

Name - Mayuri Lalwani

## Assignment 7 - VERTEX AI

a)

<https://codelabs.developers.google.com/vertex-pipelines-intro#0>

The screenshot shows two views of the Google Cloud Platform Vertex AI interface. The top view is the 'Dashboard' page, which displays a message: 'Page not viewable for organizations. To view this page, select a project.' Below this is a 'Select a recent project' section showing a 'Hello World' project. The right side of the dashboard includes a user account sidebar with details for 'Mayuri Lalwani' and links for 'Google Account', 'Add account', and 'Sign out'. The bottom view is the 'Notebooks' page, where a 'New notebook' dialog is open. The dialog fields include 'Notebook name' (set to 'tensorflow-2:3-20211203-114247'), 'Region' (set to 'us-west1 (Oregon)'), 'Zone' (set to 'us-west1-b'), and 'Environment' (set to 'TensorFlow Enterprise 2.3 (with LTS and Intel® MKL)'). Other settings like 'Machine type', 'Boot disk', 'Data disk', 'Subnetwork', 'External IP', and 'Permission' are also visible. At the bottom of the dialog are 'ADVANCED OPTIONS', 'CANCEL', and 'CREATE' buttons.

← → ⌂ https://console.cloud.google.com/vertex-ai/workbench/list/instances?authuser=2&project=vertex-ai-334019

Google Cloud Platform Vertex AI Search products and resources

Vertex AI Notebooks NEW NOTEBOOK REFRESH START STOP RESET DELETE HIDE INFO PANEL

Dashboard MANAGED NOTEBOOKS PREVIEW USER-MANAGED NOTEBOOKS EXECUTIONS PREVIEW SCHEDULES PREVIEW

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.

Notebooks have JupyterLab pre-installed and are configured with GPU-enabled machine learning frameworks. Learn more

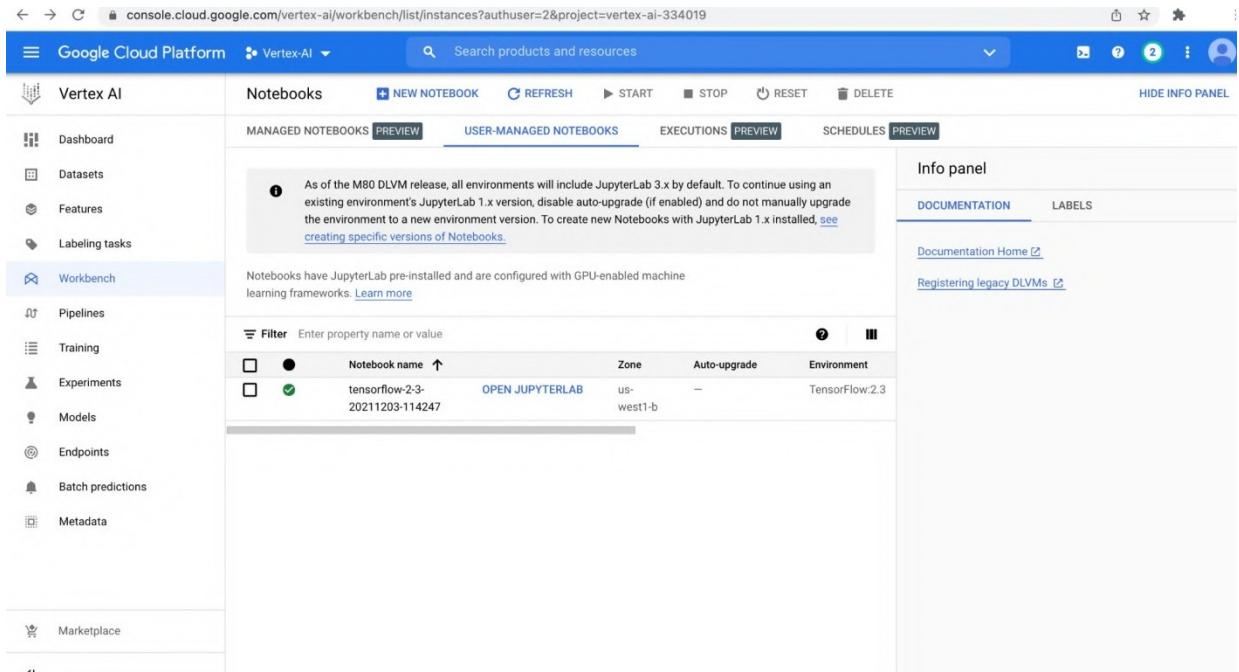
Filter Enter property name or value

	Notebook name ↑	Zone	Auto-upgrade	Environment
<input type="checkbox"/>	tensorflwo-2-3-20211203-114247	OPEN JUPYTERLAB	us-west1-b	TensorFlow:2.3

Info panel DOCUMENTATION LABELS

Documentation Home Registering legacy DLVMS

Marketplace



← → ⌂ https://757e980bcd26c11e-dot-us-west1.notebooks.googleusercontent.com/lab

File Edit View Run Kernel Git Tabs Settings Help

Launcher

Filter files by name

Name / Last Modified

src 5 minutes ago  
tutorials 5 minutes ago

Notebook

Python 3 Python [conda env:root] \*

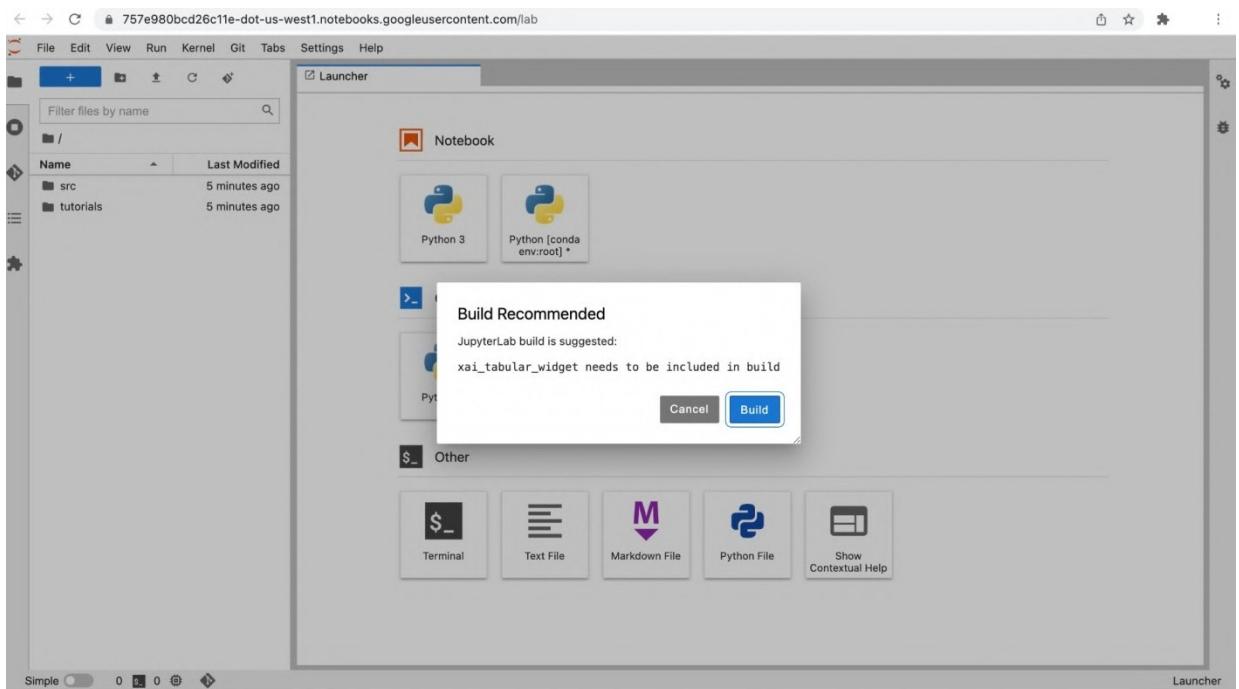
Build Recommended

JupyterLab build is suggested:  
xai\_tabular\_widget needs to be included in build

Cancel Build

Other

Terminal Text File Markdown File Python File Show Contextual Help



b)

<https://codelabs.developers.google.com/vertex-automl-tabular#0>

← → 🔒 console.cloud.google.com/vertex-ai/locations/us-central1/datasets/9199015655276281856/import?authuser=2&project=vertex-ai-334019

Google Cloud Platform Vertex-AI Search products and resources

Vertex AI fraud\_detection

Dashboard Datasets SOURCE ANALYZE

Add data to your dataset

Before you begin, read the [data guide](#) to learn how to prepare your data. Then choose a data source.

