Assignment 7: optional catchup assignment 2 - VERTEX AI - for midterm and quiz - this will catch up midterm.

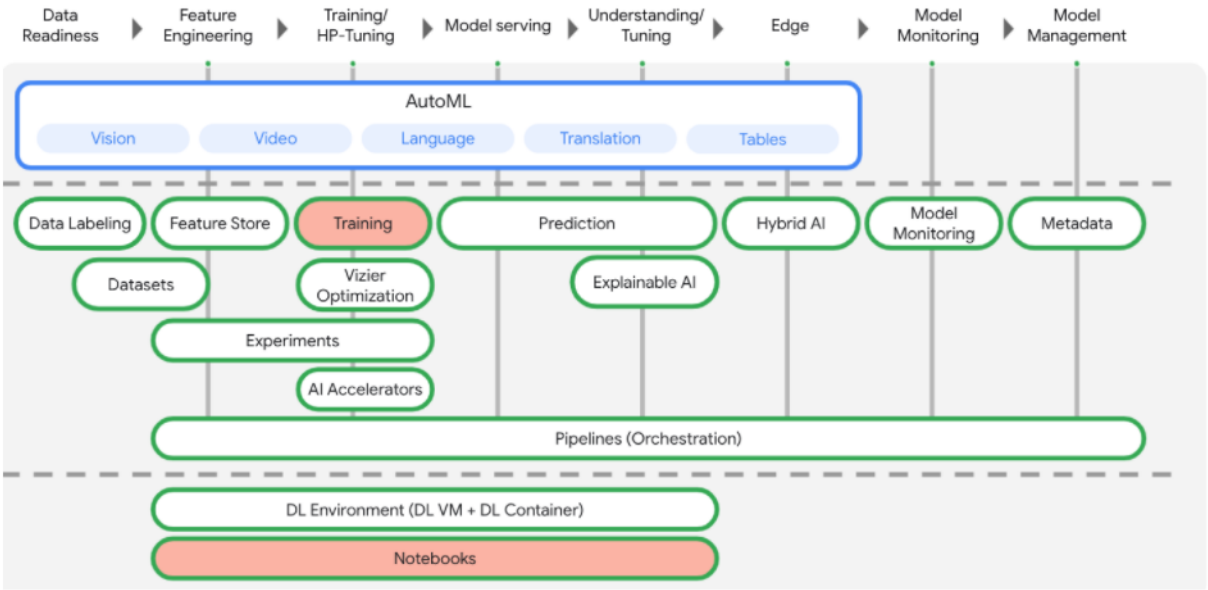
a) Vertex AI to run a hyperparameter tuning job for a TensorFlow model

Reference: https://codelabs.developers.google.com/vertex_hyperparameter_tuning#0

Objectives:

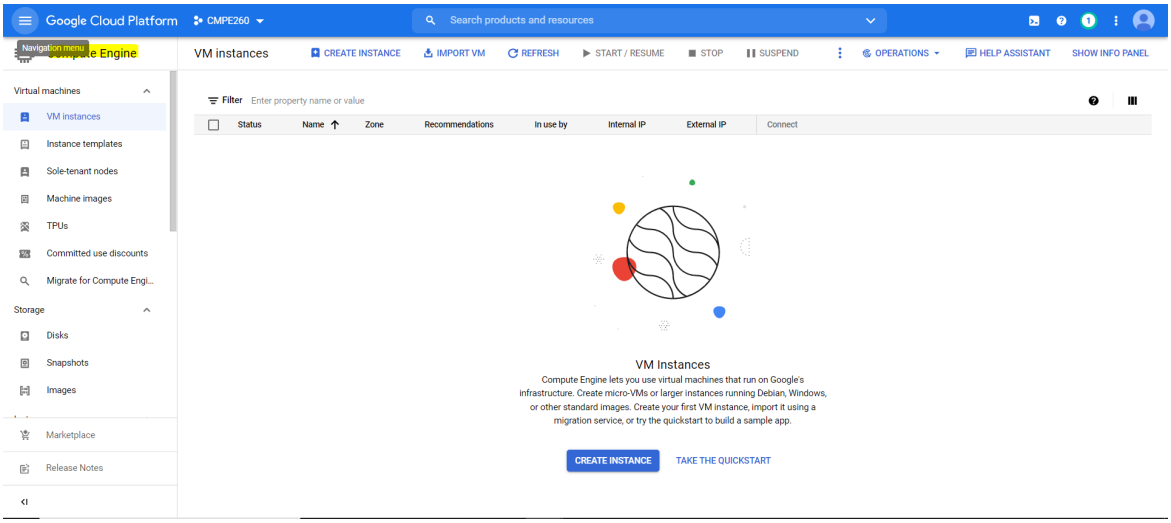
- Modify training application code for hyperparameter tuning
- Configure and launch a hyperparameter tuning job from the Vertex AI UI
- Configure and launch a hyperparameter tuning job with the Vertex SDK

Vertex AI includes many different products to support end-to-end ML workflows. This document will focus on the products highlighted below: Training/HP-Tuning and Notebooks

