### Part i - Vertex AI:

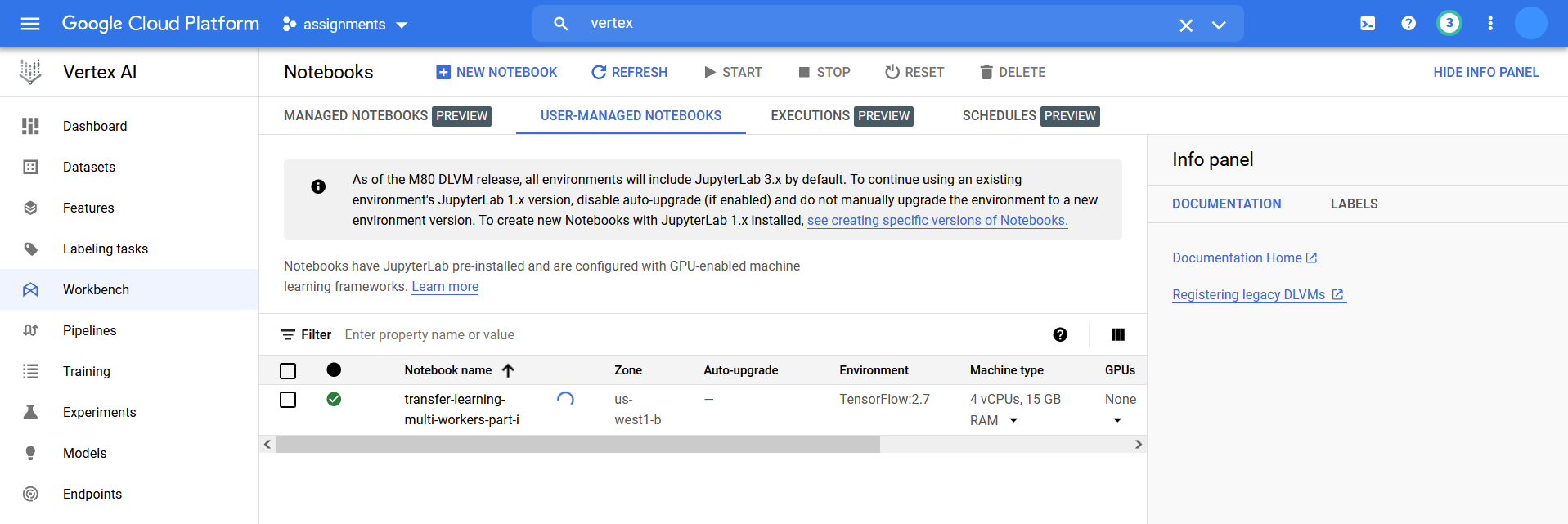
### Multi-Worker Training and Transfer Learning with TensorFlow

In this document, there are screenshots for -

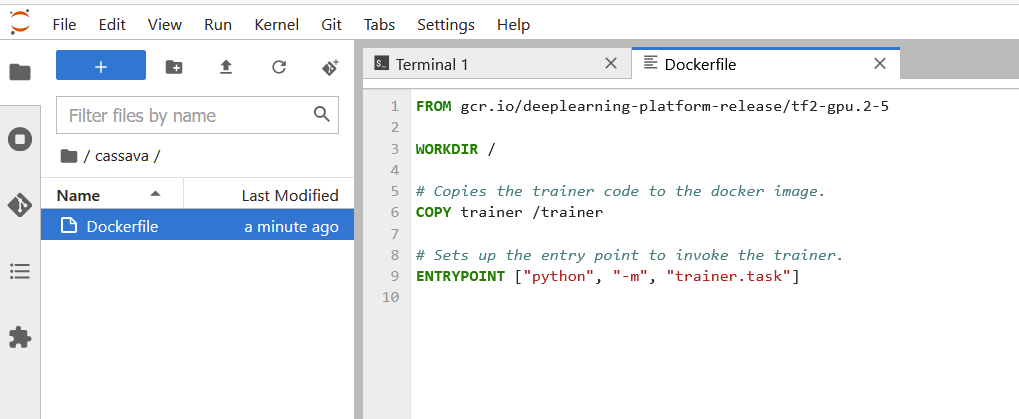
* Modify training application code for multi-worker training
* Configure and launch a multi-worker training job from the Vertex AI UI
* Configure and launch a multi-worker training job with the Vertex SDK