Select a data source

- CSV file: Can be uploaded from your computer or on Cloud Storage. [Learn more](#)
- BigQuery: Select a table or view from BigQuery. [Learn more](#)

Upload CSV files from your computer

Select CSV files from Cloud Storage

Select a table or view from BigQuery

Upload CSV files from your computer

Add up to 500 CSV files per upload. The files will be stored in a new Cloud Storage bucket ([charges apply](#)). Data from multiple files will be referenced as one dataset.

You can build two model types with tabular data. The model type is automatically chosen based on the data type of your target column.

- Regression models predict a numeric value. For example, predicting home prices or consumer spending.
- Classification models predict a category from a fixed number of categories. Examples include predicting whether an email is spam or not, or classes a student might be interested in attending.

Marketplace

€• Cg console.cloud.google.com/vertex-ai/locations/us-central1/datasets/919g0156fi5276281856/import?authuser=2&project=vertex-ai-334019

The screenshot shows the Google Cloud Platform Vertex AI interface. The top navigation bar includes the project name "Vertex AI" and a search bar. On the left, a sidebar lists "Dashboard", "Datasets" (which is selected), "Features", "Workbench", "Training", "Experiments", "Models", "Endpoints", "Batch predictions", and "Metadata". The main content area is titled "fraud\_detection" and shows the "fraud\_detection" dataset. It displays two icons: a house-like icon and a blue triangle icon. Below the icons, there are two sections: "Select a data source" (with options for CSV file upload or Cloud Storage selection) and "Add data to your dataset" (with a note about preparing data and choosing a data source). A summary table shows metrics: \$625,000 and \$975,000. At the bottom, there are sections for "Endpoints" and "Batch predictions".

€• —\* Ca console.cloud.google.com/vertex-ai/locations/us-central1/datasets/9199o15655276281856/import?authuser=2&project=vertex-ai-334019

This screenshot is similar to the one above but shows a different state. The "Datasets" section is still selected in the sidebar. In the main content area, the "Select a table or view from BigQuery" section is highlighted. A dropdown menu shows "BigQuery path" with the value "bigquery-public-data.ml\_datasets.ulb\_fraud\_detection" selected. A "BROWSE" button is visible next to the dropdown. A note at the bottom states: "referenced BigQuery table will affect the dataset before training." The rest of the interface, including the sidebar and other sections like "Add data to your dataset", remains the same.

S 0 ,0,0w.loud.google.<om&ven x-a,ooawons/«s- cn«ançda ase<s/e,sooisossi7oae era a,a;z.º»nuse =e&p qe i=vene -a-3a4o,0

Vertex AI fraud\_detection

SOURCE ANALYZE

Features Dataset Info Summary Training jobs and models

Created Oct 03, 2021 11 h All Dataset format: BigQuery

GENERATE STATISTICS TRAIN NEW MODEL

Batch predictions

Column name ↑	BigQuery type	BigQuery mode	Missing a f<<t> e	Distinct values @
Amount	FLOAT	NULLABLE		
Class	INTEGER	NULLABLE		
Time	FLOAT	NULLABLE		
V1	FLOAT	NULLABLE		
V10	FLOAT	NULLABLE		
V11	FLOAT	NULLABLE		

### Train new model

fraud\_detection\_202112319 s457



#### ④ Compute and pricing

Export test dataset to BigQuery

[W ADVANCED OPTIONS](#)

START TRAINING

CONTINUE



### Train new model

Enter the maximum number of node hours you wan1 to spend training your model.

You can train for as little as node hour. You may also be eligible to train with free node

Budget \*

1

Maximum node hours



Estimated completion date: Dec 3, 2021 1 PM GMT-8

#### Enable early stopping

Ends model training when no more improvements can be made and refunds leftover training budget. If early stopping is disabled, training continues until the budget is

Workbench	Y
Pipelines	Y
Training	Y
Experiments	

0Af1Mb



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, Models (which is selected), Endpoints, Batch predictions, and Metadata. The main area has tabs for EVALUATE, DEPLOY & TEST (which is active), BATCH PREDICTIONS, and MODEL PROPERTIES. Under DEPLOY & TEST, it says 'Model to run on a Docker container.' Below that is a 'Deploy your model' section with a table showing one endpoint named 'endpointfraud' with ID 141731446866837504, currently 'Deploying model'. The table includes columns for Name, ID, Status, Models, Region, Monitoring, Most recent monitoring job, Most recent alerts, and Last updated. The last update was on Dec 3, 2021, at 2:34:54 PM. Below the table is a 'Test your model' section with a preview message: 'Your model must be successfully deployed to an endpoint before you can test it.' It shows a feature column named 'Time' with type 'Numerical' and value '84883'. To the right, it says 'Predicted column not yet known' and 'Prediction result'.

c)

[https://codelabs.developers.google.com/vertex\\_custom\\_training\\_prediction#0](https://codelabs.developers.google.com/vertex_custom_training_prediction#0)

(Links to an external site.)

The screenshot shows a Jupyter Notebook interface. At the top, there's a menu bar with Edit, View, Run, Kernel, Git, Tabs, Settings, Help, and a toolbar with icons for file operations. The title bar says 'Perset 1'. Below the menu is a search bar and a notebook list with 'QM Notebook' selected. The notebook view shows a table of contents with 'Name' and 'Last Modified' columns. A toolbar below the notebook includes Python 3, Python [conda], Sync is on, and a Console tab. At the bottom, there's a file browser with tabs for Terminal, Text File, Markdown File, Python File, and a Show button. The terminal tab is active, showing a command line with a prompt '\$ \_' and some code. The file browser tab is also active, showing a list of files and folders.

Name - Last Modified COPY trainer /trainer

```
# Sets up the entry point to invoke the trainer
ENTRYPOINT ["python", "-m", "trainer.train"]
```

```

File Edit View Run Kernel Git Tabs Settings Help
Filter files by name
Name Last Modified
train.py 3 minutes ago
Terminal 1 train.py
1 import numpy as np
2 import pandas as pd
3 import pathlib
4 import tensorflow as tf
5
6 from tensorflow import keras
7 from tensorflow.keras import layers
8
9 print(tf.__version__)
10
11 ##### The Auto MPG dataset
12
13 The dataset is available from the [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/).
14
15 ### Get the data
16 First download the dataset.
17
18
19 dataset_path = keras.utils.get_file("auto-mpg.data", "http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data")
20 dataset_path
21
22 ##### Import it using pandas#####
23
24 column_names = ['MPG','Cylinders','Displacement','Horsepower','Weight',
25 'Acceleration', 'Model Year', 'Origin']
26 dataset = pd.read_csv(dataset_path, names=column_names,
27 na_values = "?", comment='t',
28 sep=",", skipinitialspace=True)
29
30 dataset.tail()
31
32 # TODO: replace 'your-gcs-bucket' with the name of the Storage bucket you created earlier
33 BUCKET = 'gs://ai-334019-bucket'
34
35 ##### Clean the data
36
37 The dataset contains a few unknown values.
38