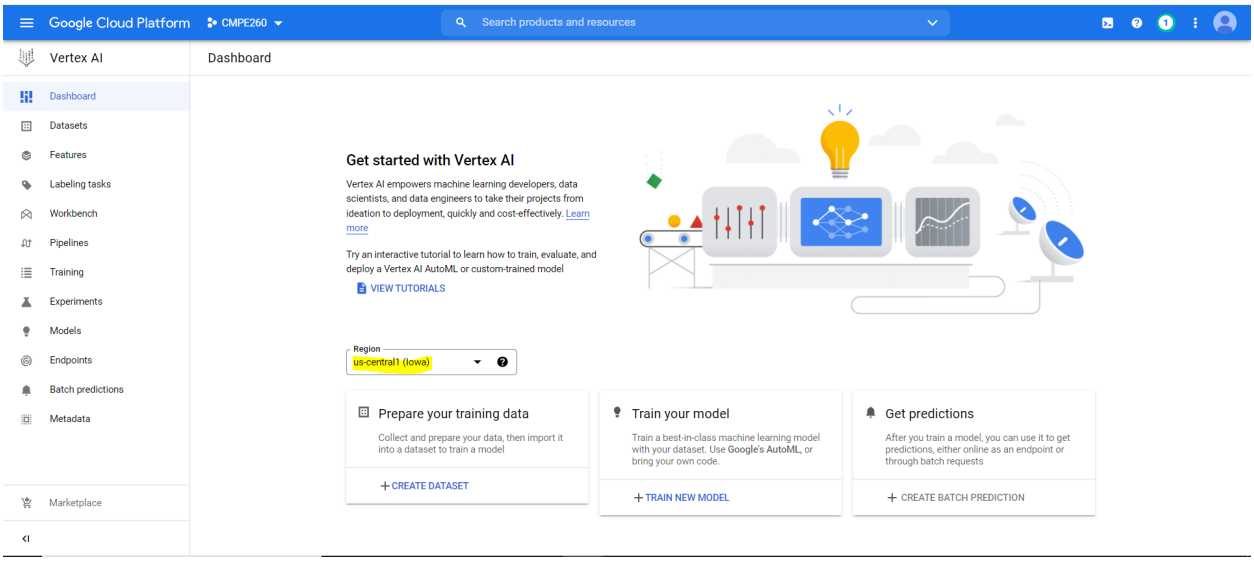


Setup your environment

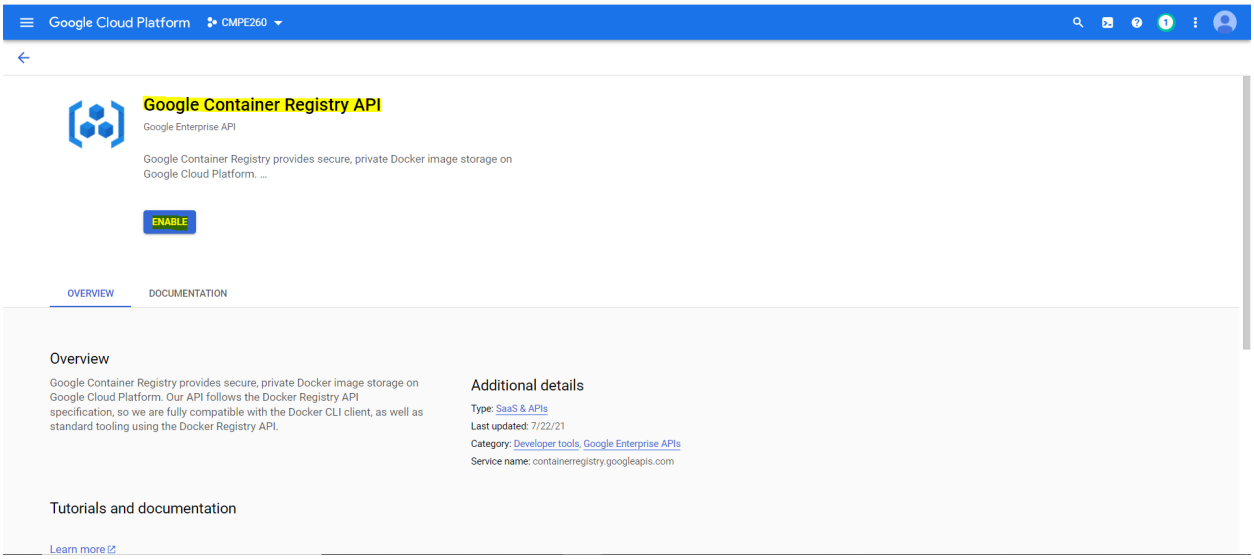
Step 1: Enable the Compute Engine API



Step 2: Enable the Vertex AI API

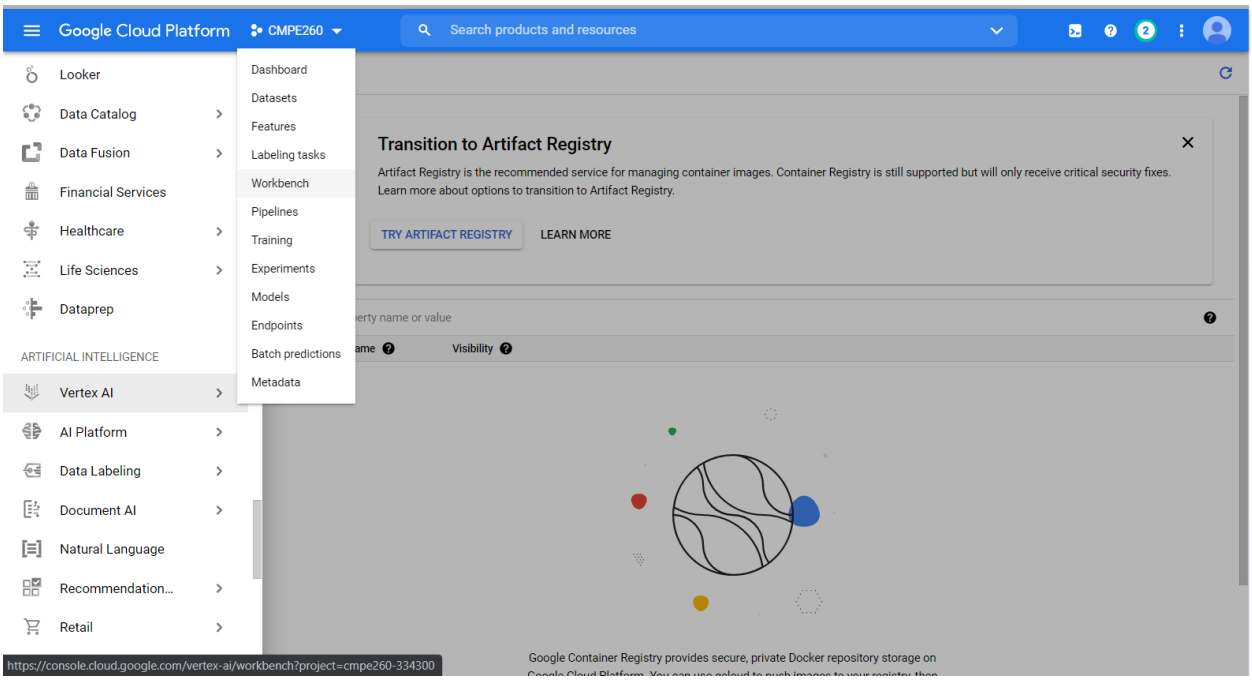


Step 3: Enable the Container Registry API

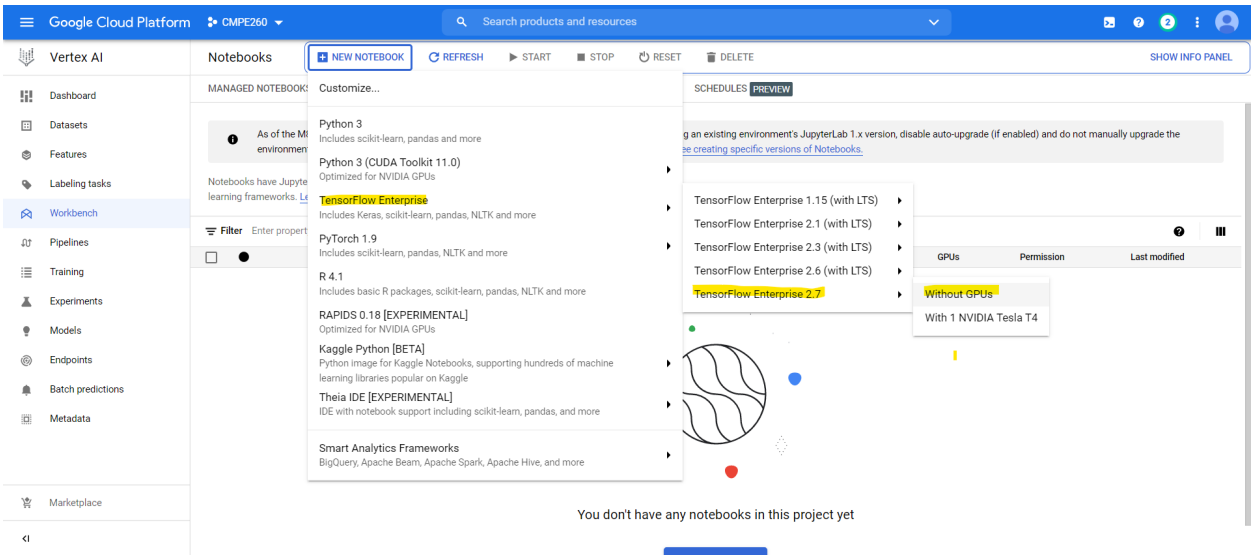


Step 4: Create a Vertex AI Workbench instance

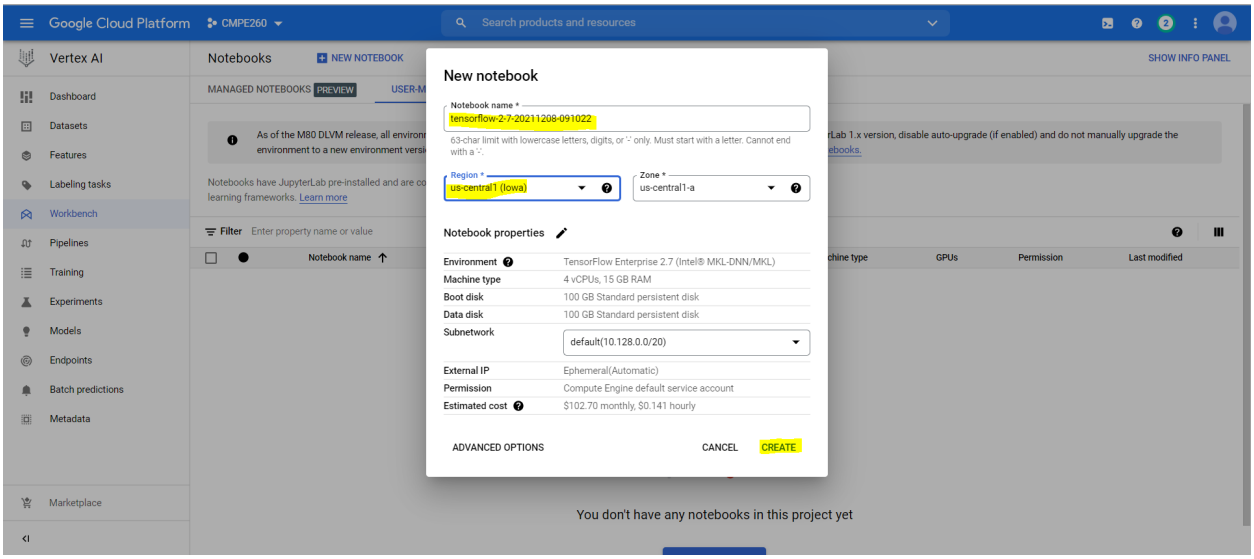
In the Left Nav, select Vertex AI → Workbench



Select **New Notebook**. Then select the **TensorFlow Enterprise 2.7** instance type **without GPUs**:



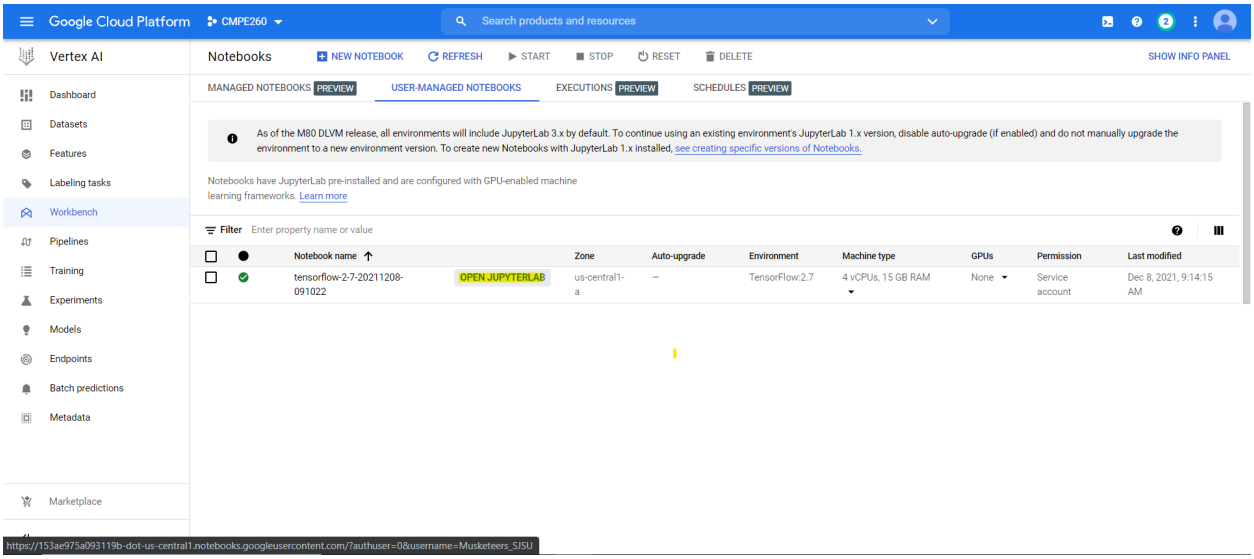
Enter the name, region and click ‘Create’



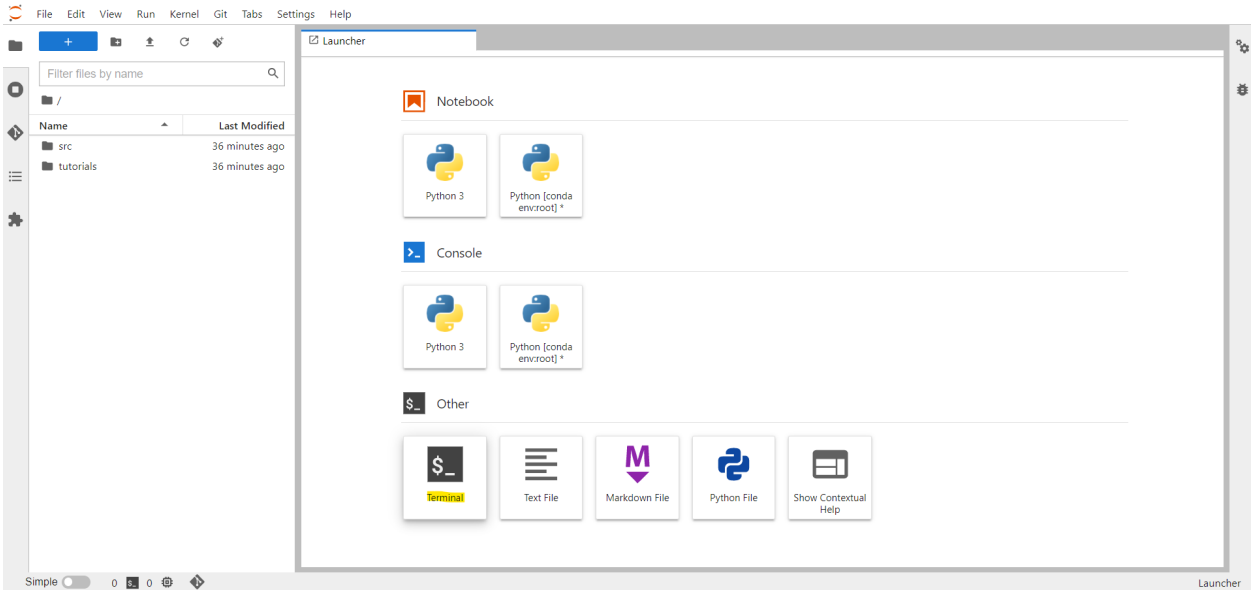
Containerize training application code

In this approach, we will submit this hyperparameter tuning job to Vertex by putting the training application code in a Docker container and pushing this container to Google Container Registry. Using this approach, you can tune hyperparameters for a model built with any framework.

1. Click ‘Open JupyterLab’

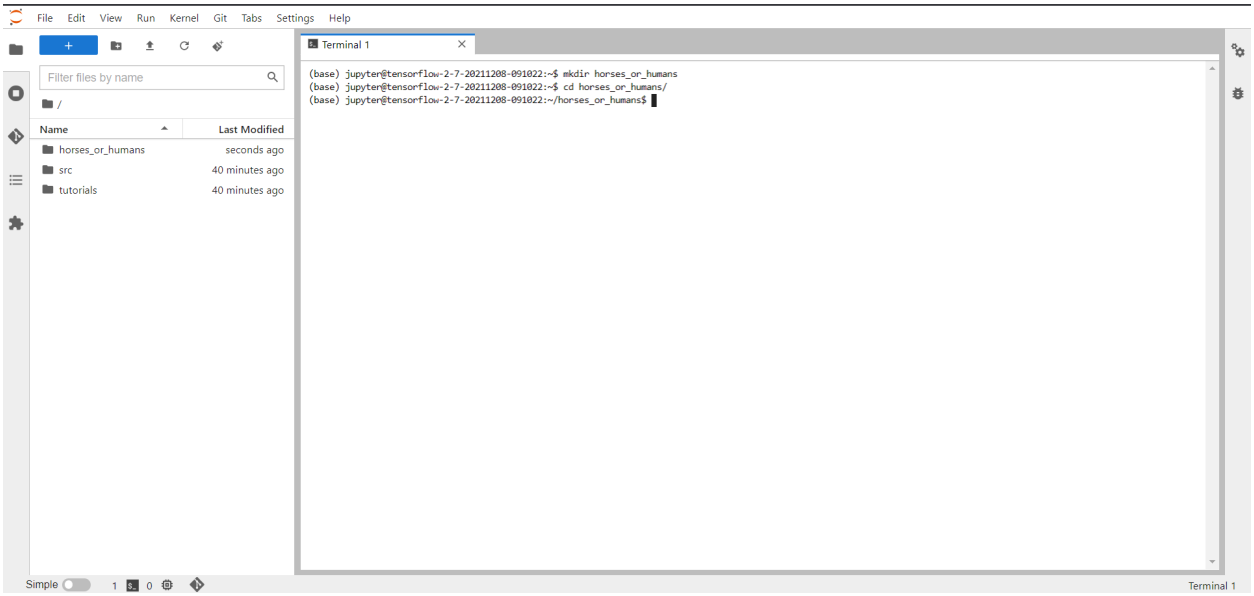


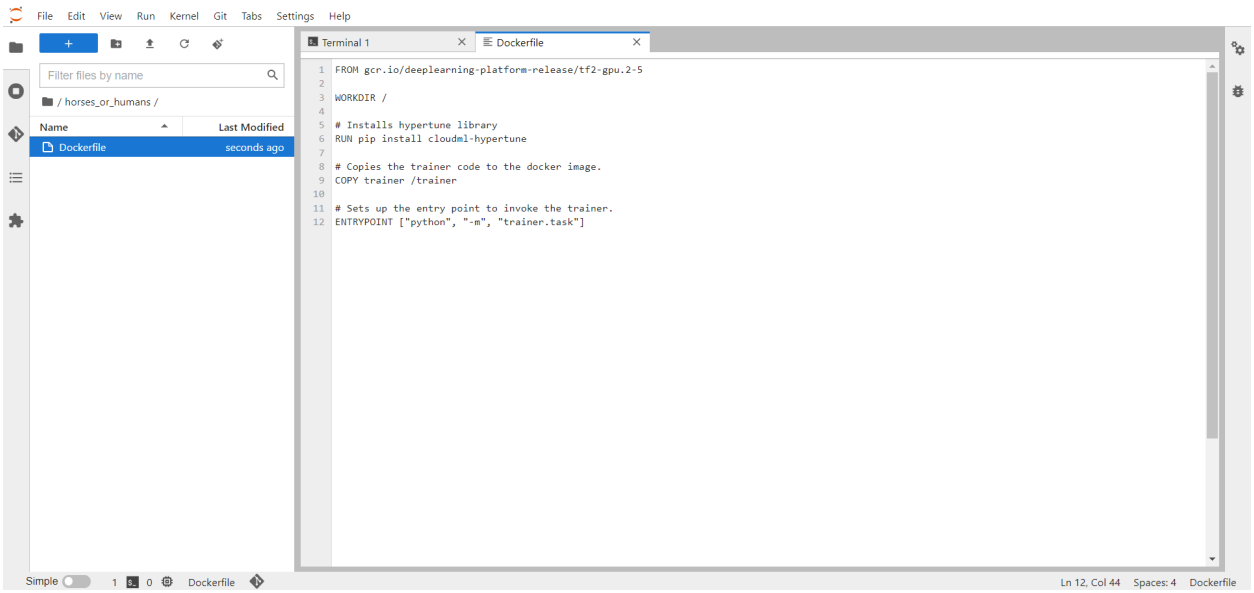
2. Click ‘Terminal’



