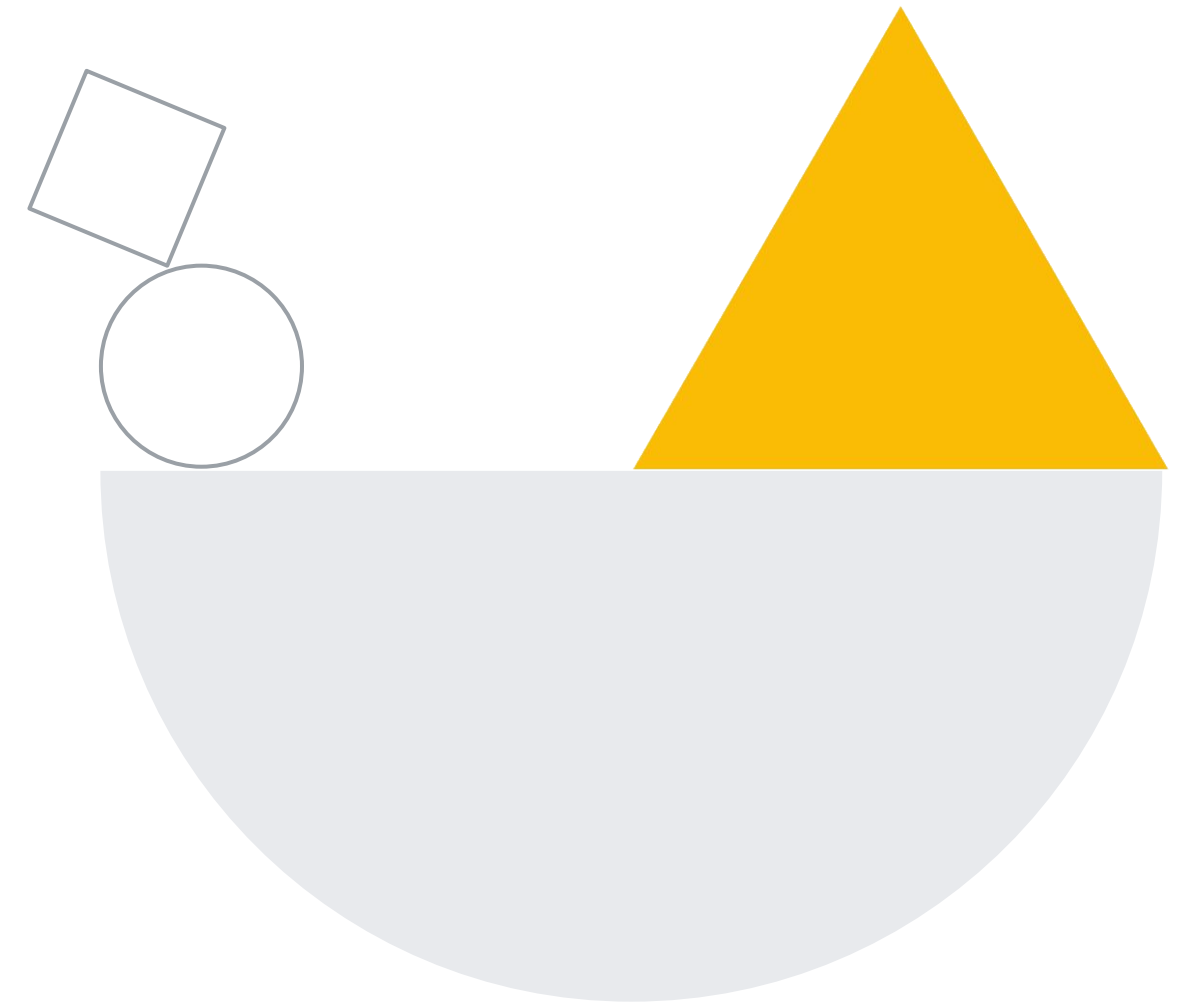
## TensorBoard

Use Vertex AI TensorBoard to visualize experiments.

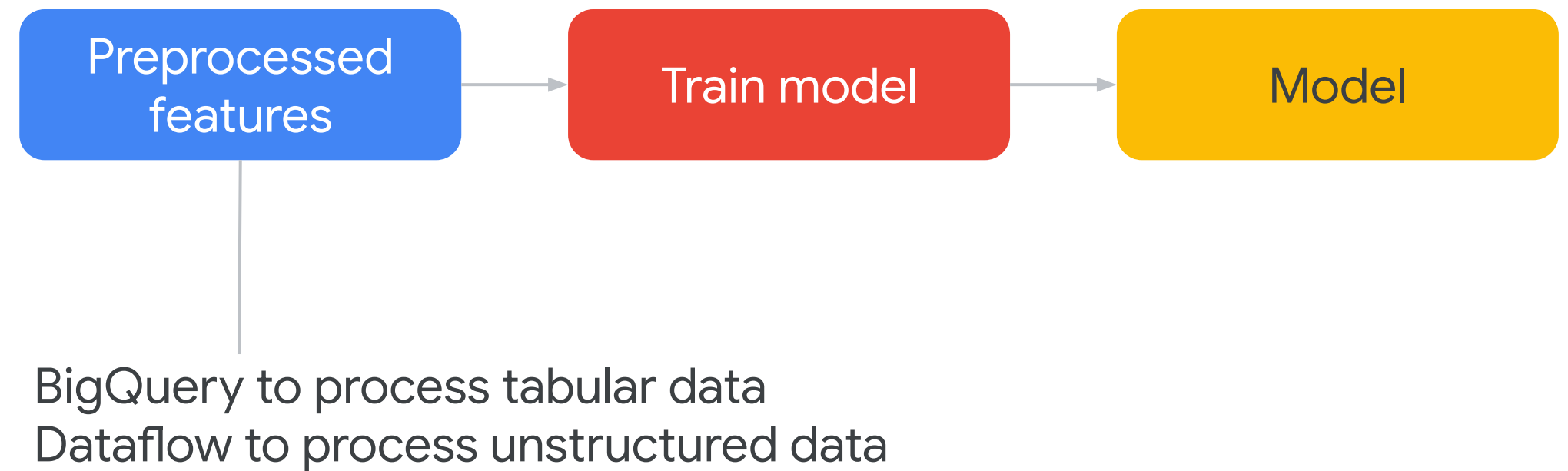


[Vertex AI TensorBoard](#) service lets you track experiment metrics such as loss and accuracy over time, visualize a model graph, project embeddings to a lower dimensional space, and much more.

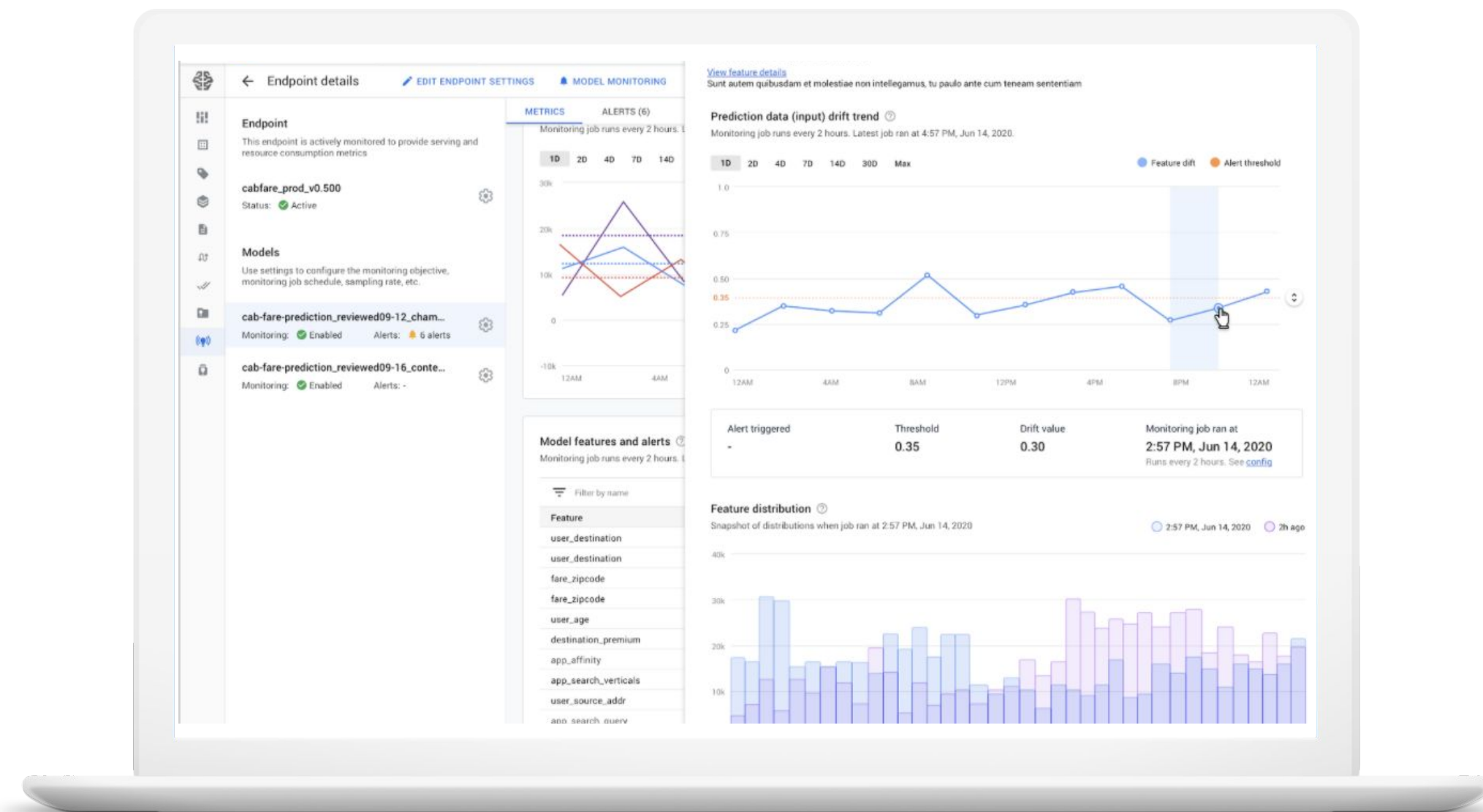
# Data preprocessing best practices



For training and evaluation, we need  
**preprocessed features**



# Best practices: Data preprocessing



## Dataflow

Use Dataflow to process unstructured data.

