

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

- a) Custom training job and prediction using managed datasets

Reference: <https://codelabs.developers.google.com/codelabs/vertex-ai-custom-code-training#0>

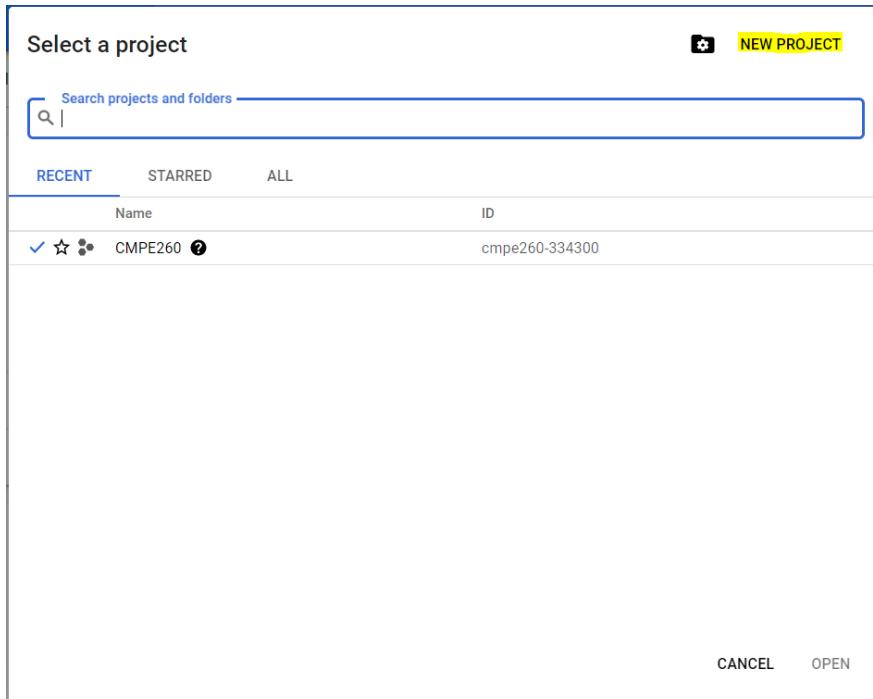
Objectives:

- Use Vertex AI to train and deploy a ML model
- Use Datasets for dataset creation and management, and custom model for training a Scikit Learn model.
- Deploy the trained model and get online predictions

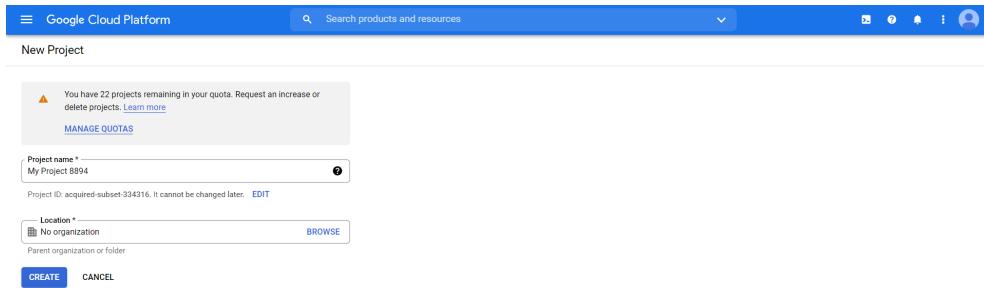
Setup the environment

Create a project

- To create a project, check if our Role has the resourcemanager.projects.create permission
- Login to Cloud Console.
- On the **Select organization** drop-down list at the top of the page, select the organization in which you want to create a project. If you are a free trial user, skip this step, as this list does not appear.
- Click **Create Project**.



- In the **New Project** window that appears, enter a project name and select a billing account as applicable. A project name can contain only letters, numbers, single quotes, hyphens, spaces, or exclamation points, and must be between 4 and 30 characters.



- Enter the parent organization or folder in the **Location** box. That resource will be the hierarchical parent of the new project.
- When you're finished entering new project details, click **Create**.

Load data in BigQuery

In order to train a Machine Learning model, you need access to data. BigQuery is a serverless, highly scalable, and cost-effective multi-cloud data warehouse and it is the perfect service for keeping your data.

Create dataset

- Make sure that you select the right project from the top of console page
- Navigate to Big Query
- Select the project you want to create the Dataset in
- Click Create Dataset

The screenshot shows the 'Create dataset' dialog in the Google Cloud Platform BigQuery interface. The dataset ID is set to 'cmpe260-334300'. The 'Data location (optional)' dropdown is set to 'Default'. Under 'Default table expiration', the 'Never' radio button is selected. In the 'Encryption' section, the 'Google-managed key' radio button is selected, with a note that no configuration is required. At the bottom, there are 'CREATE DATASET' and 'Cancel' buttons.

The screenshot shows the Google Cloud Platform interface with the 'Create dataset' dialog open. The sidebar on the left shows a pinned project 'cmpe260-334300'. The main area is titled 'Create dataset' with fields for 'Project ID' (cmpe260-334300), 'Dataset ID' (titanic), 'Data location' (us-east1 (South Carolina)), and 'Default table expiration' (checkbox unchecked). Under 'Encryption', the 'Google-managed encryption key' option is selected. At the bottom are 'CREATE DATASET' and 'CANCEL' buttons.

5. Select the 'titanic' dataset created above
6. Click 'Create Table'

From the Sidebar select the following:

7. Create table from: Upload
8. Select file: Use the downloaded titanic dataset
9. File Format: CSV
10. Table Name: survivors
11. Auto-detect: Select auto-detect checkbox - Schema and input parameters
12. Click 'Create Table' button.

The screenshot shows the 'Create table' dialog. In the 'Source' section, 'Upload' is selected and 'titanic.csv' is chosen via a 'BROWSE' button. The 'File format' is set to 'CSV'. In the 'Destination' section, the 'Project' is 'cmpe260-334300', the 'Dataset' is 'titanic', and the 'Table' is 'survivors'. The 'Schema' section has 'Auto detect' checked. At the bottom are 'CREATE TABLE' and 'CANCEL' buttons.

Create a dataset

Datasets in Vertex AI allow you to create datasets for your Machine Learning workloads. You can create datasets for structured data (CSV files or BigQuery tables) or unstructured data such as Images and Text.

Create ML dataset

1. Find Vertex AI on the GCP side menu, under Artificial Intelligence

The screenshot shows the Google Cloud Platform dashboard. The left sidebar has a 'Navigation menu' with sections like Financial Services, Healthcare, Life Sciences, and Dataprep. Under 'ARTIFICIAL INTELLIGENCE', 'Vertex AI' is selected, which has sub-options: Dashboard, Datasets, Features, Labeling tasks, Document AI, Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. The main content area shows a dataset named 'survivors'. It has a schema table with columns: survived (INTEGER, NULLABLE), name (STRING, NULLABLE), sex (STRING, NULLABLE), age (STRING, NULLABLE), sibsp (INTEGER, NULLABLE), parch (INTEGER, NULLABLE), ticket (STRING, NULLABLE), fare (STRING, NULLABLE), cabin (STRING, NULLABLE), embarked (STRING, NULLABLE), boat (STRING, NULLABLE), body (STRING, NULLABLE), and home.dest (STRING, NULLABLE). There are buttons for 'EDIT SCHEMA' and 'VIEW ROW ACCESS POLICIES'. At the bottom, it says 'PROJECT HISTORY' and 'SAVED QUERIES'.

2. Select 'Enable Vertex AI API'

The screenshot shows the 'Get started with Vertex AI' page. It features a 'ENABLE VERTEX AI API' button. Below it, there's a 'Region' dropdown set to 'us-east1 (South Carolina)'. To the right, there's a diagram illustrating machine learning components: a lightbulb, a neural network, a graph, and two satellite dishes. Below the diagram, there are three boxes: 'Prepare your training data' (with '+ CREATE DATASET' button), 'Train your model' (with '+ TRAIN NEW MODEL' button), and 'Get predictions' (with '+ CREATE BATCH PREDICTION' button).

3. Select the Region, and click 'Create Dataset' button

Get started with Vertex AI

Vertex AI empowers machine learning developers, data scientists, and data engineers to take their projects from ideation to deployment, quickly and cost-effectively. [Learn more](#)

Try an interactive tutorial to learn how to train, evaluate, and deploy a Vertex AI AutoML or custom-trained model.