Create Docker file:

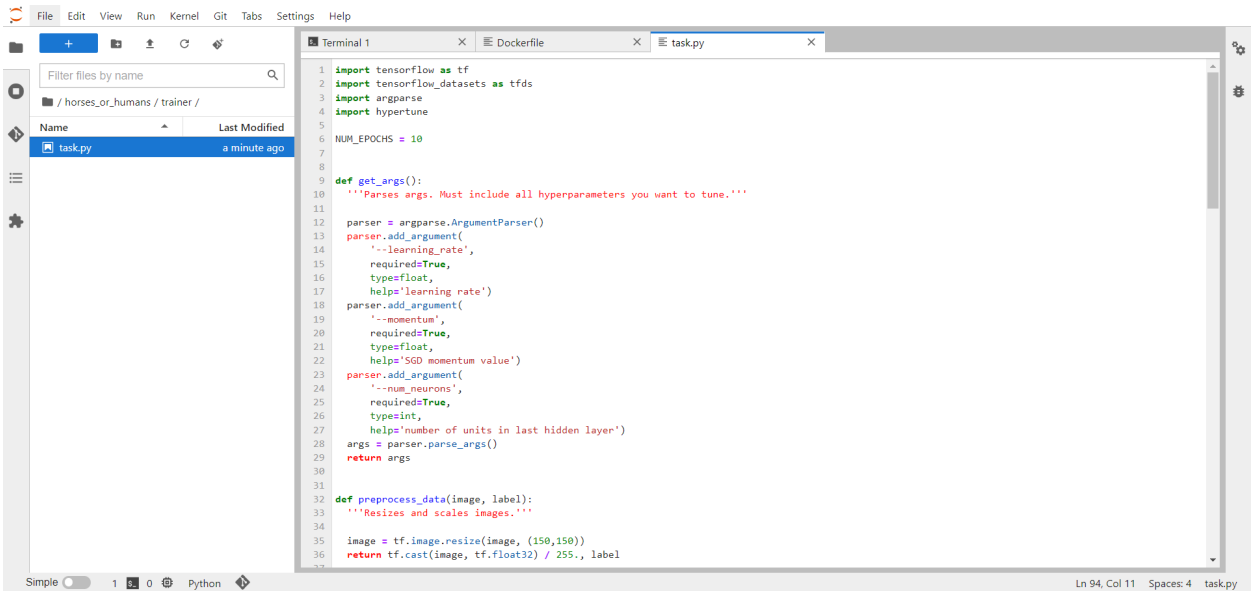
Run the following commands to create a folder
mkdir horses_or_humans
cd horses_or_humans
touch Dockerfile





Add model training code

Run the following commands:
mkdir trainer
touch trainer/task.py



There are a few components that are specific to using the hyperparameter tuning service.

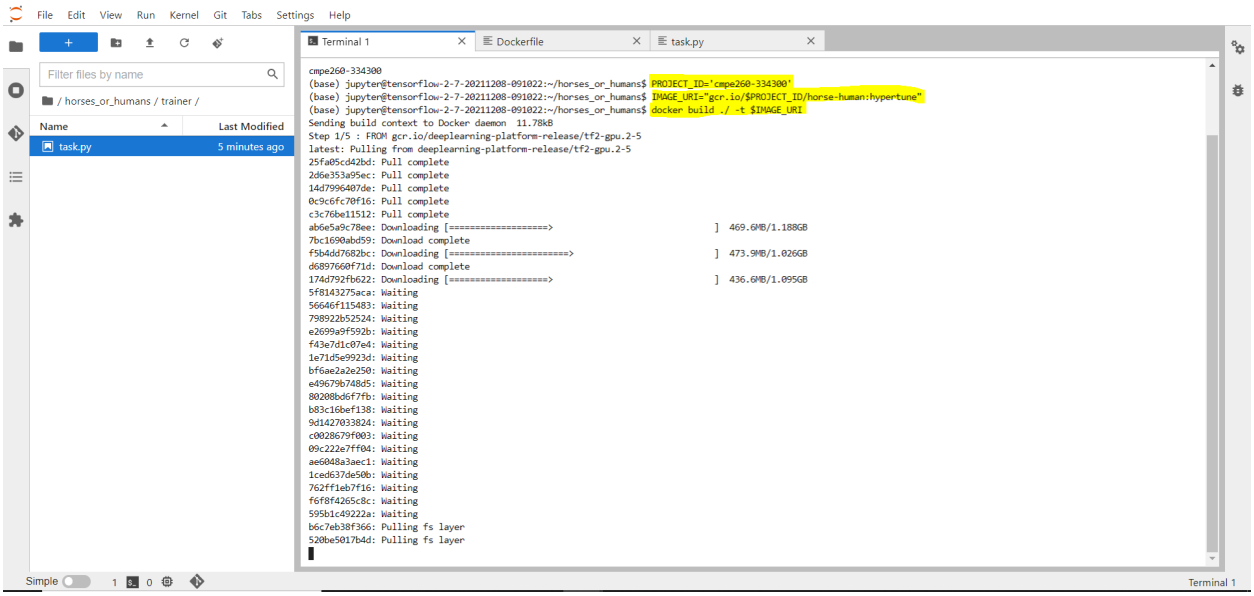
1. The script imports the hypertune library. Note that the Dockerfile from Step 1 included instructions to pip install this library.
2. The function get_args() defines a command-line argument for each hyperparameter you want to tune. In this example, the hyperparameters that will be tuned are the learning rate, the momentum value in the optimizer, and the number of neurons in the last hidden layer of the model, but feel free to experiment with others. The value passed in those arguments is then used to set the corresponding hyperparameter in the code.
3. At the end of the main() function, the hypertune library is used to define the metric you want to optimize. In TensorFlow, the keras model.fit method returns a History object. The History.history attribute is a record of training loss values and metrics values at successive epochs. If you pass validation data to model.fit the History.history attribute will include validation loss and metrics values as well. For example, if you trained a model for three epochs with validation data and provided accuracy as a metric, the History.history attribute would look similar to the following dictionary.

