Cloud ML Engine: Qwik Start

1 hour1 Credit

Rate Lab

**GSP076**



**Overview**

This lab will give you hands-on practice with [TensorFlow](http://tensorflow.org/) model training, both locally and on [Cloud ML Engine](https://cloud.google.com/ml-engine/docs/). After training, you will learn how to deploy your model to Cloud ML Engine for serving (prediction). You'll train your model to predict income category of a person using the [United States Census Income Dataset](https://archive.ics.uci.edu/ml/datasets/Census+Income).

This lab gives you an introductory, end-to-end experience of training and prediction on Cloud Machine Learning Engine. The lab will use a census dataset to:

* Create a TensorFlow training application and validate it locally.
* Run your training job on a single worker instance in the cloud.
* Run your training job as a distributed training job in the cloud.
* Optimize your hyperparameters by using hyperparameter tuning.
* Deploy a model to support prediction.
* Request an online prediction and see the response.
* Request a batch prediction.

What you will build

The sample builds a wide and deep model for predicting income category based on United States Census Income Dataset. The two income categories (also known as labels) are:

* **>50K** — Greater than 50,000 dollars
* **<=50K** — Less than or equal to 50,000 dollars

Wide and deep models use deep neural nets (DNNs) to learn high-level abstractions about complex features or interactions between such features. These models then combine the outputs from the DNN with a linear regression performed on simpler features. This provides a balance between power and speed that is effective on many structured data problems.

The sample defines the model using TensorFlow's prebuilt DNNCombinedLinearClassifier class. The sample defines the data transformations particular to the census dataset, then assigns these (potentially) transformed features to either the DNN or the linear portion of the model.

**Setup and Requirements**

**Before you click the Start Lab button**

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click Start Lab, shows how long Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access the Google Cloud Platform for the duration of the lab.

**What you need**

To complete this lab, you need:

* Access to a standard internet browser (Chrome browser recommended).
* Time to complete the lab.

***Note:*** If you already have your own personal GCP account or project, do not use it for this lab.

**How to start your lab and sign in to the Console**

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



1. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Choose an account** page.

***Tip:*** Open the tabs in separate windows, side-by-side.

1. On the Choose an account page, click **Use Another Account**.



1. The Sign in page opens. Paste the username that you copied from the Connection Details panel. Then copy and paste the password.

***Important:*** You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own GCP account, do not use it for this lab (avoids incurring charges).

1. Click through the subsequent pages:
   * Accept the terms and conditions.
   * Do not add recovery options or two-factor authentication (because this is a temporary account).
   * Do not sign up for free trials.

After a few moments, the GCP console opens in this tab.

**Note:** You can view the menu with a list of GCP Products and Services by clicking the **Navigation menu** at the top-left, next to “Google Cloud Platform”. 