[VIEW TUTORIALS](#)

Region: us-east1 (South Carolina)

Prepare your training data
Collect and prepare your data, then import it into a dataset to train a model.
[+ CREATE DATASET](#)

Train your model
Train a best-in-class machine learning model with your dataset. Use Google's AutoML, or bring your own code.
[+ TRAIN NEW MODEL](#)

Get predictions
After you train a model, you can use it to get predictions, either online as an endpoint or through batch requests.
[+ CREATE BATCH PREDICTION](#)

4. Enter dataset name as 'titanic'
5. We can create datasets for images, text or videos as well as tabular data. The 'titanic' dataset is tabular so click the Tabular tab

Create dataset

Dataset name: Can use up to 128 characters.

Select a data type and objective
First select the type of data your dataset will contain. Then select an objective, which is the outcome that you want to achieve with the trained model. [Learn more about model types](#)

IMAGE **TABULAR** TEXT VIDEO

Regression/classification Predict a target column's value. Supports tables with hundreds of columns and millions of rows.

Forecasting [PREVIEW](#) Predicting the likelihood of certain events or demand.

Region: us-central1 (Iowa)

ADVANCED OPTIONS

CREATE CANCEL

6. Select 'Tabular' and 'Regression/Classification'
7. Click the 'Create' button

Select datasource

1. As we had already loaded the titanic dataset in BigQuery, we can connect our ML dataset to our BigQuery table.
2. Select 'Select a table or view from BigQuery'
3. Select the BigQuery Path
4. Select the 'survivors' table.

5. Click 'Continue'
6. In the 'Analyze' tab we can generate statistics regarding your data. This gives you the ability to quickly have a peek at the data and check for distributions, missing values etc.

Column name	BigQuery type	BigQuery mode	Missing % (count)	Distinct values
age	STRING	NULLABLE	-	-
boat	STRING	NULLABLE	-	-
body	STRING	NULLABLE	-	-
cabin	STRING	NULLABLE	-	-
embarked	STRING	NULLABLE	-	-
fare	STRING	NULLABLE	-	-
home_dest	STRING	NULLABLE	-	-
name	STRING	NULLABLE	-	-
parch	INTEGER	NULLABLE	-	-
sibsp	INTEGER	NULLABLE	-	-

7. Click on 'Generate Statistics'

General statistics generated by Dec 06, 2021 4:35 PM [GENERATE STATISTICS](#)

Column name ↑	BigQuery type	BigQuery mode	Missing % (count) ↗	Distinct values ↗
age	STRING	NULLABLE	-	99
boat	STRING	NULLABLE	-	28
body	STRING	NULLABLE	-	122
cabin	STRING	NULLABLE	-	187
embarked	STRING	NULLABLE	-	4
fare	STRING	NULLABLE	-	282
home.dest	STRING	NULLABLE	-	370
name	STRING	NULLABLE	-	1307
parch	INTEGER	NULLABLE	-	8
pclass	INTEGER	NULLABLE	-	3
sex	STRING	NULLABLE	-	2
sibsp	INTEGER	NULLABLE	-	7
survived	INTEGER	NULLABLE	-	2
ticket	STRING	NULLABLE	-	929

Rows per page: 50 ▾ 1 – 14 of 14

- Statistics are generated.

Custom training package using Notebooks

It is a good practice to package and parameterise your code so that it becomes a portable asset.

- Create a training package with custom code using Notebooks

Create your notebook instance

- From the Vertex AI navigate to notebooks and start an instance with **Python 3**, which includes scikit-learn
- Select 'Vertex AI', Workbench
- Enable Notebooks
- On the top, select 'New Notebook'

Up upgrade your account to avoid a break in service (\$159.65 credit and 8 days left in your trial). [LEARN MORE](#) [UPGRADE](#)

Google Cloud Platform CMPE260

Search products and resources

Vertex AI Notebooks [NEW NOTEBOOK](#) [REFRESH](#) [START](#) [STOP](#) [RESET](#) [DELETE](#) [SHOW INFO PANEL](#)

MANAGED NOTEBOOKS

CUSTOMIZE...

As of the ML environment

Includes scikit-learn, pandas and more

Python 3 (CUDA Toolkit 11.0)

Optimized for NVIDIA GPUs

TensorFlow Enterprise

Includes Keras, scikit-learn, pandas, NLTK and more

PyTorch 1.9

Includes scikit-learn, pandas, NLTK and more

R 4.1

Includes basic R packages, scikit-learn, pandas, NLTK and more

RAPIDS 0.18 [EXPERIMENTAL]

Optimized for NVIDIA GPUs

Kaggle Python [BETA]

Python image for Kaggle Notebooks, supporting hundreds of machine learning libraries popular on Kaggle

Theta IDE [EXPERIMENTAL]

IDE with notebook support including scikit-learn, pandas, and more

Smart Analytics Frameworks

BigQuery, Apache Beam, Apache Spark, Apache Hive, and more

SCHEDULES PREVIEW

You don't have any notebooks in this project yet

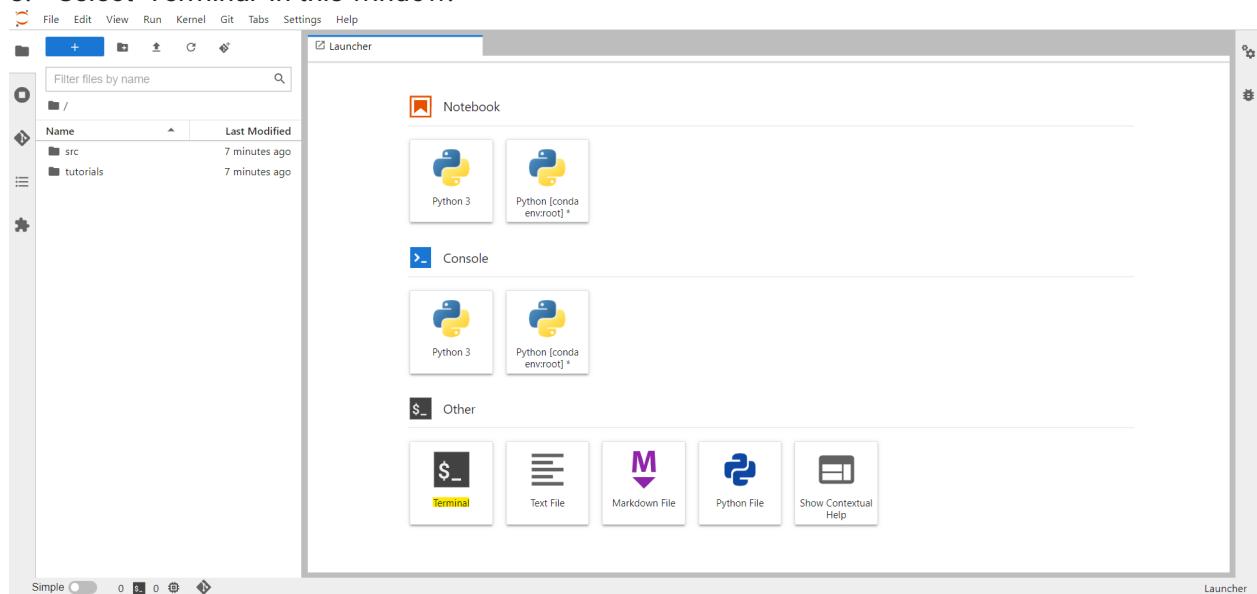
- Select 'Python3', with scikit learn module.