## TensorFlow Extended

Use TensorFlow Extended when leveraging TensorFlow ecosystem.

# Data preprocessing with BigQuery

## BigQuery

Use BigQuery to  
process tabular data.

If you're using tabular data, use BigQuery for data processing and transformation steps.

When you're working with ML, use BigQuery ML in BigQuery. Perform the transformation as a normal BigQuery query, then save the results to a [permanent table](#).

# Using managed datasets in Vertex AI

## Managed datasets

Use managed datasets to link data to your models.

Managed datasets:

- Enable you to create a clear link between your data and custom-trained models,
- Provide descriptive statistics and automatic or manual splitting into train, test, and validation sets.
- Are not required to use Vertex AI.

# Transforming unstructured data with Dataflow

## Dataflow

Use Dataflow to process unstructured data.

Use Dataflow to convert the unstructured data into binary data formats like TFRecord, which can improve performance of data ingestion during training.

If you need to perform transformations that are not expressible in Cloud SQL or are for streaming, you can use a combination of Dataflow and the [pandas](#) library.



# TensorFlow Extended

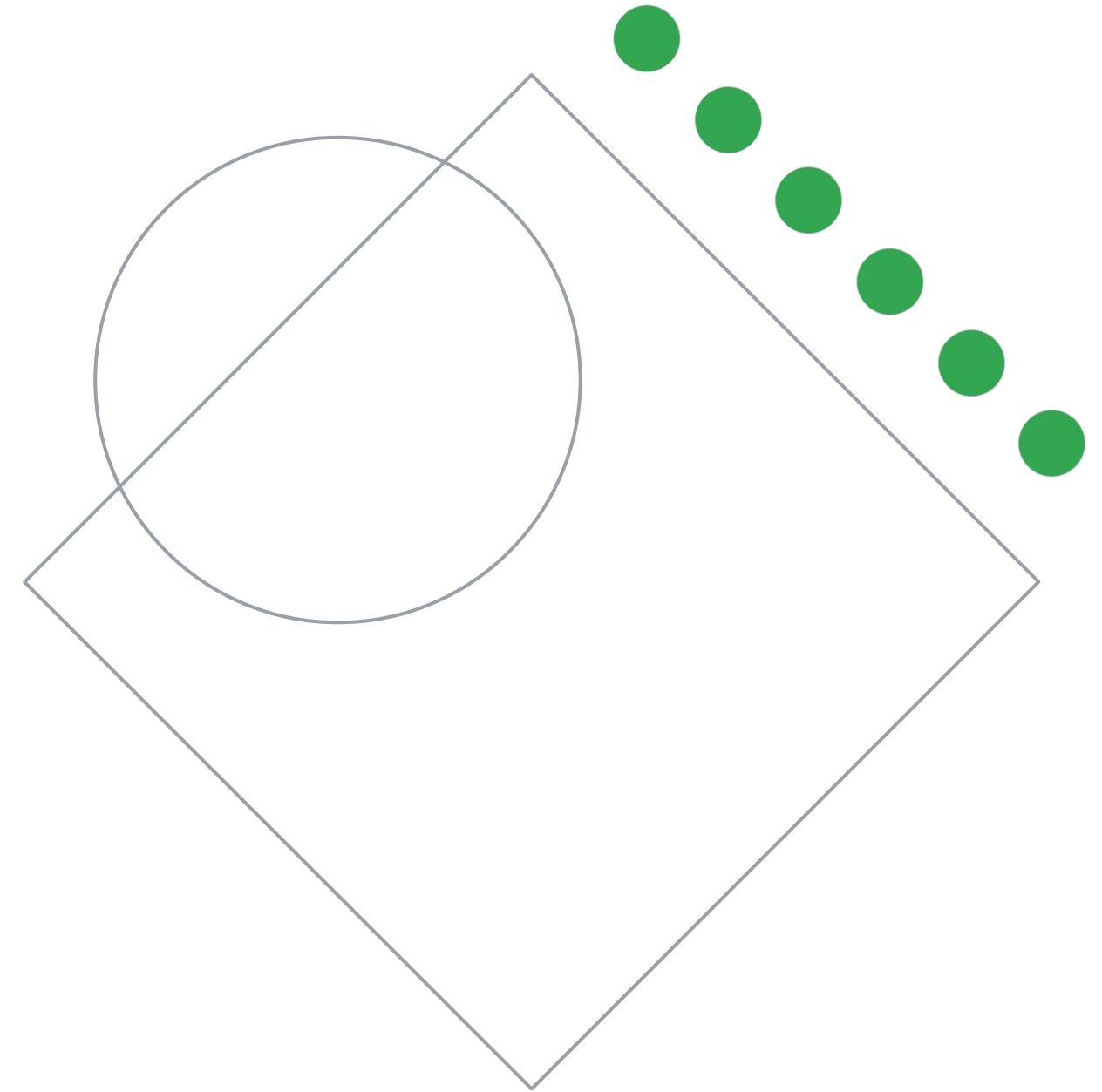
## TensorFlow Extended

Use TensorFlow Extended when leveraging TensorFlow ecosystem.

If you're using TensorFlow for model development, use [TensorFlow Extended](#) to prepare your data for training.

[TensorFlow Transform](#) is the TensorFlow component that enables defining and executing a preprocessing function to transform your data.

# Best practices for ML environment setup



# Best practices: ML environment setup

## Workbench Notebooks

Use for development and experimentation. Create NB for each team member. Use Vertex SDK for Python.

## Security

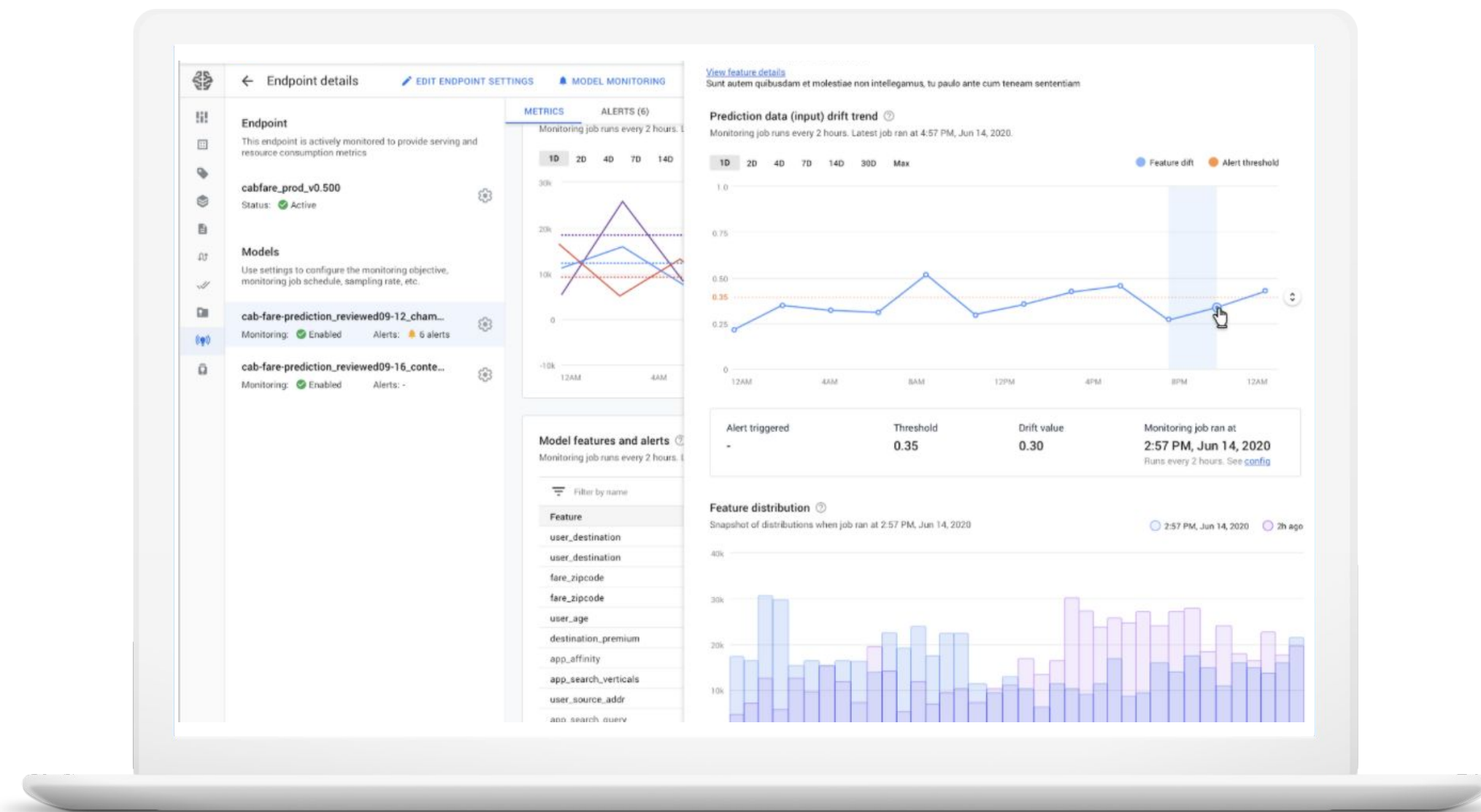
Secure PII in Notebooks.

