Jupyter Notebook on Google Cloud Compute Engine

PSTAT 135/235 (Winter 2023)

Sang-Yun Oh

Why use the cloud computing?

- Cloud computing provide ondemand computing resources.
- Resources are highly configurable: e.g. CPU, memory, disk, etc.
- Redundant systems make cloud resources reliable
- Monitoring and notification tools
- Flexible user access and management

Google Cloud Shell

Command Line Interface to Google Cloud Platform

- https://console.cloud.google.com is graphical user interface to interact with GCP
- `gcloud` command in Google Cloud Shell let's you interact with GCP on command line
- Command reference: https://cloud.google.com/sdk/gcloud/reference

CLI to Google Cloud Platform

In Cloud Shell,

- View selected project: `gcloud config get project`
- View Storage commands: `gcloud storage --help`
  Note: `h` for keyboard shortcuts, arrows to scroll, `q` to quit
- View Compute commands: `gcloud compute --help`
- Cheatsheet: https://cloud.google.com/sdk/docs/cheatsheet

Change Networking: Firewall Rule

Let's create a firewall rule for later.

- Firewall allow/deny on network ports
- Jupyter notebook server runs on port 8888
- Create firewall rule with name `notebook-server`:

```
gcloud compute firewall-rules \
    create allow-notebook \
    --allow tcp:8888 \
    --target-tags notebook-server
```

Google Cloud Storage

Create a Bucket

- Go to **Cloud Storage > Buckets > Create**
- Give it a name
  for instance `pstat135-xx`, where `xx` are your initials.
  Note: Bucket names are universally unique.
- Allowing public access makes contents public on the internet.
  Note: Everyone in your project has visibility to your bucket, avoid uploading sensitive and private information.

Upload Files

- Cloud storage buckets through the UI work very similar to a Dropbox folder.
- Upload `data/Telco-Customer-Churn.csv` to the bucket you just created.
- Access your file at `https://storage.googleapis.com/<Bucket-Name>/Telco-Customer-Churn.csv`

Google Compute Engine

Compute Instance: Compute Resources

Go to **Compute Engine > VM instances > Create instance**

- Provide a **Name**

Under **Machine Configuration** section

- Select **Region**: e.g., `us-central1`
- Select **Zone**: e.g., `us-central1-a`
- Select **Series**: `E2`
- Select **Machine type**: `e2-micro`

Compute Instance: Operating System

Under **Boot disk** section

- Click **CHANGE**
- **Operating system**: `Container Optimized OS`
- **Version**: (keep selected default)

Compute Instance: Deploy Container

Under **Container** section

- Click **Deploy Container**
- Container image: `jupyter/minimal-notebook`
- Leave everything else as default
- Click **Select**

Compute Instance: Firewall Setting

Under **Advanced options** section

- Expand **Networking**
- Enter `notebook-server` in **Network tags** then enter

Compute Instance: Launch VM

At the bottom of your screen,

- Clicking on **EQUIVALENT COMMAND LINE** shows gcloud command accomplishing the same result
- Click **CREATE** to launch your VM
- Once created, https://console.cloud.google.com/compute/instances will show your VM

Connect to Jupyter Notebook

Connect to your VM

In VM instances, https://console.cloud.google.com/compute/instances,

- Locate your VM and note the **External IP**
- click on **SSH** to launch a shell window
- `docker` command interacts with container
- Recall we deployed `jupyter/minimal-notebook`
  Note: Any image from Jupyter project will work the same way

Check container deployment

- If command `docker ps` is similar to below, container is not ready

```
syoh@instance-1 ~ $ docker ps --format "table {{.Image}}\t{{.Names}}"
IMAGE                                        NAMES
gcr.io/gce-containers/konlet:v.0.11-latest   pensive_golick
```

- If similar to below, Jupyter Lab is ready!

```
syoh@instance-1 ~ $ docker ps --format "table {{.Image}}\t{{.Names}}"
IMAGE                          NAMES
jupyter/minimal-notebook       <YOUR-CONTAINER-NAME>
```

- Note your `<YOUR-CONTAINER-NAME>` for next step

Obtain Jupyter Lab credentials

- By default, Jupyter server secures notebook access with tokens
- Using `<YOUR-CONTAINER-NAME>` from previous step, obtain currently set token inside the container:

```
syoh@instance-1 ~ $ docker exec <YOUR-CONTAINER-NAME> jupyter server list
[JupyterServerListApp] Currently running servers:
[JupyterServerListApp] http://instance-1:8888/?token=16c4bdc18e338792ed6e0609f15c842aa244b5156c6556fe :: /home/jovyan
```

- The long alphanumeric string is the token. Copy it.

Jupyter Lab!!

- Open a browser window and go to `http://<External IP>:8888`
- Under **Setup a Password** section
- Paste your **Token** and choose a **New Password**
- Access your Cloud Storage bucket content

```
jovyan@instance-1:~$ wget https://storage.googleapis.com/<Bucket-Name>/Telco-Customer-Churn.csv
```

- Access class GitHub:

```
jovyan@instance-1:~$ git clone https://github.com/UCSB-PSTAT-135-235/Winter2023
```