```

```

File Edit View Run Kernel Git Tabs Settings Help
Filter files by name
Name Last Modified
train.py 7 minutes ago
Terminal 1
(base) jupyter@tensorflow-2-3-20211203-114247:~$ mkdir mpg
(base) jupyter@tensorflow-2-3-20211203-114247:~$ cd mpg
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ touch Dockerfile
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ gcloud config list --format 'value(core.project)'
vertex-ai-334019
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ PROJECT_ID=vertex-ai-334019
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ BUCKET_NAME="gs://${PROJECT_ID}-bucket"
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ BUCKET_NAME="gs://${vertex-ai-334019}-bucket"
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ gutil mb -l us-central1 ${BUCKET_NAME}
Creating gs://ai-334019-bucket/...
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ gcloud config set component_manager/disable_update_check True
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ gcloud auth activate-service-account vertex-ai-334019@mpg:v1
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ gcloud docker build . --imageURI invalid argument "gcr.io/ai-334019/mpg:v1" for "-t, --tag" flag: invalid reference format
See 'docker build --help'.
(base) jupyter@tensorflow-2-3-20211203-114247:~/mpg$ IMAGE_URI="gcr.io/$vertex-ai-334019/mpg:v1"

```

d) Big query :

<https://codelabs.developers.google.com/vertex-workbench-intro#0>

← → C https://console.cloud.google.com/vertex-ai/workbench/list/instances?authuser=4&project=vertex-ai-334021

Google Cloud Platform Vertex-AI Search products and resources

Vertex AI Notebooks + NEW NOTEBOOK ⏪ REFRESH ▶ START ■ STOP ⏪ RESET ⏪ DELETE HIDE INFO PANEL

Dashboard Datasets Features Labeling tasks Workbench Pipelines Training Experiments Models Endpoints Batch predictions Metadata Marketplace

MANAGED NOTEBOOKS PREVIEW USER-MANAGED NOTEBOOKS EXECUTIONS PREVIEW SCHEDULES PREVIEW

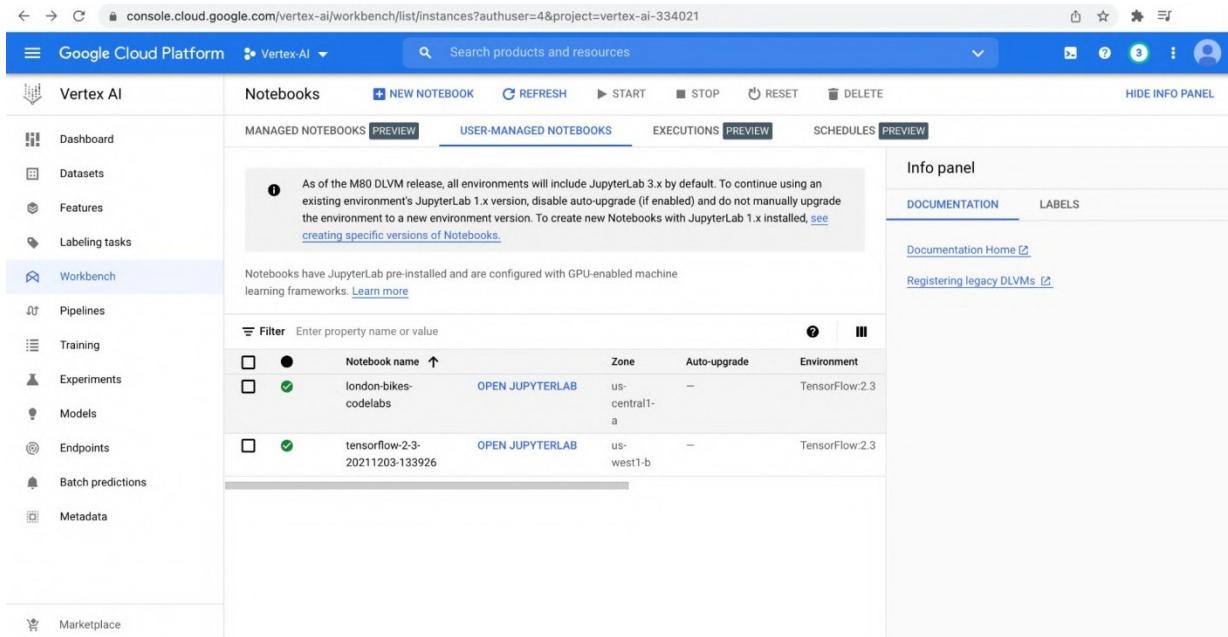
As of the M80 DLVM release, all environments will include JupyterLab 3.x by default. To continue using an existing environment's JupyterLab 1.x version, disable auto-upgrade (if enabled) and do not manually upgrade the environment to a new environment version. To create new Notebooks with JupyterLab 1.x installed, see creating specific versions of Notebooks.

Notebooks have JupyterLab pre-installed and are configured with GPU-enabled machine learning frameworks. Learn more

Filter Enter property name or value

	Notebook name ↑	Zone	Auto-upgrade	Environment
<input type="checkbox"/>	london-bikes-codelabs	OPEN JUPYTERLAB	us-central1-a	TensorFlow:2.3
<input type="checkbox"/>	tensorflow-2-3-20211203-133926	OPEN JUPYTERLAB	us-west1-b	TensorFlow:2.3

Info panel DOCUMENTATION LABELS Documentation Home Registering legacy DLVMs



← → C https://console.cloud.google.com/vertex-ai/workbench/create-managed?authuser=4&project=vertex-ai-334021

Google Cloud Platform Vertex-AI Search products and resources

Vertex AI Create a managed notebook

Dashboard Datasets Features Labeling tasks Workbench Pipelines Training Experiments Models Endpoints Batch predictions Metadata Marketplace

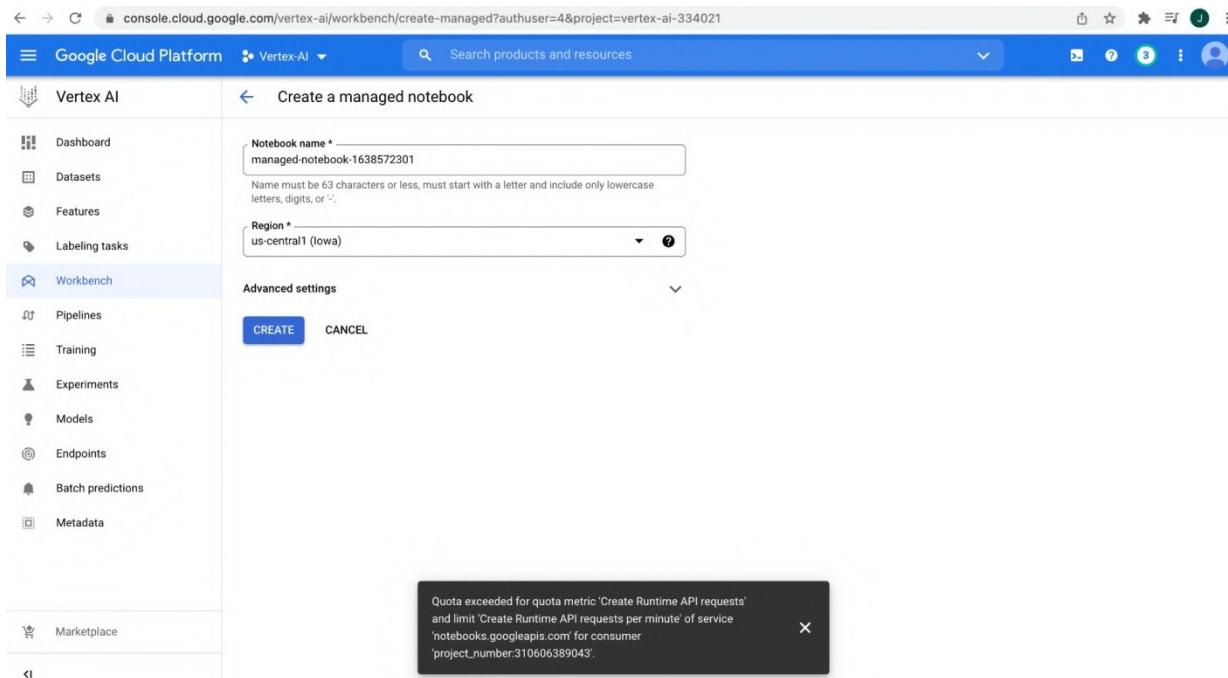
Notebook name \* managed-notebook-1638572301 Name must be 63 characters or less, must start with a letter and include only lowercase letters, digits, or '-'.