6. The Notebook instance is created.

7. Now click on 'Open JupyterLab'

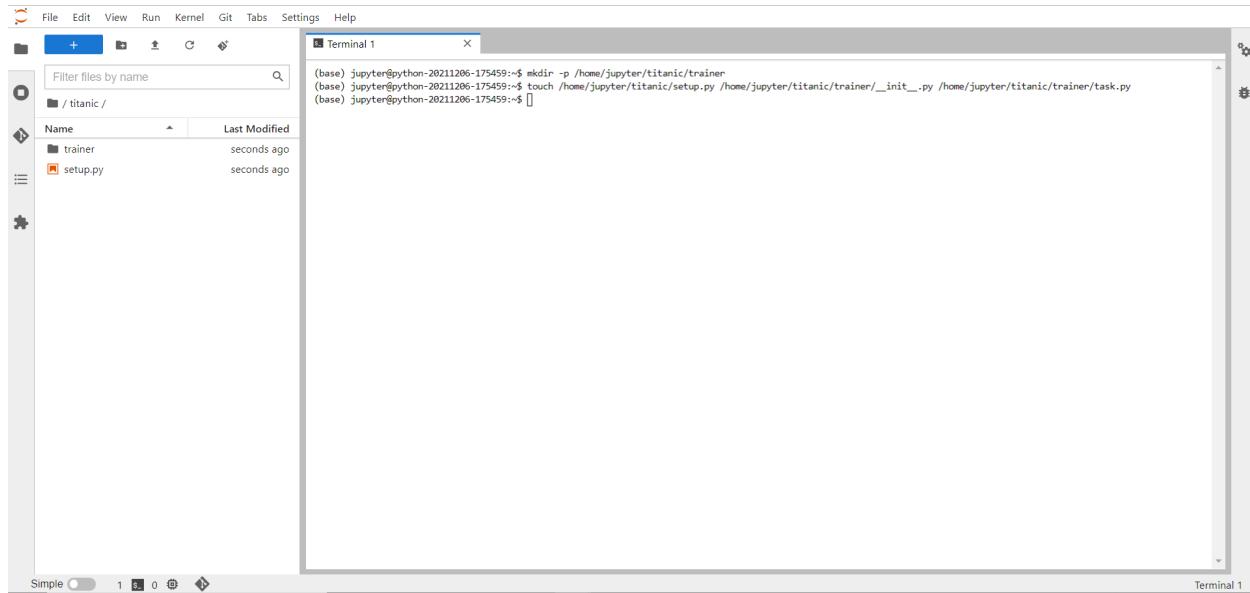
The screenshot shows the Google Cloud Platform Vertex AI interface. On the left, there's a sidebar with various options like Vertex AI, Dashboard, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. Below that is a Marketplace section. The main area is titled 'Notebooks' and has tabs for 'MANAGED NOTEBOOKS' (PREVIEW), 'USER-MANAGED NOTEBOOKS', 'EXECUTIONS' (PREVIEW), and 'SCHEDULES' (PREVIEW). A message at the top says: 'As of the M80 DLVM release, all environments will include JupyterLab 3.x by default. To continue using an existing environment's JupyterLab 1.x version, disable auto-upgrade (if enabled) and do not manually upgrade the environment to a new environment version. To create new Notebooks with JupyterLab 1.x installed, see creating specific versions of Notebooks.' Below this, it says 'Notebooks have JupyterLab pre-installed and are configured with GPU-enabled machine learning frameworks. Learn more'. There's a 'Filter' input field and a table listing notebooks. One row is selected: 'python-20211206-175459'. The 'OPEN JUPYTERLAB' button is highlighted in blue. At the bottom, the URL is https://lbg922dab56d5781-dot-us-central1.notebooks.googleapisusercontent.com/?authuser=0&username=Muskeeteers_SJSU

8. Select 'Terminal' in this window.



9. Run the following command to create the directory structure for our project.

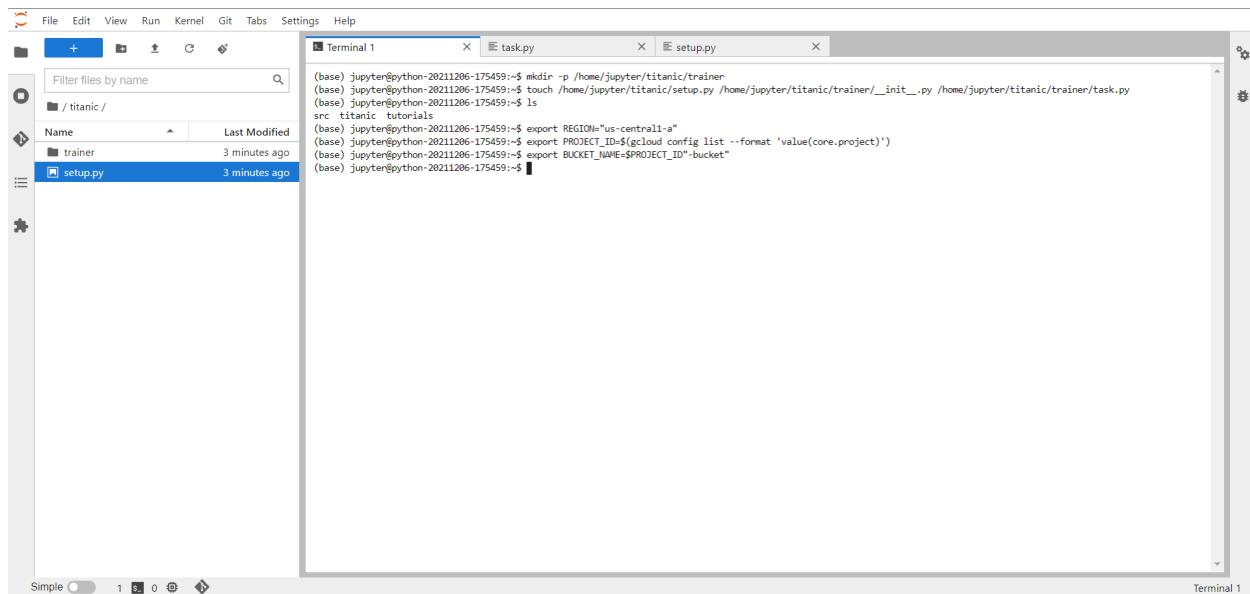
```
mkdir -p /home/jupyter/titanic/trainer  
touch /home/jupyter/titanic/setup.py /home/jupyter/titanic/trainer/__init__.py  
/home/jupyter/titanic/trainer/task.py
```



The screenshot shows the Jupyter Notebook interface. On the left is a file browser window titled 'titanic /' showing a directory structure with 'trainer' and 'setup.py'. On the right is a terminal window titled 'Terminal 1' showing the command history:

```
(base) jupyter@python-20211206-175459:~$ mkdir -p /home/jupyter/titanic/trainer  
(base) jupyter@python-20211206-175459:~$ touch /home/jupyter/titanic/setup.py /home/jupyter/titanic/trainer/__init__.py /home/jupyter/titanic/trainer/task.py  
(base) jupyter@python-20211206-175459:~$
```

10. Create the code for 'task.py' and 'setup.py', to train and run our model
11. Setup the environment variables, by running the below commands:
export REGION="us-central1"
export PROJECT_ID=\$(gcloud config list --format 'value(core.project)')
export BUCKET_NAME=\$PROJECT_ID"-bucket"