We did transfer learning to train an image classification model on the [cassava dataset](https://www.tensorflow.org/datasets/catalog/cassava) from [TensorFlow Datasets](https://www.tensorflow.org/datasets). The architecture we used was a ResNet50 model from the tf.keras.applications library pretrained on the [Imagenet dataset](https://www.image-net.org/)

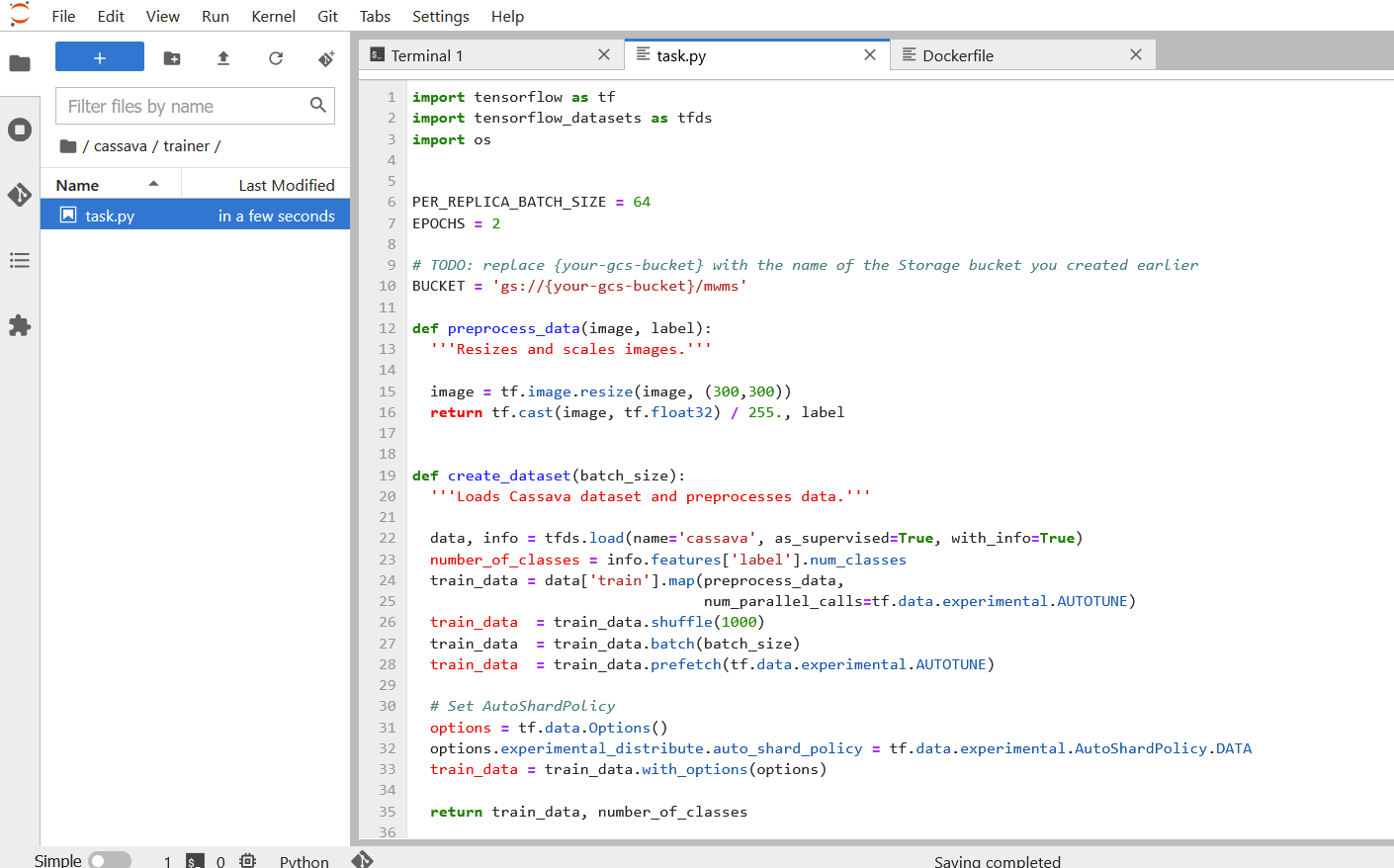
1. Creating a vertex AI notebook instance for containerizing the training code



