AI 620 Emerging Topics in Artificial Intelligence

HOS06A Inference Pipeline in SageMaker

03/14/2023 Developed by Yared Shewarade

09/26/2024 Updated by Anh Nguyen

10/7/2024 Reviewed by Jonathan Koerber

School of Technology and Computing (STC) @City University of Seattle (CityU)

Before You Start

- The directory path shown in screenshots may be different from yours.
- Some steps are not explained in the tutorial**.** If you are not sure what to do:
    1. Consult the resources listed below.
    2. If you cannot solve the problem after a few tries, ask a student worker for help.

Learning Outcomes

Students will be able to learn:

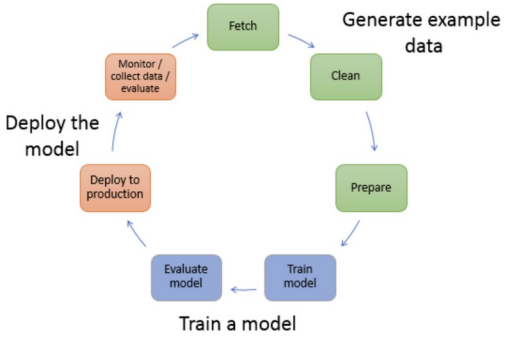
- Introduction to Amazon SageMaker
- Preprocessing big data through Spark EMR

Resources

- Tripuraneni, S., & Song, C. (2019). *Hands-on artificial intelligence on amazon web services: Decrease the time to market for AI and ML applications with the power of AWS* (1st ed.). Packt.

Introduction to Amazon SageMaker

Amazon SageMaker is a fully managed service that enables quick and easy integration of machine learning-based models into applications to generate inferences in real time and at scale. This section provides an overview of machine learning and explains how SageMaker works. If you are a first-time user of SageMaker, we recommend that you read the following sections in order:



a. **Fetch the data**— Pull the dataset or datasets into a single repository.
b. **Clean the data**— Inspect the data and clean it as needed.
c. **Prepare or transform the data**— Perform additional data transformations.
d. **Training the model**— Train a model, you need an algorithm or a pre-trained base model. The algorithm you choose depends on several factors. For a quick, out-of-the-box solution, you might be able to use one of the algorithms that SageMaker provides.
e. **Evaluating the model**—After you have trained your model, you evaluate it to determine whether the accuracy of the inferences is acceptable.

Preprocessing big data through Spark EMR

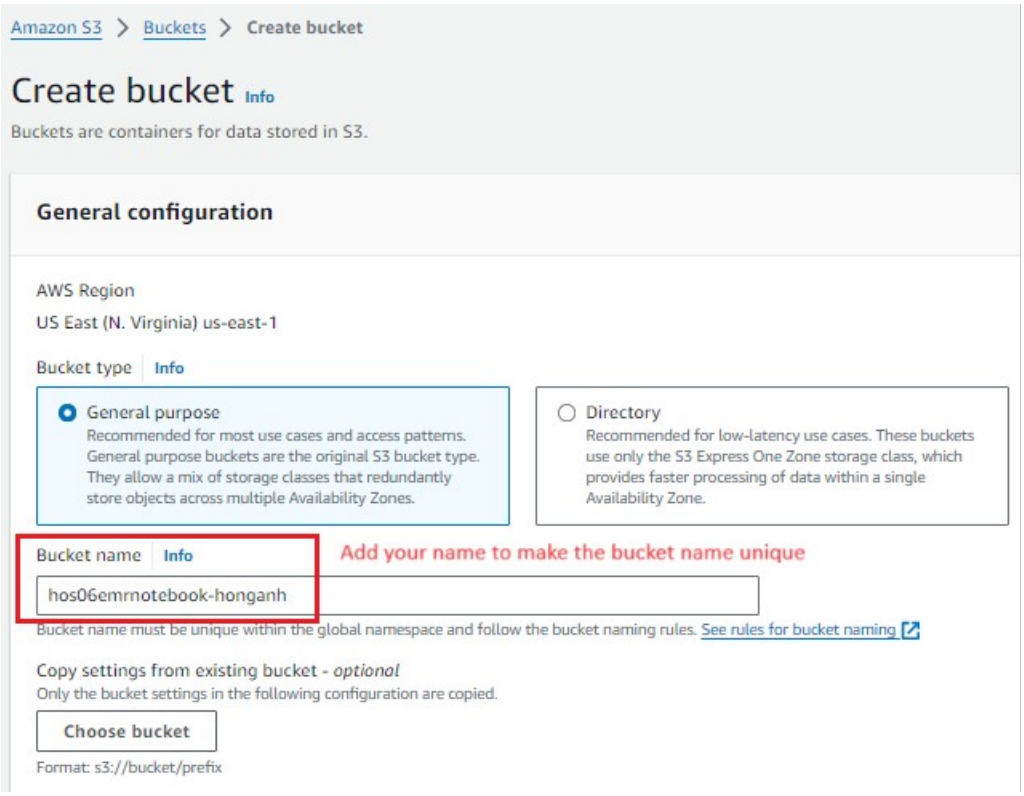
Wrangling a big dataset in Jupyter notebooks results in out-of-memory errors. Our solution is to employ AWS EMR (Elastic MapReduce) clusters to conduct distributed data processing.