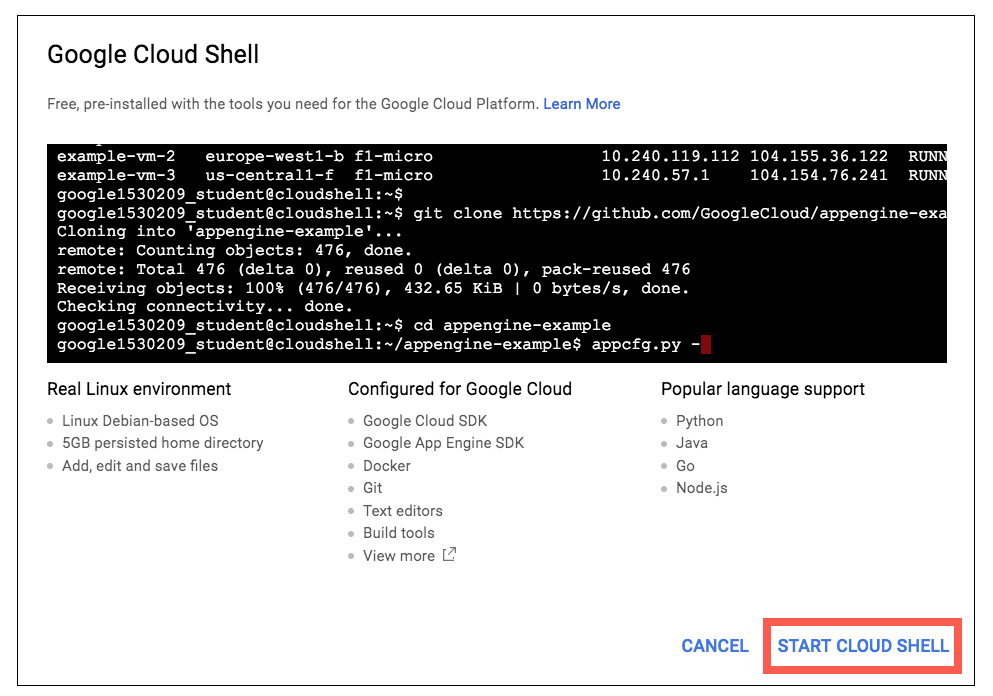
Activate Google Cloud Shell

Google Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Google Cloud Shell provides command-line access to your GCP resources.

1. In GCP console, on the top right toolbar, click the Open Cloud Shell button.



1. In the dialog box that opens, click **START CLOUD SHELL**:



You can click "START CLOUD SHELL" immediately when the dialog box opens.

It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT\_ID*. For example:



**gcloud** is the command-line tool for Google Cloud Platform. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

gcloud auth list

Output:

Credentialed accounts:

- <myaccount>@<mydomain>.com (active)

Example output:

Credentialed accounts:

- google1623327\_student@qwiklabs.net

You can list the project ID with this command:

gcloud config list project

Output:

[core]

project = <project\_ID>

Example output:

[core]

project = qwiklabs-gcp-44776a13dea667a6

Full documentation of **gcloud** is available on [Google Cloud gcloud Overview](https://cloud.google.com/sdk/gcloud).

**Install TensorFlow**

Run the following command to install TensorFlow:

pip install --user --upgrade tensorflow

Verify the installation:

python -c "import tensorflow as tf; print('TensorFlow version {} is installed.'.format(tf.VERSION))"

You can ignore any warnings that the TensorFlow library wasn't compiled to use certain instructions.

**Clone the example repo**

In Cloud Shell run the following command to clone the [cloudml-samples repo](https://github.com/GoogleCloudPlatform/cloudml-samples):

git clone https://github.com/GoogleCloudPlatform/cloudml-samples.git

Navigate to the cloudml-samples > census > estimator directory. The commands in this lab must be run from the estimator directory:

cd cloudml-samples/census/estimator

Your current directory should now be: ~/cloudml-samples/census/estimator.

The model that you'll use is in this directory, so stay inside it for the duration of the lab.

**Note:**

It's not necessary to look at the Python code to run this lab, but if you are interested, you can poke around the repo in the Cloud Shell editor.

**Develop and validate your training application locally**

Before you run your training application in the cloud, get it running locally. Local environments provide an efficient development and validation workflow so that you can iterate quickly. You also won't incur charges for cloud resources when debugging your application locally.

Get your training data

The relevant data files, adult.data and adult.test, are hosted in a public Google Cloud Storage bucket.

You can read the files directly from Cloud Storage or copy them to your local environment. For this lab you will download the samples for local training, and later upload them to your own Cloud Storage bucket for cloud training.

Run the following command to download the data to a local file directory and set variables that point to the downloaded data files:

mkdir data

gsutil -m cp gs://cloud-samples-data/ml-engine/census/data/\* data/

Now set the TRAIN\_DATA and EVAL\_DATA variables to your local file paths by running the following commands:

export TRAIN\_DATA=$(pwd)/data/adult.data.csv

export EVAL\_DATA=$(pwd)/data/adult.test.csv

To open the adult.data.csv file, run the following command:

head data/adult.data.csv

You will see that the data is stored in comma-separated value format that resembles the following:

39, State-gov, 77516, Bachelors, 13, Never-married, Adm-clerical, Not-in-family, White, Male, 2174, 0, 40, United-States, <=50K

50, Self-emp-not-inc, 83311, Bachelors, 13, Married-civ-spouse, Exec-managerial, Husband, White, Male, 0, 0, 13, United-States, <=50K

38, Private, 215646, HS-grad, 9, Divorced, Handlers-cleaners, Not-in-family, White, Male, 0, 0, 40, United-States, <=50K

53, Private, 234721, 11th, 7, Married-civ-spouse, Handlers-cleaners, Husband, Black, Male, 0, 0, 40, United-States, <=50K

...

