



# Using Google Cloud Platform for Econometrics

GOOGLE CLOUD PLATFORM

# Prerequisites

- ▶ Need a Google Cloud Platform account
  - ▶ Go to <https://cloud.google.com/> and set up a free trial account.
- ▶ Need a background in econometrics / multi-variate statistics.
- ▶ Some prior background in Python / R is helpful.



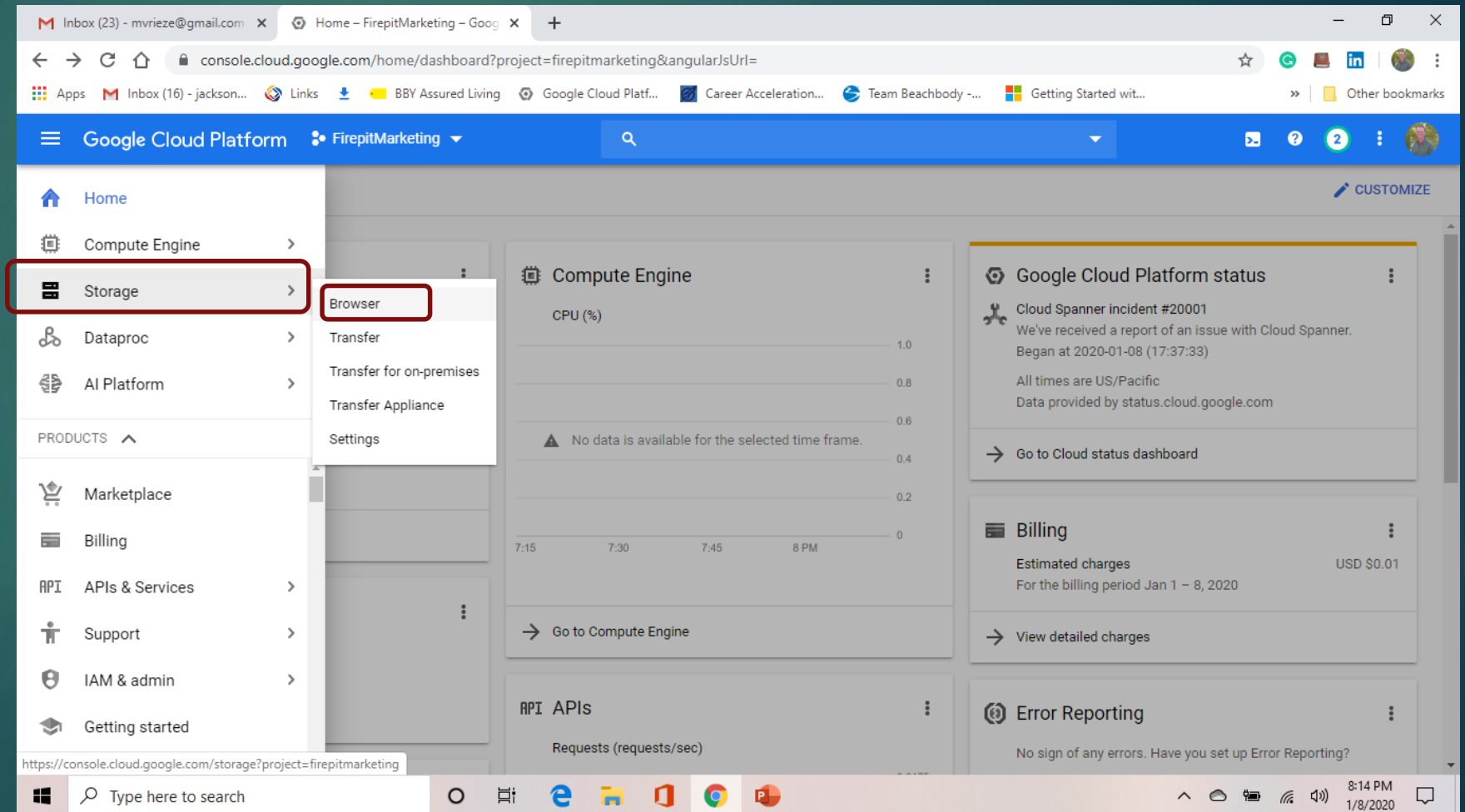
---

# Storage Bucket Setup

---

# Storage Bucket Setup

- ▶ Navigate to “Storage”
- ▶ Select “Browser”



# Storage Bucket Setup

- ▶ Select “Create Bucket”

The screenshot shows the Google Cloud Platform Storage browser interface. The left sidebar contains a menu with options: Storage, Browser, Transfer, Transfer for on-premises, Transfer Appliance, and Settings. The main area is titled 'Storage browser' and features a '+ CREATE BUCKET' button, which is highlighted with a red rectangle. Below this button are 'DELETE' and 'REFRESH' buttons. A table lists existing storage buckets, with one entry 'mvrieze-bucket' visible. The table columns include Name, Location type, Location, Default storage class, Public access, Access control, Lifecycle rules, and Labels. The bottom of the screen shows the Windows taskbar with the search bar and various application icons.

Name	Location type	Location	Default storage class	Public access	Access control	Lifecycle rules	Labels
mvrieze-bucket	Region	us-central1 (lo...	Standard	Not public	Uniform	None	

# Storage Bucket Setup

- ▶ Name your bucket
- ▶ Choose a location type - “Region”

The screenshot shows the Google Cloud Platform console for the 'FirepitMarketing' project. The 'Create a bucket' wizard is in progress. The first step, 'Name your bucket', is completed with the name 'mvrieze-bucket2'. The second step, 'Choose where to store your data', shows 'Region' selected as the location type. A 'Monthly cost estimate' sidebar on the right provides a breakdown of costs for storage, retrieval, and operations.

Category	Value	Unit	Cost
Storage size		GB	\$0.020 per GB-month
Data retrieval size		GB	Free
Class A operations		per-month	\$0.005 per 1,000 ops
Class B operations		per-month	\$0.0004 per 1,000 ops

# Storage Bucket Setup

- ▶ Choose a Location – In this case, “us-west1 (Oregon)”
- ▶ Choose the “Standard” storage class

The screenshot shows the Google Cloud Platform console for the project 'FirepitMarketing'. The 'Create a bucket' wizard is active, with the 'Location' dropdown set to 'us-west1 (Oregon)'. Below this, the 'Choose a default storage class for your data' section shows the 'Standard' class selected, with a description: 'Best for short-term storage and frequently accessed data'. Other options include 'Nearline', 'Coldline', and 'Archive'. A 'CONTINUE' button is visible at the bottom of this section. On the right, a 'Monthly cost estimate' sidebar provides a breakdown of costs: 'Storage size' (GB) at \$0.020 per GB-month, 'Data retrieval size' (GB) at Free, 'Class A operations' at \$0.005 per 1,000 ops per-month, and 'Class B operations' at \$0.0004 per 1,000 ops per-month. The sidebar also includes a link to 'Pricing details' and a note about availability (SLA: 99.9%).



# Storage Bucket Setup

- ▶ Accept the default access control (Google-managed key)
- ▶ Choose “Create” (at the bottom) to complete setup.

The screenshot shows the Google Cloud Platform console for the 'FirepitMarketing' project. The 'Storage' section is active, and the 'Create a bucket' wizard is in progress. The 'Choose how to control access to objects' step is completed, indicated by a checkmark. The 'Advanced settings (optional)' section includes 'Encryption' (Google-managed key selected), 'Retention policy' (checkbox not selected), and 'Labels' (checkbox not selected). The 'Monthly cost estimate' section shows input fields for storage and data retrieval sizes, and a table for operations costs.

Storage and retrieval	Cost
Storage size	\$0.020 per GB-month
Data retrieval size	Free

Operations	Cost
Class A operations	\$0.005 per 1,000 ops
Class B operations	\$0.0004 per 1,000 ops



# Storage Bucket Setup

- ▶ Select the storage bucket you created

The screenshot shows the Google Cloud Platform Storage browser interface. The left sidebar contains a menu with options: Storage, Browser, Transfer, Transfer for on-premises, Transfer Appliance, and Settings. The main area is titled 'Storage browser' and includes buttons for '+ CREATE BUCKET', 'DELETE', and 'REFRESH'. A search bar with the text 'Filter by name prefix' is present. Below the search bar is a table listing storage buckets. The first bucket, 'mvrieze-bucket', is highlighted with a red box. The table columns are: Name, Location type, Location, Default storage class, Public access, Access control, Lifecycle rules, and Labels.