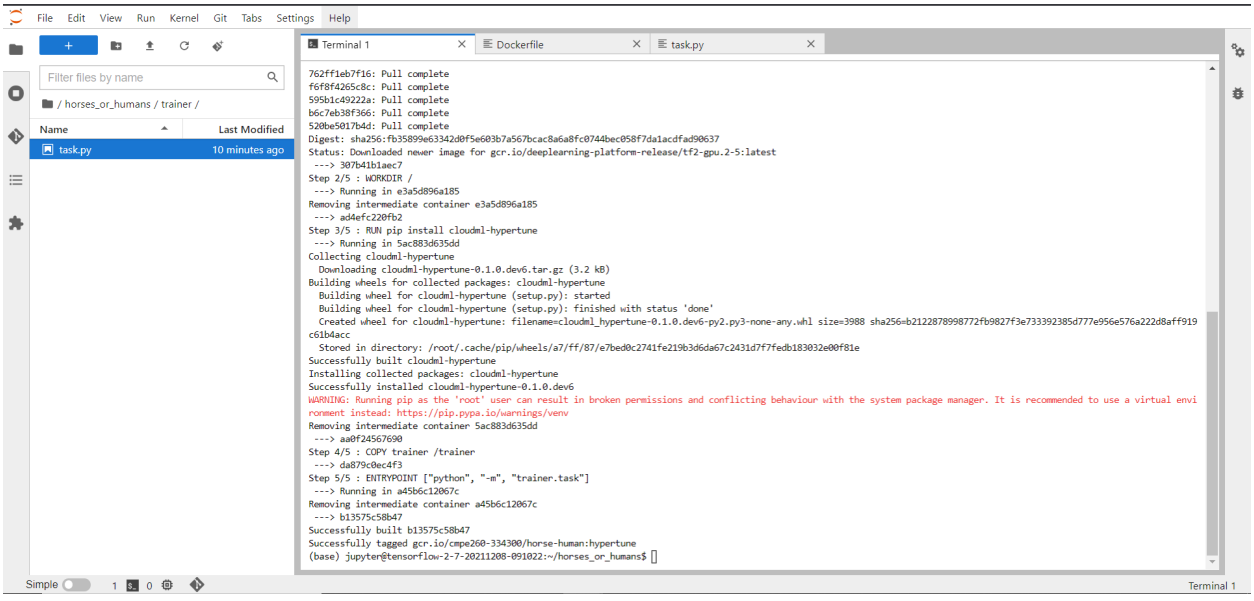
Build the container

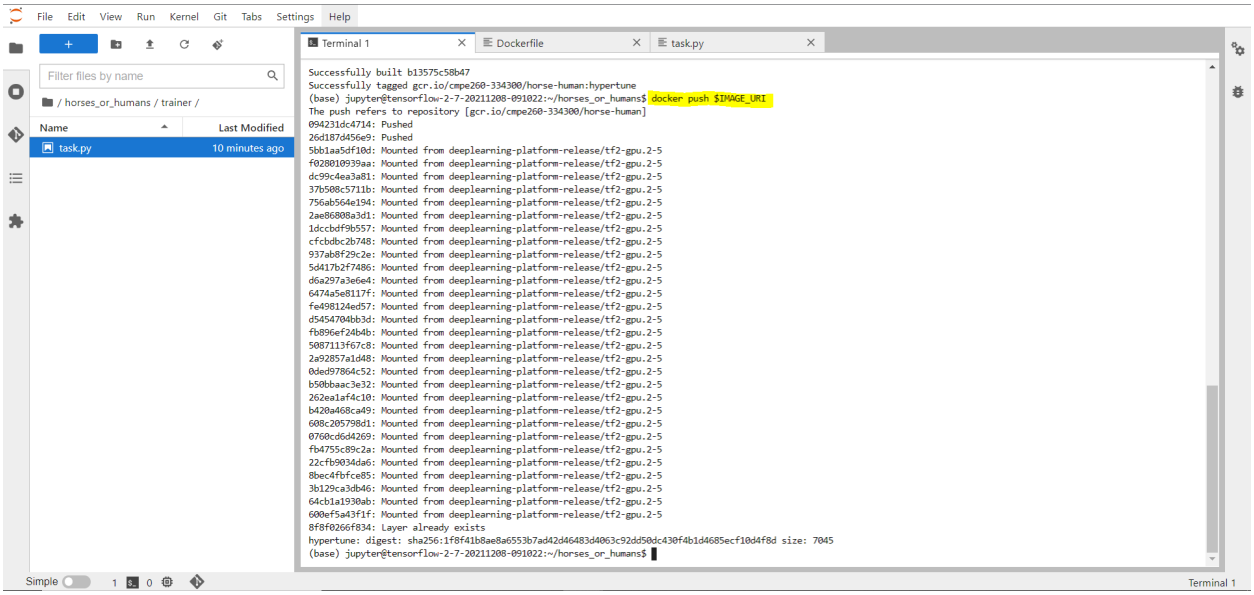
Run the following commands:

```
gcloud config list --format 'value(core.project)'
```

PROJECT_ID='cmpe260-334300'
IMAGE_URI="gcr.io/$PROJECT_ID/horse-human:hypertune"
docker build ./ -t $IMAGE_URI
docker push $IMAGE_URI