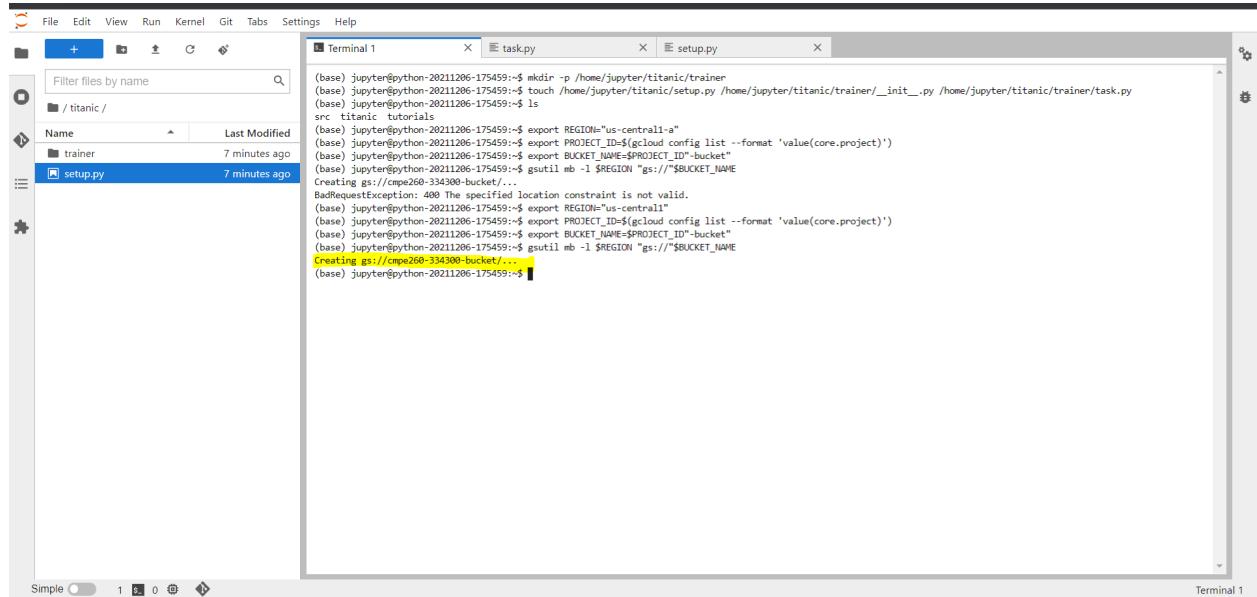


The screenshot shows the Jupyter Notebook interface. On the left is a file browser window titled 'titanic /' showing files 'trainer' and 'setup.py'. On the right is a terminal window titled 'Terminal 1' showing the command history after setting environment variables:

```
(base) jupyter@python-20211206-175459:~$ mkdir -p /home/jupyter/titanic/trainer  
(base) jupyter@python-20211206-175459:~$ touch /home/jupyter/titanic/setup.py /home/jupyter/titanic/trainer/__init__.py /home/jupyter/titanic/trainer/task.py  
(base) jupyter@python-20211206-175459:~$ ls  
src titanic tutorials  
(base) jupyter@python-20211206-175459:~$ export REGION="us-central1"  
(base) jupyter@python-20211206-175459:~$ export PROJECT_ID=$(gcloud config list --format 'value(core.project)')  
(base) jupyter@python-20211206-175459:~$ export BUCKET_NAME=$PROJECT_ID"-bucket"  
(base) jupyter@python-20211206-175459:~$
```

12. Create a bucket where you want to export your trained model

Run the command: gsutil mb -l \$REGION "gs://\$BUCKET_NAME"



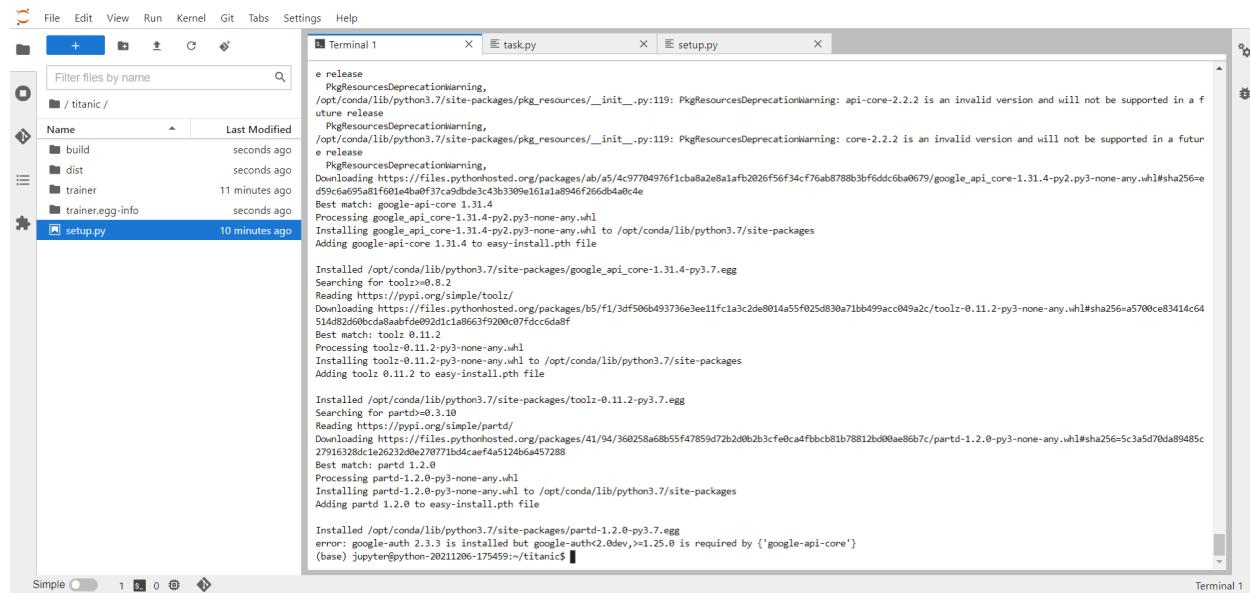
```
(base) jupyter@python-20211206-175459:~$ mkdir -p /home/jupyter/titanic/trainer
(base) jupyter@python-20211206-175459:~$ touch /home/jupyter/titanic/setup.py /home/jupyter/titanic/trainer/__init__.py /home/jupyter/titanic/trainer/task.py
(base) jupyter@python-20211206-175459:~$ ls
src titanic tutorials
(base) jupyter@python-20211206-175459:~$ export REGION="us-central1-a"
(base) jupyter@python-20211206-175459:~$ export PROJECT_ID=$(gcloud config list --format 'value(core.project)')
(base) jupyter@python-20211206-175459:~$ export BUCKET_NAME=$PROJECT_ID"-bucket"
(base) jupyter@python-20211206-175459:~$ gsutil mb -l $REGION "gs://$BUCKET_NAME"
(base) jupyter@python-20211206-175459:~$ gsutil mb -l $REGION "gs://cmpr208-334300-bucket..."
(base) jupyter@python-20211206-175459:~$
```

13. Install the required libraries

```
cd /home/jupyter/titanic
```

```
pip install setuptools
```

```
python setup.py install
```



```
e release
  PkgResourcesDeprecationWarning,
  /opt/conda/lib/python3.7/site-packages/pkg_resources/_init__.py:119: PkgResourcesDeprecationWarning: api-core-2.2.2 is an invalid version and will not be supported in a future release
  PkgResourcesDeprecationWarning,
  /opt/conda/lib/python3.7/site-packages/pkg_resources/_init__.py:119: PkgResourcesDeprecationWarning: core-2.2.2 is an invalid version and will not be supported in a future release
  PkgResourcesDeprecationWarning,
  Downloading https://files.pythonhosted.org/packages/ab/a5/c97704976f1cba8a2e1afb2026f56f34cf76ab8788b3bf6ddcc6ba0679/google_api_core-1.31.4-py3-none-any.whl#sha256=d59c6a958a81f601edbaef37c9d4de3d3b3309e161a1a8946f266db4a0c4e
Best match: google-api-core 1.31.4
Processing google_api_core-1.31.4-py3-py3-none-any.whl
Installing google_api_core-1.31.4-py3-py3-none-any.whl to /opt/conda/lib/python3.7/site-packages
Adding google-api-core 1.31.4 to easy-install.pth file

Installed /opt/conda/lib/python3.7/site-packages/google_api_core-1.31.4-py3.7.egg
Searching for toolz>=0.8.2
Reading https://pypi.org/simple/toolz/
Downloaded https://files.pythonhosted.org/packages/b5/f1/3df506b493736e3ee11fc1a3cde8014a55f025d83071bb499acc049a2c/toolz-0.11.2-py3-none-any.whl#sha256=a5700ce83414c64
514d82d60bdcda8adfded01c1a1a8663f9200c07fccc6dabf
Best match: toolz 0.11.2
Processing toolz-0.11.2-py3-none-any.whl
Installing toolz-0.11.2-py3-none-any.whl to /opt/conda/lib/python3.7/site-packages
Adding toolz 0.11.2 to easy-install.pth file

Installed /opt/conda/lib/python3.7/site-packages/toolz-0.11.2-py3.7.egg
Searching for partd>=0.3.10
Reading https://pypi.org/simple/partd/
Downloaded https://files.pythonhosted.org/packages/41/94/360258a68b55f47859d72b2d0b2b3cfe0ca4fbcb81b78812bd0ae86b7c/partd-1.2.0-py3-none-any.whl#sha256=5c3a5d70da89485c
27916328dc1e26232d0e27071bd4cae45a124b6a457288
Best match: partd 1.2.0
Processing partd-1.2.0-py3-none-any.whl
Installing partd-1.2.0-py3-none-any.whl to /opt/conda/lib/python3.7/site-packages
Adding partd 1.2.0 to easy-install.pth file

Installed /opt/conda/lib/python3.7/site-packages/partd-1.2.0-py3.7.egg
error: google-auth 2.3.3 is installed but google-auth<2.0dev,>=1.25.0 is required by {'google-api-core'}
(base) jupyter@python-20211206-175459:~/titanic$
```

14. Run the training code to verify that it executes without issues

```
python -m trainer.task -v \
--model_param_kernel=linear \
--model_dir="gs://$BUCKET_NAME/titanic/trial" \
--data_format=bigquery \
--training_data_uri="bq://$PROJECT_ID.titanic.survivors" \
--test_data_uri="bq://$PROJECT_ID.titanic.survivors" \
--validation_data_uri="bq://$PROJECT_ID.titanic.survivors"
```