## Data & model

Store prepared data and model in same project.

## Optimize performance & cost

Optimize performance and cost.



# Workbench Notebooks

## Workbench Notebooks

Use for development and experimentation. Create NB for each team member. Use Vertex SDK for Python

Use [Notebooks](#) for [experimentation](#) and development, including writing code, starting jobs, running queries, and checking status.

[Create a new notebook instance](#) for each member of your data science team.

# Secure PII in Notebooks

Security

Secure PII in Notebooks

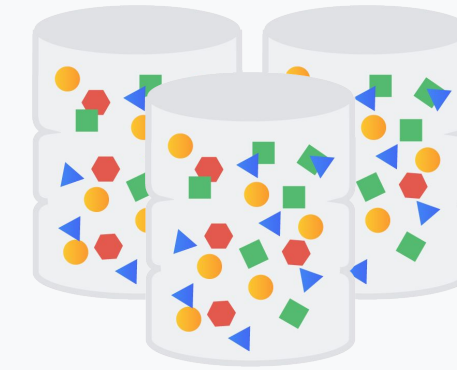
Apply data governance and security policies to help protect your Notebooks that contain personally identifiable information (PII) data - see [Notebooks security blueprint: Protecting PII data guide](#).

# Best practices: ML environment setup

## Data & model

Store prepared data and model in same project.

Your Google Cloud project



Your prepared data  
Cloud Storage or BigQuery

Access all of the datasets required for modeling.  
Store prepared data in your Google Cloud project.  
However, different parts of your organization might store their data in different projects, then rely on raw data from different projects.

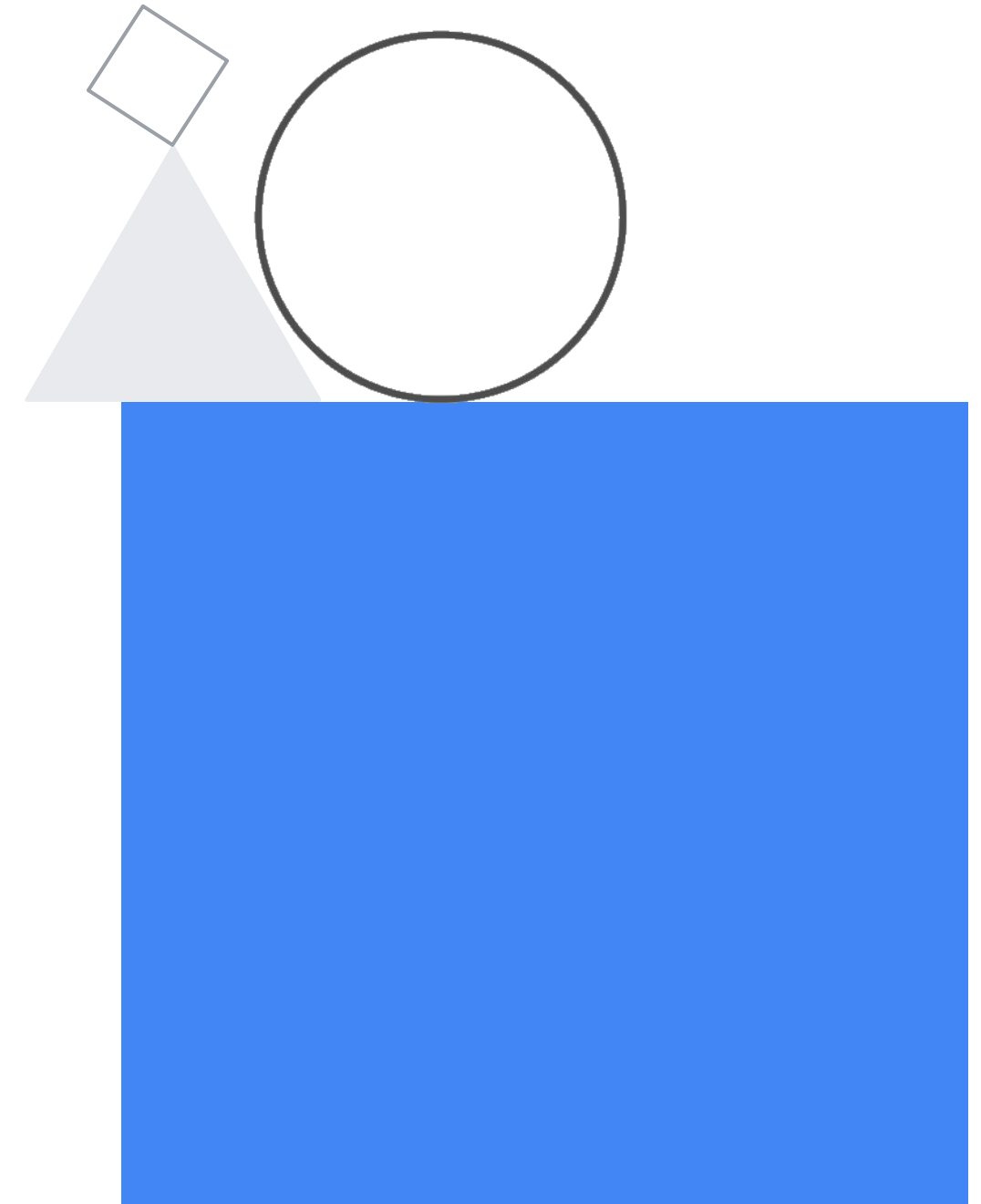
# Best practices: ML environment setup

Optimize performance & cost

Optimize performance and cost.

Enhancing the performance and decreasing the cost of your machine learning workloads is a comprehensive subject, and out of scope for this course.

# Best practices for ML model deployment and serving





# Best practices:

## Model deployment and serving

### Machine type

Specify the number and type of machines you need.

### Model inputs

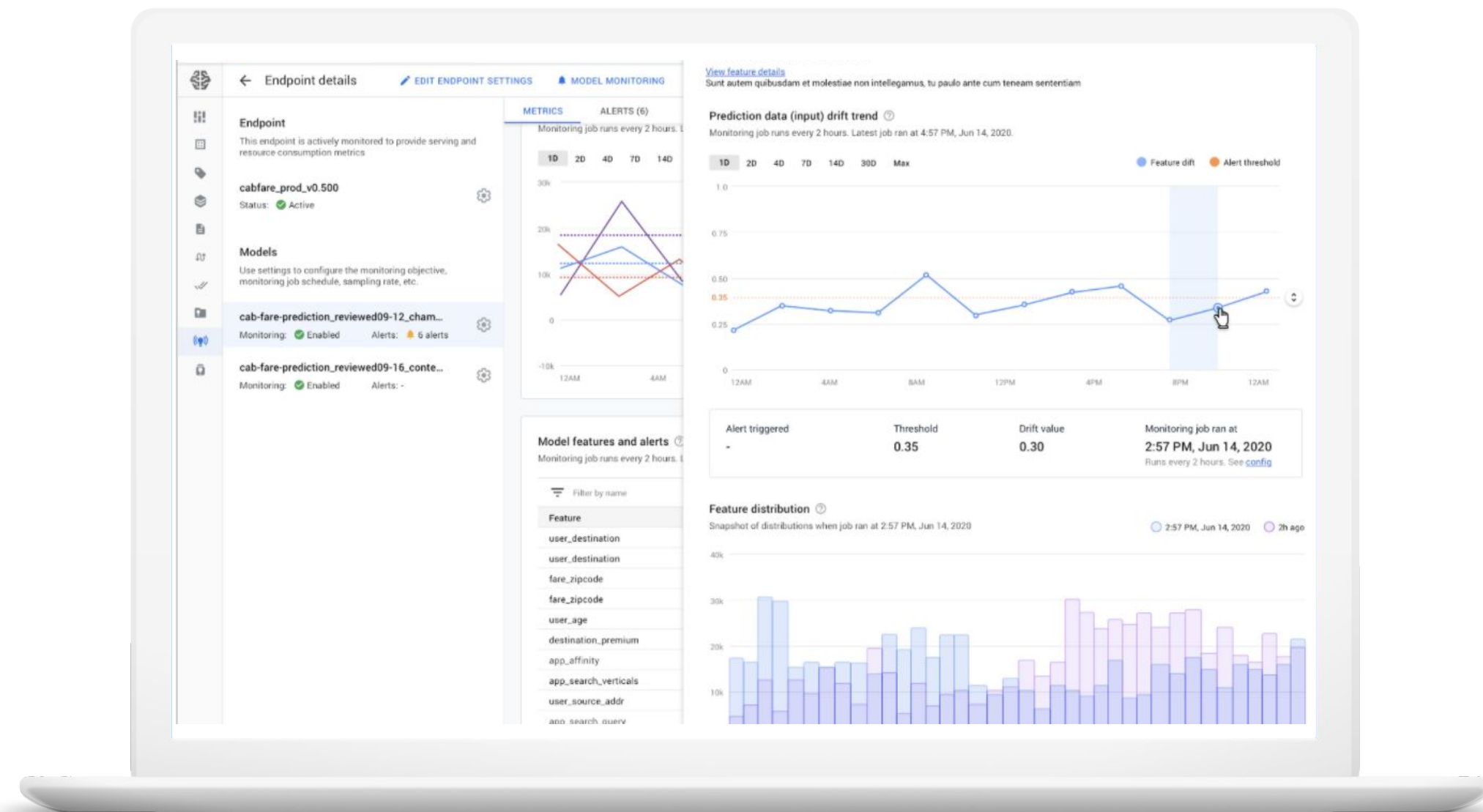
Plan inputs to the model.

