

Lab 1: Deploying and using Jupyter Notebooks

In this lab, we'll be using Amazon SageMaker. Amazon SageMaker provides developers and data scientists with the ability to build, train, and deploy machine learning models quickly. Amazon SageMaker is a fully-managed service that covers the entire machine learning workflow to label and prepare your data, choose an algorithm, train the algorithm, tune and optimize it for deployment, make predictions, and take action.

Here's what we hope to accomplish in this lab:

1. Upload a dataset into S3 (Amazon Simple Storage Service)
2. Configure and launch a managed Jupyter notebook with the Amazon SageMaker service
3. Upload a python notebook and use it to analyze the data from S3

Upload Data

To get started, we need first need to upload some data into Amazon S3. S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance. Data is stored as objects within resources called "buckets", and a single object can be up to 5 terabytes in size. S3 features include capabilities to append metadata tags to objects, move and store data across the S3 Storage Classes, configure and enforce data access controls, secure data against unauthorized users, run big data analytics, and monitor data at the object and bucket levels.

S3 is a perfect service to store your datasets. Many of our customers use S3 in their big data workflows, store and share datasets, or even use it to backup data.

Additionally, there are a number of other AWS services that have native support for S3. As you'll see here, we'll grant our Jupyter notebook instance access to S3, making it a perfect place to store our data.



Check Your Region

IMPORTANT!

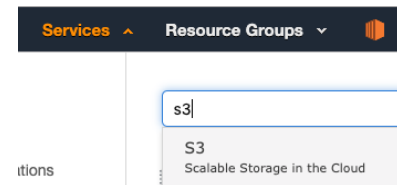
Make sure you are working in the *us-east-2, Ohio* region. You can verify this by looking at the dropdown towards the top right. It should look something like this:



1. Download these files and save them to your desktop or somewhere you can easily find them later:

- [clinical.tsv](#)
- [fpkm.tsv](#)

2. Click the top left menu called **Services**, type **S3** in the search bar, and select **S3**.

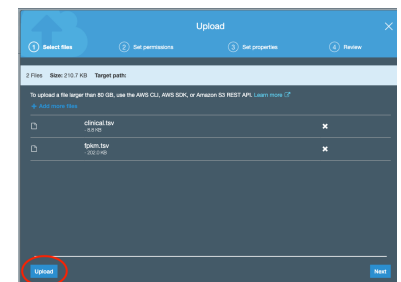


3. Here you will see your list of buckets. There should already be a bucket in your account called **sagemaker-us-east-2-12 digit number**. Click on this bucket.

3a. Open your favorite editor and **note the name of this bucket**. You will need it later on in this lab!

4. Click on the **Upload** button. Drag and drop the files in step 1.

4a. Click the **Upload** button in the bottom left corner as show in the image.



Success!

Nice work! You just uploaded files into your S3 bucket. Now let's launch a Jupyter notebook and make some sense of this data.

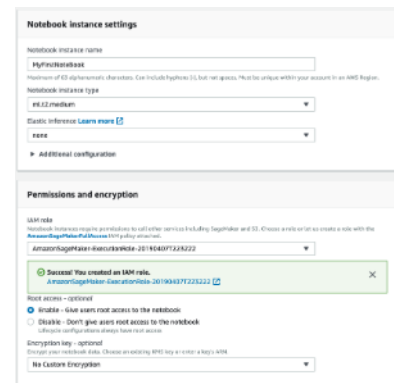
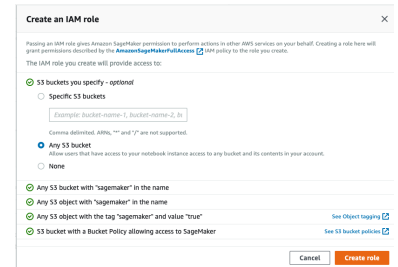
Jupyter Notebook with Amazon SageMaker

Amazon SageMaker includes three modules: Build, Train, and Deploy. The Build module provides a hosted environment to work with your data, experiment with algorithms, and visualize your output. The Train module allows for one-click model training and tuning at high-scale and low cost. The Deploy module provides a managed environment for you to easily host and test models for inference securely and with low latency.

In this lab, we will focus on the Build module. Here, we will launch a Jupyter notebook instance and use it to execute a python notebook against the data we uploaded into S3.