The screenshot shows a Jupyter Notebook environment. On the left, there is a file browser pane displaying a directory structure under 'titanic/'. The 'setup.py' file is selected. On the right, there is a terminal window titled 'Terminal 1' showing the command-line output of the training code. The output indicates that the code is executing successfully, including the download of dependencies and the execution of the training script.

```
Searching for partd>=0.3.10
Reading https://pypi.org/simple/partd/
Downloaded https://files.pythonhosted.org/packages/41/94/360258a68b55f47859d72b2d0b2b3cf0ca4fbcb81b78812bd00ae86b7c/partd-1.2.0-py3-none-any.whl#sha256=5c3a5d70da89485c27916328d1e26232d0e270771bd4caeaf4a5124b6a457288
Best match: partd 1.2.0
Processing partd-1.2.0-py3-none-any.whl
Installing partd-1.2.0-py3-none-any.whl to /opt/conda/lib/python3.7/site-packages
Adding partd 1.2.0 to easy-install.pth file

Installed /opt/conda/lib/python3.7/site-packages/partd-1.2.0-py3.7.egg
error: google-auth 2.3.3 is installed but google-auth<2.0dev,>=1.25.0 is required by {'google-api-core'}
(base) jupyter@python-20211206-175459:~/titanic$ python -m trainer.task -v
> --model_param_kernel=linear
> --model_dir="gs://$BUCKET_NAME/titanic/trial"
> --data_format=bigquery
> --training_data_uri="bq://$PROJECT_ID.titanic.survivors"
> --test_data_uri="bq://$PROJECT_ID.titanic.survivors"
> --validation_data_uri="bq://$PROJECT_ID.titanic.survivors"
INFO:root:Model artifacts will be exported here: gs://cmpe260-334300-bucket/titanic/trial
INFO:root:Data format: bigquery
INFO:root:Training data uri: bq://cmpe260-334300.titanic.survivors
INFO:root:Validation data uri: bq://cmpe260-334300.titanic.survivors
INFO:root:Test data uri: bq://cmpe260-334300.titanic.survivors
INFO:root:Loading bigquery data
bq://cmpe260-334300.titanic.survivors
INFO:root:Reading bq data: bq://cmpe260-334300.titanic.survivors
INFO:root:Reading bq data: bq://cmpe260-334300.titanic.survivors
INFO:root:Reading bq data: bq://cmpe260-334300.titanic.survivors
INFO:root:Defining model parameters
/home/jupyter/titanic/trainer/task.py:104: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError. Select only valid columns before calling the reduction.
    df = df.fillna(df.mean())
INFO:root:Running feature selection
INFO:root:Training pipelines in CV
INFO:root:Export trained pipeline and report
INFO:root:f1score: 0.857268078833539
INFO:root:Training job completed. Exiting...
(base) jupyter@python-20211206-175459:~/titanic$
```

15. The two lines indicate the f1 score which is around 0.85 and the last line indicating that the training job completed successfully

16. Run the below commands to create the distributable file.

```
cd /home/jupyter/titanic
python setup.py sdist
```

17. Check for the tar.gz file created.

```

INFO:root:Running feature selection
INFO:root:Training pipelines in CV
INFO:root:Export training pipeline and report
INFO:root:fiscore: 0.857268635339
INFO:root:Training job completed: training...
(base) jupyter@python-20211206-175459:~/titanic$ cd /home/jupyter/titanic
(base) jupyter@python-20211206-175459:~/titanic$ python setup.py sdist
running sdist
running egg_info
writing trainer.egg-info/PKG-INFO
writing dependency_links to trainer.egg-info/dependency_links.txt
writing requirements to trainer.egg-info/requirements.txt
writing top-level names to trainer.egg-info/top_level.txt
reading manifest file 'trainer.egg-info/SOURCES.txt'
writing manifest file 'trainer.egg-info/SOURCES.txt'
warning: sdist: standard file not found: should have one of README, README.rst, README.txt, README.md
running check
warning: check: missing required meta-data: url
warning: check: missing meta-data: either (author and author_email) or (maintainer and maintainer_email) must be supplied
creating trainer-0.1
creating trainer-0.1/trainer
creating trainer-0.1/trainer.egg-info
copying files to trainer-0.1...
copying setup.py > trainer-0.1
copying trainer/_init_.py > trainer-0.1/trainer
copying trainer/task.py > trainer-0.1/trainer
copying trainer.egg-info/PKG-INFO > trainer-0.1/trainer.egg-info
copying trainer.egg-info/SOURCES.txt > trainer-0.1/trainer.egg-info
copying trainer.egg-info/dependency_links.txt > trainer-0.1/trainer.egg-info
copying trainer.egg-info/requirements.txt > trainer-0.1/trainer.egg-info
copying trainer.egg-info/top_level.txt > trainer-0.1/trainer.egg-info
writing trainer-0.1/setup.cfg
Creating tar archive
removing 'trainer-0.1' (and everything under it)
(base) jupyter@python-20211206-175459:~/titanic$ []

```

18. Copy the tar.gz file to GCS so that the training service can use it to train a new model

```
gsutil cp dist/trainer-0.1.tar.gz "gs://"BUCKET_NAME"/titanic/dist/trainer-0.1.tar.gz"
```

Bucket details for cmpe260-334300-bucket

Location	Storage class	Public access	Protection
us-central1 (Iowa)	Standard	Subject to object ACLs	None

OBJECTS

Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention expiration date
trainer-0.1.tar.gz	6.3 KB	application/x-tar	Dec 6, 20...	Standard	Dec 6, 202...	Not public	-	Google-managed key	-

19. Check if the dist file is copied into the bucket.

Model Training

In this, we will train a model on Vertex AI

1. From the Google Cloud console navigate to Vertex AI -> Training

The screenshot shows the Google Cloud Platform interface with the project 'CMPE260' selected. The left sidebar has a 'Vertex AI' section expanded, showing options like Dashboard, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. The main content area is titled 'Bucket details' for 'cmpe260-334300-bucket'. It shows a file named 'titanic.tar.gz' with details: Size: 6.3 KB, Type: application/x-tar, Created: Dec 6, 2020, Storage class: Standard, Last modified: Dec 6, 2020, Public access: Not public, Version history: -, Encryption: Google-managed key, and Retention expiration date: -. There are tabs for PROJECTS, CONFIGURATION, PERMISSIONS, PROTECTION, and LIFECYCLE.

2. Select the Region and select 'Create'

The screenshot shows the 'Training' page under the 'Vertex AI' section of the sidebar. The 'TRAINING PIPELINES' tab is selected. A 'CREATE' button is visible. A 'Region' dropdown is set to 'us-central1 (Iowa)'. The main content area displays a table for training pipelines, which currently shows 'No rows to display'.

Step 1: Training method

Select the DataSet as 'titanic', Objective as 'Classification'

The screenshot shows the 'Train new model' dialog in the Google Cloud Platform Vertex AI interface. The 'Training method' step is selected. The 'Dataset' dropdown is set to 'titanic'. The 'Objective' dropdown is set to 'Classification'. A note below the dropdowns says: 'Please refer to the pricing guide for more details (and available deployment options) for each method.' There are two radio button options: 'AutoML' (disabled) and 'Custom training (advanced)' (selected). The 'Custom training' section notes: 'Train high-quality models with minimal effort and machine learning expertise. Just specify how long you want to train.' Below these are 'START TRAINING' and 'CANCEL' buttons, and a 'CONTINUE' button at the bottom.

Step 2: Model details

Enter a Model name, and Data split to 'Random Assignment'

The screenshot shows the 'Train new model' dialog in the Google Cloud Platform Vertex AI interface. The 'Model details' step is selected. The 'Model name' field contains 'titanic_202112744834'. The 'Data split' section shows 'Random assignment' selected, with a note: '80% of your data is randomly assigned for training, 10% for validation and 10% for testing.' It includes a circular progress bar and a timeline diagram showing 'Training 80%', 'Validation 10%', and 'Testing 10%' proportions. Below this are sections for 'Encryption' (checkbox for CMEK) and 'Service account' (dropdown for selection). At the bottom are 'Service account' and 'BROWSE' buttons.