### Automatic scaling

Turn on automatic scaling.

### Specify performance requirements

Define what is good and bad performance.

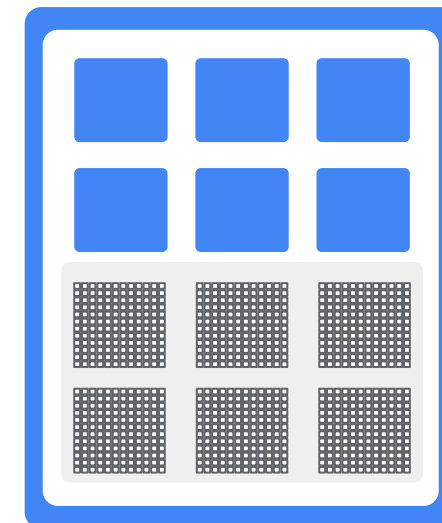


# Best practices: Model deployment and serving

## Machine type

Specify the number and type of machines you need.

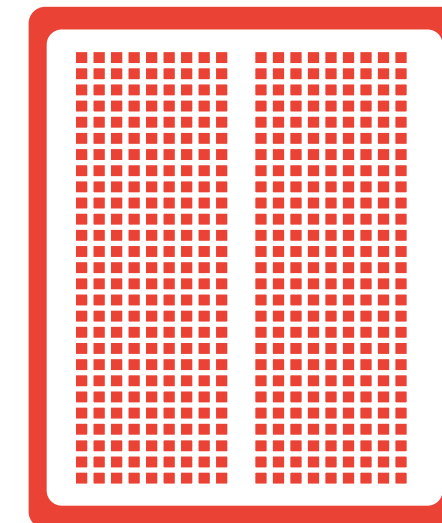
CPU



Host

Optimized for widely-used  
serial tasks

GPU



Accelerator

Optimized for many  
parallel distributed tasks

+

To deploy your model for prediction, choose hardware that is appropriate for your model, like different central processing unit (CPU) virtual machine (VM) types or graphics processing unit (GPU) types.

# Best practices: Model deployment and serving

## Model inputs

Plan inputs to the model.

If you're using batch prediction you can fetch data from the data lake, or from [Vertex AI Feature Store batch serving API](#).

If you are using online prediction, send input instances to the service and it returns your predictions in the response.

# Best practices: Model deployment and serving

Automatic scaling

Turn on automatic scaling

Turn on automatic scaling by setting minimum and maximum nodes.

# Best practices: Model deployment and serving

## Specify performance requirements

Define what is good and bad  
performance.

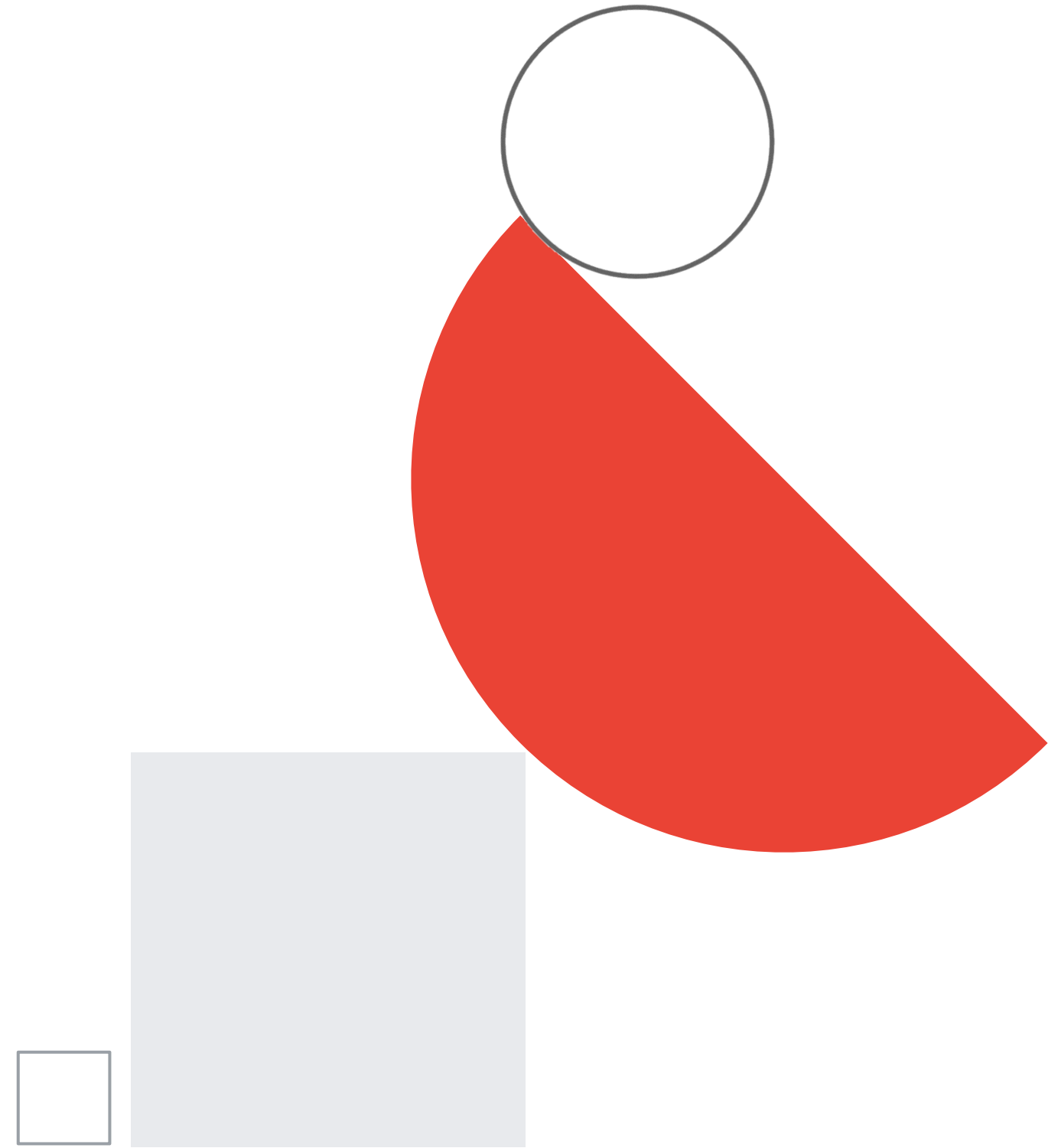
When to retrain?

What constitutes “good” and “bad” performance?

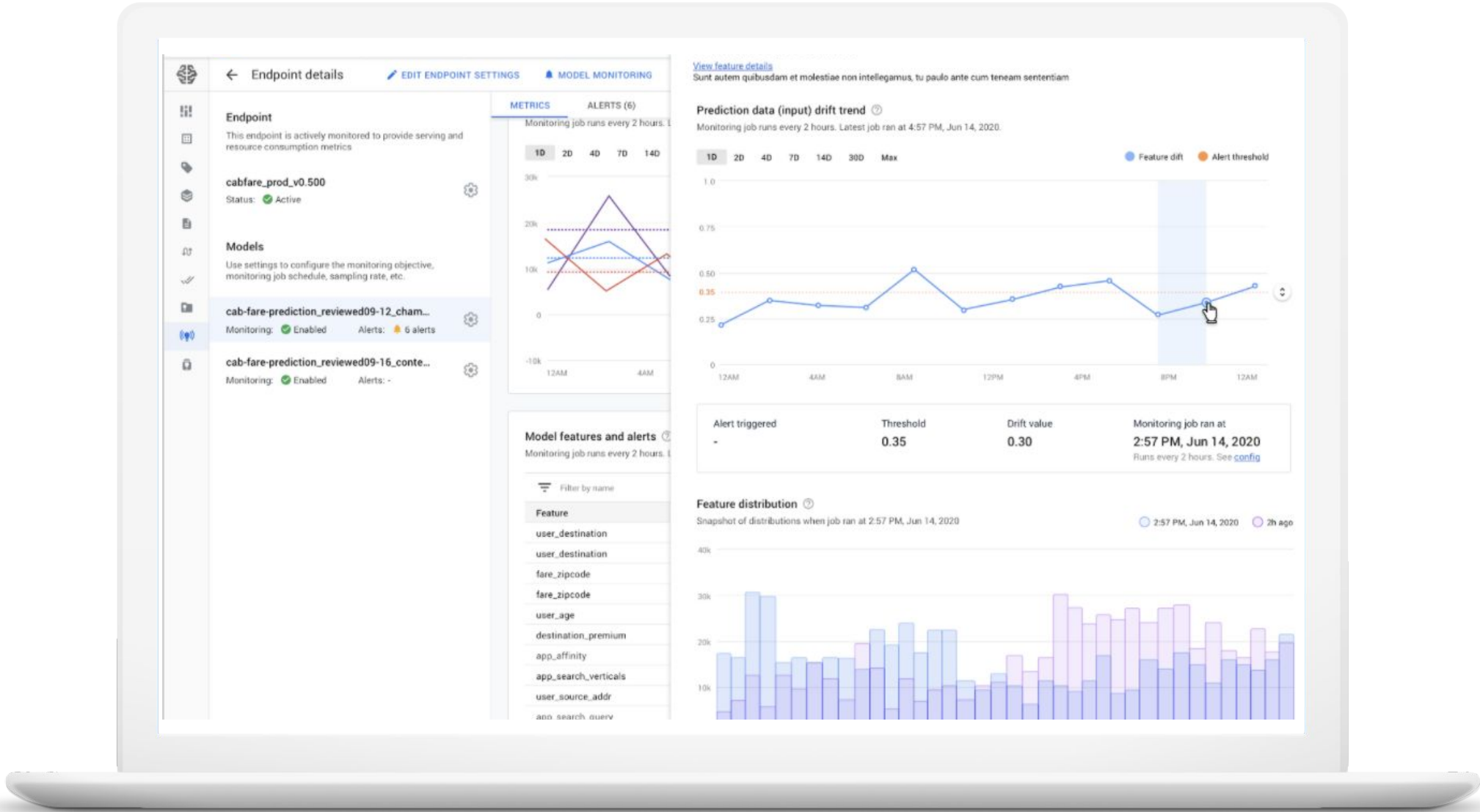
What business metric do you want to optimize?

