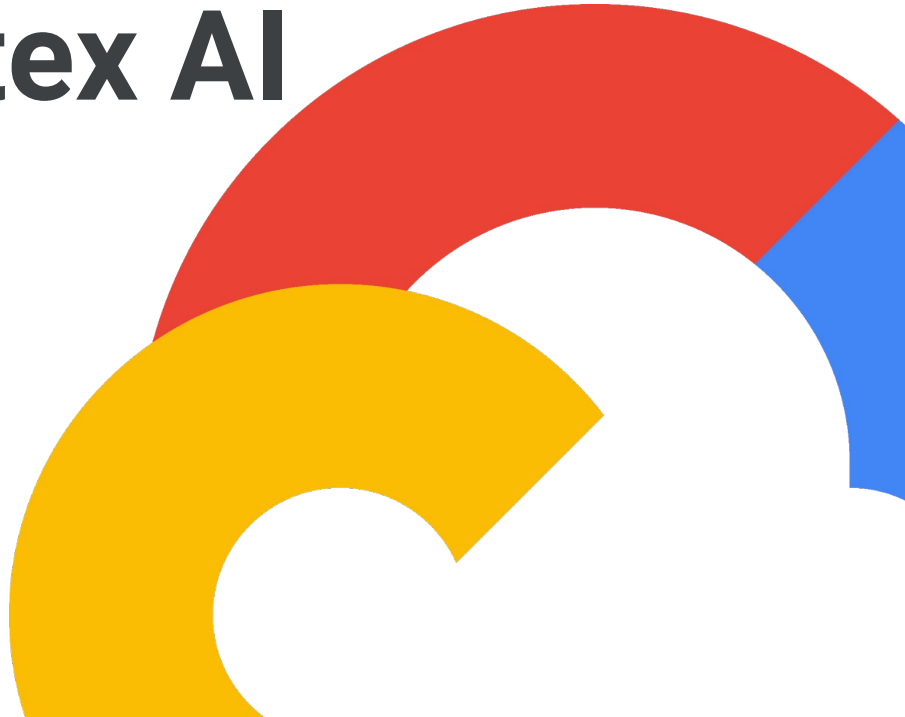
MLOps on Vertex AI

Oct 2024
Hangsik Shin

Google Cloud



Agenda

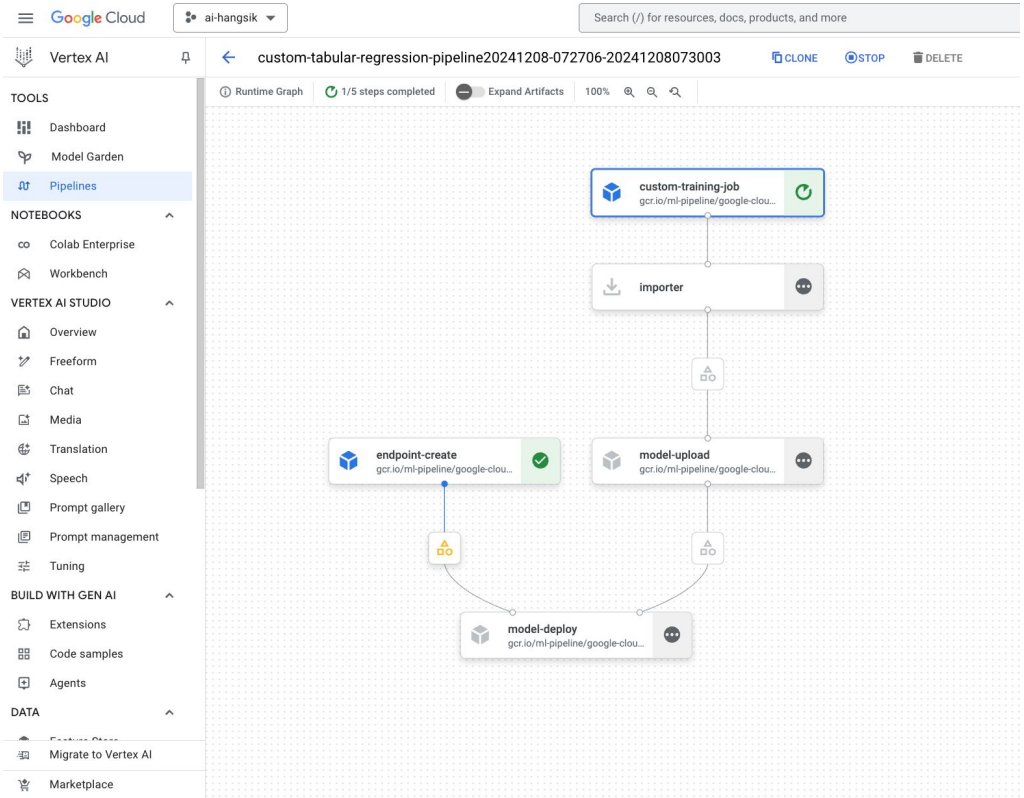
시간	Subject	Description	Time
10:00 ~ 12:00	전체적인 workshop 설명	환경셋팅, Colab 환경. AutoML training 실행.	30분
	Vertex AI Overview	Vertex AI Console. Dataset 설정	30분
	Pipeline 설명	Kubeflow, Experiments, Metadata	1시간
13:30 ~ 14:30	AutoML Pipeline 설명	AutoML, GCPC. Evaluation Log.	1시간.
14:30~16:00	Vertex AI Search and Grounding services	Search and grounding	1시간 30분
16:00~17:30	Vertex AI Agent를 활용한 Conversation Chatbot 개발	Reasoning engine / RAG engine	1시간 30분

Lab Guides

Lab - Github repositories

- MLOps
  - https://github.com/shins777/mlops_vertexai
- Generative AI
  - https://github.com/shins777/genai_workshop

Vertex AI Console



Run AutoML Pipeline

- https://github.com/shins777/mlops_vertexai/blob/main/02.pipeline/automl_tabular_regression_pipeline.ipynb

- Colab enterprise.
  - Runtime template configuration
  - Create runtime environment
- Walk through code
  - Structure of AutoML component.
- Run code.
- Check pipeline
- Check training

Run Kubeflow Pipeline

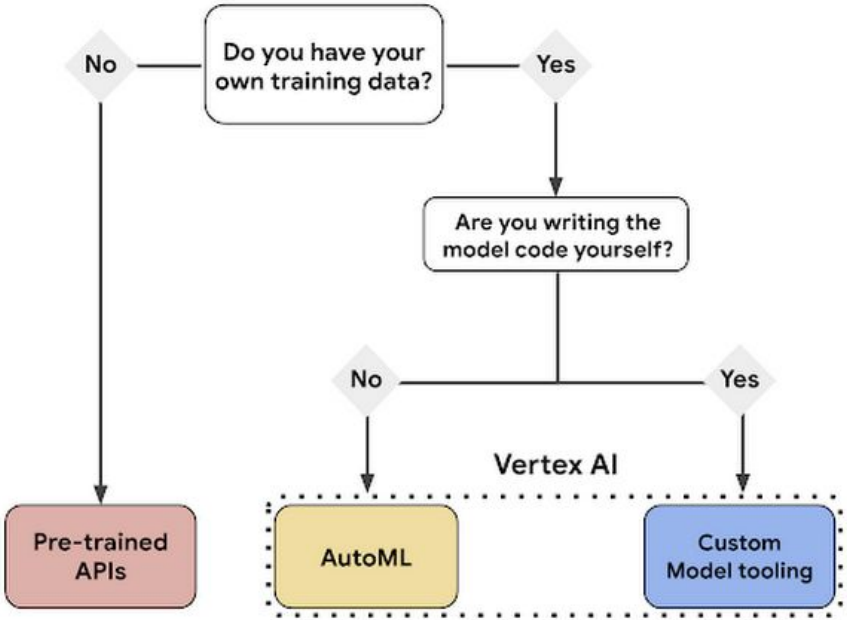
- https://github.com/shins777/mlops_vertexai/blob/main/02.pipeline/kubeflow_pipeline_vertexai.ipynb

- Colab enterprise
- Run code.
- Verify the artifacts in console.