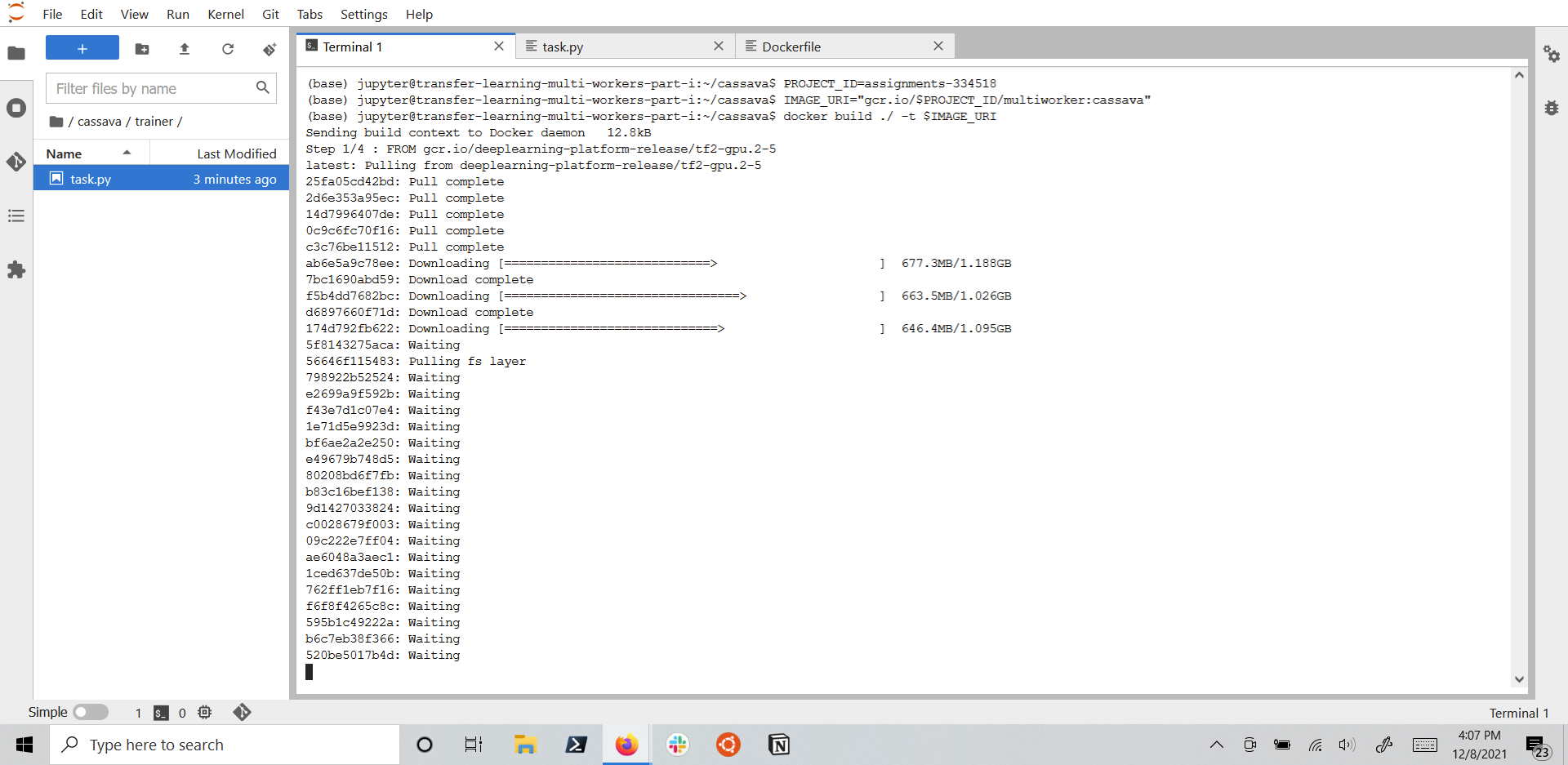
1. Creating a dockerfile which uses the [Deep Learning Container TensorFlow Enterprise 2.5 GPU Docker image](https://cloud.google.com/ai-platform/deep-learning-containers/docs/choosing-container#choose_a_container_image_type?utm_campaign=CDR_sar_aiml_ucaiplabs_011321&utm_source=external&utm_medium=web)