Name	Location type	Location	Default storage class	Public access	Access control	Lifecycle rules	Labels
mvrieze-bucket	Region	us-central1 (lo...	Standard	Not public	Uniform	None	

# Storage Bucket Setup

- ▶ Select “Create Folder” to set up a sub-folder inside your bucket and call it “data”.

Note that GCP will also use your storage bucket for other objects besides data. The screenshot also contains folders for Notebooks as well as Dataproc metadata.

The screenshot shows the Google Cloud Platform console interface for a storage bucket named 'mvrieze-bucket'. The 'Create folder' button is highlighted with a red box. Below it, a table lists existing folders: 'data/', 'google-cloud-dataproc-metadata/', and 'notebooks/'. The 'data/' folder is also highlighted with a red box.

Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds
data/	—	Folder	—	—	Not public	—	—	—
google-cloud-dataproc-metadata/	—	Folder	—	—	Not public	—	—	—
notebooks/	—	Folder	—	—	Not public	—	—	—

# Storage Bucket Setup

- ▶ Use the “Upload files” or “Upload folders” or “Create folder” buttons.
- ▶ Note we have already uploaded the “Charity.csv” dataset to our “/data” folder.

Note that upload time depends on the speed of your internet connection, the size of the file and how you connect to GCP. For larger data sets, you might consider using the Google Cloud SDK tool.

The screenshot shows the Google Cloud Platform console interface. The left sidebar contains navigation options: Storage, Browser, Transfer, Transfer for on-premises, Transfer Appliance, and Settings. The main content area displays the 'Bucket details' for 'mvrieze-bucket'. At the top, there are buttons for 'Upload files', 'Upload folder', 'Create folder', 'Manage holds', and 'Delete'. Below these is a search bar labeled 'Filter by prefix...'. A breadcrumb trail shows 'Buckets / mvrieze-bucket / data'. A table lists the objects in the bucket:

<input type="checkbox"/>	Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds
<input type="checkbox"/>	Charity.csv	173.24 KB	application/vnd.ms-excel	Standard	1/5/20, 7:03:30 PM UTC-6	Not public	Google-managed key	-	None



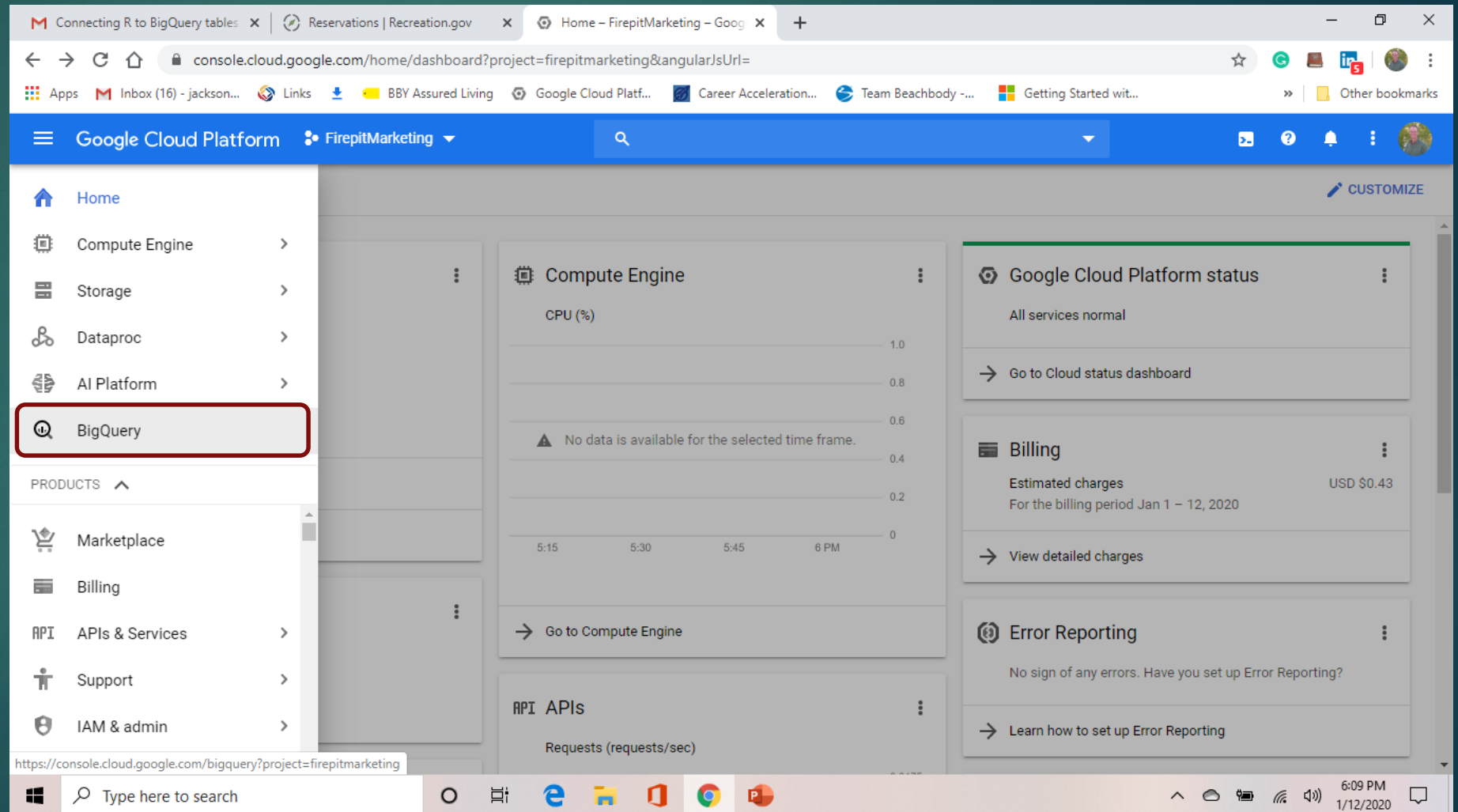
---

# Load Data to BigQuery

---

# Load Data to BigQuery

- ▶ Select “BigQuery”.