Should you retrain the model again before serving  
to capture “baseline” metrics?

# Best practices for ML model monitoring



# Best practices: Model monitoring



## Skew

Use skew detection.

## Data drift

Use feature attributions to detect data drift.

## Alert thresholds

Fine tune alert thresholds.

## Model inputs

Track model inputs.

# Best practices: Model monitoring

## Skew

Use skew detection.

Set up the model monitoring job by providing a pointer to the training data that you used to train your model.

If you do not have access to the training data, turn on drift detection so that you'll know when the inputs change over time.



# Best practices: Model monitoring

## Drift Detection

Look for drift in production data.

Drift occurs when the statistical properties of the inputs and the target, which the model is trying to predict, change over time in unforeseen ways.

This causes problems because the predictions could become less accurate as time passes.

# Best practices: Model monitoring

## Alert thresholds

Know when skew or drift occurs in your data.

Tune threshold alerts, which are determined by the use case, the user's domain expertise, and by initial model monitoring metrics.

# Best practices: Model monitoring

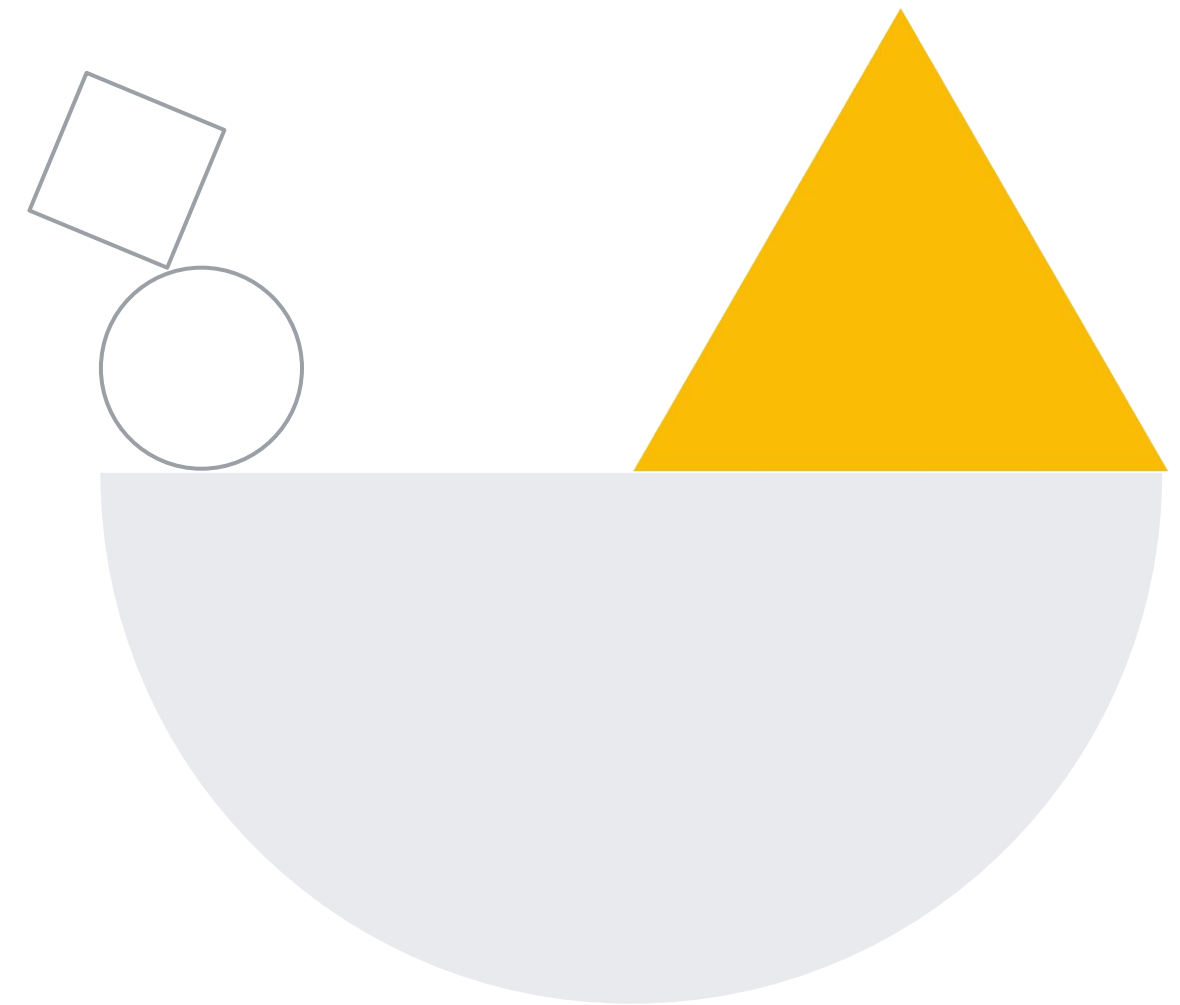
## Model Inputs

How are you going to track inputs to the model?

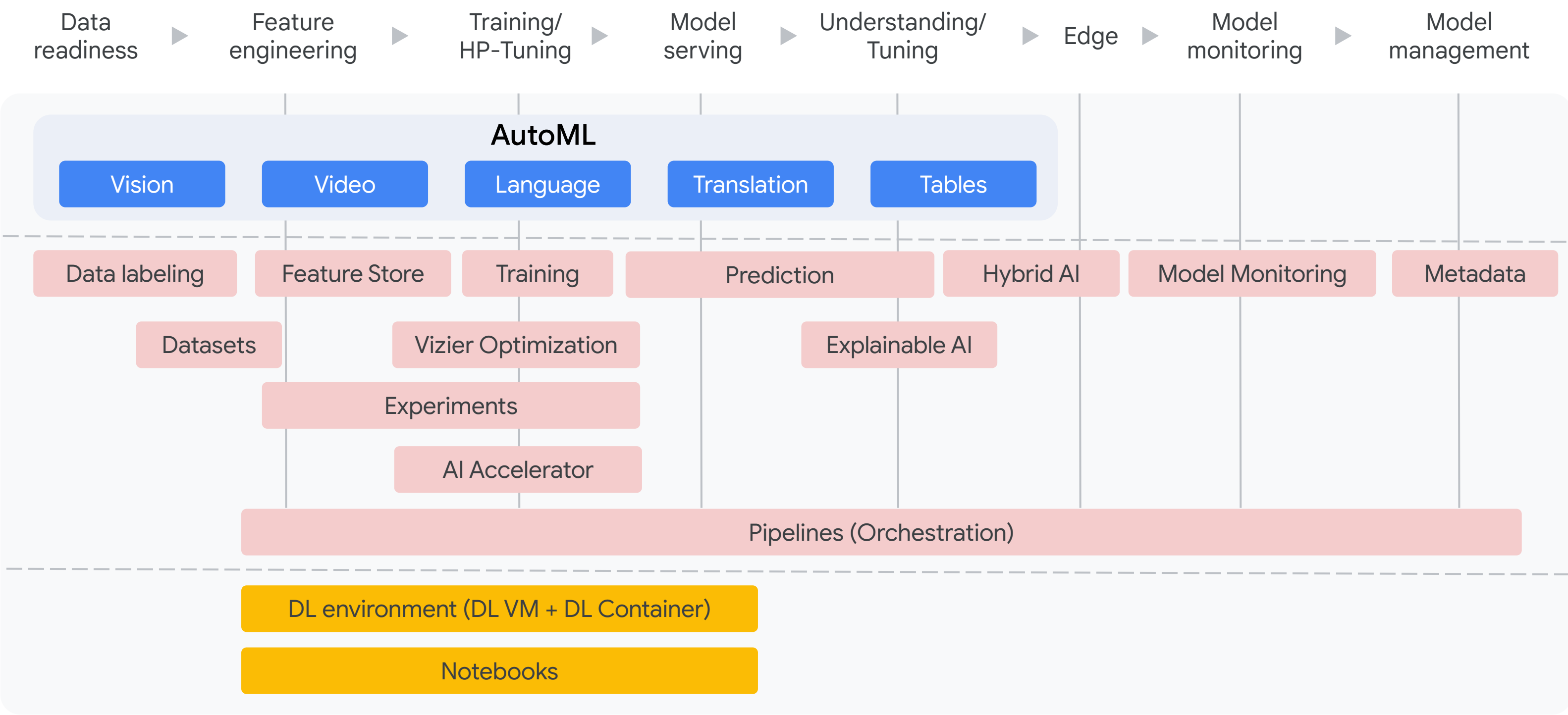
Determine how you're going to pass inputs to the model. If you're using batch prediction you can fetch data from the data lake, or from [Vertex AI Feature Store batch serving API](#).

If you are using online prediction, you can send input instances to the service and it returns your predictions in the response.