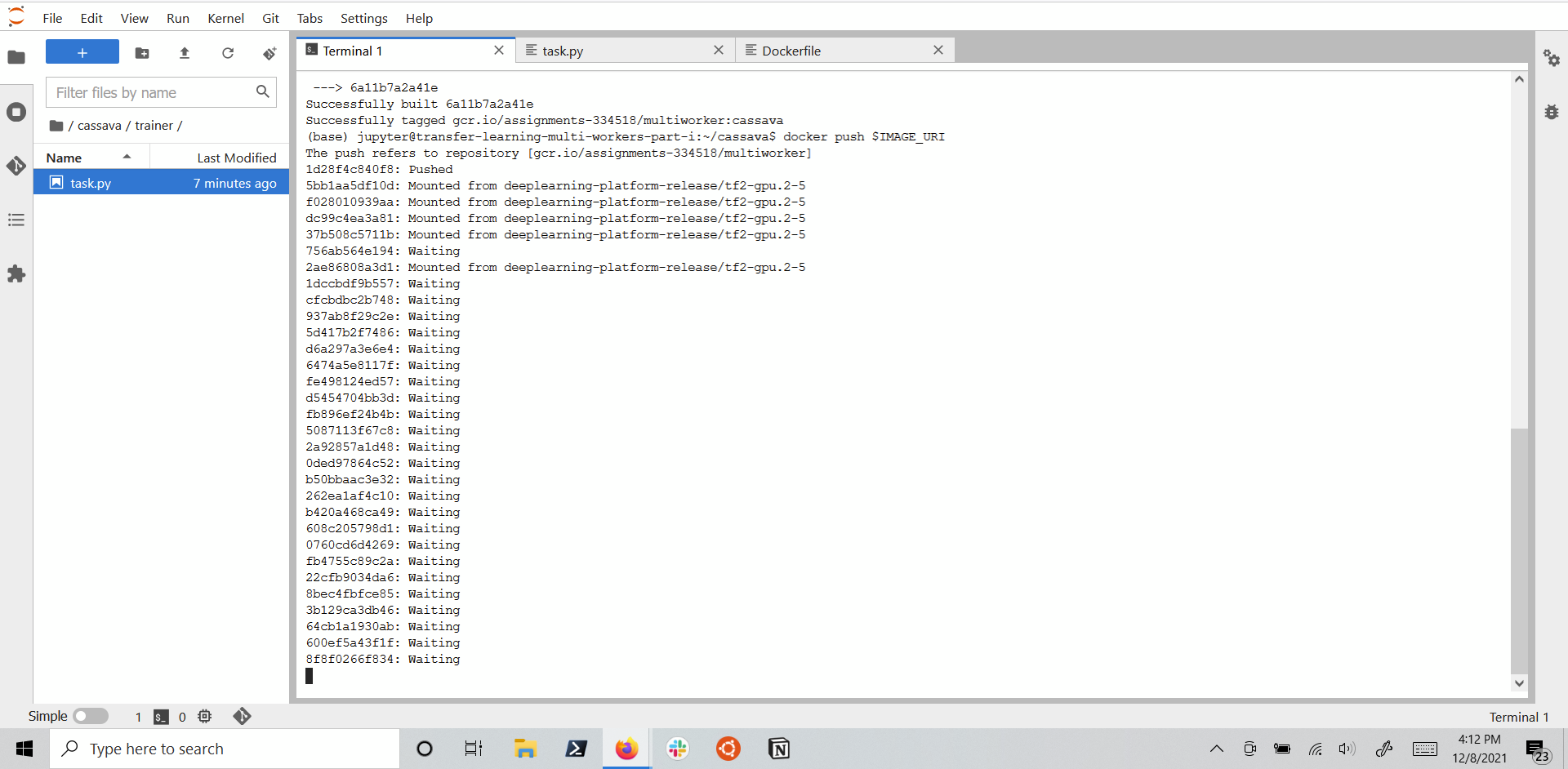
1. Adding the data preprocessing, data training code with distribution strategy (for multi workers) to task.py under cassava/trainer/. This directory will be containerized to be run later