Now that you have downloaded and inspected your training data, you will install the necessary dependencies.

**Install dependencies**

Although TensorFlow is installed on Cloud Shell, you must run the sample's requirements.txt file to ensure you are using the same version of TensorFlow required by the sample:

pip install --user -r ../requirements.txt

It will take a couple minutes for this command to complete. You will receive a similar output when it does:

Successfully installed Keras-2.2.4 future-0.16.0 numexpr-2.6.9 numpy-1.14.5 pandas-0.24.1 python-dateutil-2.8.0 pyyaml-3.13 scipy-1.2.1 setuptools-39.1.0 tensorboard-1.10.0 tensorflow-1.10.0

**Run a local training job**

A local training job loads your Python training program and starts a training process in an environment that's similar to that of a live Cloud ML Engine cloud training job.

Specify an output directory and set a MODEL\_DIR variable by running the following command:

export MODEL\_DIR=output

Run this training job locally by running the following command:

gcloud ai-platform local train \

--module-name trainer.task \

--package-path trainer/ \

--job-dir $MODEL\_DIR \

-- \

--train-files $TRAIN\_DATA \

--eval-files $EVAL\_DATA \

--train-steps 1000 \

--eval-steps 100

**Note:**

When you run the same training job on CMLE later in the lab, you'll see that the command is not much different from the above.

By default, verbose logging is turned off. You can enable it by setting the --verbosity tag to DEBUG. A later example shows you how to enable it.

Inspect the summary logs using Tensorboard

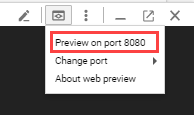
To see the evaluation results, you can use the visualization tool called[TensorBoard](https://www.tensorflow.org/programmers_guide/summaries_and_tensorboard). With TensorBoard, you can visualize your TensorFlow graph, plot quantitative metrics about the execution of your graph, and show additional data like images that pass through the graph. Tensorboard is available as part of the TensorFlow installation.

Follow the steps below to launch TensorBoard and point it at the summary logs produced during training, both during and after execution.

Launch TensorBoard:

tensorboard --logdir=$MODEL\_DIR --port=8080

Click on the **Web Preview** icon, then **Preview on port 8080**. A new tab will open with TensorBoard running.



Click on **Accuracy** to see graphical representations of how accuracy changes as your job progresses.

Type CTRL+C in Cloud Shell to shut down TensorBoard.

Running model prediction locally (in Cloud Shell)

The output/export/census directory holds the model exported as a result of running training locally. List that directory to see the generated timestamp subdirectory:

ls output/export/census/

You should receive a similar output:

1547669983

Copy the timestamp that was generated.

Run the following command in Cloud Shell, replacing <timestamp> with your timestamp:

gcloud ai-platform local predict \

--model-dir output/export/census/<timestamp> \

--json-instances ../test.json

You should see a result that looks something like the following:

CLASS\_IDS CLASSES LOGISTIC LOGITS PROBABILITIES

[0] [u'0'] [0.0583675391972065] [-2.780855178833008] [0.9416324496269226, 0.0583675354719162]

Where class 0 means income <= 50k and class 1 means income >50k.

**Use your trained model for prediction**

Once you've trained your TensorFlow model, you can use it for prediction on new data. In this case, you've trained a census model to predict income category given some information about a person.

For this section we'll use a predefined prediction request file, test.json, included in the GitHub repository in the census directory. You can inspect the JSON file in the Cloud Shell editor if you're interested in learning more.

**Run your training job in the cloud**

Now that you've validated your model by running it locally, you will now get practice training is using Cloud ML Engine. You'll remain in the Cloud Shell for this section.

**Note:**The initial job request will take several minutes to start, but subsequent jobs run more quickly. This enables quick iteration as you develop and validate your training job.

Set up a Google Cloud Storage bucket

The Cloud ML Engine services need to access [Google Cloud Storage](https://cloud.google.com/storage/) (GCS) to read and write data during model training and batch prediction.

First, set the following variables:

PROJECT\_ID=$(gcloud config list project --format "value(core.project)")

BUCKET\_NAME=${PROJECT\_ID}-mlengine

echo $BUCKET\_NAME

REGION=us-central1

Create the new bucket:

gsutil mb -l $REGION gs://$BUCKET\_NAME

Test Completed Task

Click **Check my progress** to verify your performed task.