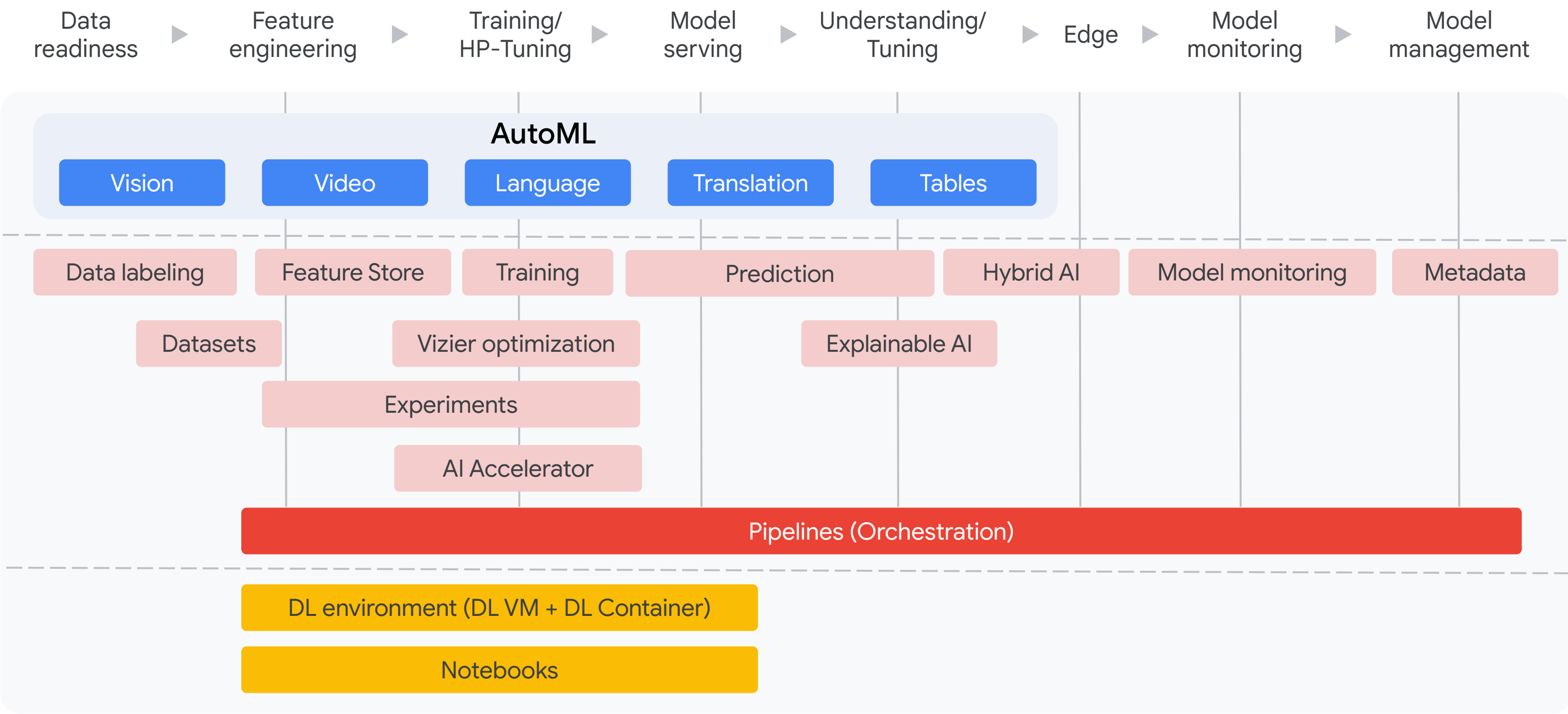
# Vertex AI Pipeline best practices



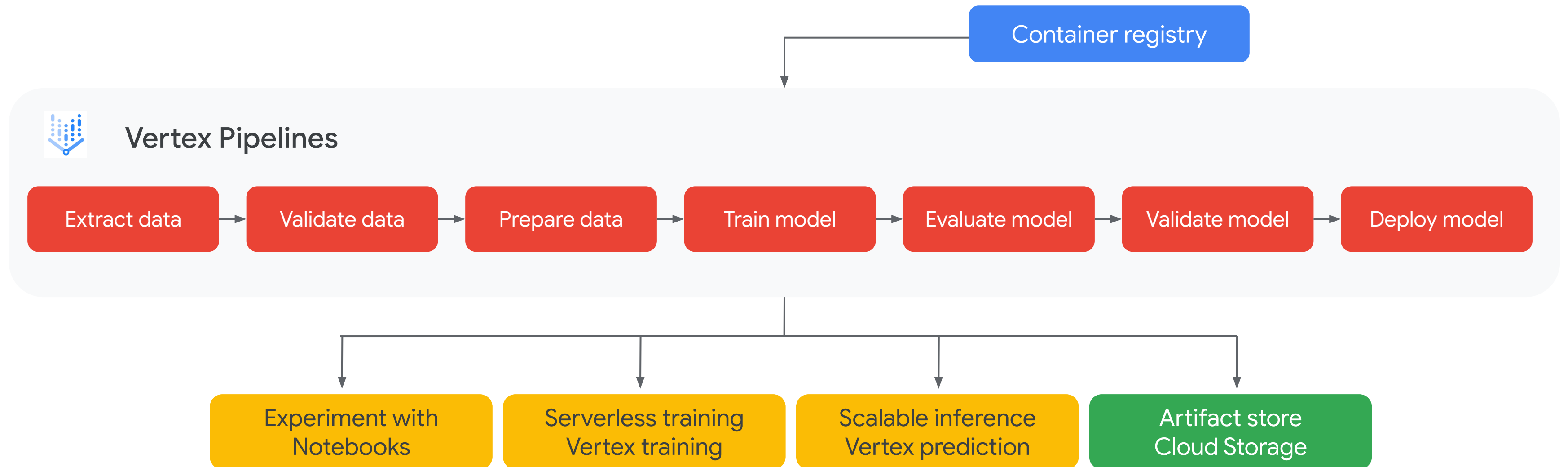
# What's included in Vertex AI?