AutoML

Two types ML training on Vertex AI

- **AutoML**: Create and train models with minimal technical knowledge and effort. AutoML beginner's guide.
- **Custom training**: Create and train models at scale using any ML framework. Custom training overview.



Types of models you can build using AutoML (Link)

Data type	Supported objectives
Image data	Classification, object detection.
Video data	Action recognition, classification, object tracking.
Text data	Classification, entity extraction, sentiment analysis. → Gemini model
Tabular data	Classification/regression, forecasting.

AutoML resources

- Vertex AI Manual
  - https://cloud.google.com/vertex-ai/docs
- aiplatform api
  - https://cloud.google.com/python/docs/reference/aiplatform/1.18.2/google.cloud.aiplatform
- Google Cloud Pipeline Components
  - https://google-cloud-pipeline-components.readthedocs.io/en/google-cloud-pipeline-components-2.16.0/api/v1/automl/index.html

Tabular data (Link)

- **Binary classification** models predict a binary outcome (one of two classes). Use this model type for yes or no questions. For example, you might want to build a binary classification model to predict whether a customer would buy a subscription. Generally, a binary classification problem requires less data than other model types.
- **Multi-class classification** models predict one class from three or more discrete classes. Use this model type for categorization. For example, as a retailer, you might want to build a multi-class classification model to segment customers into different personas.
- **Regression** models predict a continuous value. For example, as a retailer, you might want to build a regression model to predict how much a customer will spend next month.
- **Forecasting** models predict a sequence of values. For example, as a retailer, you might want to forecast daily demand of your products for the next 3 months so that you can appropriately stock product inventories in advance.

View model architecture (Link)

Provides information about how to use Cloud Logging to view details about a Vertex AI model. Using Logging, you can see:

- The hyperparameters of the final model as key-value pairs.
- The hyperparameters and object values used during model training and tuning, as well as an objective value.

Prebuilt container

- Pre-built containers for training
  - https://cloud.google.com/vertex-ai/docs/training/pre-built-containers
- Pre-built containers for prediction
  - https://cloud.google.com/vertex-ai/docs/predictions/pre-built-containers.

Development tools

Vertex AI Workbench

A one-stop surface for Data Science

Fully managed compute with admin control

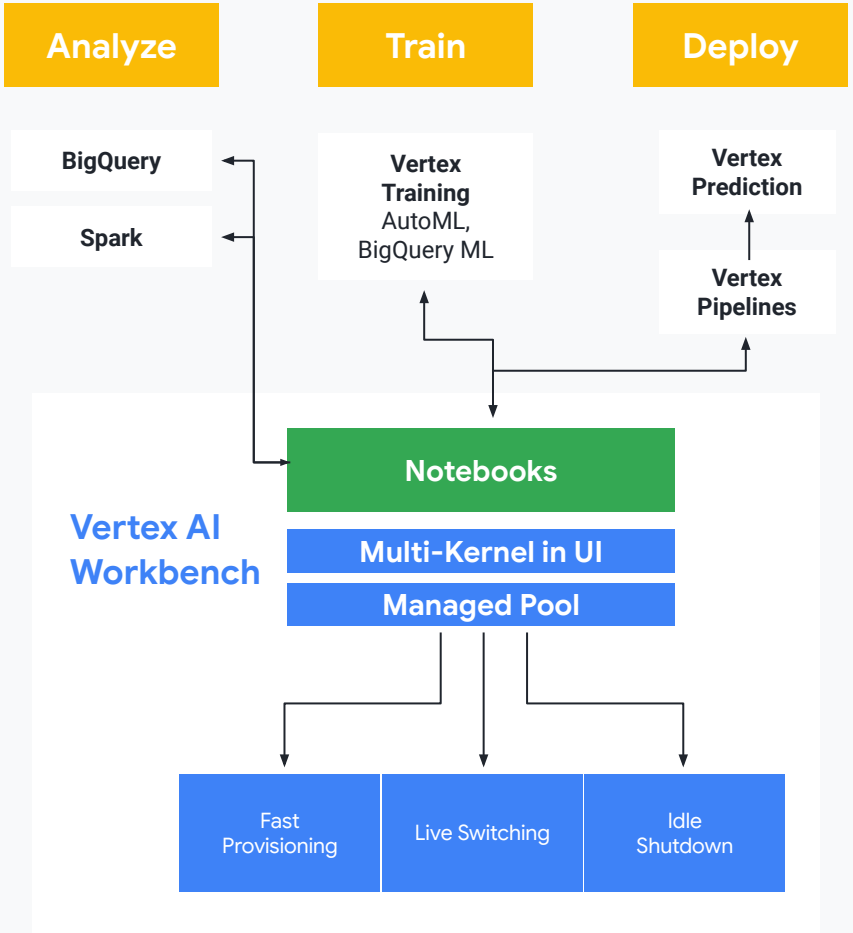
A Jupyter-based fully managed, scalable, enterprise-ready compute infrastructure with easily enforceable policies and user management

Unified Workbench for Productivity

Seamless visual and code-based integrations with data & analytics services

At-your-fingertips integration for MLOps

Load and share notebooks alongside your AI and data tasks. Run tasks without extra code



Colab Enterprise on Vertex AI

Colab Enterprise combines the ease of use of Google Colab notebooks with the enterprise-level security and compliance capabilities of Google Cloud

Collaboration & Productivity

IAM based notebook sharing

Automatic Versioning

Commenting (coming soon!)

Co-editing (coming soon!)