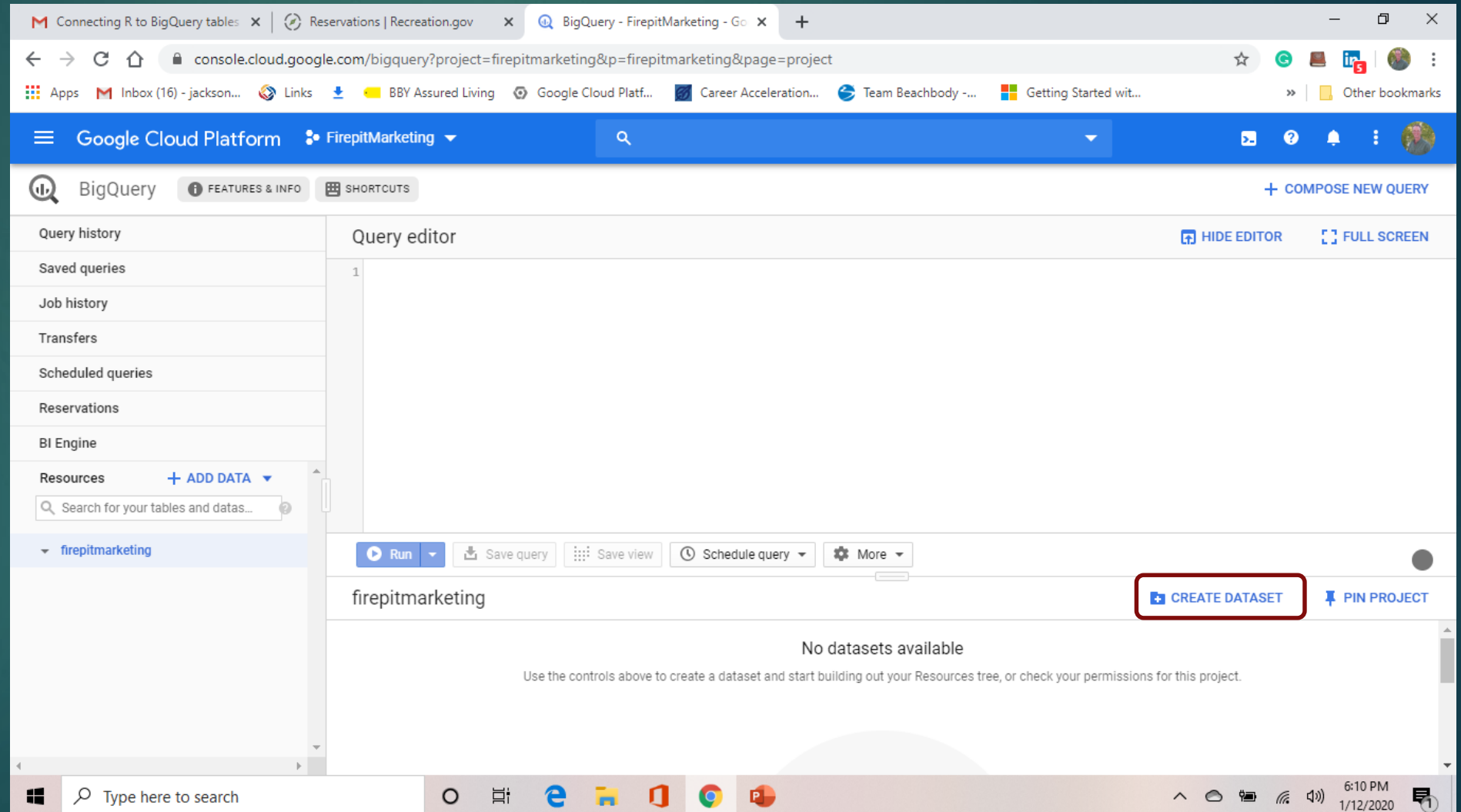
# Load Data to BigQuery

- ▶ Click on your project name.

The screenshot displays the Google Cloud Platform BigQuery console interface. The browser's address bar shows the URL `console.cloud.google.com/bigquery?project=firepitmarketing&p=firepitmarketing&page=project`. The left sidebar contains a navigation menu with options like 'Query history', 'Saved queries', 'Job history', 'Transfers', 'Scheduled queries', 'Reservations', 'BI Engine', and 'Resources'. The 'Resources' section is expanded, showing a search bar and a list of resources. The 'firepitmarketing' project is highlighted with a red rectangle. The main area is titled 'Query editor' and contains a large text area for writing queries. Below the query editor, there are buttons for 'Run', 'Save query', 'Save view', 'Schedule query', and 'More'. The bottom of the console shows the 'firepitmarketing' project name and a message stating 'No datasets available'. The Windows taskbar at the bottom indicates the time is 6:10 PM on 1/12/2020.

# Load Data to BigQuery

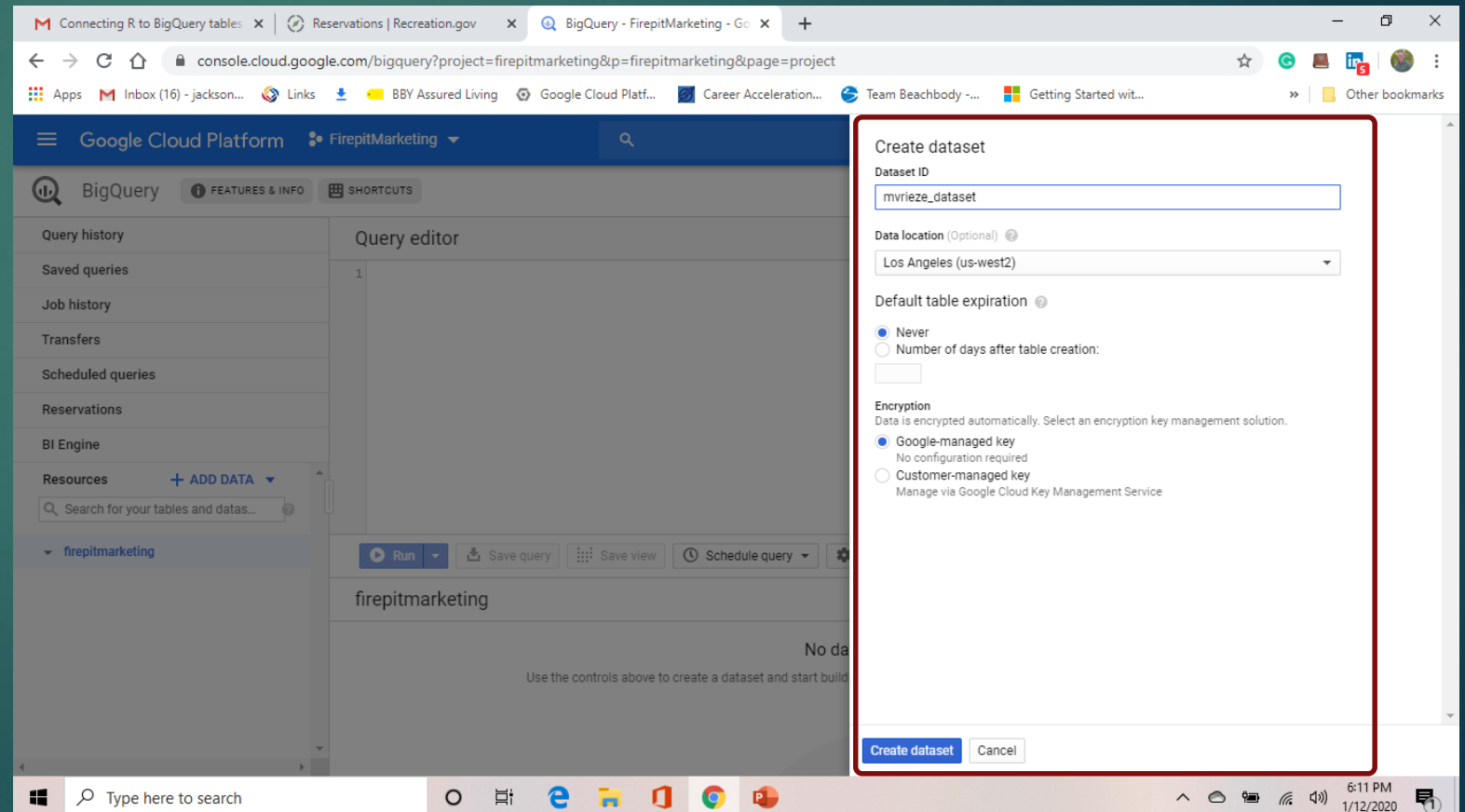
- ▶ Select “Create Dataset”