Set up a Google Cloud Storage bucket.

Check my progress

Upload the data files to your Cloud Storage bucket.

Use gsutil to copy the two files to your Cloud Storage bucket:

gsutil cp -r data gs://$BUCKET\_NAME/data

Set the TRAIN\_DATA and EVAL\_DATA variables to point to the files:

TRAIN\_DATA=gs://$BUCKET\_NAME/data/adult.data.csv

EVAL\_DATA=gs://$BUCKET\_NAME/data/adult.test.csv

Use gsutil again to copy the JSON test file test.json to your Cloud Storage bucket:

gsutil cp ../test.json gs://$BUCKET\_NAME/data/test.json

Set the TEST\_JSON variable to point to that file:

TEST\_JSON=gs://$BUCKET\_NAME/data/test.json

Test Completed Task

Click **Check my progress** to verify your performed task.

Upload the data files to your Cloud Storage bucket.

Check my progress

**Run a single-instance trainer in the cloud**

With a validated training job that runs in both single-instance and distributed mode, you're now ready to run a training job in the cloud. You'll start by requesting a single-instance training job.

Use the default BASIC scale tier to run a single-instance training job. The initial job request can take a few minutes to start, but subsequent jobs run more quickly. This enables quick iteration as you develop and validate your training job.

Select a name for the initial training run that distinguishes it from any subsequent training runs. For example, you can append a number to represent the iteration:

JOB\_NAME=census\_single\_1

Specify a directory for output generated by Cloud ML Engine by setting an OUTPUT\_PATH variable to include when requesting training and prediction jobs. The OUTPUT\_PATH represents the fully qualified Cloud Storage location for model checkpoints, summaries, and exports. You can use the BUCKET\_NAME variable you defined in a previous step. It's a good practice to use the job name as the output directory:

OUTPUT\_PATH=gs://$BUCKET\_NAME/$JOB\_NAME

Run the following command to submit a training job in the cloud that uses a single process. This time, set the --verbosity tag to DEBUG so that you can inspect the full logging output and retrieve accuracy, loss, and other metrics. The output also contains a number of other warning messages that you can ignore for the purposes of this sample:

gcloud ai-platform jobs submit training $JOB\_NAME \

--job-dir $OUTPUT\_PATH \

--runtime-version 1.10 \

--module-name trainer.task \

--package-path trainer/ \

--region $REGION \

-- \

--train-files $TRAIN\_DATA \

--eval-files $EVAL\_DATA \

--train-steps 1000 \

--eval-steps 100 \

--verbosity DEBUG

You can monitor the progress of your training job by watching the logs on the command line:

gcloud ai-platform jobs stream-logs $JOB\_NAME

Or monitor in the Console: **ML Engine** > **Jobs**.

Test Completed Task

Click **Check my progress** to verify your performed task.

Run a single-instance trainer in the cloud.

Check my progress

**Inspect the output**

If you have been monitoring the job in Cloud Shell, you will know you're done when your output resembles the following:

INFO 2019-01-16 12:58:34 -0800 master-replica-0 Task completed successfully.

In cloud training, outputs are produced in Cloud Storage. In this sample, outputs are saved to OUTPUT\_PATH; to list them, run:

gsutil ls -r $OUTPUT\_PATH

The outputs should be similar to the outputs from training locally (above).

**Optional:** You can point TensorBoard to this directory, either during or after training using the **Web Preview** menu at the top of the Cloud Shell and again select **Preview on port 8080**:

tensorboard --logdir=$OUTPUT\_PATH --port=8080

You can shut down TensorBoard by typing **ctrl** + **c** on the command line.

**Deploy your model to support prediction**

By deploying your trained model to Cloud ML Engine to serve online prediction requests, you get the benefit of scalable serving. This is useful if you expect your trained model to be hit with many prediction requests in a short period of time.

Wait until your CMLE training job is done. It is finished when you see a green check mark by the jobname in the Cloud Console, or when you see the message **Job completed successfully** in the Cloud Shell command line.

Create a Cloud ML Engine model:

MODEL\_NAME=census

Create a Cloud ML Engine model:

gcloud ai-platform models create $MODEL\_NAME --regions=$REGION

Test Completed Task

Click **Check my progress** to verify your performed task.

Create a Cloud ML Engine model.