Region \* us-central1 (Iowa)

Advanced settings

CREATE CANCEL

Quota exceeded for quota metric 'Create Runtime API requests' and limit 'Create Runtime API requests per minute' of service 'notebooks.googleapis.com' for consumer 'project\_number:310606389043'. X



f) [https://codelabs.developers.google.com/vertex\\_hyperparameter\\_tuning#0](https://codelabs.developers.google.com/vertex_hyperparameter_tuning#0)

File Edit View Run Kernel Git Tabs Settings Help

Terminal 2 task.py Dockerfile

```

1 FROM gcr.io/deeplearning-platform-release/tf2-gpu.2-5
2
3 WORKDIR /
4
5 # Installs hypertune library
6 RUN pip install cloudml-hypertune
7
8 # Copies the trainer code to the docker image.
9 COPY trainer /trainer
10
11 # Sets up the entry point to invoke the trainer.
12 ENTRYPOINT ["python", "-m", "trainer.task"]

```

Simple 2 0 Dockerfile

File Edit View Run Kernel Git Tabs Settings Help

Terminal 2 task.py Dockerfile

```

1 import tensorflow as tf
2 import tensorflow_datasets as tfds
3 import argparse
4 import hypertune
5
6 NUM_EPOCHS = 10
7
8 def get_args():
9     '''parse args. Must include all hyperparameters you want to tune.'''
10    parser = argparse.ArgumentParser()
11    parser.add_argument(
12        '--learning_rate',
13        required=True,
14        type=float,
15        help='learning rate')
16    parser.add_argument(
17        '--momentum',
18        required=True,
19        type=float,
20        help='SGD momentum value')
21    parser.add_argument(
22        '--num_neurons',
23        required=True,
24        type=int,
25        help='number of units in last hidden layer')
26    args = parser.parse_args()
27    return args
28
29
30
31 def preprocess_data(image, label):
32     '''Resizes and scales images.'''
33
34     image = tf.image.resize(image, (150,150))
35     return tf.cast(image, tf.float32) / 255., label
36
37
38
39 def create_dataset():

```

The screenshot shows a Jupyter Notebook environment. On the left, a file browser displays a directory structure with files: Dockerfile, trainer, and task.py. The Dockerfile and task.py files are selected. The right side contains two terminal windows. The top terminal window, titled 'Terminal 2', shows the content of the task.py file:

```

59     inputs = tf.keras.Input(shape=(150, 150, 3))
60     x = tf.keras.layers.Conv2D(16, (3, 3), activation='relu')(inputs)
61     x = tf.keras.layers.MaxPooling2D((2, 2))(x)
62     x = tf.keras.layers.Conv2D(32, (3, 3), activation='relu')(x)
63     x = tf.keras.layers.MaxPooling2D((2, 2))(x)
64     x = tf.keras.layers.Conv2D(64, (3, 3), activation='relu')(x)
65     x = tf.keras.layers.MaxPooling2D((2, 2))(x)
66     x = tf.keras.layers.Flatten()(x)
67     x = tf.keras.layers.Dense(num_neurons, activation='relu')(x)
68     output = tf.keras.layers.Dense(1, activation='sigmoid')(x)
69     model = tf.keras.Model(inputs, outputs)
70     model.compile(
71         loss='binary_crossentropy',
72         optimizer=tf.keras.optimizers.SGD(learning_rate=learning_rate, momentum=momentum),
73         metrics=['accuracy'])
74     return model
75
76
77 def main():
78     args = get_args()
79     train_data, validation_data = create_dataset()
80     model = create_model(args.num_neurons, args.learning_rate, args.momentum)
81     history = model.fit(train_data, epochs=NUM_EPOCHS, validation_data=validation_data)
82
83 # DEFINE METRIC
84 hp_metric = history.history['val_accuracy'][-1]
85
86 hpt = hypertune.HyperTune()
87 hpt.report_hyperparameter_tuning_metric(
88     hyperparameter_metric_tag='accuracy',
89     metric_value=hp_metric,
90     global_step=NUM_EPOCHS)
91
92
93 if __name__ == "__main__":
94     main()

```

The bottom terminal window, also titled 'Terminal 2', shows the command-line output of running the Dockerfile:

```

(base) jupyter@tensorflow-2-3-20211203-133926:~$ mkdir horses_or_humans
(base) jupyter@tensorflow-2-3-20211203-133926:~$ cd horses_or_humans
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ touch Dockerfile
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ mkdir trainer
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ touch trainer/task.py
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ PROJECT_ID='vertex-ai-334021'
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ IMAGE_URI="gcr.io/vertex-ai-334021/horse-human:hypertune"
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ docker build ./ -t $IMAGE_URI
invalid argument "gcr.io/ai-334021/horse-human:hypertune" for "-t, --tag" flag: invalid reference format
See 'docker build --help'.
(base) jupyter@tensorflow-2-3-20211203-133926:~/horses_or_humans$ docker push $IMAGE_URI

```

This screenshot shows the same Jupyter Notebook environment as the previous one, but the Dockerfile has been run. The terminal output now includes the error message from the docker build command, indicating that the image name is invalid due to a missing colon in the tag.