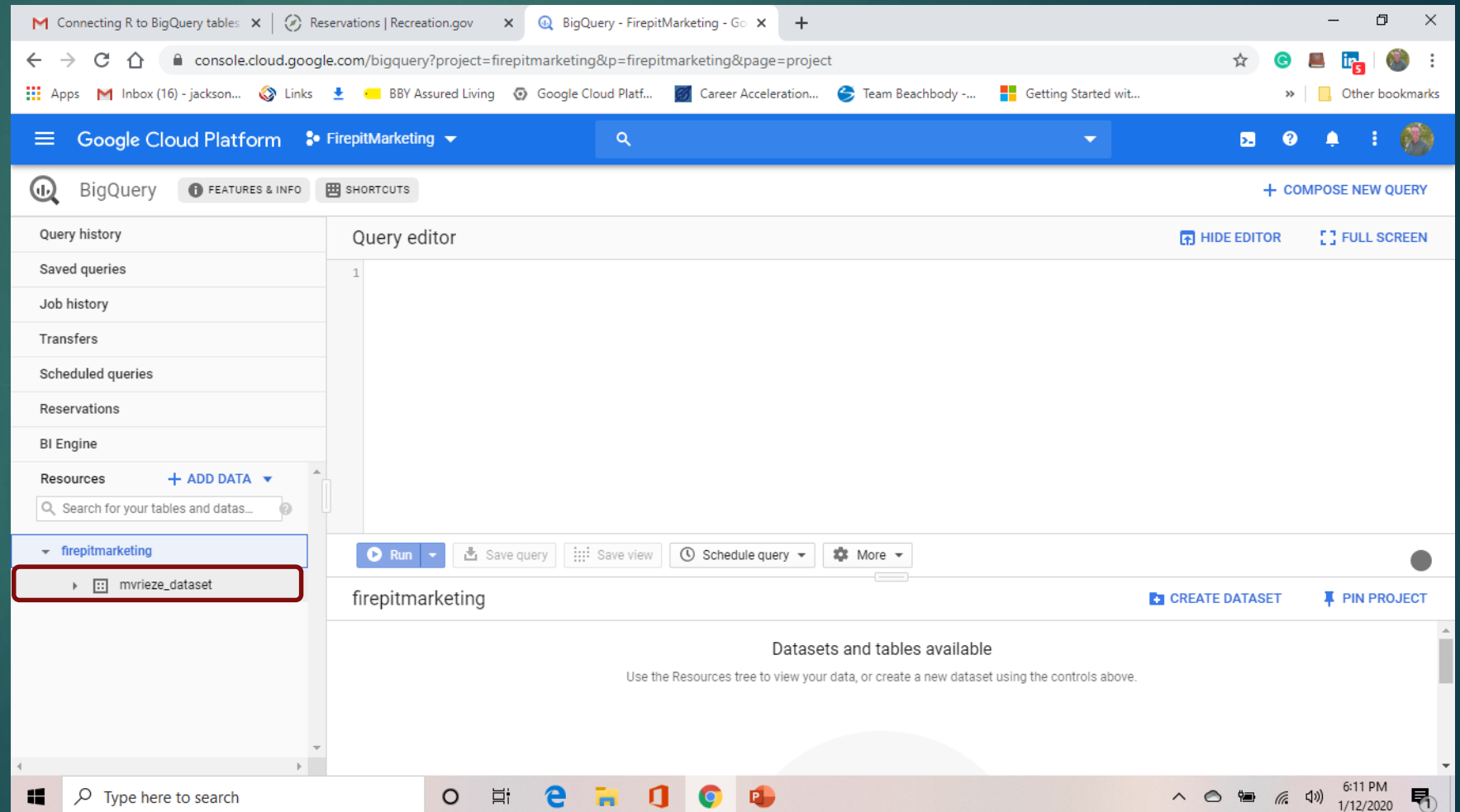
# Load Data to BigQuery

- ▶ Set a name for the dataset
- ▶ Choose a region – We recommend the generic “United States” region
- ▶ Leave the default table expiration and encryption defaults and select “Create Dataset.”



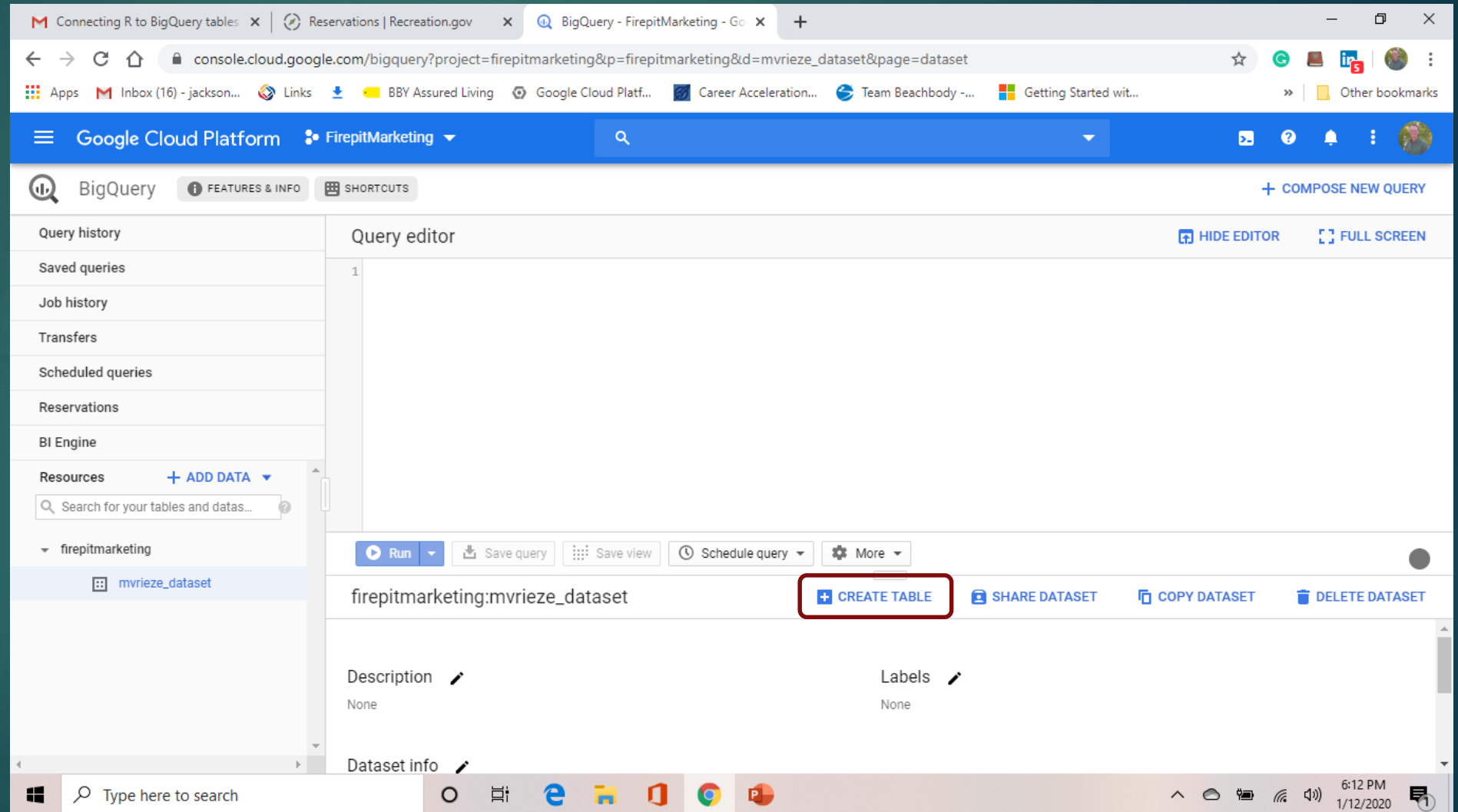
# Load Data to BigQuery

- ▶ Your dataset now appears below your project name.
- ▶ Click on your new dataset.