Generative AI powered code completion and generation

Zero-Config & Flexible Compute

Provides both zero-config compute options, as well as access to a wide range of machine-shapes and compute

Enterprise Ready

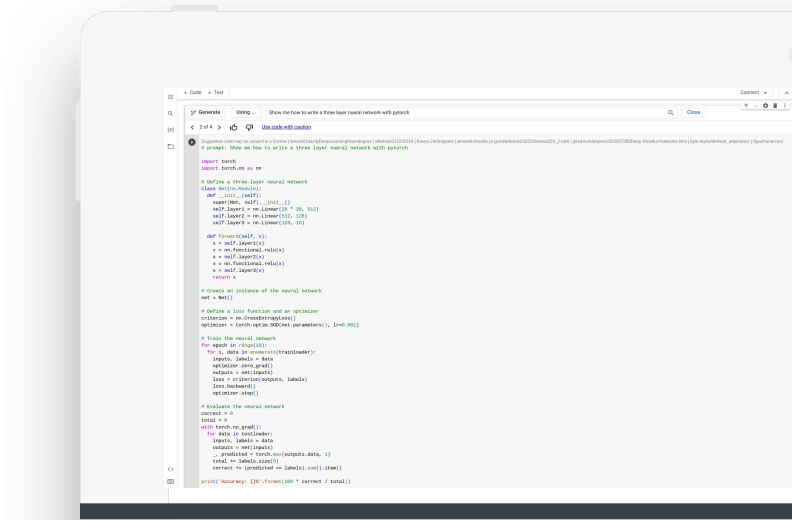
Will support a wide range of security and management capabilities including:

- VPC-SC
- CMEK
- Regionalization
- Cloud Monitoring
- Cloud Logging

Available across Google Cloud

Available in BigQuery and Vertex AI (Dataproc and Dataflow coming soon), making it easy to work across data and AI workloads

Use Cases: Data science, data analysis, data engineering, ML engineering



Machine Types (Link)

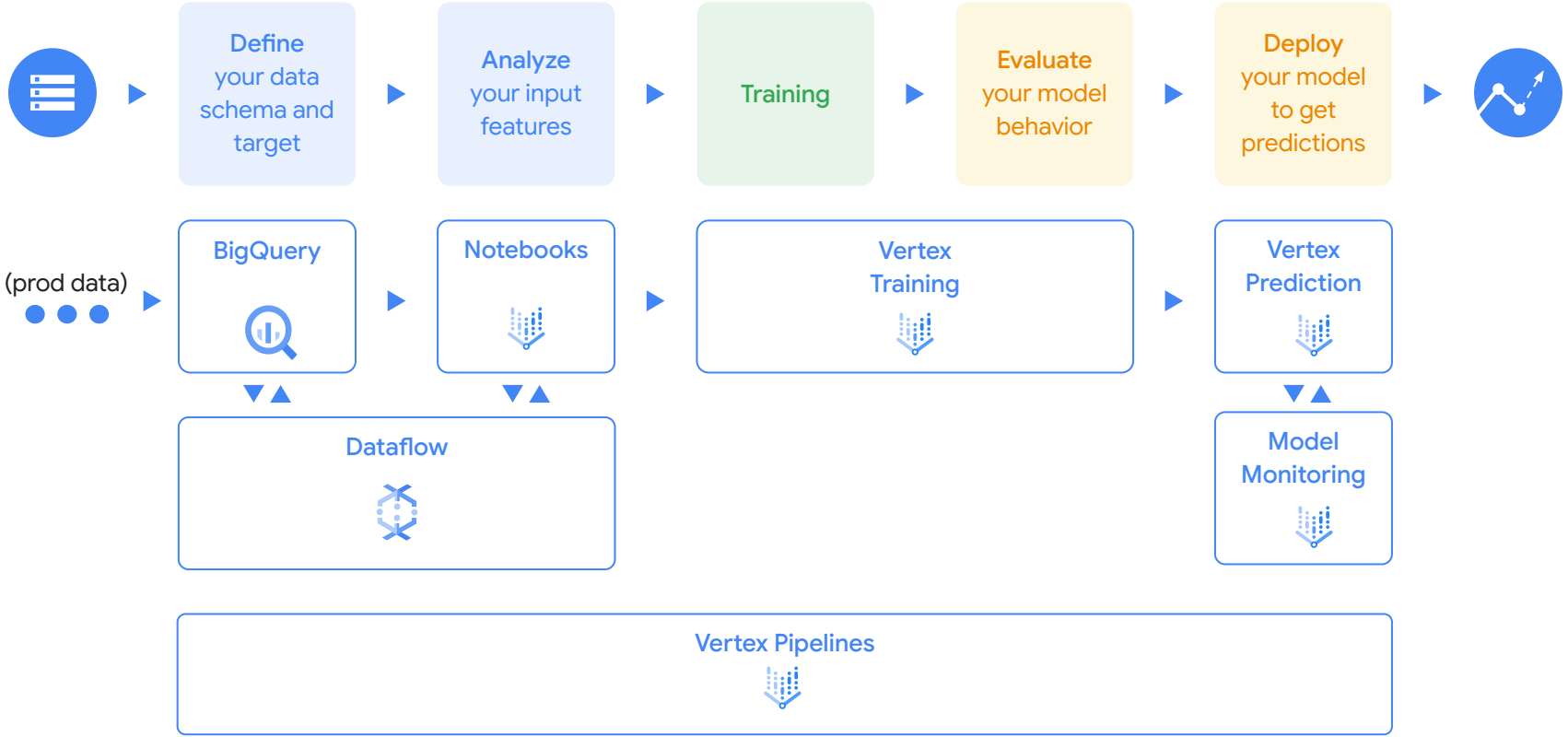
General-purpose workloads			
N4, N2, N2D, N1	**C4A, C4, C3, C3D**	**E2**	**Tau T2D, Tau T2A**
Balanced price/performance across a wide range of machine types	Consistently high performance for a variety of workloads	Day-to-day computing at a lower cost	Best per-core performance/cost for scale-out workloads
• Medium traffic web and app servers • Containerized microservices • Business intelligence apps • Virtual desktops • CRM applications • Development and test environments • Batch processing • Storage and archive	• High traffic web and app servers • Databases • In-memory caches • Ad servers • Game Servers • Data analytics • Media streaming and transcoding • CPU-based ML training and inference	• Low-traffic web servers • Back office apps • Containerized microservices • Microservices • Virtual desktops • Development and test environments	• Scale-out workloads • Web serving • Containerized microservices • Media transcoding • Large-scale Java applications

Vertex AI Overview

Tasks on Vertex AI

- AutoML lets you train tabular, image, text, or video data without writing code or preparing data splits. These models can be deployed for online prediction or queried directly for batch prediction.
- Custom training gives you complete control over the training process, including using your preferred ML framework, writing your own training code, and choosing hyperparameter tuning options. You can import your custom-trained model into the Model Registry and deploy it to an endpoint for online prediction using prebuilt or custom containers. Or you can query it directly for batch predictions.
- Model Garden lets you discover, test, customize, and deploy Vertex AI and select open-source (OSS) models and assets.
- Generative AI gives you access to Google's large generative AI models for multiple modalities (text, code, images, speech). You can tune Google's LLMs to meet your needs, and then deploy them for use in your AI-powered applications.