# What's included in Vertex AI?

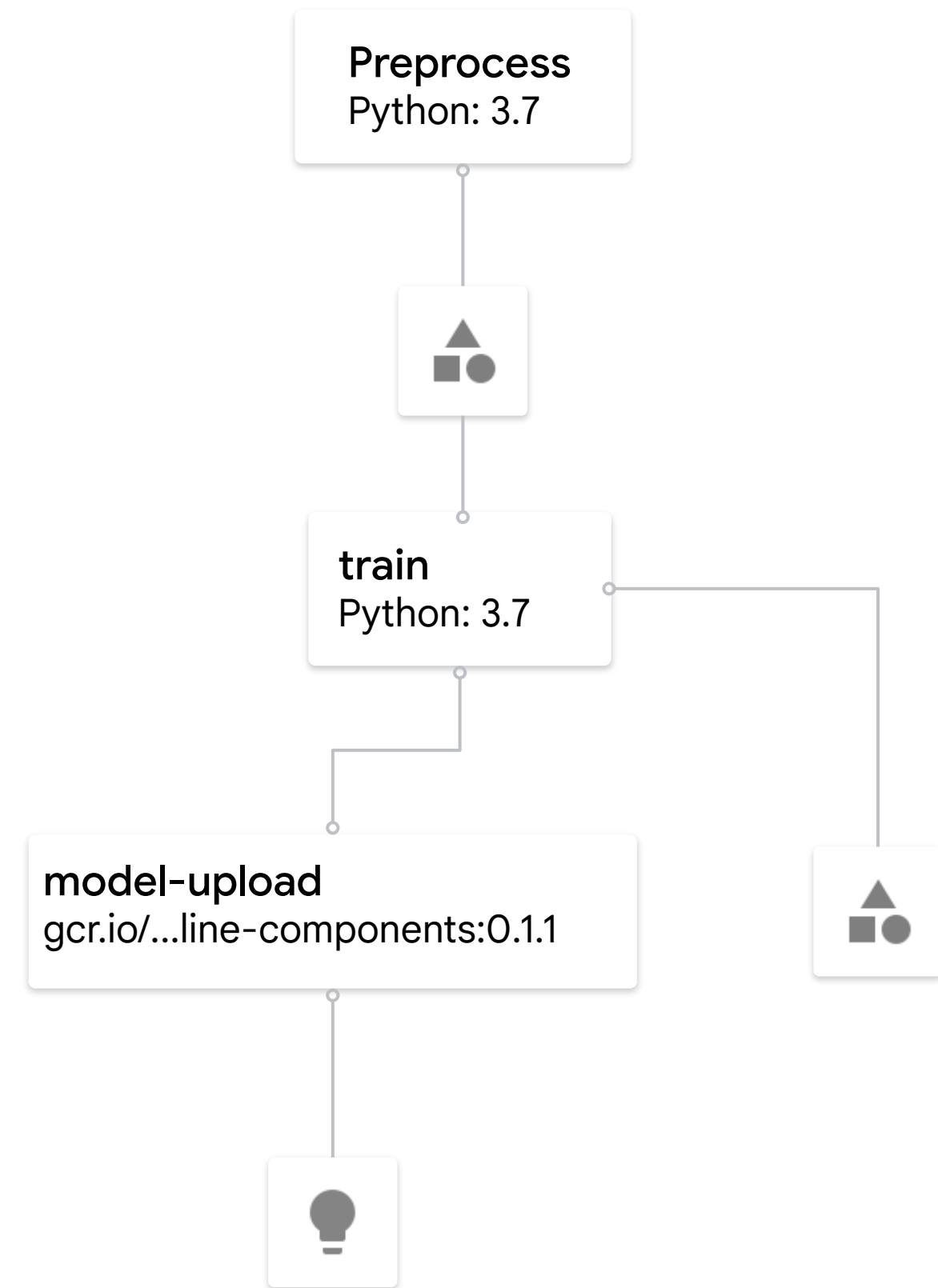


# Pipelines automate the training and deployment of models



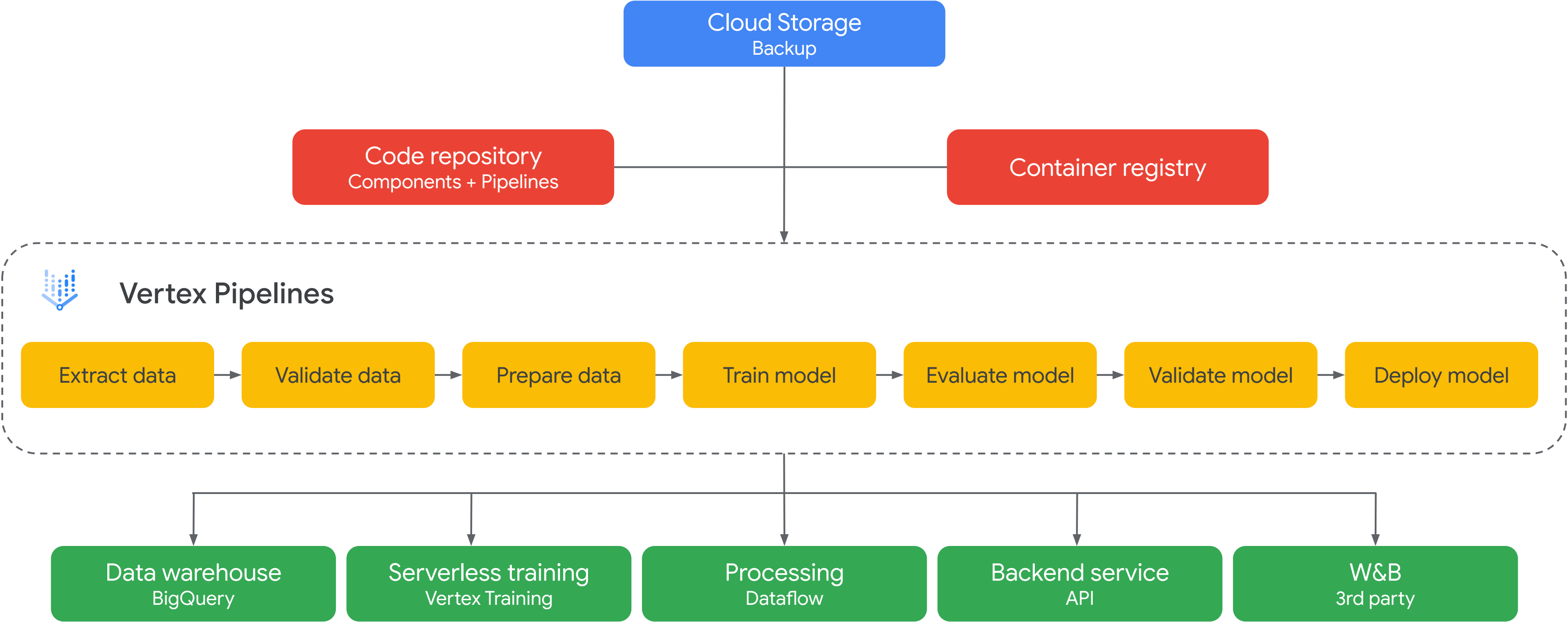
# Pipeline

- A pipeline is composed of modular components
- A pipeline offers automation and orchestration
- Components are chained with dsl to form a pipeline



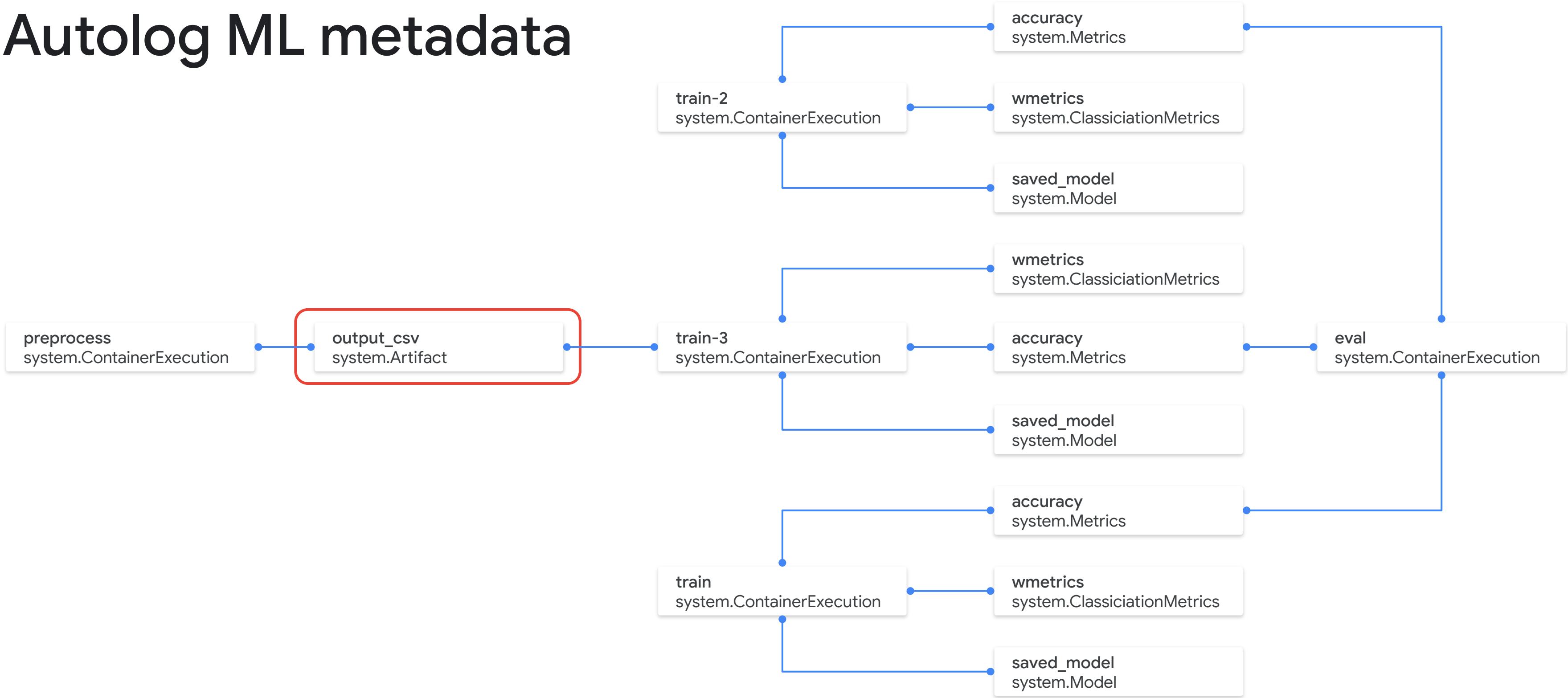


# Supporting architecture



# Data-rich formats

## Autolog ML metadata



# Best practices: Vertex AI Pipelines

## Assess perfection

Why did a pipeline produce an especially accurate model?

## Compare pipelines

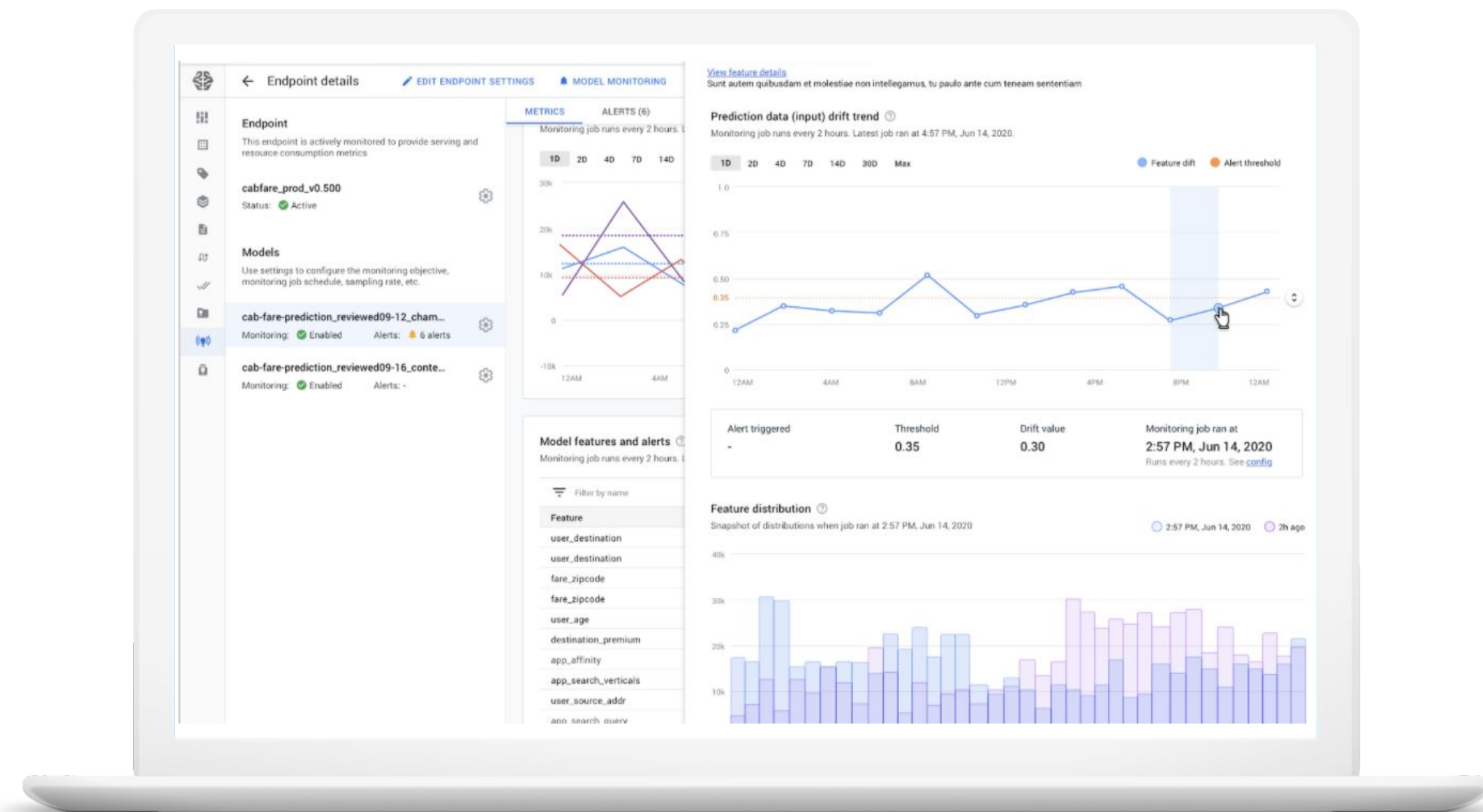
Which pipeline run produced the most accurate model and parameters used?