Amazon Elastic MapReduce (Amazon EMR) is a web service that makes it easy to process big data. Hadoop will be used as the underlying distributed filesystem while Spark will be used as the distributed computing framework. It provides a managed notebook environment, based on Jupyter Notebook which can be used to interactively wrangle large data, visualize the same, and prepare analytics-ready datasets. These EMR notebooks can also be saved periodically to a persistent data store, S3, so the saved work can be retrieved later.

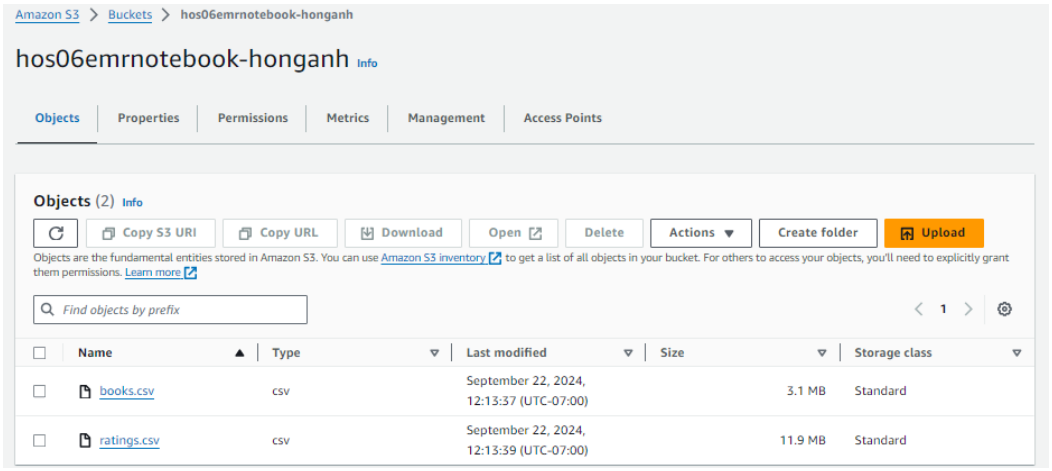
Note: For submission, take the screenshot for all steps and save it in your local repository along with your code.

A. Prepare input data on Amazon S3

1. Sign in to your AWS Management Console. Go to S3 and create a bucket named "**hos06emrnotebook-<yourname>**".



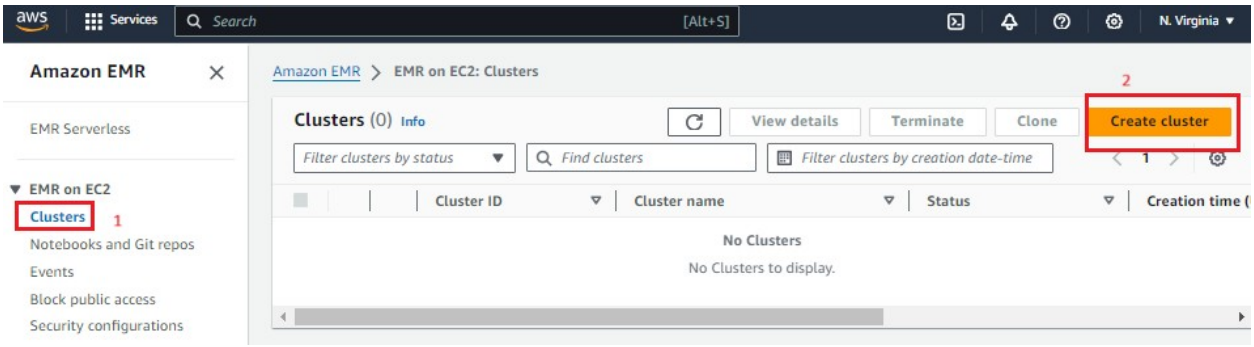
2. Download books.csv and ratings.csv from here and upload it to the above created S3 bucket.