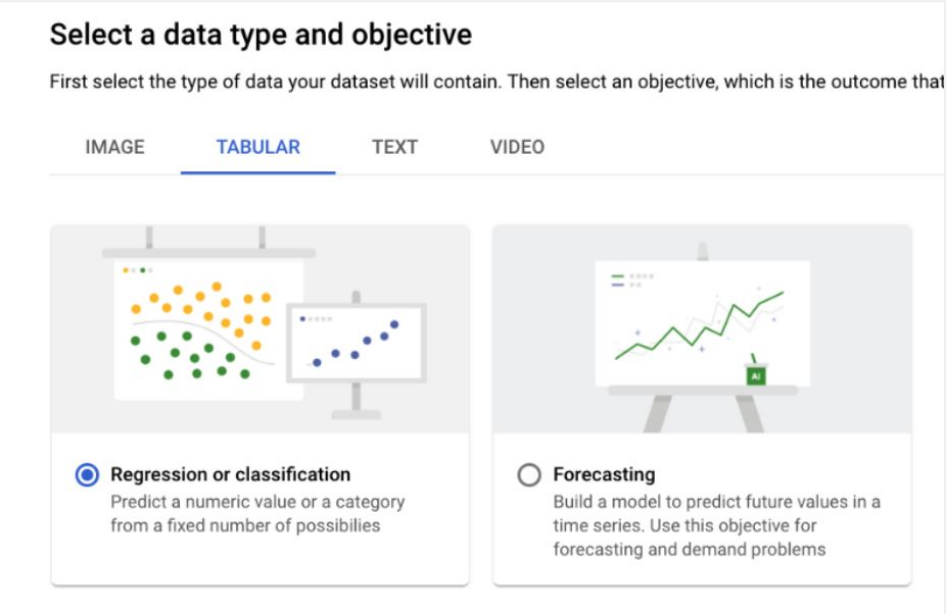
MLOps process on Vertex AI



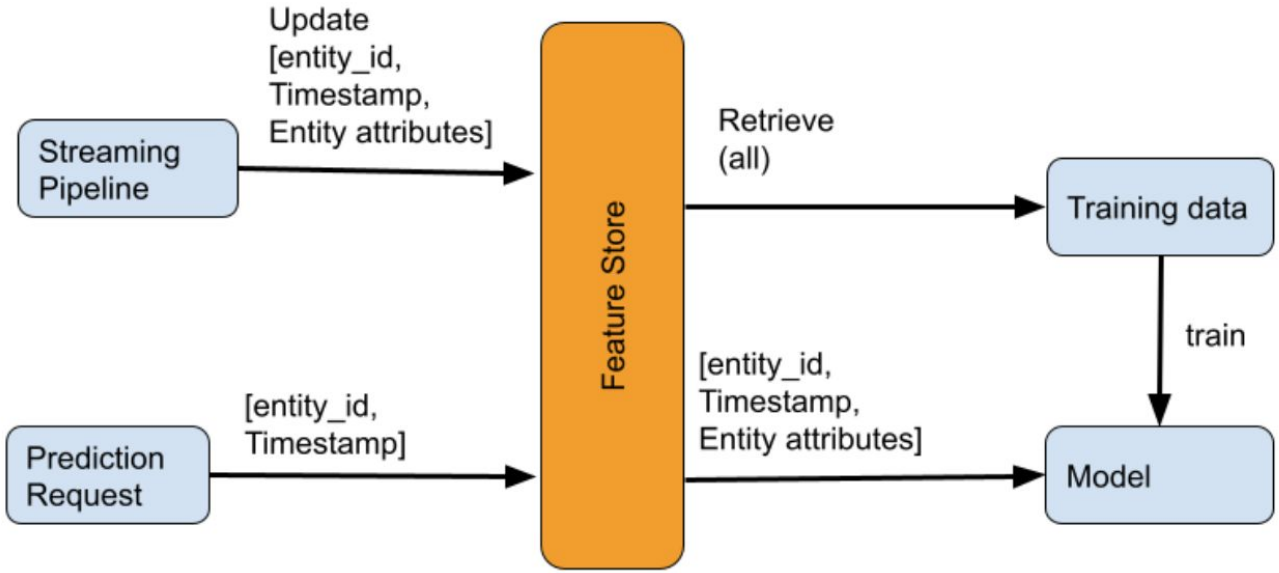
Dataset / Feature store

Dataset types

Data type	Supported objectives
Image data	Classification, object detection.
Video data	Action recognition, classification, object tracking.
Text data	Classification, entity extraction, sentiment analysis.
Tabular data	Classification/regression, forecasting.



Use feature store to train and predict



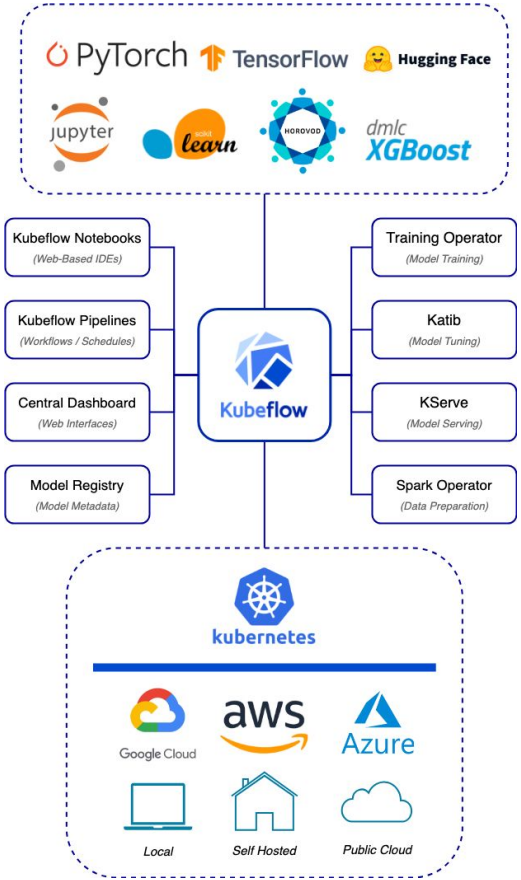
Pipeline

Kubeflow

- Kubeflow is a community and ecosystem of open-source projects to address each stage in the machine learning (ML) lifecycle with support for best-in-class open source tools and frameworks. Kubeflow makes AI/ML on Kubernetes simple, portable, and scalable.