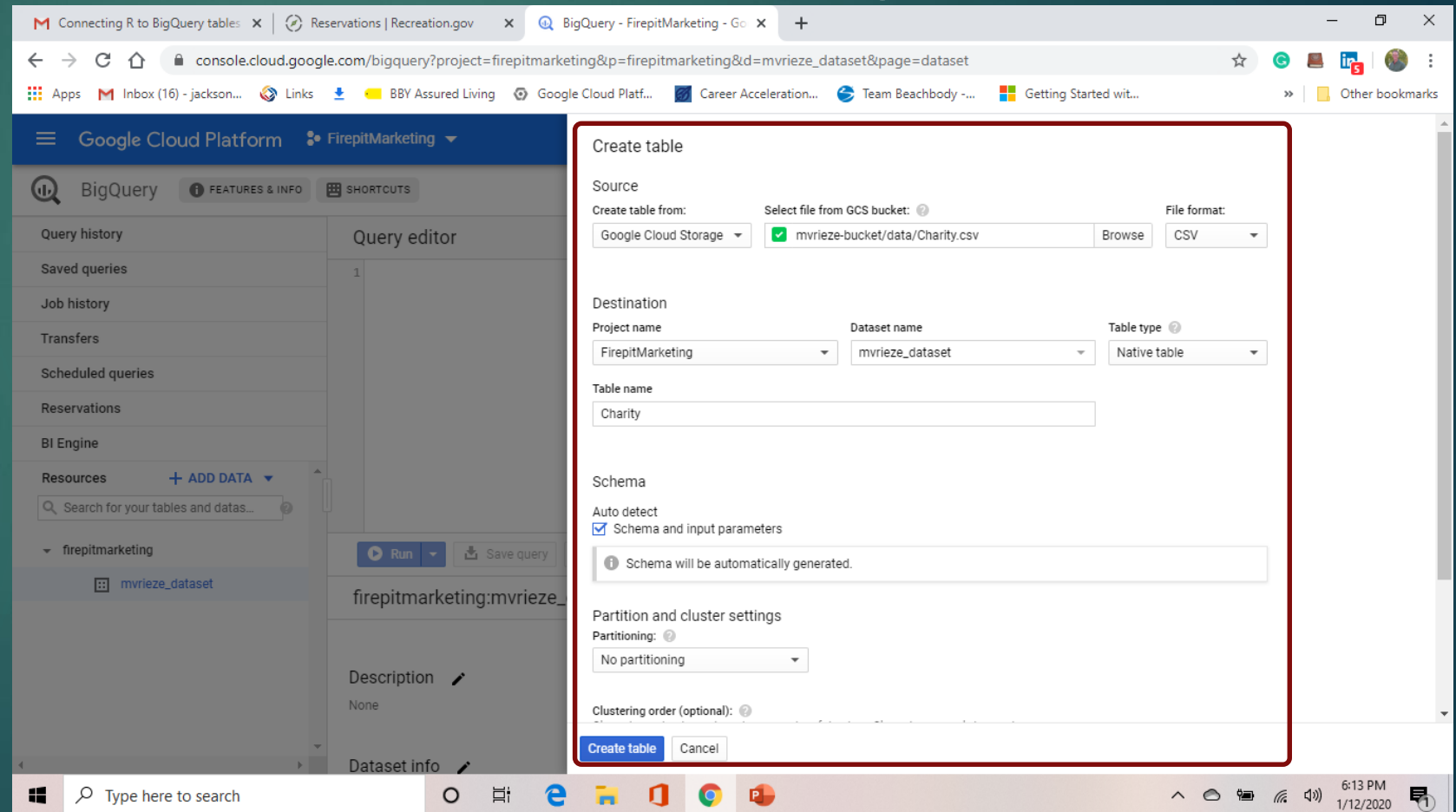
# Load Data to BigQuery

- ▶ Click on “Create Table”.



# Load Data to BigQuery

- ▶ Select Google Cloud Storage and browse/select the Charity.csv file we uploaded earlier.
- ▶ Be sure to check the “Auto detect” box and then select “Create Table”



# Load Data to BigQuery

- ▶ In the query editor, write a simple select query and select “Run”
- ▶ Results appear in bottom box.

The screenshot shows the Google Cloud Platform BigQuery console interface. The browser address bar indicates the URL: `console.cloud.google.com/bigquery?project=firepitmarketing&j=bq:US:bquxjob_1d9617f2_16f9c473be1&page=queryresults`. The left sidebar contains a navigation menu with options like Query history, Saved queries, Job history, Transfers, Scheduled queries, Reservations, BI Engine, and Resources. The main area is divided into a 'Query editor' and a 'Query results' section. The 'Query editor' contains a single SQL query: `select * from mvrieze_dataset.Charity`, which is highlighted with a red box. Below the editor, there are buttons for 'Run', 'Save query', 'Save view', 'Schedule query', and 'More'. A status message indicates: 'This query will process 300.1 KB when run.' The 'Query results' section shows a table with 10 columns: Row, OBS, RESPONSE, GIFT, RESPLASTMAIL, WEEKSLASTRESP, PROPRESPONSE, MAILSPERYEAR, GIFTLASTRESP, and AVERAGEGIFT. The table displays three rows of data. At the bottom, there are pagination controls showing 'Rows per page: 100' and '1 - 100 of 4268'.

Row	OBS	RESPONSE	GIFT	RESPLASTMAIL	WEEKSLASTRESP	PROPRESPONSE	MAILSPERYEAR	GIFTLASTRESP	AVERAGEGIFT
1	809	0	0	0	158.0	0.125	2.0	50	50.0
2	810	0	0	0	158.0	0.125	2.0	50	50.0
3	815	0	0	0	158.0	0.125	2.0	50	50.0



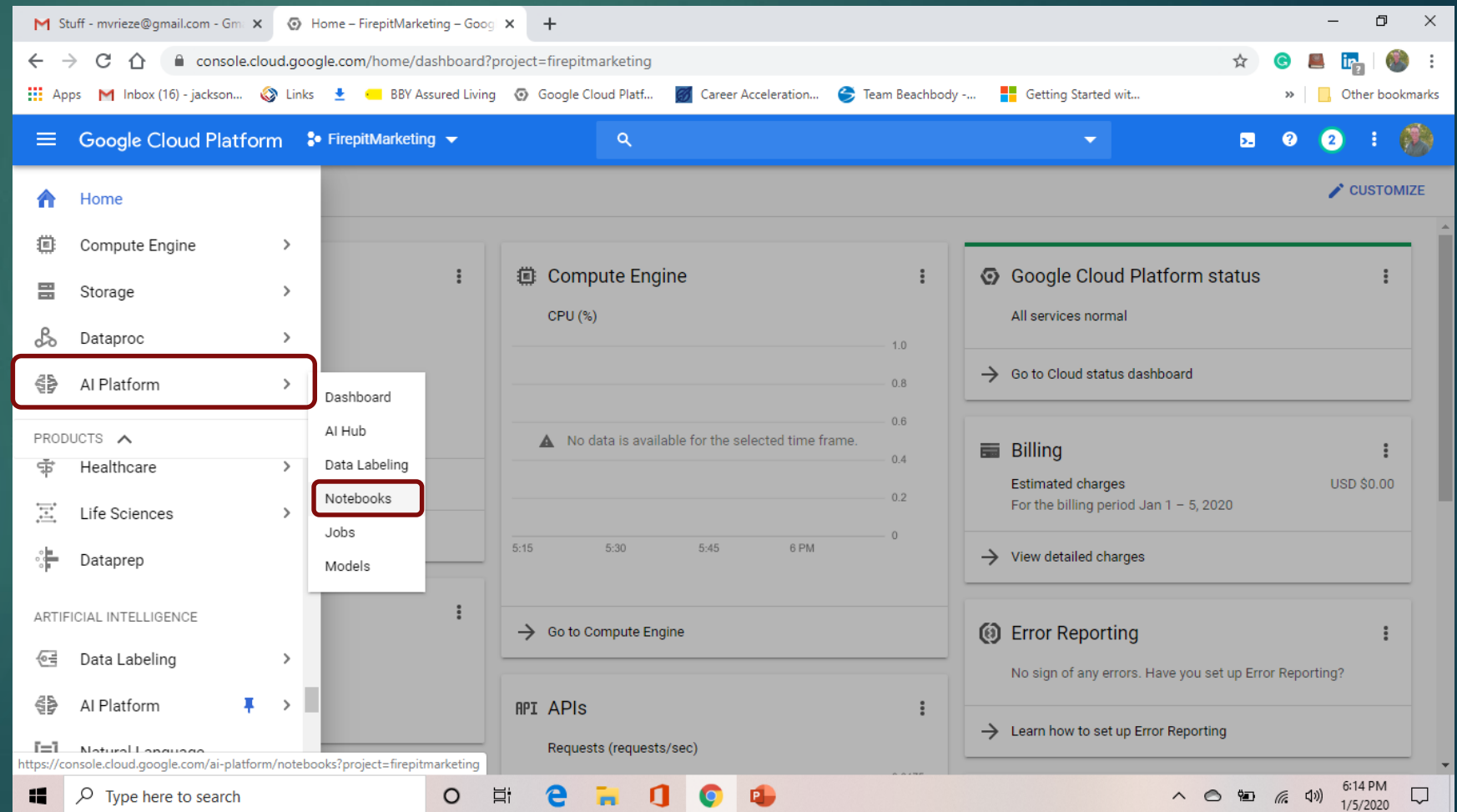
---

# Virtual Machine Setup

---

# Virtual Machine Setup

- ▶ Navigate to “AI Platform”
- ▶ Select “Notebooks”





# Virtual Machine Setup

- ▶ Select “+ New Instance”

The screenshot shows the Google Cloud Platform console for the 'FirepitMarketing' project. The left sidebar contains navigation links: AI Platform, Dashboard, AI Hub, Data Labeling, Notebooks (selected), Jobs, and Models. The main content area is titled 'Notebook instances BETA' and includes a '+ NEW INSTANCE' button, which is highlighted with a red box. Other buttons include REFRESH, START, STOP, RESET, DELETE, and SHOW INFO PANEL. Below the buttons, there is a table with columns: Instance name, Region, Environment, Machine type, GPUs, Permission, and Labels. The table currently displays 'No notebook instances to display'. The bottom of the image shows the Windows taskbar with the search bar and various application icons.

