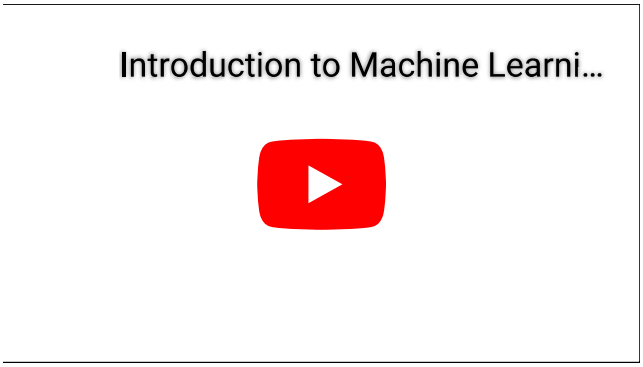
Try Gemini 1.5 models (https://console.cloud.google.com/freetrial/?redirectPath=/vertex-ai/generative/multimodal/create/text), our newest multimodal models in Vertex AI, and see what you can build with a 1M token context window.

Introduction to Vertex AI

Vertex AI is a machine learning (ML) platform that lets you train and deploy ML models and AI applications, and customize large language models (LLMs) for use in your AI-powered applications. Vertex AI combines data engineering, data science, and ML engineering workflows, enabling your teams to collaborate using a common toolset and scale your applications using the benefits of Google Cloud.



Vertex AI provides several options for model training (/vertex-ai/docs/start/training-methods) and deployment:

- AutoML (/vertex-ai/docs/beginner/beginners-guide) lets you train tabular, image, text, or video data without writing code or preparing data splits.
- Custom training (/vertex-ai/docs/training/overview) gives you complete control over the training process, including using your preferred ML framework, writing your own training code, and choosing hyperparameter tuning options.
- Model Garden (/vertex-ai/docs/start/explore-models) lets you discover, test, customize, and deploy Vertex AI and select open-source (OSS) models and assets.
- Generative AI (/vertex-ai/generative-ai/docs/learn/overview) 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.

After you deploy your models, use Vertex AI's end-to-end MLOps tools to automate and scale projects throughout the ML lifecycle. These MLOps tools are run on fully-managed infrastructure that you can customize based on your performance and budget needs.

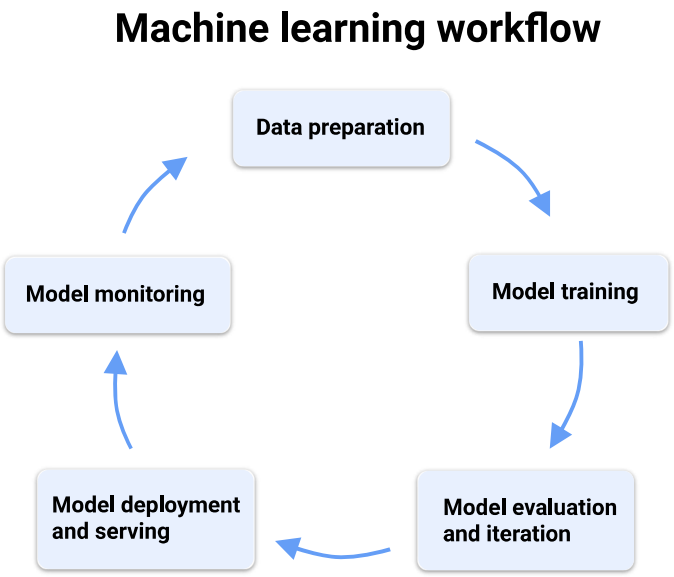
You can use the Vertex AI SDK for Python (/vertex-ai/docs/python-sdk/use-vertex-ai-python-sdk) to run the entire machine learning workflow in Vertex AI Workbench (/vertex-ai/docs/workbench/introduction), a Jupyter notebook-based development environment. You can collaborate with a team to develop your

model in Colab Enterprise (/colab/docs/introduction), a version of Colaboratory (https://colab.google/) that is integrated with Vertex AI. Other available interfaces (/vertex-ai/docs/start/introduction-interfaces) include the Google Cloud Console, the gcloud command line tool, client libraries, and Terraform (limited support).