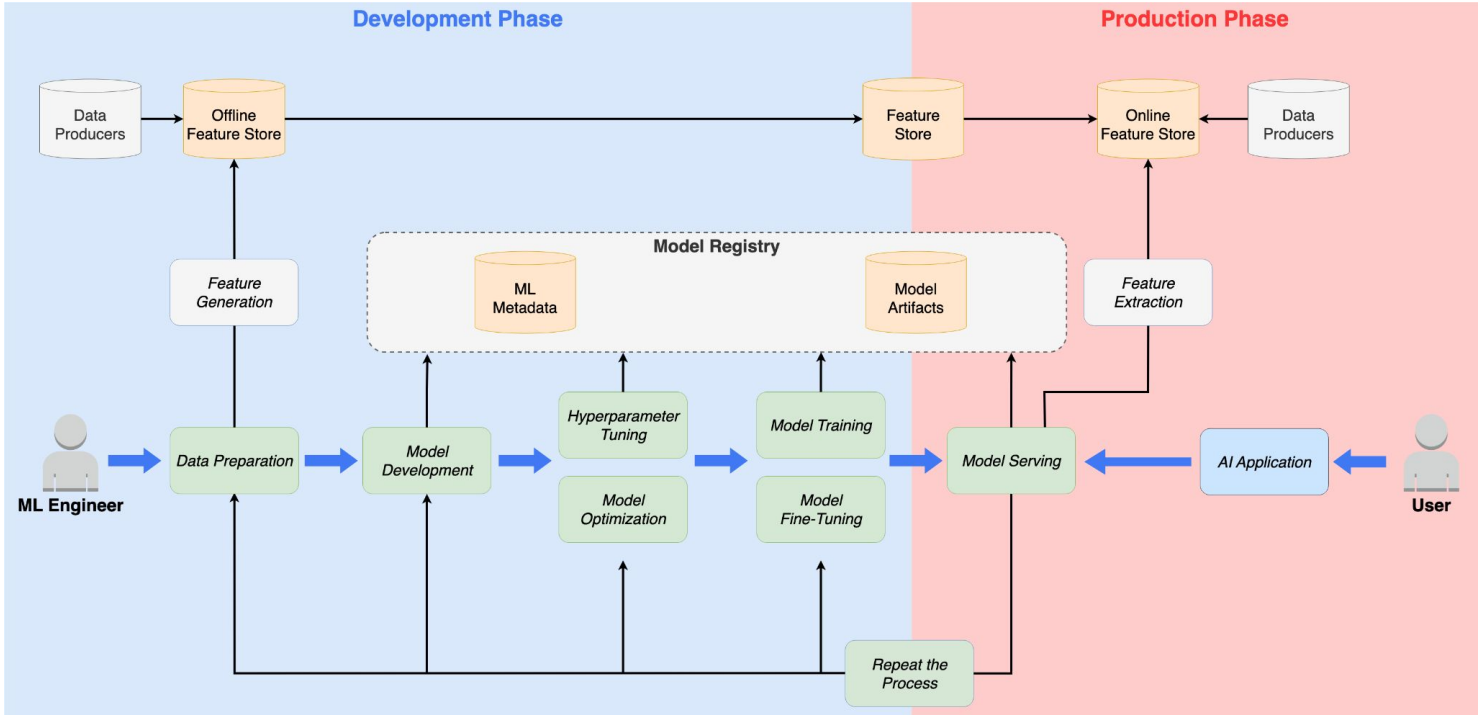
- https://googlecloudplatform.github.io/kubeflow-gke-docs/dev/docs/

- https://github.com/googlecloudplatform/kubeflow-distribution



ML Lifecycle for Production and Development Phases

The ML lifecycle for AI applications may be conceptually split between development and production phases, this diagram explores which stages fit into each phase:



Kubeflow references

Community and Support
Where to get help, contribute, and learn more

Version Compatibility
Version compatibility between KFP Runtime and KFP SDK

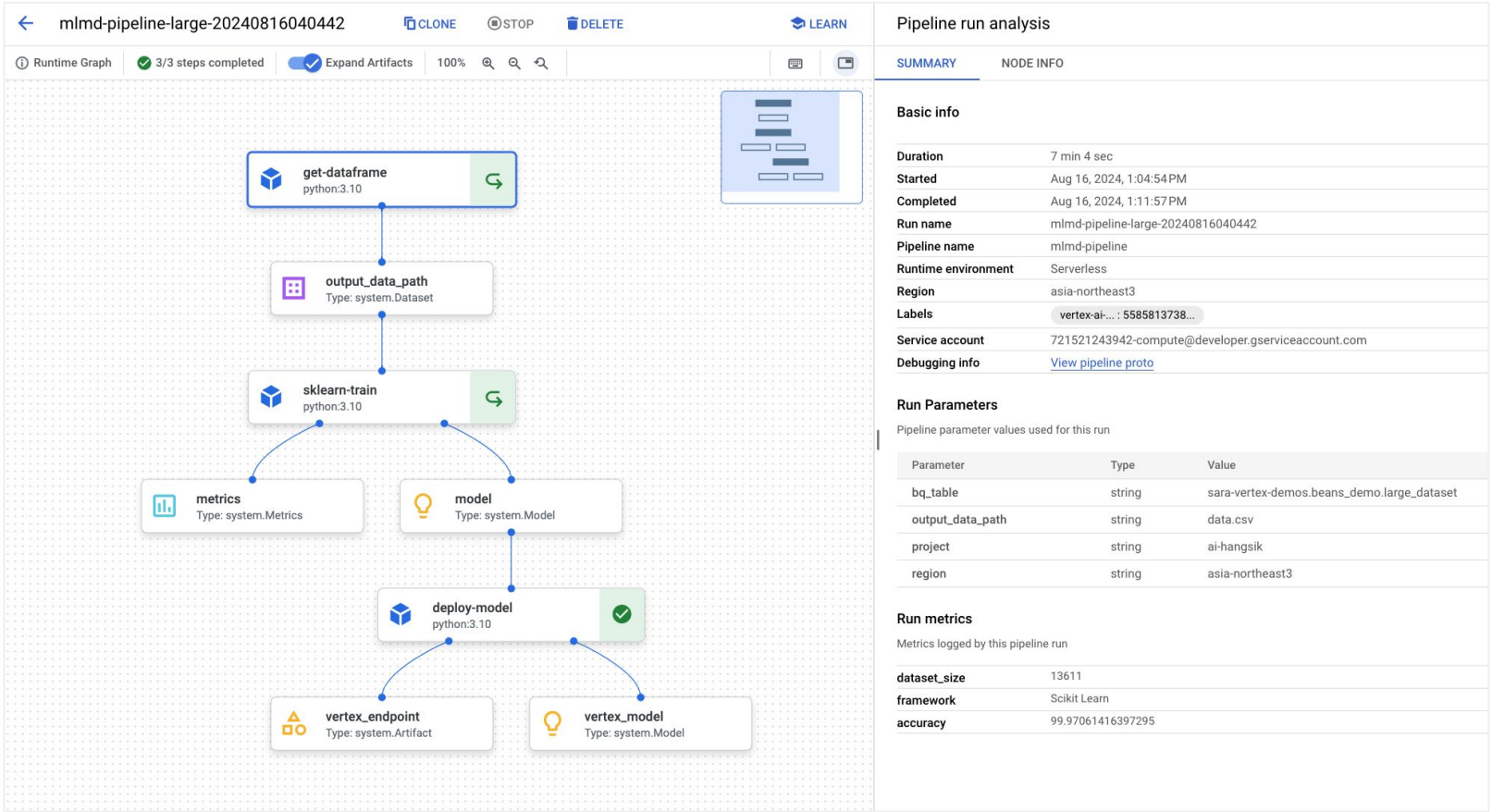
Pipelines API Reference (v2beta1)
API Reference for Kubeflow Pipelines API - v2beta1

Component Specification
Definition of a Kubeflow Pipelines component

Pipelines SDK Reference
Reference documentation for the Kubeflow Pipelines SDK Version 2

Kubernetes Platform-specific Features
Reference documentation for the `kfp-kubernetes` Python library

Pipeline on Vertex AI



Vertex Pipelines

Automate, monitor, and govern your ML systems by orchestrating your ML workflow in a serverless manner, and storing your workflow's artifacts using Vertex ML Metadata