# Virtual Machine Setup

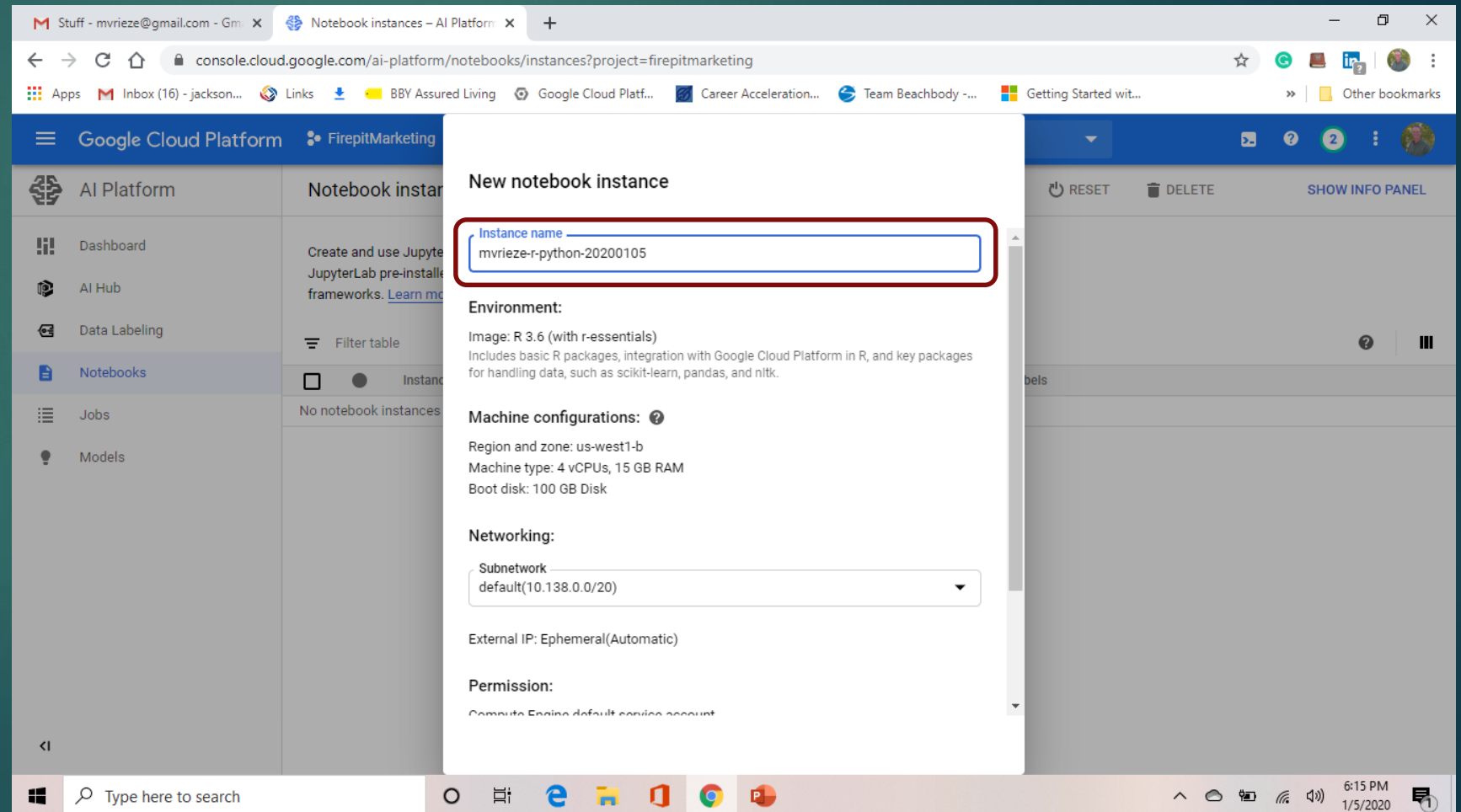
- ▶ Select “R 3.x” (If you wanted Python, don’t worry – Python comes along for the ride)

The screenshot shows the Google Cloud Platform AI Platform interface. The left sidebar contains navigation links: Dashboard, AI Hub, Data Labeling, Notebooks (selected), Jobs, and Models. The main content area is titled 'Notebook instances' with a 'BETA' badge. It includes buttons for '+ NEW INSTANCE', 'REFRESH', 'START', 'STOP', 'RESET', 'DELETE', and 'SHOW INFO PANEL'. Below these is a description: 'Create and use Jupyter Notebooks with a no JupyterLab pre-installed and are configured frameworks. [Learn more](#)'. A 'Filter table' section is present, followed by a table header with 'Instance name' and 'Reg'. The table currently shows 'No notebook instances to display'. The 'Customize instance' dropdown menu is open, listing the following options:

- R 3.6**  
R 3.6 and key libraries pre-installed
- Python  
Python 2 and 3 with Pandas, SciKit Learn and other key packages pre-installed
- TensorFlow Enterprise 1.15  
TensorFlow Enterprise 1.15 pre-installed with support for Keras
- TensorFlow 2.0  
TensorFlow 2.0 pre-installed with support for Keras
- PyTorch 1.3  
PyTorch 1.3 pre-installed
- RAPIDS XGboost [EXPERIMENTAL]  
XGboost optimized for NVIDIA GPUs
- CUDA 10.1  
Optimized for NVIDIA GPUs

# Virtual Machine Setup

- ▶ Name your instance
- ▶ Choose your configuration
  - ▶ Rule of thumb: You need 3x-5x the size of your data in memory to handle your project.



# Virtual Machine Setup

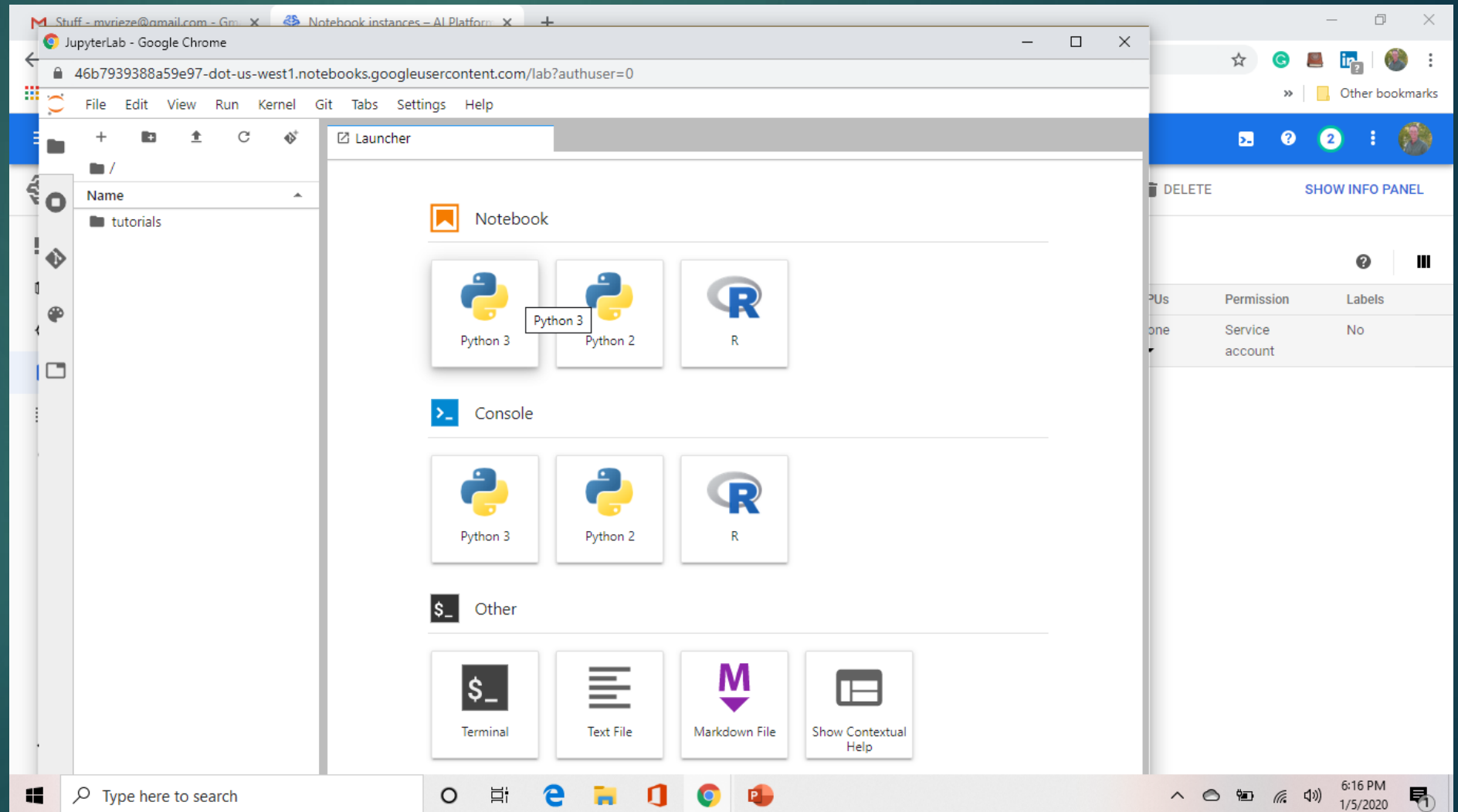
- ▶ Select “Open Jupyterlab”