Vertex AI and the machine learning (ML) workflow

This section provides an overview of the machine learning workflow and how you can use Vertex AI to build and deploy your models.

1. **Data preparation**: After extracting and cleaning your dataset, perform exploratory data analysis (EDA)



(/vertex-ai/docs/glossary#exploratory_data_analysis) to understand the data schema and characteristics that are expected by the ML model. Apply data transformations and feature engineering to the model, and split the data into training, validation, and test sets.

   - Explore and visualize data using Vertex AI Workbench (/vertex-ai/docs/workbench/introduction) notebooks. Vertex AI Workbench integrates with Cloud Storage and BigQuery to help you access and process your data faster.
   - For large datasets, use Dataproc Serverless Spark (/dataproc-serverless/docs/overview) from a Vertex AI Workbench notebook to run Spark workloads without having to manage your own Dataproc clusters.

2. **Model training**: Choose a training method to train a model and tune it for performance.

   - To train a model without writing code, see the AutoML overview (/vertex-ai/docs/training-overview#automl). AutoML supports tabular, image, text, and video

data.

- To write your own training code and train custom models using your preferred ML framework, see the Custom training overview (/vertex-ai/docs/training/overview).
- Optimize hyperparameters for custom-trained models using custom tuning jobs (/vertex-ai/docs/training/using-hyperparameter-tuning).
- Vertex AI Vizier (/vertex-ai/docs/vizier/overview) tunes hyperparameters for you in complex machine learning (ML) models.
- Use Vertex AI Experiments (/vertex-ai/docs/experiments/intro-vertex-ai-experiments) to train your model using different ML techniques and compare the results.
- Register your trained models in the Vertex AI Model Registry (/vertex-ai/docs/model-registry/introduction) for versioning and hand-off to production. Vertex AI Model Registry integrates with validation and deployment features such as model evaluation and endpoints.

3. **Model evaluation and iteration**: Evaluate your trained model, make adjustments to your data based on evaluation metrics, and iterate on your model.
    - Use model evaluation (/vertex-ai/docs/evaluation/introduction) metrics, such as precision and recall, to evaluate and compare the performance of your models. Create evaluations through Vertex AI Model Registry, or include evaluations in your Vertex AI Pipelines (/vertex-ai/docs/pipelines/introduction) workflow.
4. **Model serving**: Deploy your model to production and get predictions.
    - Deploy your custom-trained model using prebuilt (/vertex-ai/docs/predictions/pre-built-containers) or custom (/vertex-ai/docs/predictions/use-custom-container) containers to get real-time *online predictions* (/vertex-ai/docs/predictions/overview#online_predictions) (sometimes called HTTP prediction).
    - Get asynchronous *batch predictions* (/vertex-ai/docs/predictions/overview#batch_predictions), which don't require deployment to endpoints.
    - Optimized TensorFlow runtime (/vertex-ai/docs/predictions/optimized-tensorflow-runtime) lets you serve TensorFlow models at a lower cost and with lower latency than open source based prebuilt TensorFlow Serving containers.
    - For online serving cases with tabular models, use Vertex AI Feature Store (/vertex-ai/docs/featurestore/overview) to serve features from a central repository and monitor feature health.

- Vertex Explainable AI (/vertex-ai/docs/explainable-ai/overview) helps you understand how each feature contributes to model prediction (*feature attribution*) and find mislabeled data from the training dataset (*example-based explanation*).
- Deploy and get online predictions for models trained with BigQuery ML (/vertex-ai/docs/beginner/bqml).

5. **Model monitoring**: Monitor the performance of your deployed model. Use incoming prediction data to retrain your model for improved performance.
    - Vertex AI Model Monitoring (/vertex-ai/docs/model-monitoring/overview) monitors models for training-serving skew and prediction drift and sends you alerts when the incoming prediction data skews too far from the training baseline.

What's next

- Learn about Vertex AI's MLOps features (/vertex-ai/docs/start/introduction-mlops).
- Learn about interfaces that you can use to interact with Vertex AI (/vertex-ai/docs/start/introduction-interfaces).