g) <https://codelabs.developers.google.com/vertex-mlmd-pipelines#0>

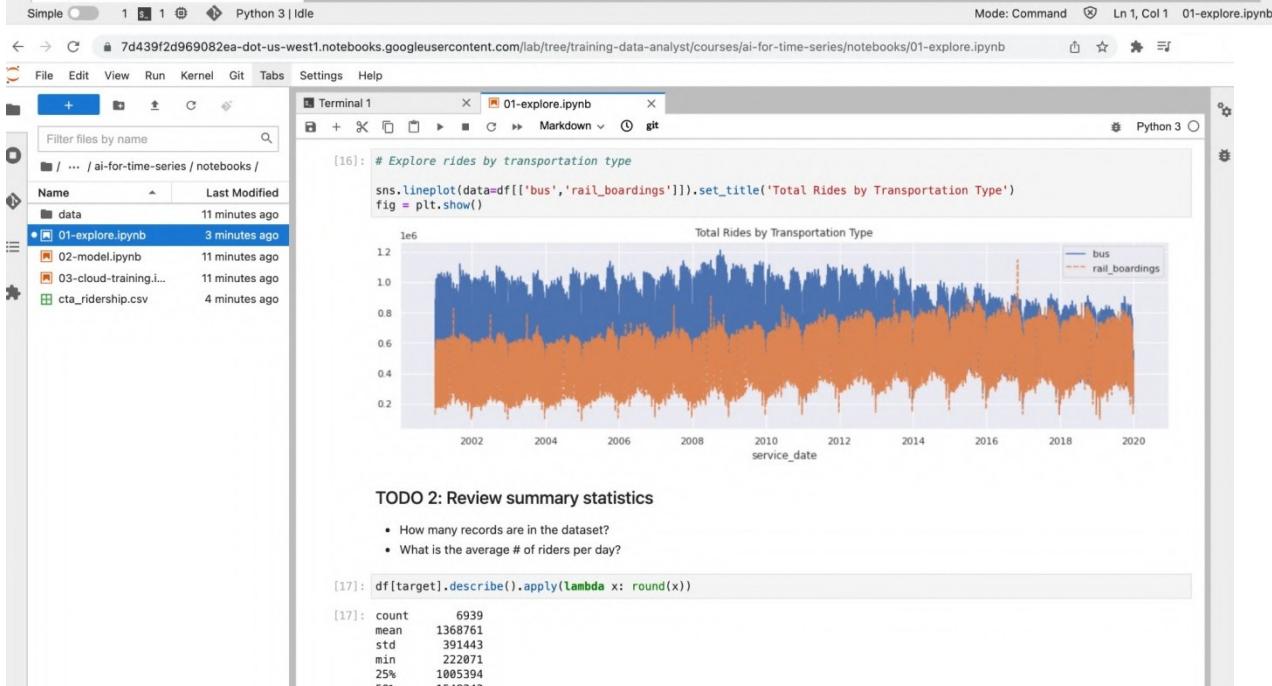
← → 🔒 console.cloud.google.com/vertex-ai/workbench/create-managed?authuser=4&project=vertex-ai-334021

The screenshot shows the Vertex AI Workbench interface. On the left sidebar, 'Workbench' is selected. In the main area, a form is being filled out to create a managed notebook. The 'Notebook name' field contains 'managed-notebook-1638572301'. The 'Region' dropdown is set to 'us-central1 (Iowa)'. A modal dialog box is displayed, stating: 'Quota exceeded for quota metric 'Create Runtime API requests' and limit 'Create Runtime API requests per minute' of service 'notebooks.googleapis.com' for consumer 'project\_number:310606389043''. Below the modal, there are 'CREATE' and 'CANCEL' buttons.

← → 🔒 console.cloud.google.com/vertex-ai/locations/us-central1/models/5366254460290990080/deploy?authuser=2&project=vertex-ai-334019

The screenshot shows the Vertex AI Model page for a model named 'fraud\_detection\_2021123195457'. The 'DEPLOY & TEST' tab is selected. The 'Deploy your model' section shows a table of endpoints. One endpoint, 'endpointfraud', is listed with ID 141731446866837504, status Active, 0 models, region us-central1, monitoring Disabled, and last updated Dec 3, 2021, 2:34:54 PM. The 'Test your model' section includes a 'PREVIEW' button and a table for inputting feature values. The first row has 'Time' as the feature column name, type Numerical, required, value 84883, and a note 'Predicted column not yet known'. The second row has 'V1' as the feature column name, type Numerical, required, value 0.0218339018594994, and a note 'Prediction result --'. The third row has 'V2' as the feature column name, type Numerical, required, value 0.0672124249198848, and a note ' --'. The fourth row has 'V3' as the feature column name, type Numerical, required, value 0.0000000000000002, and a note ' --'.

h) <https://codelabs.developers.google.com/codelabs/automl-forecasting-with-vertex-ai/#0>



```

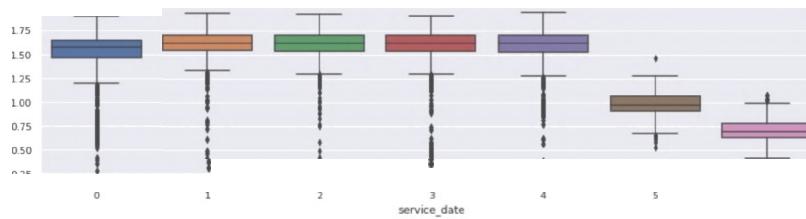
← G é 7d4fl9I?d96908?ea-donus-west.no+ebooksgoogIeuserconientcom/abjr e,'waiing-dna-anas ouwu \idor-liukri 'uo+ book\04-Score.p ab
fe Edit View Ru Kernel Gis labs Ss4ings Help
File Edit View Run Kernel Git Tabs Settings Help
RTermia4 X ñJbe, Qoe.ipab Markdown git
min 222071
Name Last Modified 75% 1660947
02- mod.1.p yab 11 minutes ago
cta_ridership.csv 5 minutes ago

```

**TODO 3: Explore seasonality**

- Is there much difference between months\*
- can you extract the trend and seasonal pattern from the data

days of free k = d1.index.to\_series(), d1['day of free k']

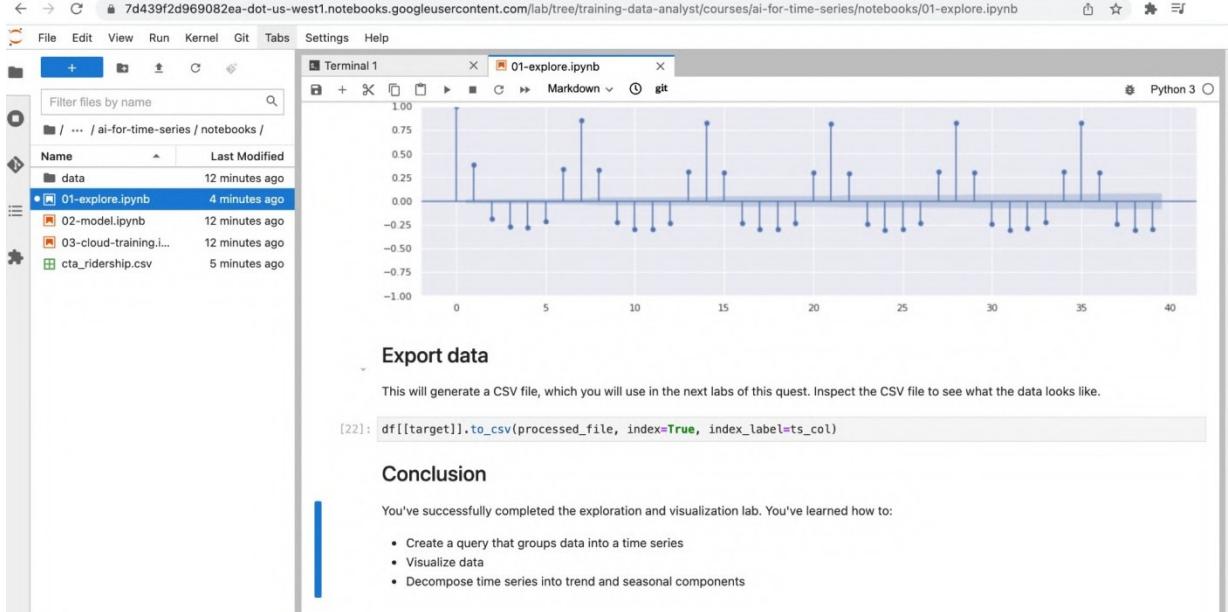
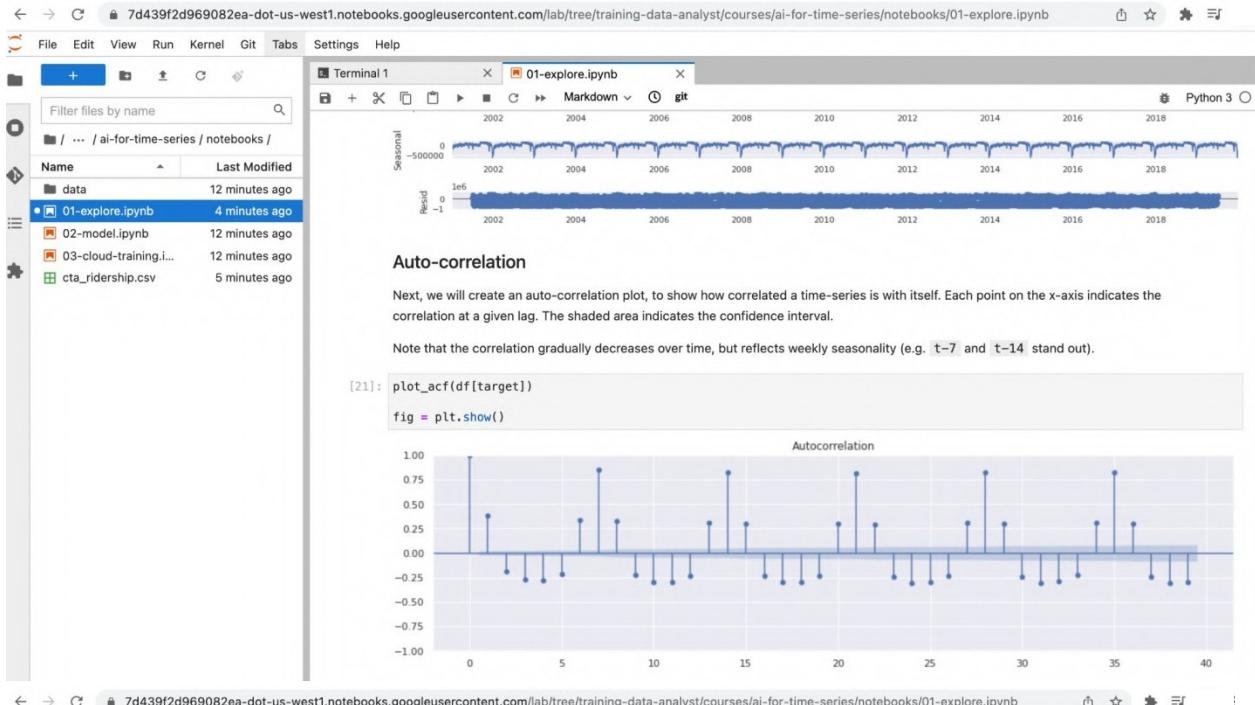


```

File Edit View Run Kernel G It Tabs Se ttings Help
File Edit View Run Kernel G It Tabs Se ttings Help
01-explore.ipynb
Name Last Modified
meseries}xébooñ,
fo fe %s:ridnios of vales far eat svrh » s bo piñ:
months = d .nde to_series().d:nook
02-model.i ab 12 minutes ago
FB cta_ridership.csv 5 minutes ago