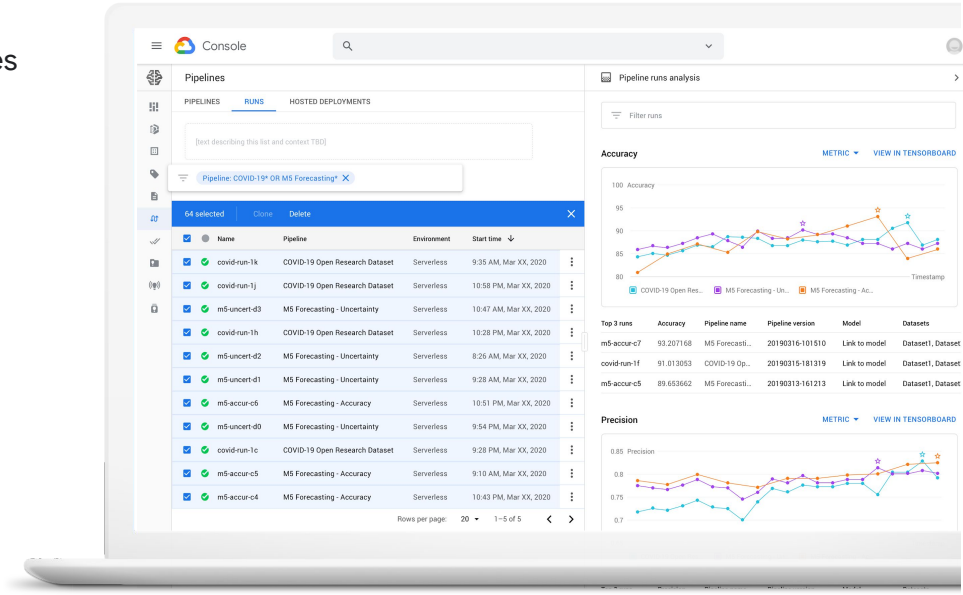
Scalable: Run as many pipelines on as much data as you want without having to worry about compute resources

Metadata Tracking and Lineage: Automatically store metadata about every artifact produced by the Pipelines.

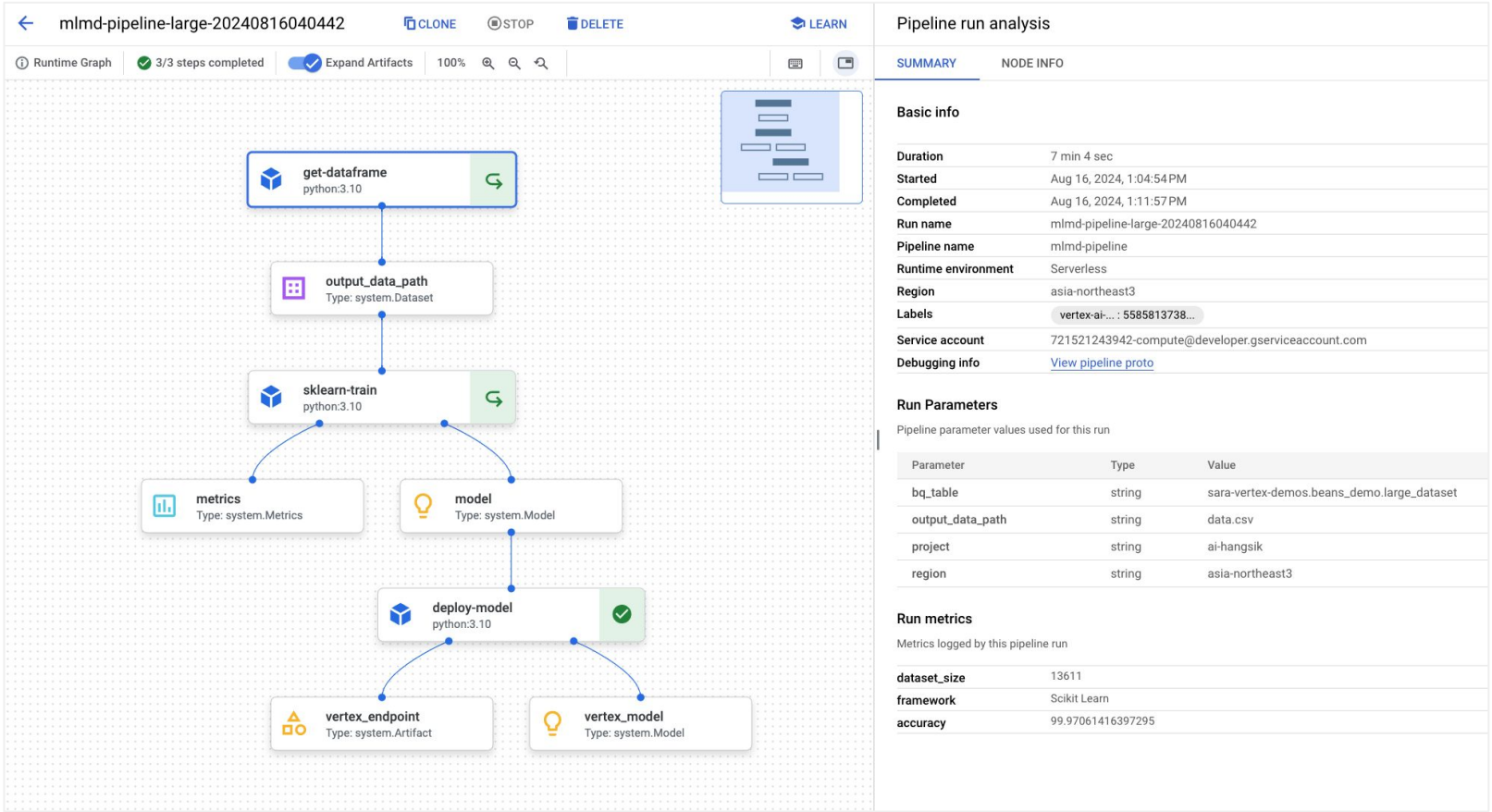
Cost-effective: Pay for the pipelines you run and the resources they use.

Secure: Integrated with GCP security features like IAM, VPC-SC, and CMEK.

Easy to use Python SDKs: Build your Pipelines using the battle-tested and easy-to-use KFP SDK and TFX SDK



Pipeline in GCP Console



Pipeline comparison

Compare Runs

REFRESH LEARN

Parameters

Run	input:bq_table	input:output_data_path	input:project	input:region
mlmd-pipeline-large-20240816040442	sara-vertex-demos.beans_demo.large_dataset	data.csv	ai-hangsik	asia-northeast3
mlmd-pipeline-small-20240816040442	sara-vertex-demos.beans_demo.small_dataset	data.csv	ai-hangsik	asia-northeast3

Metrics

Run	accuracy	dataset_size	framework
mlmd-pipeline-large-20240816040442	99.97061416397295	13611	Scikit Learn
mlmd-pipeline-small-20240816040442	99.42857142857143	700	Scikit Learn

Vertex AI Pipeline

Kubeflow Pipelines (KFP) and **TensorFlow Extended (TFX) SDKs**

- **KFP:** https://www.kubeflow.org/docs/pipelines/
  - Kubeflow Pipelines SDK v1.8 or later (v2 is recommended) on Vertex AI
- **TFX:** https://www.tensorflow.org/tfx
  - TensorFlow Extended v0.30.0 or later on Vertex AI
- Both **open-source** SDKs are supported by both **KFP OSS**/Hosted KFP and **Vertex Pipelines**.
- Both SDKs support use of both prebuilt and custom *components* (pipeline step definitions).
- 'v2 compatibility mode': (soon) run the same pipelines on OSS KFP and Vertex Pipelines
- If you use TensorFlow in an ML workflow that processes terabytes of structured data or text data, we recommend that you build your pipeline using TFX.