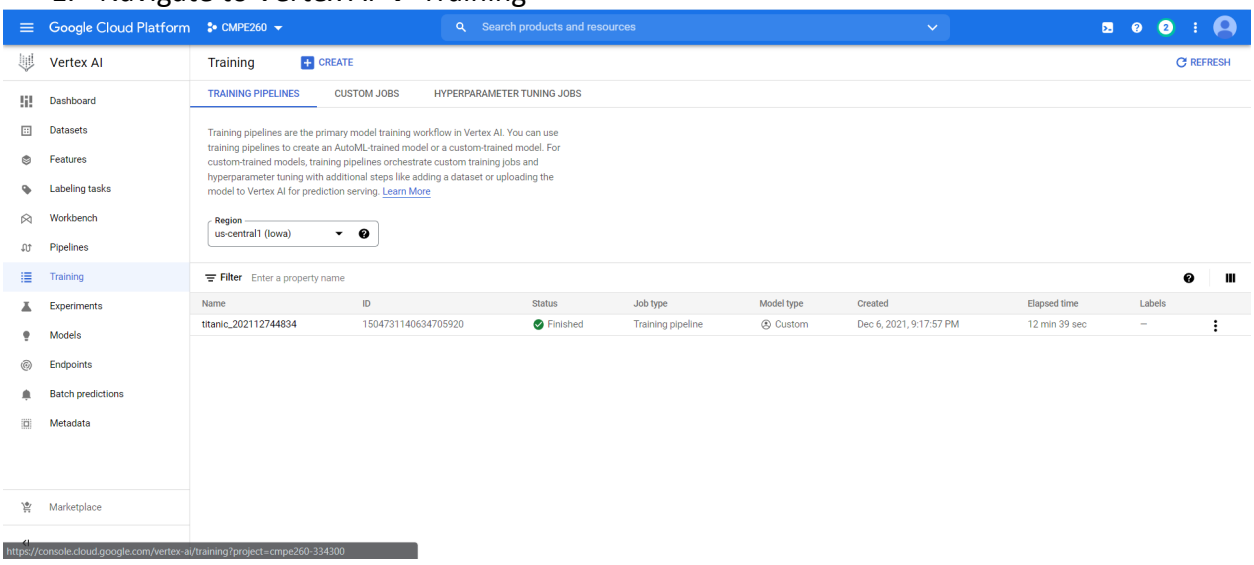






Run a hyperparameter tuning job on Vertex AI

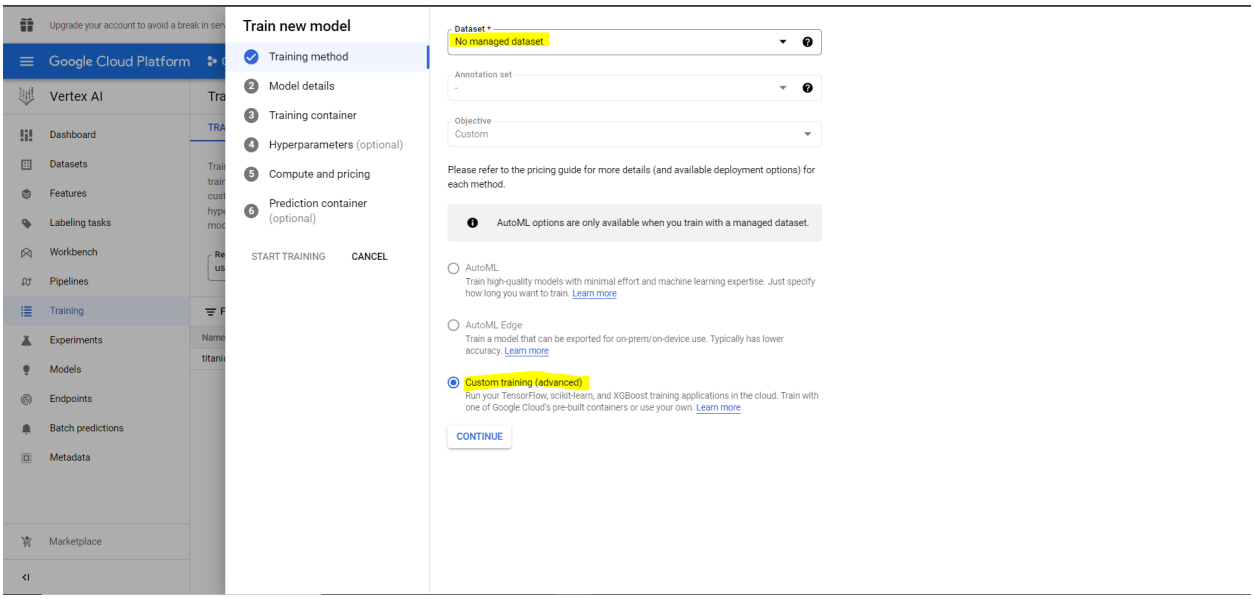
1. Navigate to Vertex AI → Training



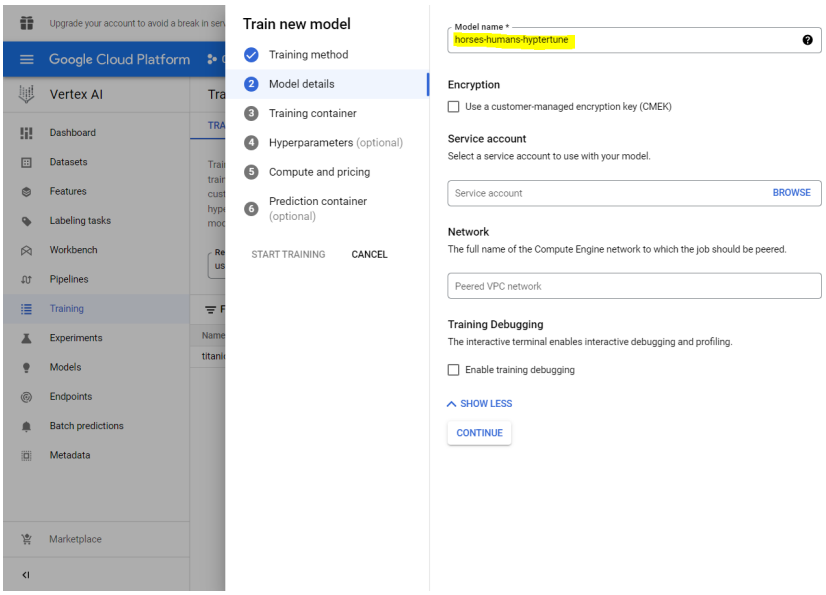
Step 1: Configure training job

Click **Create** to enter the parameters for your hyperparameter tuning job.

- Under **Dataset**, select **No managed dataset**
- Then select **Custom training (advanced)** as your training method and click **Continue**.
- Click **Continue**

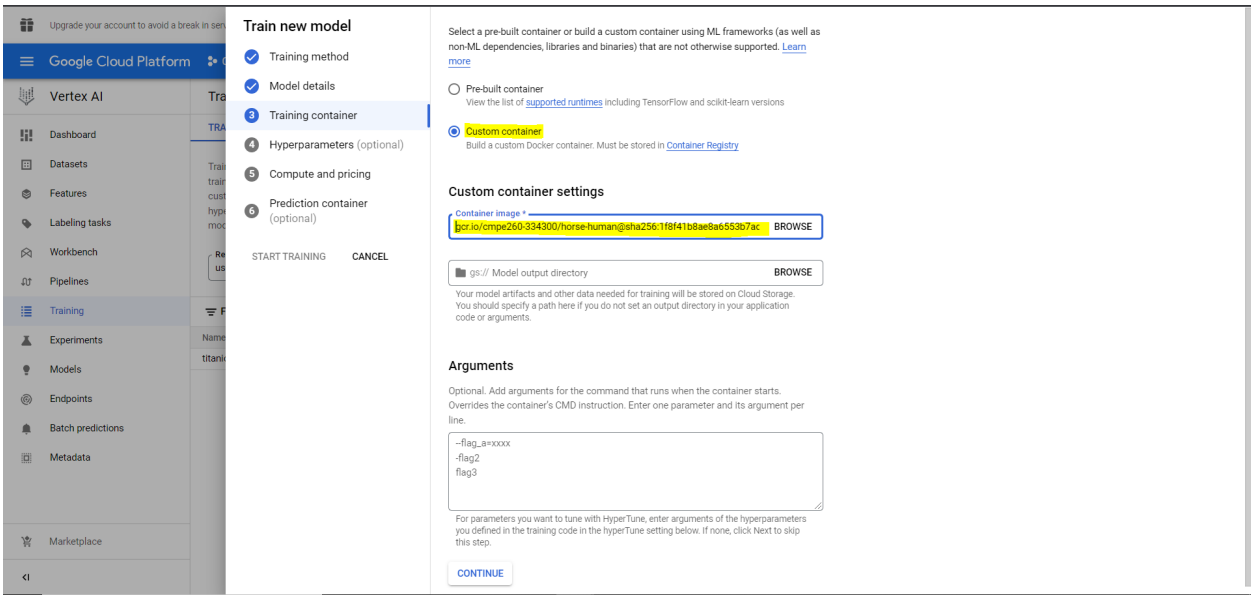


- Enter `horses-humans-hyptertune` (or whatever you'd like to call your model) for **Model name**
- Click **Continue**



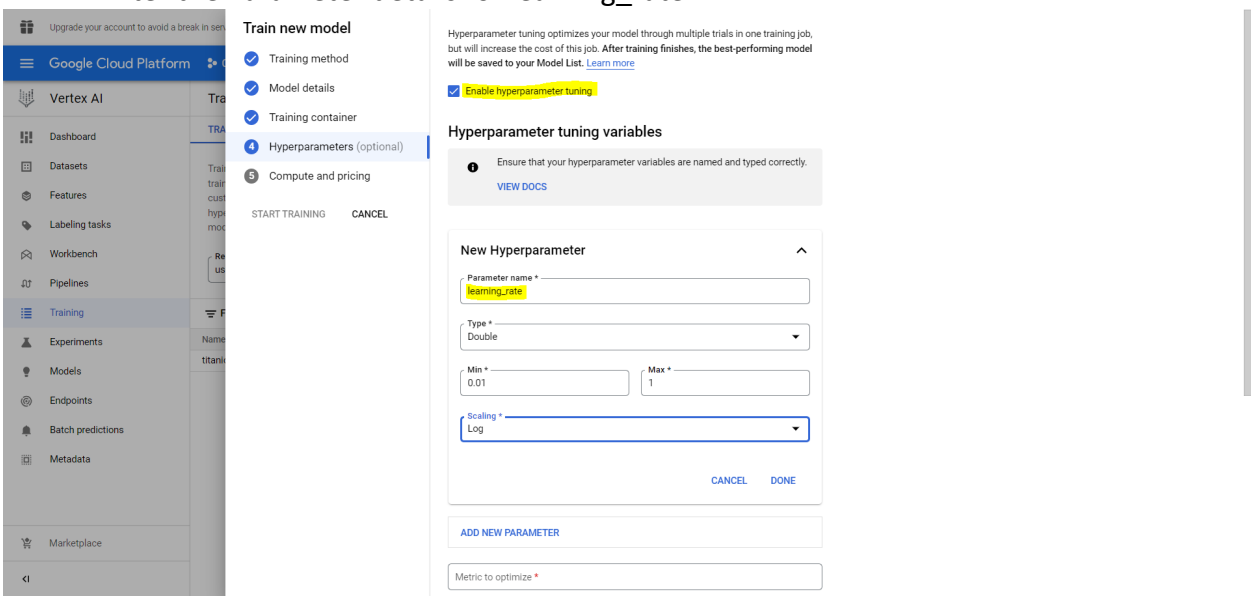
- Select ‘Custom Container’
- Enter **Container Image: URI to the Image**

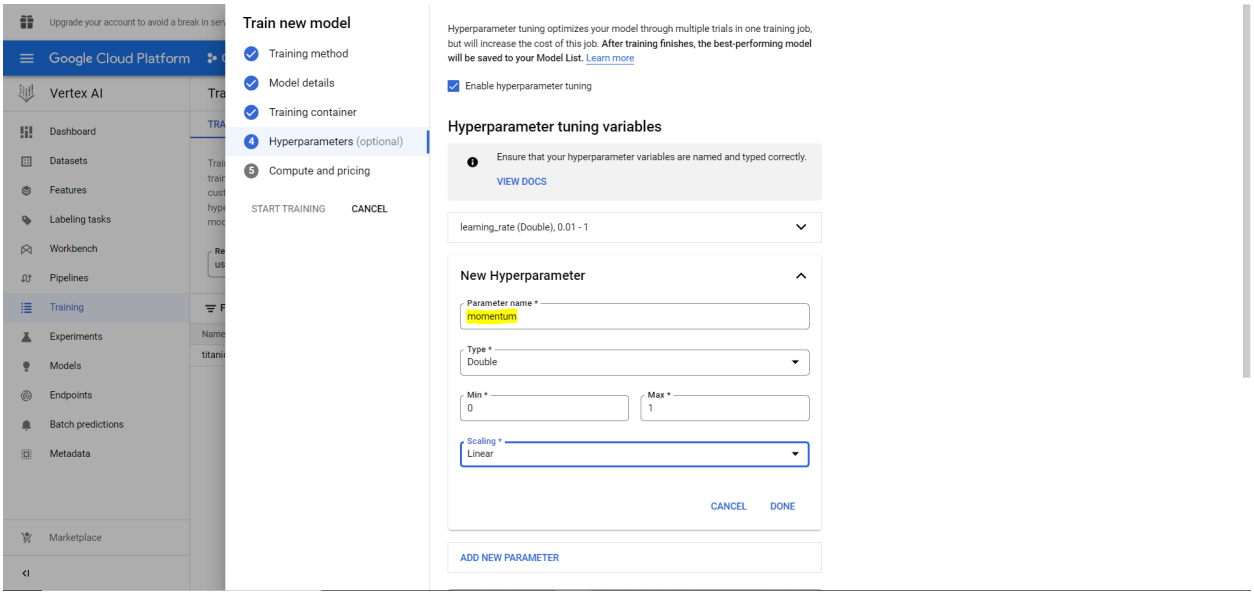
  gcr.io/cmpe260-334300/horse-human@sha256:1f8f41b8ae8a6553b7ad42d46483d4063c92dd50dc430f4b1d4685ecf10d4f8d