Step 3: Training container

Define your training environment

1. **Pre-built container:** Google cloud offers a set of prebuilt containers that make it easy to train your models. Those containers support frameworks such as Scikit-Learn, Tensorflow and XGBoost. Your model is based on scikit-learn and prebuilt container already exists.
2. Model framework: **Scikit-learn.** This is the library you used for model training.
3. Model framework version: Your code is compatible with **0.23.**
4. Package location: You can browse to the location of your training package. This is the bucket location where the training-0.1.tar.gz got uploaded.
5. Python Module: The python module you created in Notebooks. It will correspond to the folder that has your training code/module and the name of the entry file. This should be trainer.task
6. BigQuery project for exporting data:

The screenshot shows the 'Train new model' dialog in the Google Cloud Platform Vertex AI interface. The left sidebar shows various AI services like Vertex AI, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. The main area is titled 'Training' and shows a 'Training pipelines' section with a 'CREATE' button. The 'Train new model' dialog is open, showing the following steps:

- Training method:** Pre-built container (selected)
- Model details:** Model name is 'CMPE260'
- Training container:** Pre-built container selected
- Hyperparameters (optional):** Not specified
- Compute and pricing:** Not specified
- Prediction container (optional):** Not specified

Below these steps, there are sections for 'Model framework' (set to 'scikit-learn') and 'Model framework version' (set to '0.23').

Pre-built container settings:

- Package location (Cloud Storage path):** A dropdown menu shows the path: 'gs://cmpe260-334300-bucket/titanic/trainer/trainer-0.1.tar.gz' with a 'BROWSE' button.
- Python module:** 'trainer.task'
- BigQuery project for exporting data:** 'cmpe260-334300'
- Model output directory:** 'gs://cmpe260-334300-bucket/titanic/assets'

Step 4: Hyperparameter tuning

The screenshot shows the 'Train new model' wizard in the Vertex AI interface. The current step is 'Hyperparameters (optional)'. A note states: '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.' There is a checkbox for 'Enable hyperparameter tuning' which is unchecked. A 'CONTINUE' button is visible.

Train new model

- Training method
- Model details
- Training container
- 4 Hyperparameters (optional)
- 5 Compute and pricing
- 6 Prediction container (optional)

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

CONTINUE

START TRAINING CANCEL

Step 5: Compute and pricing

The screenshot shows the 'Train new model' wizard in the Vertex AI interface. The current step is 'Compute and pricing'. A note states: 'Model training pricing is based on the length of time spent training, machine types, and any accelerators used.' There is a dropdown for 'Region' set to 'us-central1 (Iowa)'. A 'Compute settings' section allows adding up to 4 worker pools. The first pool is configured with 'Machine type' as 'n1-standard-4, 4 vCPUs, 15 GiB memory', 'Worker count' as '1', 'Disk type' as 'SSD', and 'Disk size (GB)' as '100'. A 'CONTINUE' button is visible.

Train new model

- Training method
- Model details
- Training container
- Hyperparameters (optional)
- 5 Compute and pricing
- 6 Prediction container (optional)

Model training pricing is based on the length of time spent training, machine types, and any accelerators used. [Learn more](#)

Region
us-central1 (Iowa) ?

Compute settings

Select the type of virtual machine to use for your worker pool. You can add up to 4 worker pools. To learn about compute costs and how to map your ML framework's roles to specific worker pools, consult the [documentation](#).

Worker pool 0

Machine type *
n1-standard-4, 4 vCPUs, 15 GiB memory

Worker count
1

Disk type
SSD

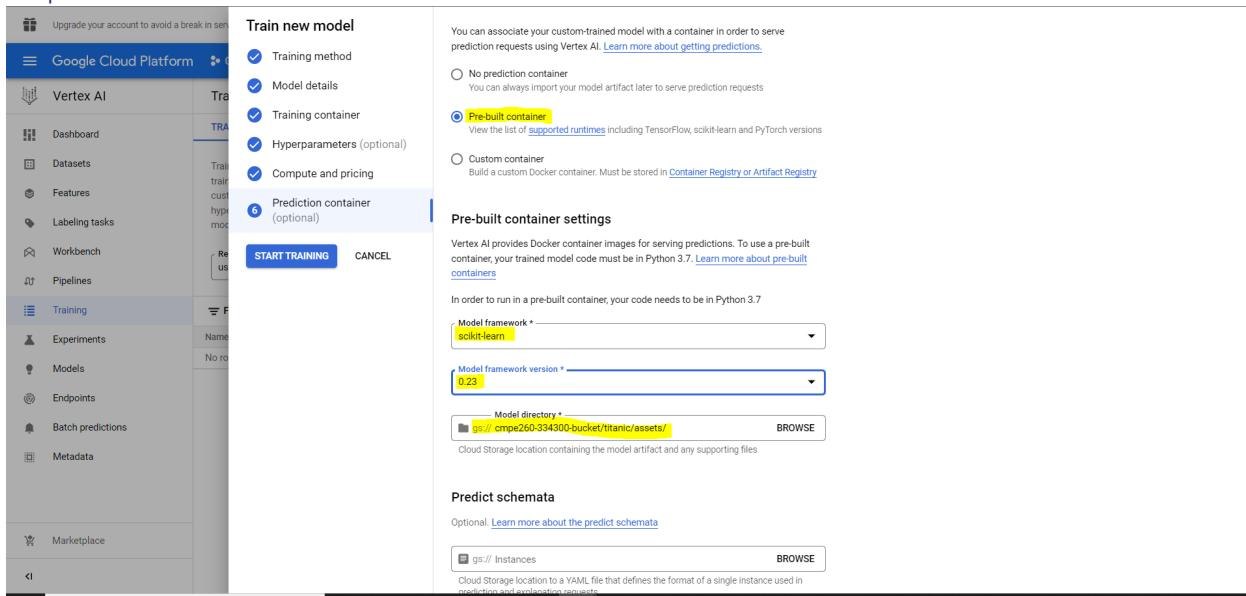
Disk size (GB)
100

ADD MORE WORKER POOLS (OPTIONAL)

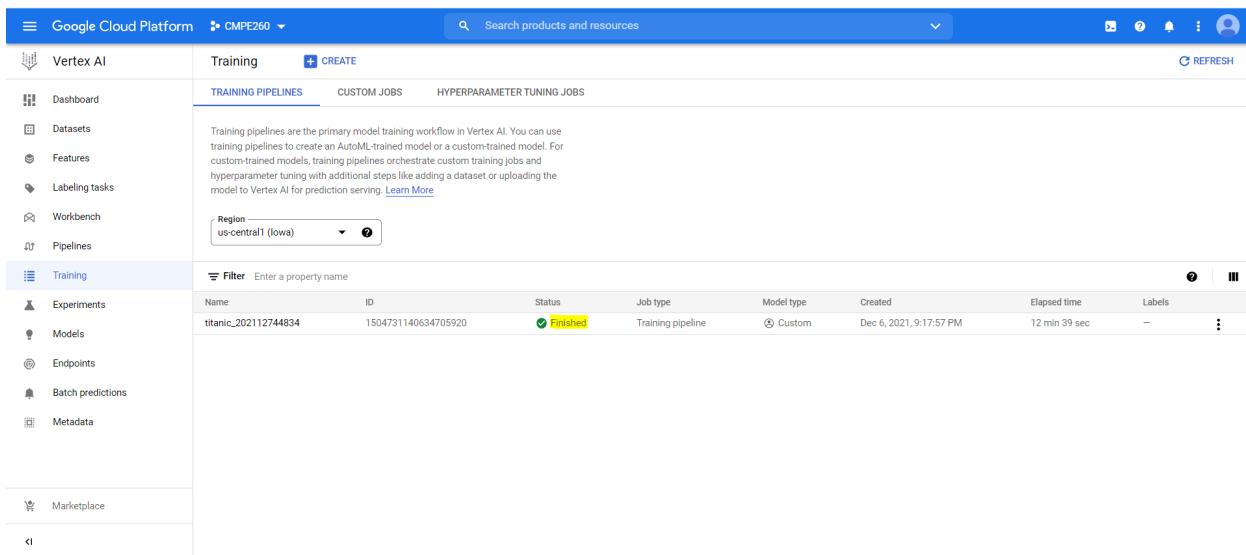
CONTINUE

START TRAINING CANCEL

Step 6: Prediction container



Click on 'Start Training'.



Check for the training to be 'Finished'

Model Evaluation

1. After the training job completion artifacts will be exported under gs://YOUR-BUCKET-NAME/training/assets
2. Check for the report.txt, under this folder.

The screenshot shows the 'Bucket details' page for 'cmpe260-334300-bucket'. The 'OBJECTS' tab is selected, displaying two files: 'model.pkl' and 'report.txt'. Both files are text/plain type, created on Dec 6, 2021, at 9:30:35. They are stored in the 'Standard' storage class and have 'Not public' public access. The 'report.txt' file is highlighted with a yellow box.