The screenshot shows the Google Cloud Platform AI Platform Notebook instances page. The page has a sidebar on the left with navigation links: Dashboard, AI Hub, Data Labeling, Notebooks (selected), Jobs, and Models. The main content area is titled 'Notebook instances BETA' and includes buttons for '+ NEW INSTANCE', 'REFRESH', 'START', 'STOP', 'RESET', 'DELETE', and 'SHOW INFO PANEL'. Below these buttons is a table of notebook instances. The table has columns: Instance name, Region, Environment, Machine type, GPUs, Permission, and Labels. One instance is listed: 'mvrieze-r-python-20200105' in the 'us-west1-b' region, with environment 'R:3.6' and machine type '4 vCPUs, 15 GB RAM'. A red box highlights the 'OPEN JUPYTERLAB' button next to this instance.

Instance name	Region	Environment	Machine type	GPUs	Permission	Labels
mvrieze-r-python-20200105	us-west1-b	R:3.6	4 vCPUs, 15 GB RAM	None	Service account	No

# Virtual Machine Setup

- ▶ Select whichever tool you wish to use (R or Python – notebook or console)





---

# Connect Python to GCP Storage Bucket

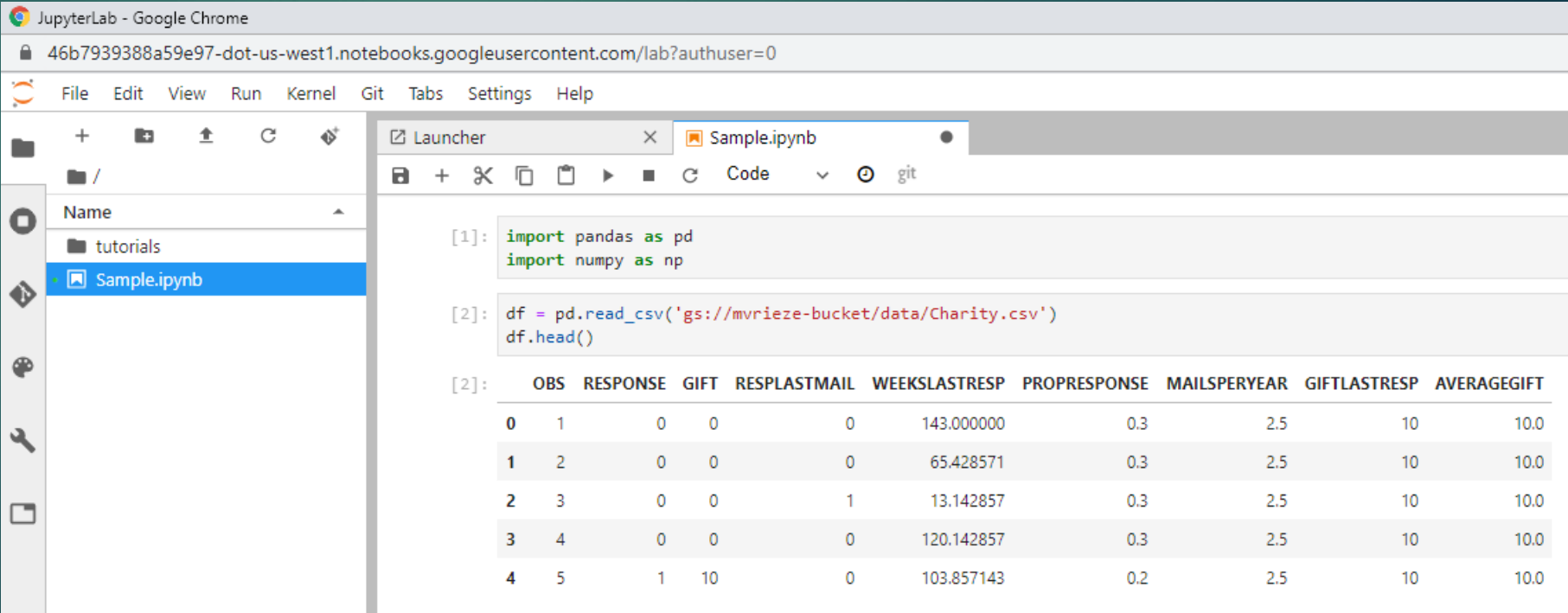
---

# Connect Python to GCP Storage Bucket

<https://stackoverflow.com/questions/49357352/read-csv-from-google-cloud-storage-to-pandas-dataframe>

## Code:

```
# Note how easy it is to connect to a csv file in This is really easy
import pandas as pd
import numpy as np
df = pd.read_csv('gs://bucket-name/folder-name/file-name.csv')
```



The screenshot shows a JupyterLab interface in Google Chrome. The browser address bar displays the URL: `46b7939388a59e97-dot-us-west1.notebooks.googleusercontent.com/lab?authuser=0`. The JupyterLab interface includes a file browser on the left, a top menu bar (File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help), and a central code editor. The code editor has two tabs: 'Launcher' and 'Sample.ipynb'. The 'Sample.ipynb' tab is active, showing two code cells. The first cell contains the imports: `import pandas as pd` and `import numpy as np`. The second cell contains the code to read a CSV file from Google Cloud Storage and display its head: `df = pd.read_csv('gs://mvrieze-bucket/data/Charity.csv')` followed by `df.head()`. The output of the second cell is a table with 10 columns: OBS, RESPONSE, GIFT, RESPLASTMAIL, WEEKSLASTRESP, PROPPRESPONSE, MAILSPERYEAR, GIFTLASTRESP, and AVERAGEGIFT. The table shows 5 rows of data.

	OBS	RESPONSE	GIFT	RESPLASTMAIL	WEEKSLASTRESP	PROPPRESPONSE	MAILSPERYEAR	GIFTLASTRESP	AVERAGEGIFT
0	1	0	0	0	143.000000	0.3	2.5	10	10.0
1	2	0	0	0	65.428571	0.3	2.5	10	10.0
2	3	0	0	1	13.142857	0.3	2.5	10	10.0
3	4	0	0	0	120.142857	0.3	2.5	10	10.0
4	5	1	10	0	103.857143	0.2	2.5	10	10.0





---

# Connect Python to Google BigQuery

---

# Connect Python to Google BigQuery

<https://pypi.org/project/google-cloud-bigquery/>

<https://cloud.google.com/bigquery/docs/datalab-migration>

Code:

```
import numpy as np
import pandas as pd

# Import the library
from google.cloud import bigquery

# Instantiate the client
client = bigquery.Client()

#Write the query - note the ticks surrounding project.dataset.table
sql = """
SELECT * from `firepitmarketing.mvrieze_dataset.Charity`
"""

#Execute the query and set results to Pandas dataframe
df1 = client.query(sql).to_dataframe()
print(df1.head())
```

# Connect Python to Google BigQuery

The screenshot shows a JupyterLab environment running in Google Chrome. The browser address bar displays the URL: `46b7939388a59e97-dot-us-west1.notebooks.googleusercontent.com/lab?authuser=0`. The JupyterLab interface includes a file browser on the left, a code editor in the center, and a console at the bottom.

The file browser shows a directory structure with a folder named `tutorials` and a file named `Sample.ipynb`, which is currently selected.