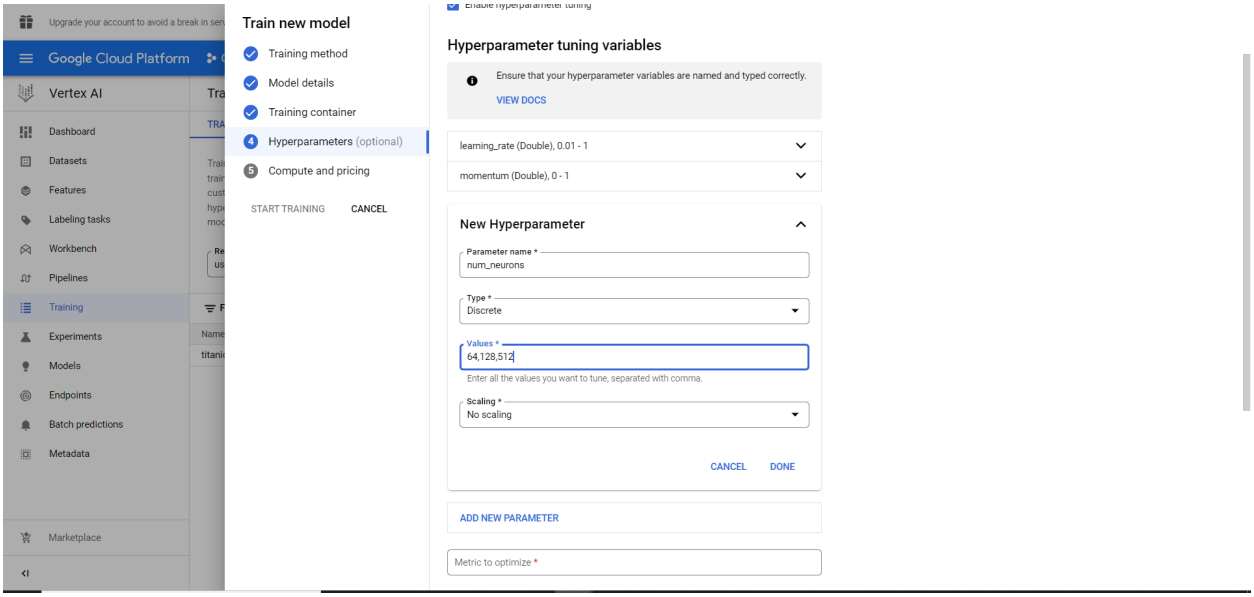


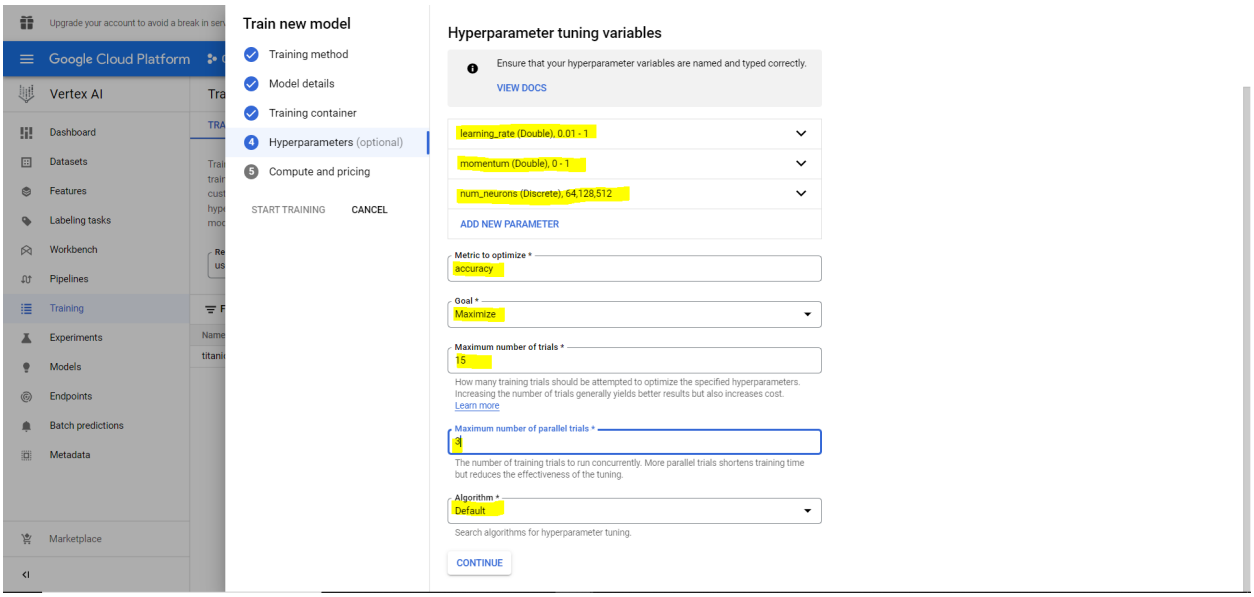
Step 2: Configure hyperparameter tuning job

- Select ‘Enable Hyperparameter tuning
- Enter the Parameter details for learning_rate’

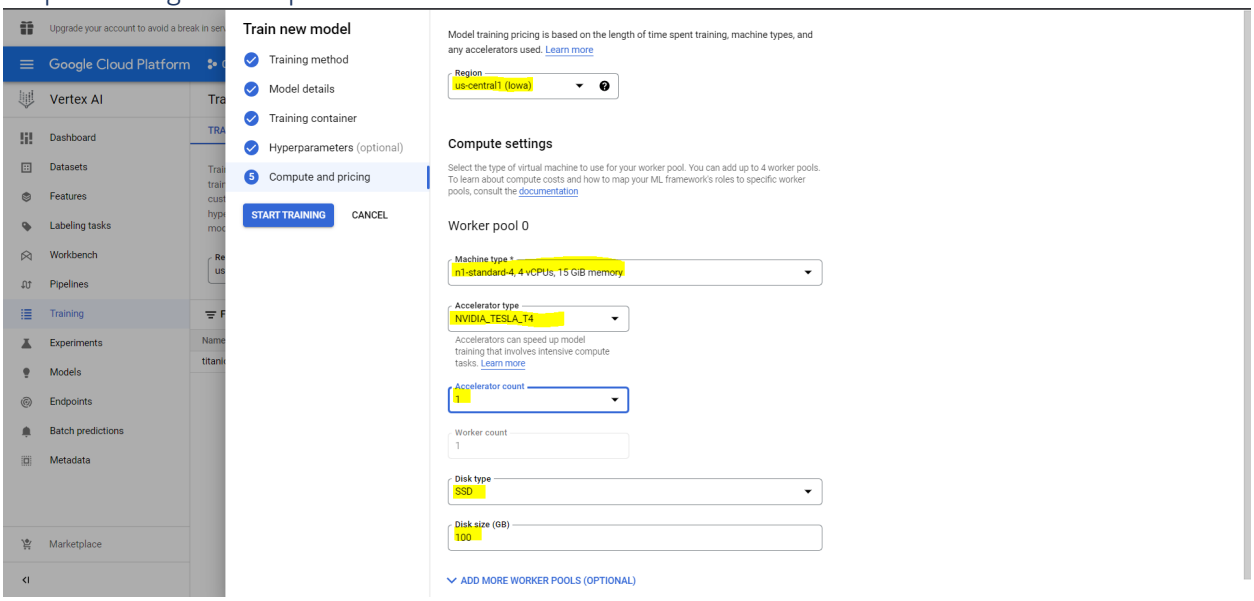



Upgrade your account to avoid a break in serv
Google Cloud Platform
Vertex AI
Dashboard
Datasets
Features
Labeling tasks
Workbench
Pipelines
Training
Experiments
Models
Endpoints
Batch predictions
Metadata
Marketplace
Train new model
Training method
Model details
Training container
4 Hyperparameters (optional)
5 Compute and pricing
START TRAINING
CANCEL
Hyperparameter tuning optimizes your model through multiple trials in one training job, but will increase the cost of this job. After training finishes, the best-performing model will be saved to your Model List. Learn more
Enable hyperparameter tuning
Hyperparameter tuning variables
Ensure that your hyperparameter variables are named and typed correctly.
VIEW DOCS
learning_rate (Double), 0.01 - 1
New Hyperparameter
Parameter name *
momentum
Type *
Double
Min *
0
Max *
1
Scaling *
Linear
CANCEL
DONE
ADD NEW PARAMETER