Pipelines Product Portfolio

Kubeflow

Kubeflow Pipelines

- Kubernetes native, open source product.
- The industry standard for running ML Pipelines.

Google Cloud Platform and Vertex AI

AI Platform Pipelines - Hosted* Beta

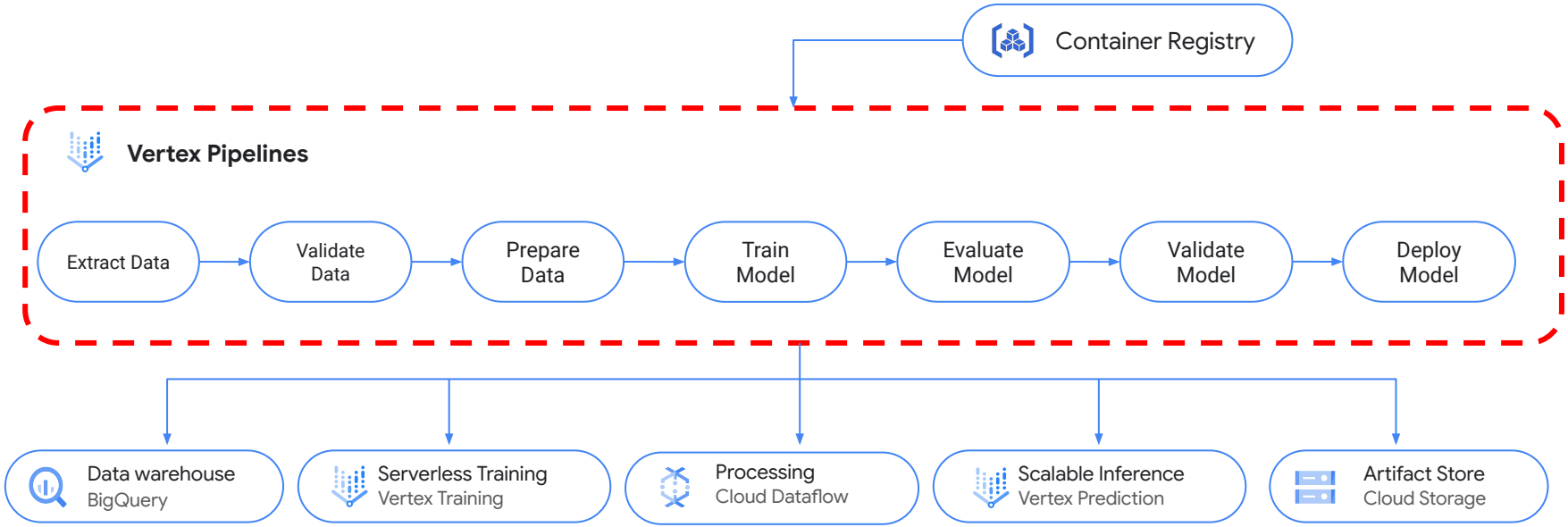
- Kubeflow Pipelines running on GCP.
- Optimized for GKE, integrated with GCP Services.

Vertex Pipelines - Managed

- Fully managed and serverless.
- Allows users to focus on building their pipelines, scale easily, and pay only for what they use.

Vertex AI Pipeline

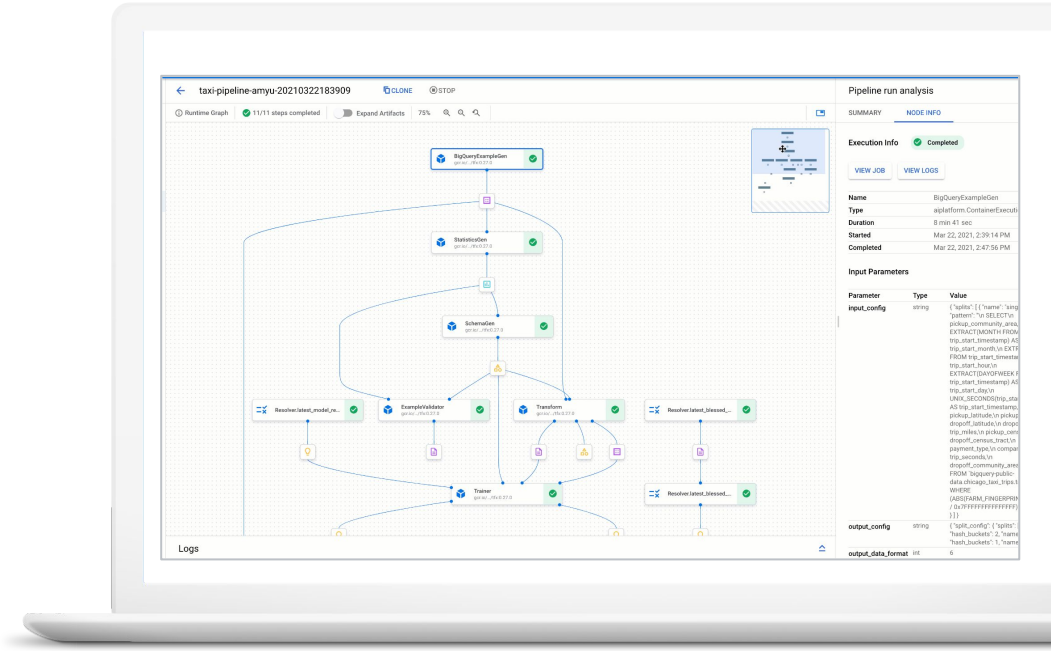
- Vertex Pipelines is the backbone of any MLOps workflow.
- Pipelines is a orchestrator tool, responsible for managing a comprehensive ML workflow in a standardized way.
- In practice, it is a managed service that runs either Kubeflow Pipeline or TFX workloads.



Pipelines are the backbone of MLOps

Simplify and streamline with AI Platform Pipelines.

- Rapid, scalable experimentation through a fully-managed, serverless service.
- Detailed metadata tracking for reproducibility, audit, and governance.
- Continuous monitoring and triggered model retraining
- Automatically log metadata for every artifact produced by the pipeline
- Track artifacts, lineage, metrics, and execution across your ML workflow



Pipeline tasks and components

- A pipeline task is an instantiation of a pipeline component with specific inputs. While defining your ML pipeline, you can interconnect multiple tasks to form a DAG, by routing the outputs of one pipeline task to the inputs for the next pipeline task in the ML workflow.

- Pipeline component
  - A pipeline component is a self-contained set of code that performs a specific step of an ML workflow, such as data preprocessing, model training, or model deployment. Components are the basis of defining tasks in an ML pipeline.
  - Two types of components : Predefined components and custom components

- Pipeline task
  - A pipeline task is the instantiation of a pipeline component and performs a specific step in your ML workflow. You can author ML pipeline tasks either using Python or as prebuilt container images.