```

**total\_riders** = seasonal\_decompose(darge), period=3f



- i) [https://codelabs.developers.google.com/vertex\\_multiworker\\_training#0](https://codelabs.developers.google.com/vertex_multiworker_training#0)

File Edit View Run Kernel Git Tabs Settings Help

Terminal 2 task.py Dockerfile

```
1 FROM gcr.io/deeplearning-platform-release/tf2-gpu.2-5
2
3 WORKDIR /
4
5 # Installs hypertune library
6 RUN pip install cloudml-hypertune
7
8 # Copies the trainer code to the docker image.
9 COPY trainer /trainer
10
11 # Sets up the entry point to invoke the trainer.
12 ENTRYPOINT ["python", "-m", "trainer.task"]
```

Simple 2 0 Dockerfile

Ln 1, Col 1 Spaces: 4 Dockerfile

File Edit View Run Kernel Git Tabs Settings Help

Terminal 2 task.py Dockerfile

```
1 import tensorflow as tf
2 import tensorflow_datasets as tfds
3 import argparse
4 import hypertune
5
6 NUM_EPOCHS = 10
7
8 def get_args():
9     '''parse args. Must include all hyperparameters you want to tune.'''
10    parser = argparse.ArgumentParser()
11    parser.add_argument(
12        '--learning_rate',
13        required=True,
14        type=float,
15        help='learning rate')
16    parser.add_argument(
17        '--momentum',
18        required=True,
19        type=float,
20        help='SGD momentum value')
21    parser.add_argument(
22        '--num_neurons',
23        required=True,
24        type=int,
25        help='number of units in last hidden layer')
26    args = parser.parse_args()
27    return args
28
29
30
31 def preprocess_data(image, label):
32     '''Resizes and scales images.'''
33
34     image = tf.image.resize(image, (150,150))
35     return tf.cast(image, tf.float32) / 255., label
36
37
38
39 def create_dataset():
```

Training pipelines are the primary model training mechanism in Vertex AI. You can use training pipelines to create an AutoML-trained model or a custom-trained model. For custom-trained models, training pipelines orchestrate custom training jobs and hyperparameter tuning with additional steps like adding a data set or uploading the model to Vertex AI for prediction serving. [Learn more](#)

Region  
us-central1 (Iowa) ▾ ?

Filter Enter a property name

Name	ID	Job type	Model type
multiworker-cassava	2119053575840595968	Training pipeline	Custom

The training job will take about 30-35 minutes to complete.

Congratulations! You've learned how to use Vertex AI to:

- Launch a multi-worker training job for training code provided in a custom container. You used a TensorFlow model in this example, but you can train a model built with any framework using custom or built-in containers.

j) <https://codelabs.developers.google.com/codelabs/bqml-vertex-prediction#0>

training pipelines to create an AutoML-trained model or a custom-trained model. For custom-trained models, training pipelines orchestrate custom training jobs and hyperparameter tuning with additional steps like adding a data set or uploading the model to Vertex AI for prediction serving. [Learn more](#)

Region

us-central1 (Iowa)



Filter Enter a property name

Name	ID	Job type	Model type
j multiworker-cassava	2119053575840595968	Training pipeline	@ CUSTOffl

The training job will take about 30-35 minutes to complete.

@ Congratulations! 6

You've learned how to use Vertex AI to:

- Launch a multi-worker training job for training code provided in a custom container. You used a TensorFlow model in this example, but you can train a model built with any framework using custom or built-in containers.

A screenshot of a Jupyter Notebook interface. On the left, there's a file browser showing a directory structure with files: 'horses\_or\_humans', 'trainer', 'Dockerfile', and 'Untitled.ipynb'. The 'Untitled.ipynb' file is selected. In the center, there are three tabs: 'Terminal 2', 'Untitled.ipynb', and 'Dockerfile'. The 'Untitled.ipynb' tab shows Python code. The code includes a JSON dump of data and a command to set the REGION variable.

```
%writefile default-pred.json
{
    "instances": [
        {"age": 39,
         "bill_amt_1": 47174,
         "bill_amt_2": 47974,
         "bill_amt_3": 48630,
         "bill_amt_4": 50803,
         "bill_amt_5": 30789,
         "bill_amt_6": 15874,
         "education_level": "1",
         "limit_balance": 50000,
         "marital_status": "2",
         "pay_0": 0,
         "pay_2": 0,
         "pay_3": 0,
         "pay_4": 0,
         "pay_5": "0",
         "pay_6": "0",
         "pay_amt_1": 1800,
         "pay_amt_2": 2000,
         "pay_amt_3": 3000,
         "pay_amt_4": 2000,
         "pay_amt_5": 2000,
         "pay_amt_6": 2000,
         "sex": "1"
    ]
}

REGION="us-central1" # either us-central1, europe-west4, or asia-east1
```

The screenshot shows a Jupyter Notebook interface with the following components:

- File Tree:** On the left, it shows a directory structure under "/horses\_or\_humans/". The files listed are:
  - trainer (modified 22 minutes ago)
  - Dockerfile (modified 23 minutes ago)
  - Untitled.ipynb (selected, modified seconds ago)
- Code Editor:** The main area contains Python code. It includes a JSON object definition and a curl command for making a POST request to a prediction endpoint.
- Terminal:** A terminal tab labeled "Terminal 2" is open, showing the following command and its output:

```

{
    "instances": [
        {"age": 39,
        "bill_amt_1": 47174,
        "bill_amt_2": 47974,
        "bill_amt_3": 48638,
        "bill_amt_4": 58883,
        "bill_amt_5": 38789,
        "bill_amt_6": 15874,
        "education_level": "1",
        "limit_balance": 58000,
        "marital_status": "2",
        "pay_0": 0,
        "pay_2": 0,
        "pay_3": 0,
        "pay_4": 0,
        "pay_5": 0,
        "pay_6": 0,
        "pay_amt_1": 1800,
        "pay_amt_2": 2000,
        "pay_amt_3": 3000,
        "pay_amt_4": 2000,
        "pay_amt_5": 2000,
        "pay_amt_6": 2000,
        "sex": "1"
    ]
}

[ ]: REGION="us-central1" # either us-central1, europe-west4, or asia-east1

[ ]: !curl \
-X POST \
-H "Authorization: Bearer $(gcloud auth print-access-token)" \
-H "Content-Type: application/json" \
https://us-central1-prediction-aiplatform.googleapis.com/v1alpha1/projects/$PROJECT_ID/locations/$REGION/endpoints/$ENDPOINT \
-d "@default-pred.json"
  
```

## k) <https://codelabs.developers.google.com/vertexx-xgb-wit#0>

The screenshot shows a Jupyter Notebook interface with the following components:

- File Tree:** On the left, it shows a directory structure under "/". The files listed are:
  - src (modified 25 minutes ago)
  - tutorials (modified 25 minutes ago)
- Code Editor:** The main area contains Python code. It includes a pip3 command for installing xgboost==1.2.
- Terminal:** A terminal tab labeled "Terminal 1" is open, showing the following command and its output:

```

(base) jupyter@london-bikes-codelabs:~$ pip3 install xgboost==1.2
Collecting xgboost==1.2
  Downloading xgboost-1.2.0-py3-none-manylinux2010_x86_64.whl (148.9 MB)
Requirement already satisfied: ecipy in /opt/conda/lib/python3.7/site-packages (from xgboost==1.2) (1.7.3)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages (from xgboost==1.2) (1.19.5)
Installing collected packages: xgboost
Successfully installed xgboost-1.2.0
(base) jupyter@london-bikes-codelabs:~$ 
  
```

g c27f2704da25242-dot-us-central1.notebooks.googleusercontent.com /lab/tree/e ntitled.ipynb



File + X □ ▢ ▣ ▤ ▥ ▦ ▨ Code ▾ ⓘ

Python 3 □

⚠️ extension community. However, we do not review third-party extensions, and some extensions may introduce security risks or contain malicious code that runs on your system.

```
import collections  
import wtitwidge
```

```
from wtitwidge import tebo ok, vzsua, i_z a l No n, mpo rt, WtTw id, get, III Co nf zgBu i lde, r
```

```
cocuru mAmzs = cone<<io>s.ordeceJoi<<(c
```

← G a c27f2704da25242-dot-us-central1.notebooks.googleusercontent.com, lau/kree/ui titieu ipyou  
file Edit view Ran kernel cii cabs sewings i-Jelp



⚠️ review third-party extensions, and some extensions may introduce security risks or contain malicious code that runs on your system.

```
'purchaser_type': 'category',  
'hoepa_status': 'category',
```

```
'preapproval': 'category',  
'county_code': np.float64,
```

Enable

```
'lien_status': 'category',
```

```
data = pd.read_csv(
```

← W g c27f27 04da 25242-dot-us-central1. notebooks.googleusercontent.com/Untitled.ipynb  
file Ewe yiez Fun Kernel sir Rao stings neip

Untitled.ipynb

Code applicable

**WARNING**

The Jupyter Notebook development team is extension community. Howe'ver we do not extensions may introduce security risks or

**Enable**

[?] dummy\_columns = list(data.dtypes[data.dtypes == 'category'].index)  
data = pd.get\_dummies(data, columns=dummy\_columns)

as=year o u#aa loanamq=<ss=s =u=<>e app lsan\_ nome\_i ousa s g Pm aion me me=n amp ome iras<n «

719588	2016	1	10 0 0	127 0	70.0	2422 0	46400.0
--------	------	---	--------	-------	------	--------	---------

The screenshot shows two JupyterLab sessions side-by-side. Both sessions have the same interface with a top navigation bar (File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help) and a sidebar on the left containing a search bar, a warning message about third-party extensions, and sections for Installed and Discover packages.

**Code Cell 1 (Top):**

```
[9]: x,y = data.values,labels
x_train,x_test,y_train,y_test = train_test_split(x,y)

[10]: model = xgb.XGBClassifier(
        objective='reg:logistic'
    )

[11]: model.fit(x_train, y_train)

[12]: y_pred = model.predict(x_test)
acc = accuracy_score(y_test, y_pred.round())
print(acc, '\n')
0.874616

[13]: model.save_model('model.bst')

[14]: num_wit_examples = 500
test_examples = np.hstack((x_test[:num_wit_examples],y_test[:num_wit_examples].reshape(-1,1)))

[16]: config_builder = (WitConfigBuilder(test_examples.tolist(), data.columns.tolist() + ['mortgage_status'])
.set_custom_predict_fn(model.predict_proba)
.set_target_feature('mortgage_status')
.set_label_vocab(['denied', 'approved']))
WitWidget(config_builder, height=800)
```

**Code Cell 2 (Bottom):**

```
[11]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bytree=1,
                   colsample_bynode=1, colsample_bylevel=1, gamma=0, gpu_id=-1,
                   importance_type='gain', interaction_constraints='',
                   learning_rate=0.300000012, max_delta_step=0, max_depth=6,
                   min_child_weight=1, missing=nan, monotone_constraints=''),
                   n_estimators=100, n_jobs=0, num_parallel_tree=1,
                   objective='reg:logistic', random_state=0, reg_alpha=0,
                   reg_lambda=1, scale_pos_weight=1, subsample=1,
                   tree_method='exact', validate_parameters=1, verbosity=None)

[12]: y_pred = model.predict(x_test)
acc = accuracy_score(y_test, y_pred.round())
print(acc, '\n')
0.874616

[13]: model.save_model('model.bst')

[14]: num_wit_examples = 500
test_examples = np.hstack((x_test[:num_wit_examples],y_test[:num_wit_examples].reshape(-1,1)))

[16]: config_builder = (WitConfigBuilder(test_examples.tolist(), data.columns.tolist() + ['mortgage_status'])
.set_custom_predict_fn(model.predict_proba)
.set_target_feature('mortgage_status')
.set_label_vocab(['denied', 'approved']))
WitWidget(config_builder, height=800)
```

**Data Analysis Tools:**

Below the code cells, there is a data analysis interface with tabs for "Datapoint editor", "Performance & Fairness", and "Features". The "Datapoint editor" tab is active, showing a "Visualize" section with options for "Datapoints" (radio button selected), "Partial dependence plots", and "Nearest counterfactual". Below this, a note states: "Datapoints and their inference results will be displayed here." There are also dropdown menus for "Binning I...", "Color By", "Label By", and "Scatter I...".

l) h [https://codelabs.developers.google.com/vertex\\_notebook\\_executor#0](https://codelabs.developers.google.com/vertex_notebook_executor#0)