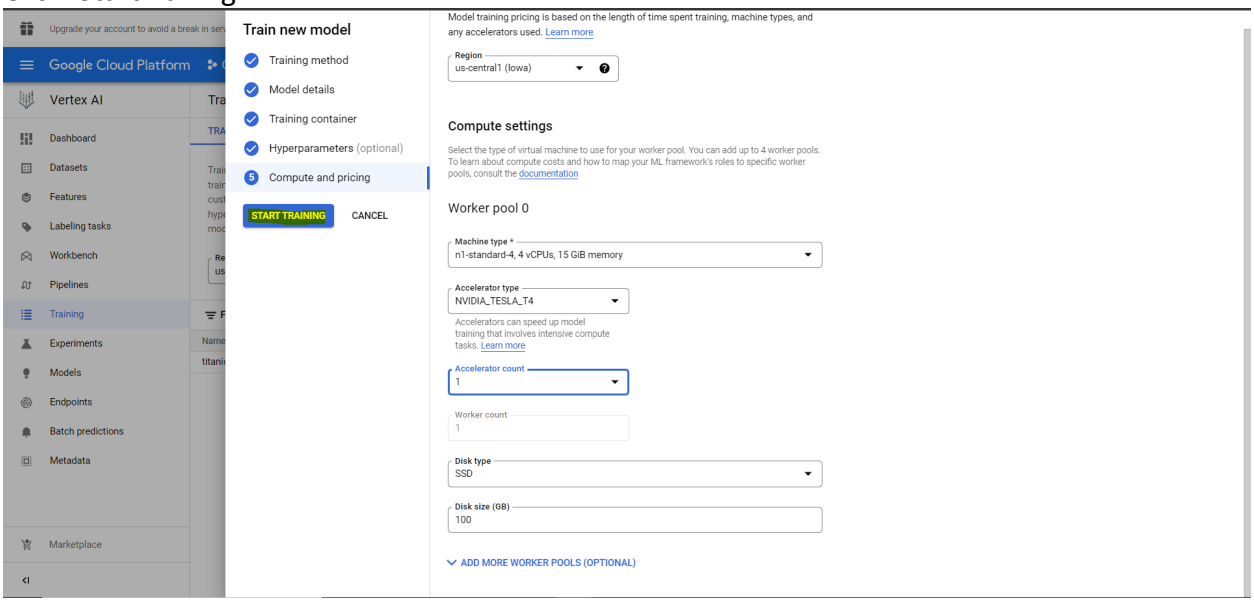

Upgrade your account to avoid a break in serv
Google Cloud Platform
Vertex AI
Dashboard
Datasets
Features
Labeling tasks
Workbench
Pipelines
Training
Experiments
Models
Endpoints
Batch predictions
Metadata
Marketplace
Train new model
Training method
Model details
Training container
4 Hyperparameters (optional)
5 Compute and pricing
START TRAINING
CANCEL
Hyperparameter tuning variables
Ensure that your hyperparameter variables are named and typed correctly.
VIEW DOCS
learning_rate (Double), 0.01 - 1
momentum (Double), 0 - 1
New Hyperparameter
Parameter name *
num_neurons
Type *
Discrete
Values *
64,128,512
Enter all the values you want to tune, separated with comma.
Scaling *
No scaling
CANCEL
DONE
ADD NEW PARAMETER
Metric to optimize *



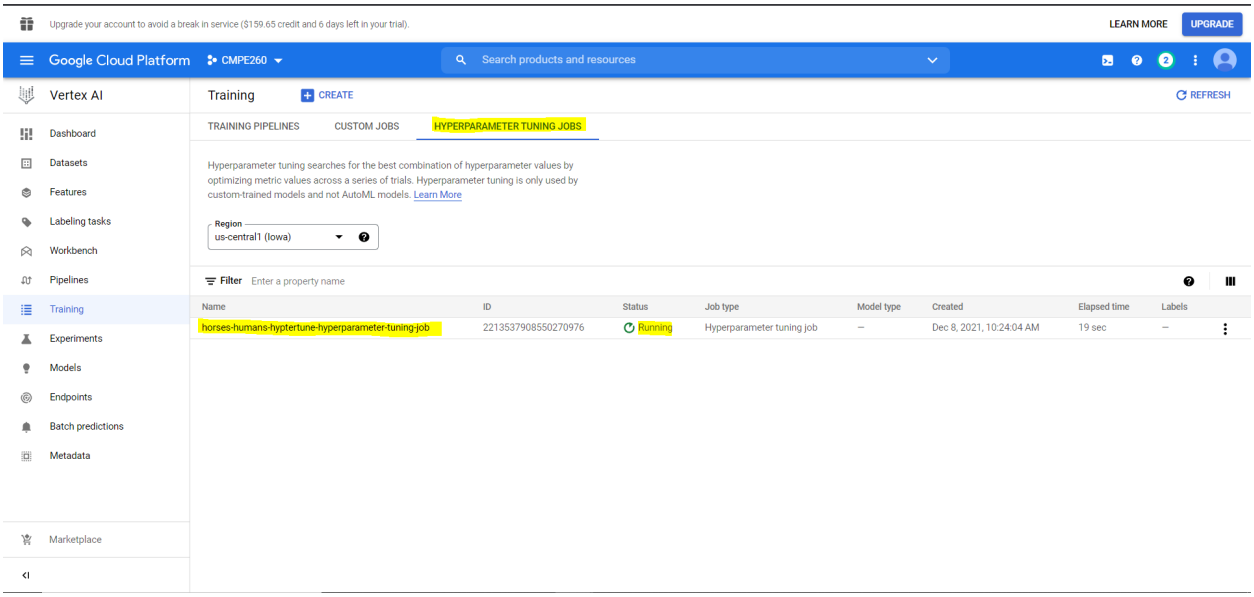
Step 3: Configure compute



Click ‘Start Tuning’

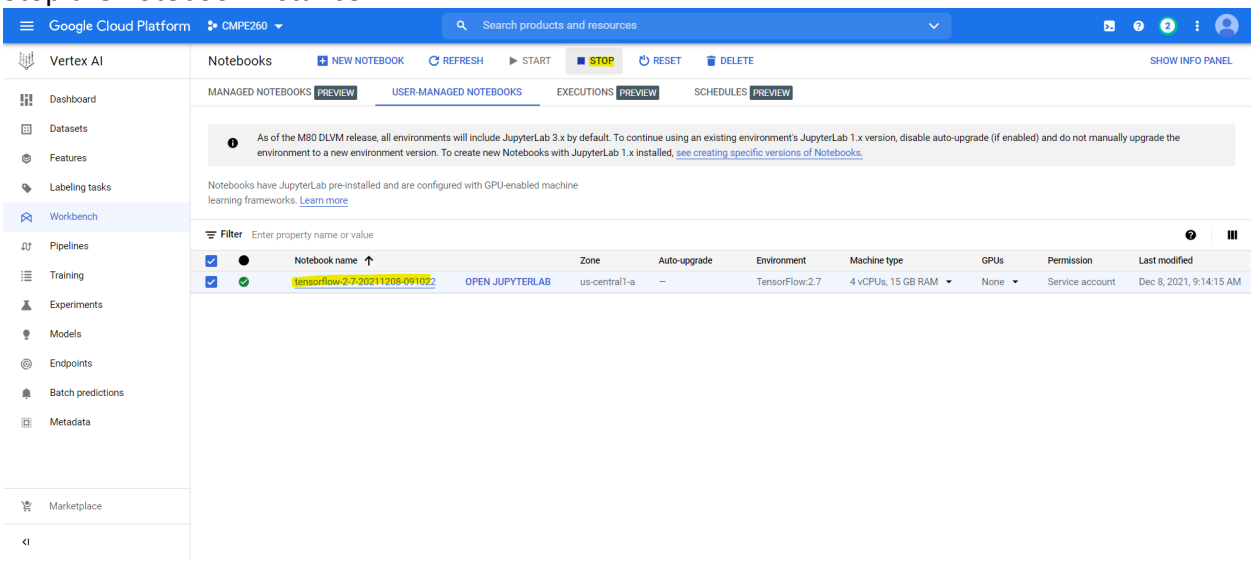


HyperParameter Tuning job is in progress:



Cleanup

Stop the notebook instance



Delete the Storage Bucket