The code editor displays the following Python code:

```
[6]: import numpy as np
import pandas as pd

# Import the library
from google.cloud import bigquery

# Instantiate the client
client = bigquery.Client()

# Write the query - note the ticks surrounding project.dataset.table
sql = """
SELECT * from `firepitmarketing.mvrieze_dataset.Charity`
"""

# Execute the query and set results to Pandas dataframe
df1 = client.query(sql).to_dataframe()
print(df1.head())
```

The output of the code is displayed in the console, showing the first five rows of the query results:

	OBS	RESPONSE	GIFT	RESPLASTMAIL	WEEKSLASTRESP	PROPRESPONSE	\
0	809	0	0	0	158.0	0.125	
1	810	0	0	0	158.0	0.125	
2	815	0	0	0	158.0	0.125	
3	820	0	0	0	143.0	0.125	
4	893	0	0	0	158.0	0.125	

	MAILSPERYEAR	GIFTLASTRESP	AVERAGEGIFT
0	2.0	50	50.0
1	2.0	50	50.0
2	2.0	50	50.0
3	2.0	50	50.0
4	2.0	71	71.0

The bottom status bar indicates the current mode is `Command`, the cursor is at `Ln 1, Col 1`, and the file is `Sample.ipynb`. The system tray at the bottom shows the date and time as `2:44 PM 1/19/2020`.



---

# Connect R to GCP Storage Bucket

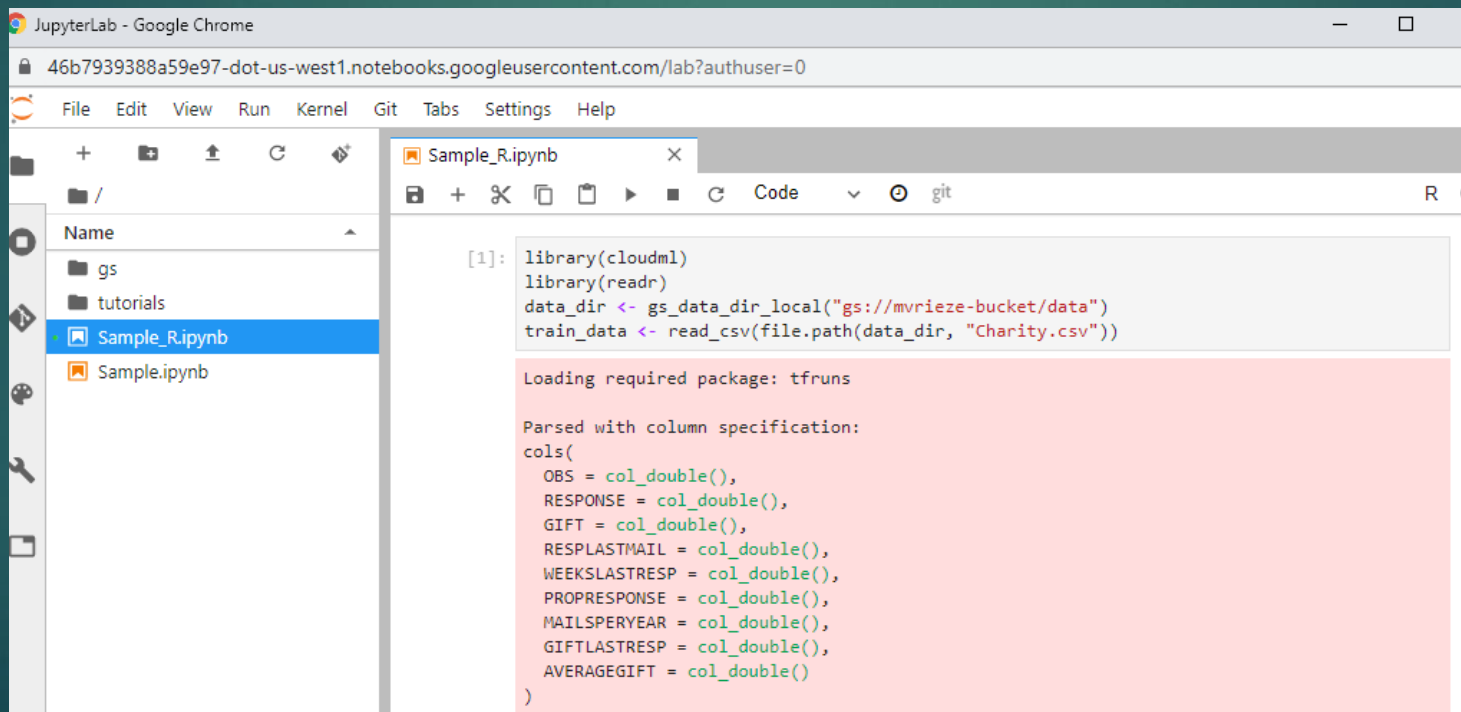
---

# Connect R to GCP Storage Bucket

<https://cran.r-project.org/web/packages/cloudml/vignettes/storage.html>

## Code:

```
library(cloudml)
library(readr)
data_dir <- gs_data_dir_local("gs://bucket/path_in_bucket")
train_data <- read_csv(file.path(data_dir, "filename.csv"))
```



```
[1]: library(cloudml)
library(readr)
data_dir <- gs_data_dir_local("gs://mvrieze-bucket/data")
train_data <- read_csv(file.path(data_dir, "Charity.csv"))

Loading required package: tfruns

Parsed with column specification:
cols(
  OBS = col_double(),
  RESPONSE = col_double(),
  GIFT = col_double(),
  RESPLASTMAIL = col_double(),
  WEEKSLASTRESP = col_double(),
  PROPRESPONSE = col_double(),
  MAILSPERYEAR = col_double(),
  GIFTLASTRESP = col_double(),
  AVERAGEGIFT = col_double()
)
```



---

# Connect R to Google BigQuery

---

# Connect R to Google BigQuery

<https://cloud.google.com/ai-platform/notebooks/docs/use-r-bigquery>

<https://bigrquery.r-dbi.org/>

## Code:

```
# Load the package
```

```
library(bigrquery)
```

```
# Add another notebook block and paste in the following block
```

```
# Provide authentication
```

```
bq_auth(use_oob = TRUE)
```

```
# Add another notebook block
```

```
# Store the project id
```

```
projectid = "firepitmarketing"
```

```
# Set your query - note the ticks around the <project name>.<data set name>.<table name>
```

```
sql <- "SELECT * from `firepitmarketing.mvrieze_dataset.Charity`"
```

```
# Run the query and store the data in a dataframe
```

```
df <- query_exec(sql, projectid, use_legacy_sql = FALSE)
```

```
# Print the query result
```

```
df
```



# Connect R to Google BigQuery

The screenshot shows a JupyterLab environment in Google Chrome. The browser address bar displays the URL: `46b7939388a59e97-dot-us-west1.notebooks.googleusercontent.com/lab?authuser=0`. The JupyterLab interface includes a file browser on the left, a code editor in the center, and a console/output area at the bottom.

The file browser shows a directory structure with folders `gs` and `tutorials`, and files `Sample_R.ipynb` (selected) and `Sample.ipynb`.

The code editor displays the following R code:

```
[2]: # Load the package
library(bigrquery)

# Add another notebook block and paste in the following block
# Provide authentication
bq_auth(use_oob = TRUE)

# Add another notebook block
# Store the project id
projectid = "firepitmarketing"

# Set your query - note the ticks around the <project name>.<data set name>.<table name>
sql <- "SELECT * from `firepitmarketing.mvrieze_dataset.Charity`"

# Run the query and store the data in a dataframe
df <- query_exec(sql, projectid, use_legacy_sql = FALSE)

# Print the query result
df
```

The output area shows a message: "10.0 megabytes processed". Below this, a data frame is displayed with the following structure:

A data.frame: 4268 x 9

OBS	RESPONSE	GIFT	RESPLASTMAIL	WEEKSLASTRESP	PROPRRESPONSE	MAILSPERYEAR	GIFTLASTRESP	AVERAGEGIFT
<int>	<int>	<int>	<int>	<dbl>	<dbl>	<dbl>	<int>	<dbl>
809	0	0	0	158.00000	0.125	2	50	50
810	0	0	0	158.00000	0.125	2	50	50
815	0	0	0	158.00000	0.125	2	50	50

The bottom status bar indicates the current mode is "Command", the cursor is at "Ln 1, Col 1", and the file is "Sample\_R.ipynb". The system tray shows the time as 3:42 PM on 1/19/2020.

# Next Steps

1. Go to: [https://github.com/mvrieze/gcp\\_econometrics](https://github.com/mvrieze/gcp_econometrics)
2. Pull down Sample\_Python.ipynb notebook and import into your GCP notebook environment.
3. Go ahead and run the code blocks:
  - GCP Storage Bucket: Change the <bucket name>.<folder name>.<file name>.csv pointers in the code that pulls in the data.
  - BigQuery: Change the <project>.<dataset name>.<table name> pointers in the code that pulls in the data.