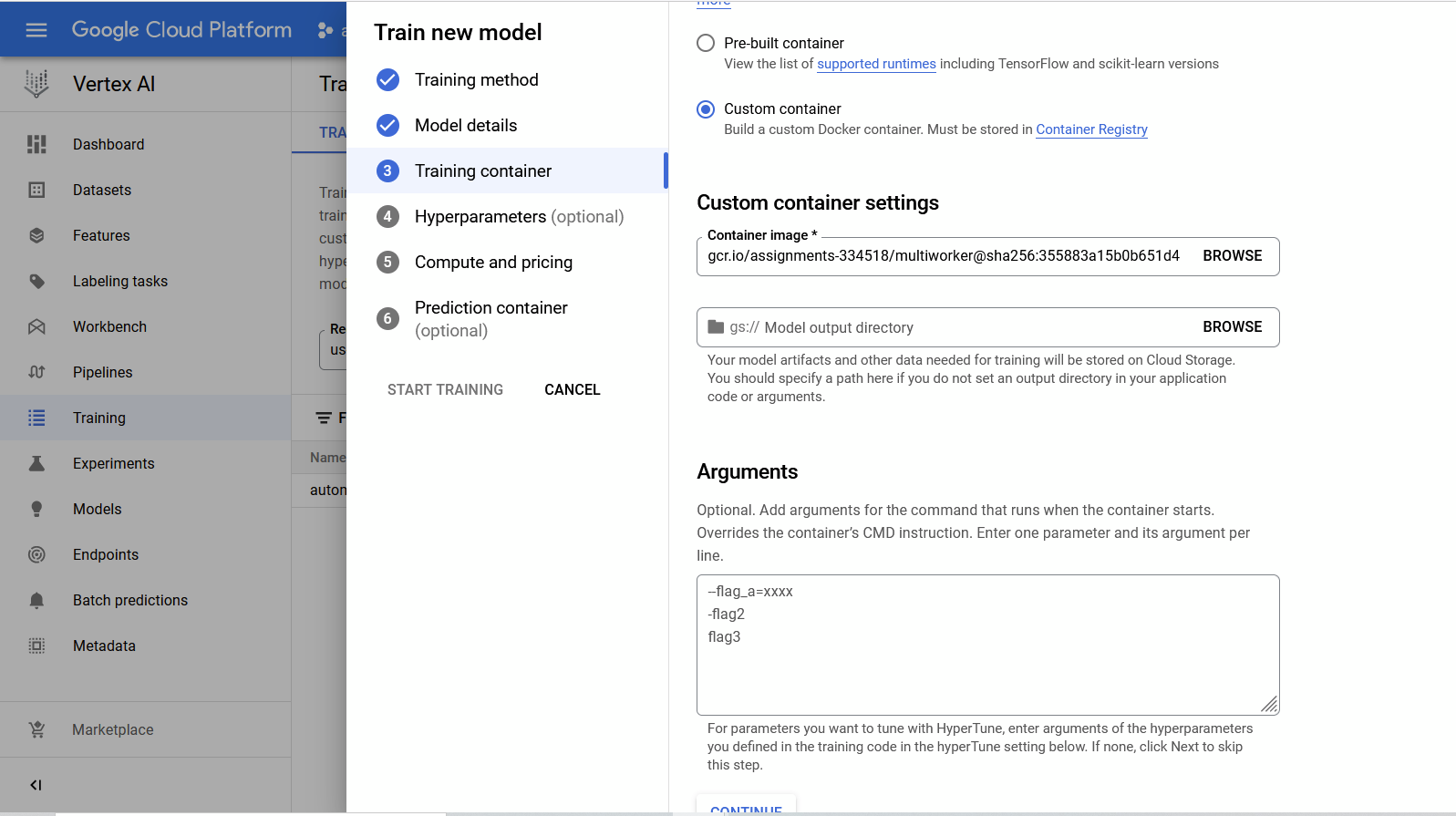
1. Defining a variable with the URI of the container image for google container registry, building the container