Name	Type	Created	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds
model.pkl	text/plain	Dec 6, 2021, 9:30:35...	Standard	Dec 6, 2021, 9:30:35...	Not public	Google-managed key	—	None
report.txt	text/plain	Dec 6, 2021, 9:30:35...	Standard	Dec 6, 2021, 9:30:35...	Not public	Google-managed key	—	None

Model Deployment

1. Last step is model deployment
2. Click on the trained model and DEPLOY TO ENDPOINT

The screenshot shows the 'Training' page in Vertex AI. The 'TRAINING PIPELINES' tab is selected, showing a single pipeline named 'Titanic_202112744834' which has completed successfully. The pipeline was created on Dec 6, 2021, at 9:17:57 PM, with an elapsed time of 12 min 39 sec. The status is 'Finished' and the job type is 'Training pipeline'. The pipeline is associated with a 'Custom' model type and is deployed to a 'us-central1 (Iowa)' region. The URL for the pipeline is visible at the bottom of the screen.

https://console.cloud.google.com/vertex-ai/locations/us-central1/models/4465534534816890880/deploy?taskTo=training&trainingPipelinesId=1504731140634705920&projectId=cmpe260-334300

The screenshot shows the Google Cloud Platform Vertex AI interface. On the left, there's a sidebar with options like Dashboard, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models (which is selected), Endpoints, Batch predictions, and Metadata. The main area has tabs for DEPLOY & TEST, BATCH PREDICTIONS, and MODEL PROPERTIES. Under DEPLOY & TEST, there's a 'Deploy your model' section with a sub-section 'Test your model'. A prominent blue button labeled 'DEPLOY TO ENDPOINT' is visible. Below it, a message says 'In order to test your model, you will need to deploy it first.' A 'Pricing guide' link is also present. At the bottom of the main area, there's a table header for 'Name', 'ID', 'Status', 'Models', 'Region', 'Monitoring', 'Most recent monitoring job', 'Most recent alerts', 'Last updated', 'API', 'Notification', 'Labels', and 'Encryption'.

- **Endpoint name:** titanic-endpoint Endpoint URL where the model is served.
- **Traffic split:** Defines the percentage of traffic that you want to direct to this model. An endpoint can have multiple models and you can despite how to split the traffic among them. In this case you are deploying a single model so the traffic has to be 100 percent.
- **Minimum number of compute nodes:** The minimum number of nodes required to serve model predictions. Start with 1. Additionally the prediction service will autoscale in case there is traffic
- **Maximum number of compute nodes:** In case of autoscaling, this variable defines the upper limit of nodes. It helps protecting against unwanted costs that autoscaling might result in. Set this variable to 2
- **Machine type:** Google cloud offers a set of machine types you can deploy your model to. Each machine has its own memory and vcpu specs. Your model is simple so serving on an n1-standard-4 instance will do the job

The screenshot shows the 'Deploy to endpoint' dialog box. It has three steps: 1. Define your endpoint (with 'Create new endpoint' selected and 'Endpoint name' set to 'titanic-endpoint'), 2. Model settings, and 3. Model monitoring. Step 1 is highlighted. Below the steps are 'DEPLOY' and 'CANCEL' buttons. To the right of the dialog, there are sections for 'Location' (set to 'us-central1 (Iowa)') and 'Access' (with 'Standard' selected). The 'Access' section includes a note about REST API availability and links for 'private services access' and 'Learn more'. At the bottom right of the dialog is a 'CONTINUE' button.

The screenshot shows the Google Cloud Platform Vertex AI interface. On the left is a sidebar with options like Dashboard, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models (which is selected), Endpoints, Batch predictions, and Metadata. The main area shows a model named 'titanic_202112744834'. Below it are tabs for DEPLOY & TEST, BATCH PREDICTIONS, and MODEL PROPERTIES. Under 'DEPLOY & TEST', there's a section titled 'Deploy your model' with a 'DEPLOY TO ENDPOINT' button. To the right, a modal window titled 'Deploy to endpoint' is open, showing three steps: 1. Define your endpoint (selected), 2. Model settings, 3. Model monitoring. It includes fields for 'Traffic split' (set to 100%), 'Compute resources' (minimum and maximum nodes set to 1), and 'Advanced scaling options' (machine type set to 'n1-standard-4, 4 vCPUs, 15 GiB memory').

3. Click CONTINUE and DEPLOY

The screenshot shows the Google Cloud Platform Vertex AI interface after deployment has started. The 'DEPLOY & TEST' tab is active, showing the 'Deploy your model' section. A table lists the deployed endpoint: Name: titanic-endpoint, ID: 432424729064767488, Status: Deploying model, Models: 0, Region: us-central1. The status bar at the bottom indicates the deployment is in progress.

The screenshot shows the Google Cloud Platform Vertex AI interface. On the left, there's a sidebar with various options like Dashboard, Datasets, Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, and Models. Under Models, 'Endpoints' is selected. The main area shows a deployed endpoint named 'titanic_endpoint' with ID 432424729064767488, status Active, 1 model, and region us-central1. It has monitoring disabled and no recent alerts or monitoring jobs. The last update was Dec 6, 2021, at 9:49:36 PM. The API is set to 'Sample request'. There are tabs for DEPLOY & TEST, BATCH PREDICTIONS, and MODEL PROPERTIES. Below the endpoint table, there's a 'Test your model' section with a JSON request editor and a response viewer.

Model Prediction

Under **Models** test the model prediction endpoint.

Try for different payloads:

Request:

```
{
  "instances": [
    ["male", 29.8811345124283, 26.0, 1, "S", "New York, NY", 0, 0],
    ["female", 48.0, 39.6, 1, "C", "London / Paris", 0, 1]
  ]
}
```

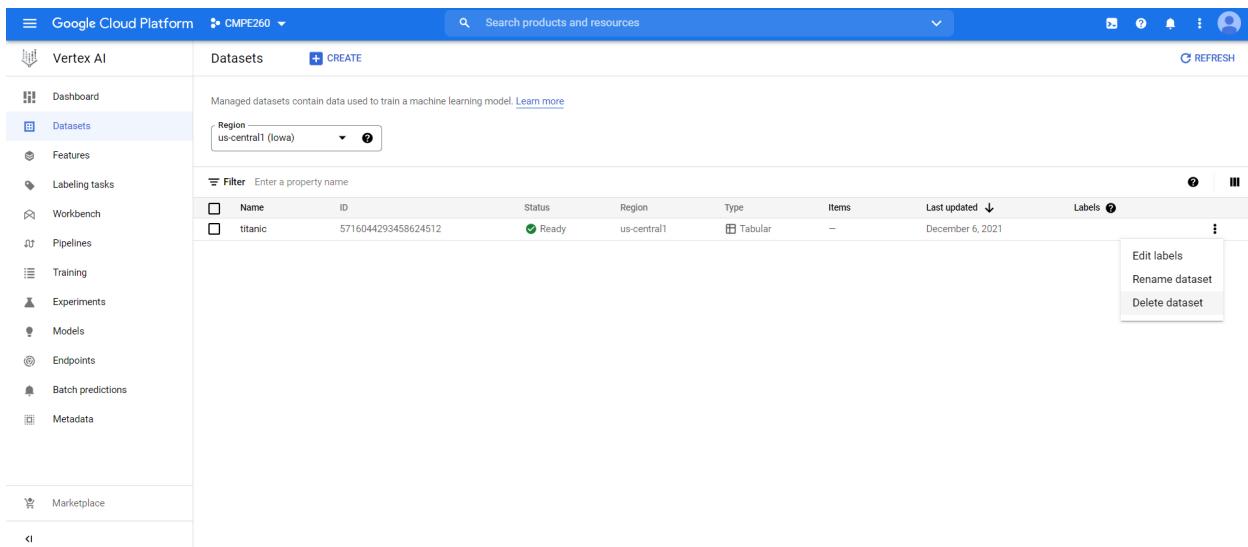
This screenshot is similar to the one above, showing the same deployed endpoint 'titanic_endpoint'. In the 'Test your model' section, the JSON request now contains the payload from the previous step. A blue 'PREDICT' button is visible below the request editor. To the right, the 'Response' panel shows the predicted results. The response JSON includes the 'predictions' array, the deployed model ID, the model location, and the model display name.