The screenshot shows a JupyterLab interface with a warning message from the development team about third-party extensions. The main area displays Python code for importing TensorFlow, loading a dataset, and creating train-validation splits.

```
[1]: import tensorflow as tf
import tensorflow_datasets as tfds
import tensorflow_hub as hub

[ ]: data, info = tfds.load(name='deep_weeds', as_supervised=True, with_info=True)
NUM_CLASSES = info.features['label'].num_classes
DATASET_SIZE = info.splits['train'].num_examples

[ ]: def preprocess_data(image, label):
    image = tf.image.resize(image, (300,300))
    return tf.cast(image, tf.float32) / 255., label

[ ]: # Create train/validation splits

# Shuffle dataset
dataset = data['train'].shuffle(1000)

train_split = 0.8
val_split = 0.2
train_size = int(train_split * DATASET_SIZE)
val_size = int(val_split * DATASET_SIZE)

train_data = dataset.take(train_size)
train_data = train_data.map(preprocess_data)
train_data = train_data.batch(64)

validation_data = dataset.skip(train_size)
validation_data = validation_data.map(preprocess_data)
validation_data = validation_data.batch(64)

[ ]: feature_extractor_model = "inception_v3"

[ ]: tf_hub_uri = f"https://tfhub.dev/google/imagenet/{feature_extractor_model}/feature_vector/5"
```

The screenshot shows the continuation of the JupyterLab session. The code now includes the creation of a Keras layer using the loaded feature extractor, the definition of a Sequential model, its compilation with Adam optimizer and SparseCategoricalCrossentropy loss, and finally its training with 20 epochs.

```
[ ]: # Shuffle dataset
dataset = data['train'].shuffle(1000)

train_split = 0.8
val_split = 0.2
train_size = int(train_split * DATASET_SIZE)
val_size = int(val_split * DATASET_SIZE)

train_data = dataset.take(train_size)
train_data = train_data.map(preprocess_data)
train_data = train_data.batch(64)

validation_data = dataset.skip(train_size)
validation_data = validation_data.map(preprocess_data)
validation_data = validation_data.batch(64)

[ ]: feature_extractor_model = "inception_v3"

[ ]: tf_hub_uri = f"https://tfhub.dev/google/imagenet/{feature_extractor_model}/feature_vector/5"

[ ]: feature_extractor_layer = hub.KerasLayer(
    tf_hub_uri,
    trainable=False)

[ ]: model = tf.keras.Sequential([
    feature_extractor_layer,
    tf.keras.layers.Dense(units=NUM_CLASSES)
])

[ ]: model.compile(
    optimizer=tf.keras.optimizers.Adam(),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['acc'])

model.fit(train_data, validation_data=validation_data, epochs=20)
```

m) h <https://codelabs.developers.google.com/codelabs/time-series-forecasting-with-cloud-ai-platform#0>

```
■ /          remote: Counting objects: 100, done.  
■ Name       remote: Compressing objects: 100% (24/24), done.  
■ Last Modified  remote: Total 54506 (delta 147), reused 256 (delta 83), pack-reused 54131  
  
■ tutorials  Checking out files: 100% (12334/12334), done.  
■ data-analyst  (base) jupyter@tensorflow-2-3-20211203-133926:~$ ■  
■ 26 minutes ago
```

```

g 7d439f2d 96g082 ea--dot-u-s we st1.notebooks.google.com/about+content.html about+content.html analysis+data+systems for training+creation+notebooks70j+exploratory
File Edit View Run Kernel Git Tabs Settings Help
+ X □ ▢ ▣ ▤ ▥ Markdown git
$ Python 3
■ / ... /ai-for-time-series / notebooks /
restarting the kernel may be required to use new packages.

data 7 minutes ago
01-explore.ipynb 1 minute ago
02-model.ipynb 7 minutes ago

```

Up zip is installed: click installing... download analysis for training creation notebooks70j+exploratory  
Requirement already satisfied: lumpy>=1.?? in /opt/czntv/lib/python3.11/site-packages (from some packages (from site-packages)) (1.19.1)

Requirement already satisfied: lumpy>=1.?? in /opt/czntv/lib/python3.11/site-packages (from some packages (from site-packages)) (1.19.1)

#### Import libraries and define constants

```

from pandas.plotting import register_matplotlib_converters
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import grangercausalitytests

import seaborn as sns

```

```

File Edit View Run Kernel Git Tabs Settings Help
+ Terminal 1 01-explore.ipynb

```

Name	Last Modified
data	7 minutes ago
01-explore.ipynb	1 minute ago
02-model.ipynb	7 minutes ago
03-cloud-training.i...	7 minutes ago
cta_ridership.csv	seconds ago

```

[5]: # Enter your project and region. Then run the cell to make sure the
# Cloud SDK uses the right project for all the commands in this notebook.
PROJECT = 'Vertex-AI' # REPLACE WITH YOUR PROJECT NAME
REGION = 'us-central-1' # REPLACE WITH YOUR REGION e.g. us-central1

#Don't change the following command - this is to check if you have changed the project name above.
#assert PROJECT != 'Vertex-AI', 'Don't forget to change the project variables!'

target = 'total_rides' # The variable you are predicting
target_description = 'Total Rides' # A description of the target variable
features = {'day_type': 'Day Type'} # Weekday = W, Saturday = A, Sunday/Holiday = U
ts_col = 'service_date' # The name of the column with the date field

raw_data_file = 'https://data.cityofchicago.org/api/views/6iiy-9s97/rows.csv?accessType=DOWNLOAD'
processed_file = 'cta_ridership.csv' # Which file to save the results to

```

#### Load data

```

[6]: # Import CSV file
df = pd.read_csv(raw_data_file, index_col=ts_col, parse_dates=[ts_col])

df = df[df.index > '2029-01-01']

df = df.drop_duplicates()

```

File Edit View Run Kernel Git Tabs Settings Help

Explore data

```
[12]: # Print the top 5 rows
df.head()
```

	day_type	bus	rail_boardings	total_rides
service_date				
2001-01-01	U	297192	126455	423647
2001-01-02	W	780827	501952	1282779
2001-01-03	W	824923	536432	1361355
2001-01-04	W	870021	550011	1420032
2001-01-05	W	890426	557917	1448343

**TODO 1: Analyze the patterns**

- Is ridership changing much over time?
- Is there a difference in ridership between the weekday and weekends?
- Is the mix of bus vs rail ridership changing over time?

```
[13]: # Initialize plotting
register_matplotlib_converters() # Addresses a warning
sns.set(rc={'figure.figsize':(16,4)})
```

```
[14]: # Explore total rides over time
sns.lineplot(data=df, x=df.index, y=df[target]).set_title('Total Rides')
```



File Edit View Run Kernel Git Tabs Settings Help

Terminal 1 01-explore.ipynb

```
[16]: # Explore rides by transportation type
sns.lineplot(data=df[['bus','rail_boardings']].set_title('Total Rides by Transportation Type')
fig = plt.show()
```

**TODO 2: Review summary statistics**

- How many records are in the dataset?
- What is the average # of riders per day?

```
[17]: df[target].describe().apply(lambda x: round(x))
```

	count	mean	std	min	25%	50%	75%	max
target	6939	1368761	391443	222071	1005394	1548343	1660947	2049519

File Edit View Run Kernel Git Tabs Settings Help

Terminal 1 01-explore.ipynb

```
[17]: df[target].describe().apply(lambda x: round(x))
```

	count	mean	std	min	25%	50%	75%	max
target	6939	1368761	391443	222071	1005394	1548343	1660947	2049519

**TODO 3: Explore seasonality**

- Is there much difference between months?
- Can you extract the trend and seasonal pattern from the data?

```
[18]: # Show the distribution of values for each day of the week in a boxplot:
# Min, 25th percentile, median, 75th percentile, max
daysofweek = df.index.to_series().dt.dayofweek
fig = sns.boxplot(x=daysofweek, y=df[target])
```