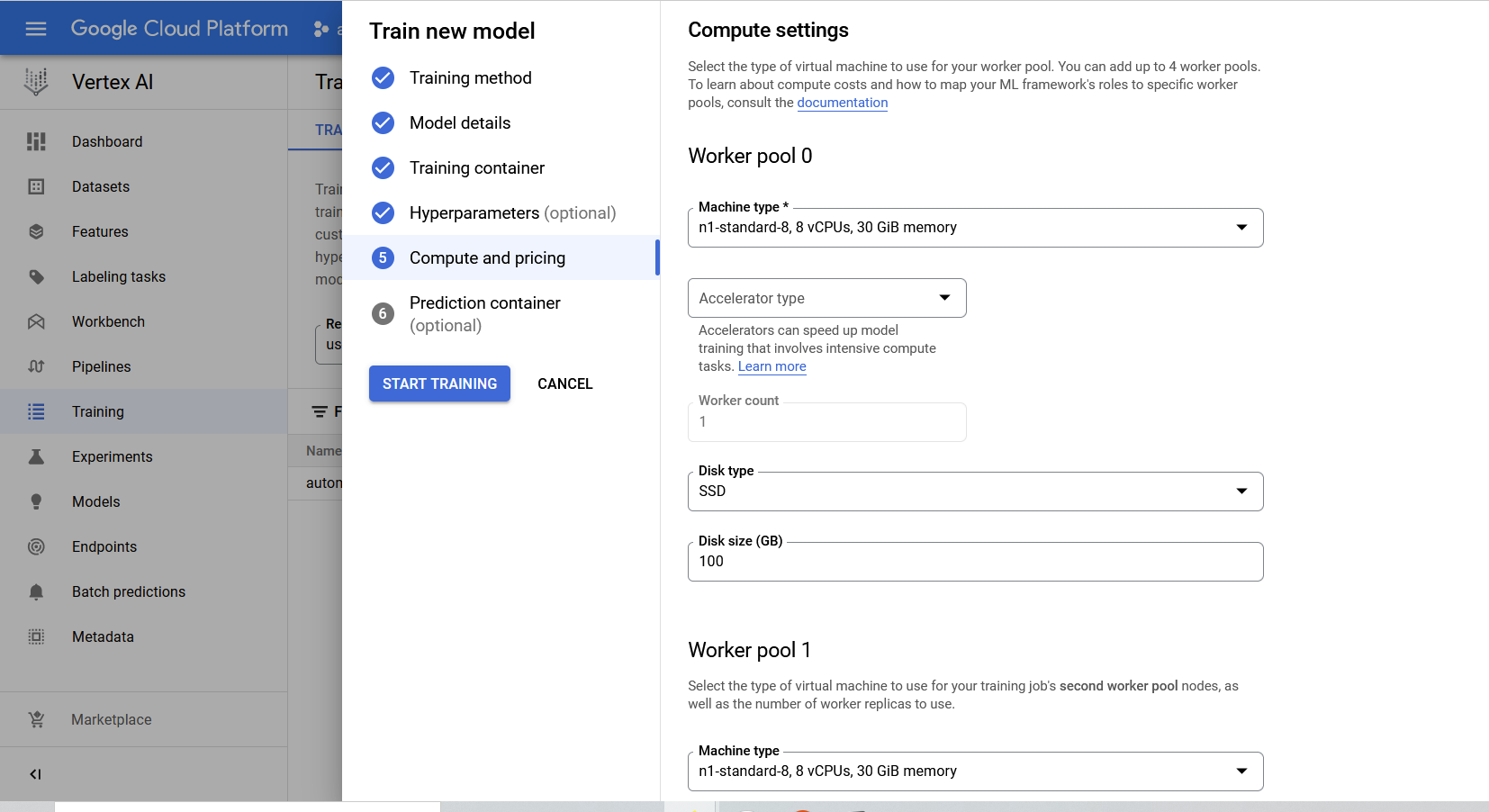
1. Pushing the just created container to the google container registry



1. Using a custom container image to train the model



1. Configuring compute to use multiple workers during training for distribution strategy



1. Training complete using distribution strategy with multiple workers defined in a custom training container