## System governance

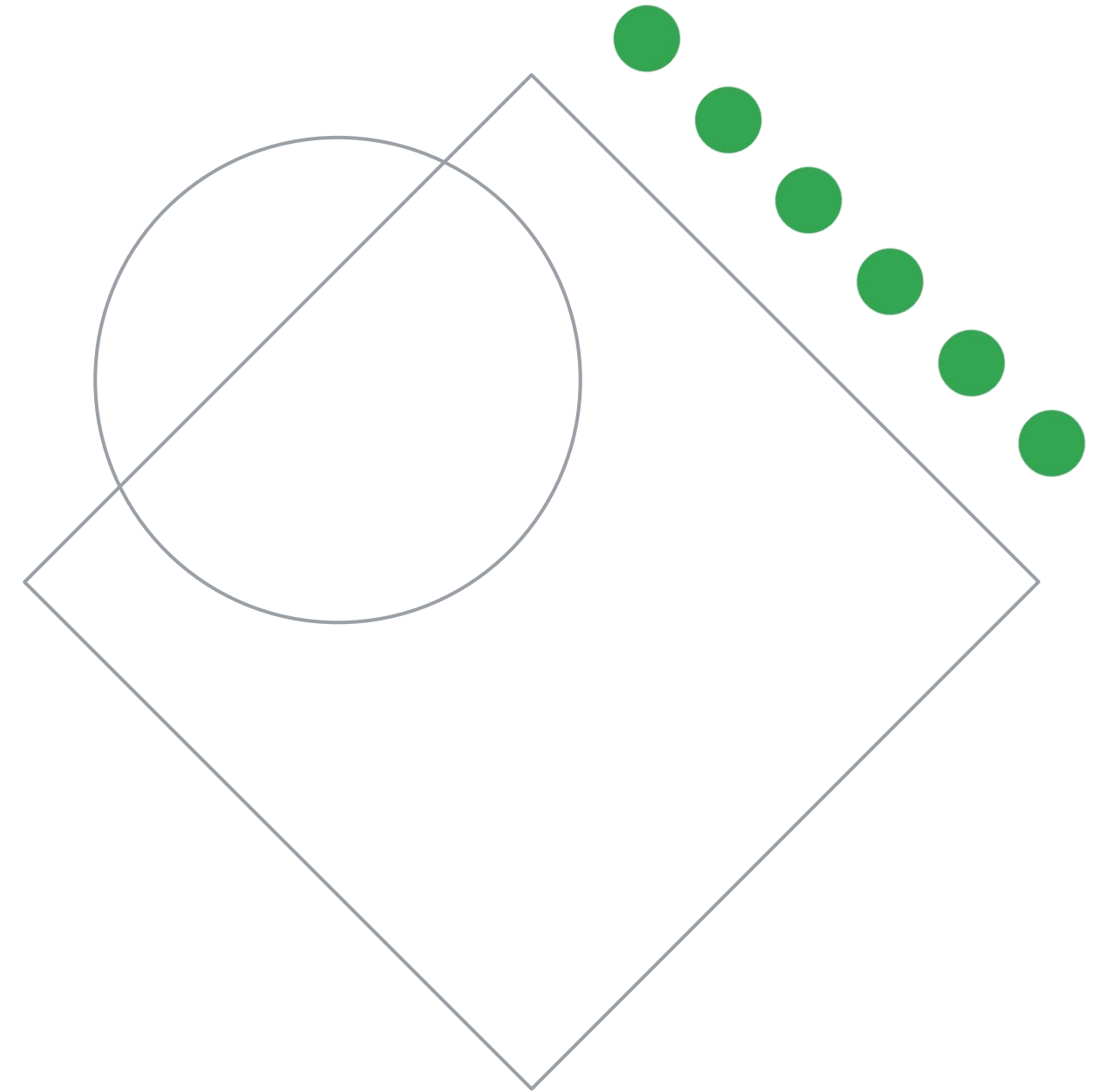
Which version of your model is in production at a given time?

## Pipeline SDK

Kubeflow SDK  
TensorFlow Extended



# Artifact organization best practices



# Best practices: Artifact organization

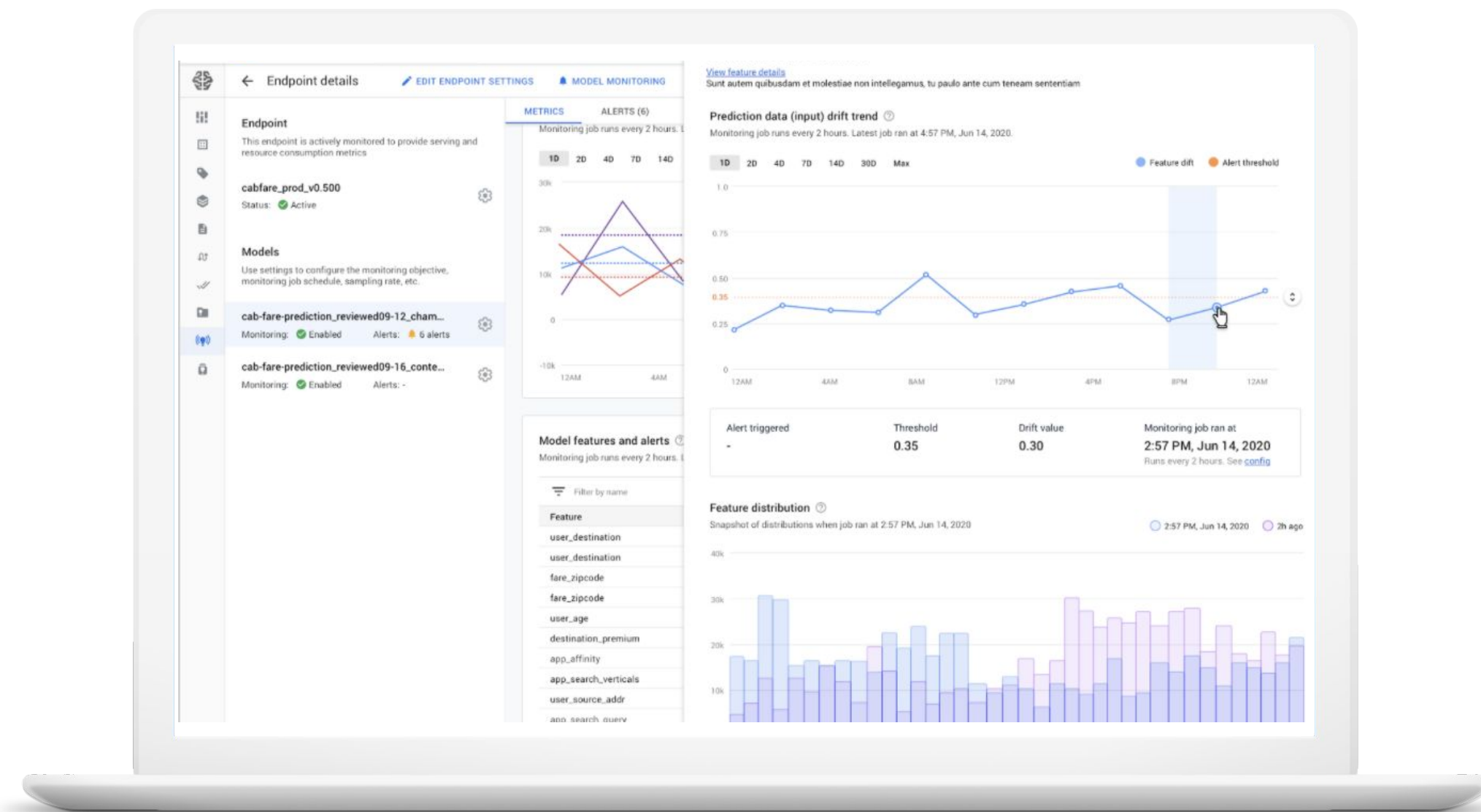
Artifact lineage describes all the factors that resulted in an artifact.

You can understand differences in performance or accuracy over several pipeline runs.

A model's lineage could include the following:

- The training, test, and evaluation data used to create the model.
- The hyperparameters used during model training.
- The code that was used to train the model.
- Metadata recorded from the training and evaluation process.
- Artifacts that descend from this model.

# Best practices: Artifact organization



## Artifacts

Organize your ML model artifacts.

## Git repo

Use a Git repo for pipeline definitions and training code.

Artifacts are outputs resulting from each step in the ML workflow.

# Best practices:

## Artifact organization

### Git repo

Use a Git repo for pipeline definitions and training code.

You can use Git to version control your ML pipelines and the custom components you build for those pipelines.

# Best practices: Artifact organization

## Git repo

Use a Git repo for pipeline definitions and training code