Response:

```
{
  "predictions": [
    0,
    1
  ],
  "deployedModelId": "5093685527765319680",
  "model": "projects/196373151126/locations/us-central1/models/4465534534816890880",
  "modelDisplayName": "titanic_202112744834"
}
```

Cleaning up

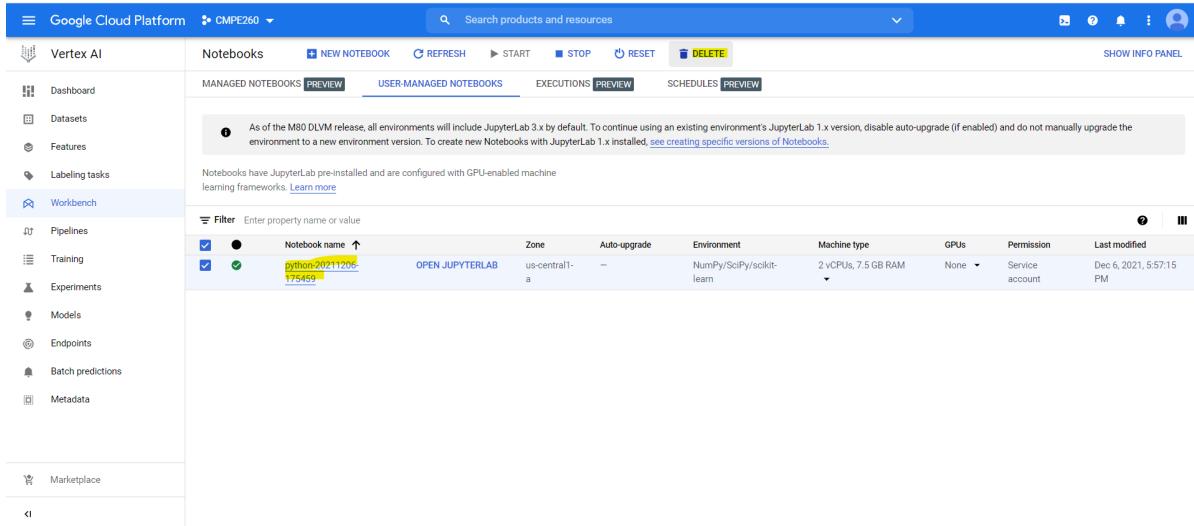
Delete ML Dataset



The screenshot shows the Google Cloud Platform Vertex AI Datasets page. The left sidebar includes options like Dashboard, Datasets (which is selected), Features, Labeling tasks, Workbench, Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. The main area displays a table of datasets. One row for 'titanic' is selected, showing details: Name (titanic), ID (5716044293458624512), Status (Ready), Region (us-central1), Type (Tabular), Items (—), Last updated (December 6, 2021). A context menu is open over this row, containing options: Edit labels, Rename dataset, and Delete dataset.

Name	ID	Status	Region	Type	Items	Last updated	Labels
titanic	5716044293458624512	Ready	us-central1	Tabular	—	December 6, 2021	

Delete Notebook



The screenshot shows the Google Cloud Platform Vertex AI Notebooks page. The left sidebar includes options like Dashboard, Datasets, Features, Labeling tasks, Workbench (which is selected), Pipelines, Training, Experiments, Models, Endpoints, Batch predictions, and Metadata. The main area displays a table of notebooks. One row for 'python30211206' is selected, showing details: Notebook name (python30211206), Zone (us-central1-a), Auto-upgrade (—), Environment (NumPy/SciPy/sklearn), Machine type (2 vCPUs, 7.5 GB RAM), GPUs (None), Permission (Service account), and Last modified (Dec 6, 2021, 5:57:15 PM). A context menu is open over this row, containing options: Edit notebook, Stop, Start, Delete, and Preview.

Notebook name	Zone	Auto-upgrade	Environment	Machine type	Gpus	Permission	Last modified
python30211206	us-central1-a	—	NumPy/SciPy/sklearn	2 vCPUs, 7.5 GB RAM	None	Service account	Dec 6, 2021, 5:57:15 PM

Delete Endpoint

Google Cloud Platform CMPE260

Search products and resources

titanic_202112744834

DEPLOY & TEST BATCH PREDICTIONS MODEL PROPERTIES

Deploy your model

Endpoints are machine learning models made available for online prediction requests. Endpoints are useful for timely predictions from many users (for example, in response to an application request). You can also request batch predictions if you don't need immediate results.

DEPLOY TO ENDPOINT

Name	ID	Status	Models	Region	Monitoring	Most recent monitoring job	Most recent alerts	Last updated	API	Notification	Labels	Encryption
titanic_endpoint	432424729054767488	Active	1	us-central1	Disabled	—	—	Dec 6, 2021, 9:49:36 PM	Sample request			Google-managed key

Test your model PREVIEW

Your JSON request must contain an `instances` field and an optional `parameters` field if you're using a custom container. No other fields can be present in the JSON request. [Learn how to format your JSON request.](#)

JSON request

```
{"instances": [ { "sample_key": "sample_value" } ]}
```

Response

View logs
Edit labels
Undeploy model
Rename endpoint
Remove endpoint

Delete Model

Google Cloud Platform CMPE260

Search products and resources

REFRESH

Models CREATE IMPORT

Models are built from your datasets or unmanaged data sources. There are many different types of machine learning models available on Vertex AI, depending on your use case and level of experience with machine learning. [Learn more](#)

Region us-central1 (Iowa)

Filter Enter a property name

Name	ID	Status	Data	Endpoints	Region	Type	Created	Notifications	Labels
titanic_202112744834	4465534534816890880	Ready	—	0	us-central1	Custom trained Custom training	Dec 6, 2021, 9:17:57 PM		

Add to endpoint
Edit labels
Rename model
Delete model

Delete Objects, GCP Bucket

Google Cloud Platform CMPE260																																							
Search products and resources																																							
Browser CREATE BUCKET DELETE REFRESH SHOW INFO PANEL																																							
Filter Filter buckets																																							
<table border="1"> <thead> <tr> <th>Name</th> <th>Created</th> <th>Location type</th> <th>Location</th> <th>Default storage class</th> <th>Last modified</th> <th>Public access</th> <th>Access control</th> <th>Protection</th> <th>Life</th> </tr> </thead> <tbody> <tr> <td>cloud-e-platform-65473780-0280-46f...</td> <td>Dec 6, 2021, 4:17:06 PM</td> <td>Region</td> <td>us-central1 (Io...</td> <td>Regional</td> <td>Dec 6, 2021, 4:17:06 PM</td> <td>Subject to object ACLs</td> <td>Fine-grained</td> <td>None</td> <td>None</td> </tr> <tr> <td>cmpe260-334300-project</td> <td>Dec 6, 2021, 6:17:52 PM</td> <td>Region</td> <td>us-central1 (Io...</td> <td>Standard</td> <td>Dec 6, 2021, 6:17:52 PM</td> <td>Subject to object ACLs</td> <td>Fine-grained</td> <td>None</td> <td>None</td> </tr> </tbody> </table>										Name	Created	Location type	Location	Default storage class	Last modified	Public access	Access control	Protection	Life	cloud-e-platform-65473780-0280-46f...	Dec 6, 2021, 4:17:06 PM	Region	us-central1 (Io...	Regional	Dec 6, 2021, 4:17:06 PM	Subject to object ACLs	Fine-grained	None	None	cmpe260-334300-project	Dec 6, 2021, 6:17:52 PM	Region	us-central1 (Io...	Standard	Dec 6, 2021, 6:17:52 PM	Subject to object ACLs	Fine-grained	None	None
Name	Created	Location type	Location	Default storage class	Last modified	Public access	Access control	Protection	Life																														
cloud-e-platform-65473780-0280-46f...	Dec 6, 2021, 4:17:06 PM	Region	us-central1 (Io...	Regional	Dec 6, 2021, 4:17:06 PM	Subject to object ACLs	Fine-grained	None	None																														
cmpe260-334300-project	Dec 6, 2021, 6:17:52 PM	Region	us-central1 (Io...	Standard	Dec 6, 2021, 6:17:52 PM	Subject to object ACLs	Fine-grained	None	None																														

Delete BigQuery dataset

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar displays a project named "cmpe260-334300" which contains two datasets: "dataset_57160442934586245..." and "titanic". A context menu is open over the "titanic" dataset, with the "Delete" option highlighted in yellow. The main editor area is empty, showing a single row number "1". At the bottom, there are tabs for "PERSONAL HISTORY", "PROJECT HISTORY", and "SAVED QUERIES".