Pipeline lifecycle

1. **Define**: The process of defining an ML pipeline and its task is also called building a pipeline. In this stage, you need to perform the following steps:
   a. Choose an ML framework
   b. Define pipeline tasks and configure pipeline
2. **Compile**: In this stage, you need to perform the following steps:
   a. Generate your ML pipeline definition in a compiled YAML file
   b. You can upload the compiled YAML file as a pipeline template to a repository and reuse it to create ML pipeline runs.
3. **Run**: Create an execution instance of your ML pipeline using the compiled YAML file or a pipeline template. The execution instance of a pipeline definition is called a pipeline run.
4. **Monitor, visualize, and analyze runs**: After you create a pipeline run, you can do the following to monitor the performance, status, and costs of pipeline runs:
   a. Configure email notifications for pipeline failures.
   b. Use Cloud Logging to create log entries for monitoring events.
   c. Visualize, analyze, and compare pipeline runs.
5. Optional: **stop or delete** pipeline runs

Google Cloud Pipeline Components

Google Cloud Pipeline Components

- The Google Cloud Pipeline Components (GCPC) SDK provides a set of prebuilt Kubeflow Pipelines components that are production quality, performant, and easy to use. You can use Google Cloud Pipeline Components to define and run ML pipelines in Vertex AI Pipelines and other ML pipeline execution backends conformant with Kubeflow Pipelines.
- For example, you can use these components to complete the following:
  - Create a new dataset and load different data types into the dataset (image, tabular, text, or video).
  - Export data from a dataset to Cloud Storage.
  - Use AutoML to train a model using image, tabular, text, or video data.
  - Run a custom training job using a custom container or a Python package.
  - Upload an existing model to Vertex AI for batch prediction.
  - Create a new endpoint and deploy a model to it for online predictions.
- Package install
  - pip install --upgrade google-cloud-pipeline-components

Google Cloud Pipeline Components

- The Google Cloud Pipeline Components (GCPC) SDK provides a set of prebuilt Kubeflow Pipelines components that are production quality, performant, and easy to use.

- You can use Google Cloud Pipeline Components to define and run ML pipelines in Vertex AI Pipelines and other ML pipeline execution backends conformant with Kubeflow Pipelines.

- Benefits of GCPC
  - Easier debugging
  - Standardized artifact types (Link)
  - Understand pipeline costs with billing labels
  - Cost efficiencies

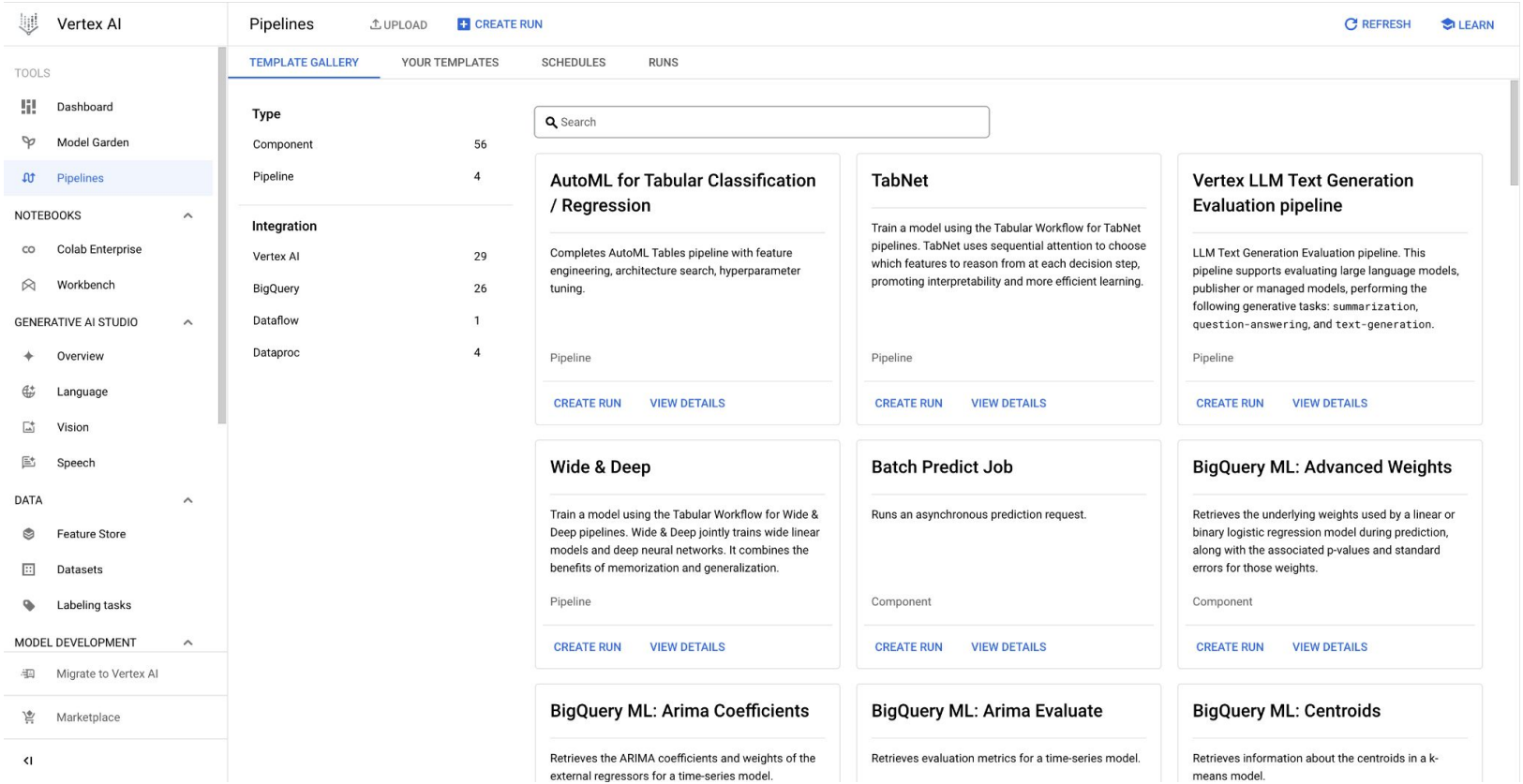
Components on GCP As of Aug 2024

Generally available (1.0 and later) components
- AutoML components
- Batch Prediction components
- BigQuery ML components
- Custom Job components
- Dataflow components
- Dataproc Serverless components
- Dataset components
- Endpoint components
- Forecasting components
- Hyperparameter Tuning components
- Model components
- Model evaluation components
- Email notification components
- Components related to wait on resources

Preview components
- AutoML components
- Custom Job components
- Dataflow components
- Large-language model (LLM) components
- Model evaluation components

https://google-cloud-pipeline-components.readthedocs.io/en/google-cloud-pipeline-components-2.16.0/api/index.html

Pipeline templates



Pipeline options : Configure execution caching

- If there is a matching execution in Vertex ML Metadata, the outputs of that execution are used and the step is skipped. This helps to reduce costs by skipping computations that were completed in a previous pipeline run.
- The cached result doesn't have a time-to-live (TTL), and can be reused as long as the entry is not deleted from the Vertex ML Metadata. If the entry is deleted from Vertex ML Metadata, the task will rerun to regenerate the result again.

```
pl = PipelineJob(
    display_name="My first pipeline",

    # Whether or not to enable caching
    # True = enable the current run to use caching results from previous runs
    # False = disable the current run's use of caching results from previous runs
    # None = defer to cache option for each pipeline component in the pipeline definition
    enable_caching=False,

    template_path="pipeline.yaml",
    parameter_values=parameter_values,
    pipeline_root=pipeline_root,
)
```

Thank you!