1. From the top left menu called **Services**, type *SageMaker*, and select **Amazon SageMaker**.
2. Double check you're in the **Ohio** region!! You should see **Ohio** towards the top right of your screen. If not, click the dropdown menu and select **US East (Ohio)**.
3. From the left-hand column, under **Notebook**, select **Notebook instances**.
4. Towards the top-right, click the orange button labeled **Create notebook instance**.
5. Configure details for this notebook:

- 5a. Give this notebook a name - something like *MyFirstNotebook*
- 5b. Under **Permissions and encryption**, in the **IAM Role** dropdown, select **Create a new role**
- 5c. In the **Create an IAM role** dialogue window, select the **Any S3 Bucket** radio button and click the **Create role** button
- 5d. Leave all the defaults, scroll to the bottom and click the **Create notebook instance** button



6. You will be directed to a page that will list all of your current notebook instances. Select the one you just created to view the details.
7. Once you see the status change from *Pending* to *InService*, click the **Open Jupyter** button towards the top right of the page.
8. Download this [example notebook](#).
9. In the Jupyter notebook instance, click the **Upload** button towards the top right.
 - 9a. Select the notebook you downloaded in step 8.
 - 9b. Click the **Upload** button.
10. Open the notebook by selecting it in the file viewer.
11. To execute each cell, click the **Run** icon at the top, or simply type *shift + enter*.

- 11a. Pay close attention to the 5th cell as you'll need to replace the value with the name of your S3 bucket that you captured earlier. See image to the right:

```
import boto3

bucket = "CHANGE HERE" # Use the name of your s3 bucket here
clinical_key = "clinical.tsv"
fpkm_key = "fpkm.tsv"

boto3.resource('s3').Bucket(bucket).download_file(clinical_key, 'clinical.tsv')
boto3.resource('s3').Bucket(bucket).download_file(fpkm_key, 'fpkm.tsv')
```

Oops!

Did you see an error like this?

```
boto3.resource('s3').bucket(bucket).download_file(ipkm_key, ipkm.tsv)

~/anaconda3/envs/mxnet_p36/lib/python3.6/site-packages/botocore/hooks.py in _emit(self, event_name, response)
    209     for handler in handlers_to_call:
    210         logger.debug('Event %s: calling handler %s', event_name, handler)
--> 211         response = handler(**kwargs)
    212     responses.append((handler, response))
    213     if stop_on_response and response is not None:

~/anaconda3/envs/mxnet_p36/lib/python3.6/site-packages/botocore/handlers.py in validate_bucket_name
    221     'Invalid bucket name %s'.format(bucket_name))
    222     'the regex %s' % (bucket, VALID_BUCKET_PATTERN))
--> 223     raise ParamValidationError(report=error_msg)
    224
    225

ParamValidationError: Parameter validation failed:
Invalid bucket name "CHANGE ME!!": Bucket name must match the regex "[a-zA-Z0-9.-]_{1,255}"
```

That's because you didn't change the bucket name 😊

Take a look at the previous step again!

Read Carefully!

The notebook contains 3 types of cells:

- A markdown cell which contains useful information about subsequent cells
- A code cell which will execute python code
- An output cell which will display data (if applicable) from the previous code cell's execution

Make sure you read through all of the cells so you have an understanding of what's going on!

Pro Tip

As a cell is being executed, an asterisk will appear in the cell like this:

```
In [*]:
```

While that is there, be patient, things are running in the background. When complete, the asterisk will change to a number - then you can move on!

12. Did you get a graph that looks like this? If so, congrats! Now it's time to clean up after ourselves.

13. Go to **File Close and halt**

14. Click the **Quit** button at the top right of the page.

15. Go back to the Amazon SageMaker service page. This should already be open in a previous window / tab.

15a. Can't find it? No worries. From the **Services** dropdown, search for *SageMaker*, then click on the link.

16. Click **Notebook instances** from the left hand column.

17. Select the radio button next to the notebook instance.

17a. From the **Actions** button dropdown, select **Stop**.

17b. When the status changes from *Stopping* to *Stopped*, from the **Actions** button dropdown, select **Delete**.

18. That's it! Feel free to jump to the [next lab](#).