Check my progress

Select the exported model to use, by looking up the full path of your exported trained model binaries.

gsutil ls -r $OUTPUT\_PATH/export

Scroll through the output to find the value of $OUTPUT\_PATH/export/census/<timestamp>/. Copy timestamp and add it to the following command to set the environment variable MODEL\_BINARIES to its value:

MODEL\_BINARIES=$OUTPUT\_PATH/export/census/<timestamp>/

**Note:**The timestamp for this training run will not be the same as the timestamp generated during your local training run above. Be sure to scroll through the gsutil ls output to find this new timestamp.

You'll deploy this trained model.

Run the following command to create a version v1 of your model:

gcloud ai-platform versions create v1 \

--model $MODEL\_NAME \

--origin $MODEL\_BINARIES \

--runtime-version 1.10

Test Completed Task

Click **Check my progress** to verify your performed task.

Create a version v1 of your model.

Check my progress

It may take several minutes to deploy your trained model. When done, you can see a list of your models using the models list command:

gcloud ai-platform models list

**Send an online prediction request to your deployed model**

You can now send prediction requests to your deployed model. The following command sends a prediction request using the test.json file included in the GitHub repository.

gcloud ai-platform predict \

--model $MODEL\_NAME \

--version v1 \

--json-instances ../test.json

The response includes the probabilities of each label (**>50K** and **<=50K**) based on the data entry in test.json, thus indicating whether the predicted income is greater than or less than 50,000 dollars

CLASS\_IDS CLASSES LOGISTIC LOGITS PROBABILITIES

[0] [u'0'] [0.06142739951610565] [-2.726504325866699] [0.9385725855827332, 0.061427392065525055]

Where class 0 means *income <= 50k* and class 1 means *income >50k*.

**Note:** CMLE supports batch prediction too, but it's not included in this lab. See the documentation for more info.

**Test your Understanding**

Below are a multiple choice questions to reinforce your understanding of this lab's concepts. Answer them to the best of your abilities.

A model version is an instance of a machine learning solution stored in the Cloud ML Engine model service.

True

False

Cloud ML Engine offers training jobs and batch prediction jobs.

True

False

**Congratulations!**

In this lab, you've learned how to train a [TensorFlow](http://tensorflow.org/) model both locally and on[Cloud ML Engine](https://cloud.google.com/ml-engine/docs/), and then how to use your trained model for prediction.

**Finish your quest**

This self-paced lab is part of the Qwiklabs Quests [Machine Learning APIs](https://google.qwiklabs.com/quests/32),[Baseline: Data, ML, AI](https://google.qwiklabs.com/quests/34), [Intro to ML: Language Processing](https://google.qwiklabs.com/quests/82) and [Intro to ML: Image Processing](https://google.qwiklabs.com/quests/85). A Quest is a series of related labs that form a learning path. Completing a Quest earns you a badge to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in a Quest and get immediate completion credit if you've taken this lab. [See other available Qwiklabs Quests](http://google.qwiklabs.com/catalog).

**Take your next lab**

Try out another lab on Machine Learning APIs, like [Extract, Analyze, and Translate Text from Images with the Cloud ML APIs](https://google.qwiklabs.com/catalog_lab/1111) or [Awwvision: Cloud Vision API from a Kubernetes Cluster](https://google.qwiklabs.com/catalog_lab/1041).

This lab is also part of a series of labs called Qwik Starts. These labs are designed to give you a little taste of the many features available with Google Cloud. Search for "Qwik Starts" in the [lab catalog](https://google.qwiklabs.com/catalog) to find the next lab you'd like to take!

**Next steps**

* Sign up for the full [Coursera Course on Machine Learning](https://www.coursera.org/learn/serverless-machine-learning-gcp/).
* Learn more about [TensorFlow Wide & Deep](https://www.tensorflow.org/tutorials/wide_and_deep)
* You can read more about wide and deep models in the Google Research Blog post named [Wide & Deep Learning: Better Together with TensorFlow](https://ai.googleblog.com/2016/06/wide-deep-learning-better-together-with.html).
* Get your [own version of Tensorflow](https://www.tensorflow.org/install/).

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. [Our classes](https://cloud.google.com/training/courses) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](https://cloud.google.com/certification/)help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated: July 1, 2019

Lab Last Tested: May 3, 2019

Copyright 2019 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.