Screenshot Of Bucket. *

B. Launch an Amazon EMR Cluster

1. Under EMR on EC2 in the left navigation pane, choose **Clusters**, and then choose **Create cluster**.

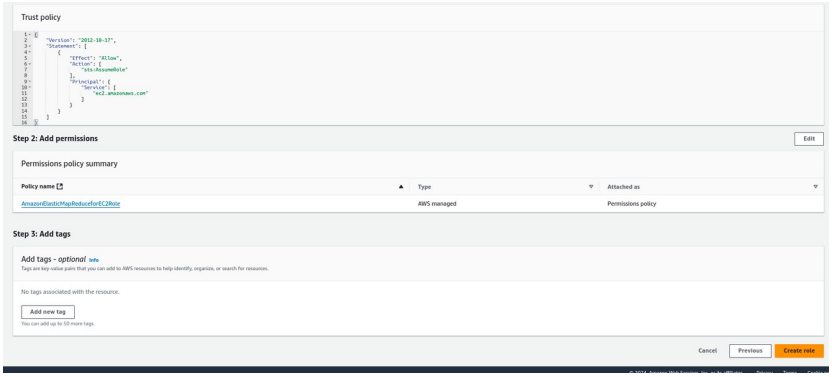


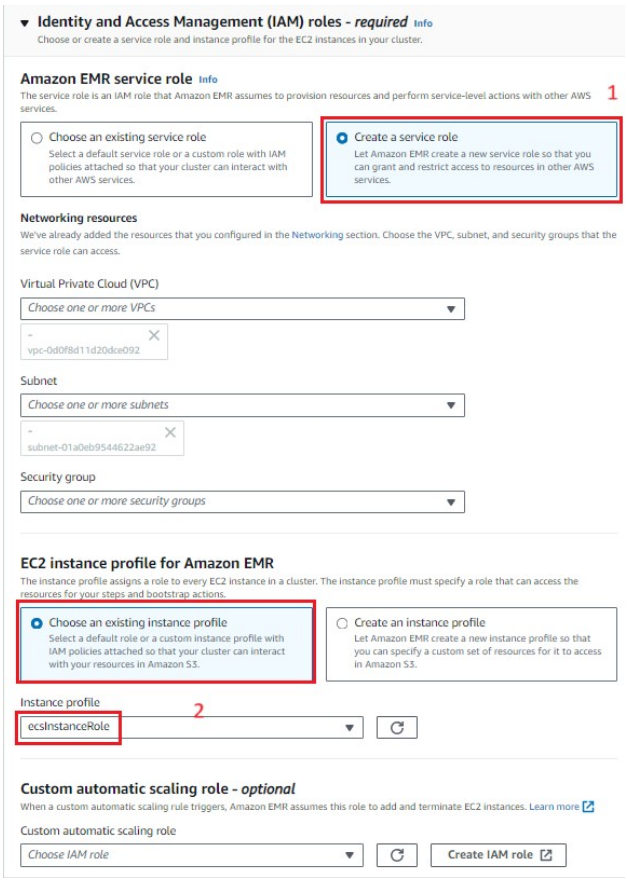
2. Under **Cluster termination and node replacement**, make sure Automatically terminate cluster after idle time is chosen. The idle time should be 1 hour.

3. Under Identity and Access management (IAM) roles:

- Choose Create a service role
- In EC2 instance profile for Amazon EMR:
    - Choose an existing instance profile
    - Instance profile: open the dropdown and choose the created profile "ecsInstanceRole".

**If you have not created an IAM role with EMR default access you can create one. Navigate to the AWS Identity and Access Managment (IAM) by search IAM in the search bar. In the left gutter select Roles. On the left side of the page click create role. This will be for an AWS service so select that on the next screen enter EMR into service or use case you. Select EMR Role for EC2 click next and next again adding AmazonElasticMapReducerforEC2Role permission and name your role. Then create a role.





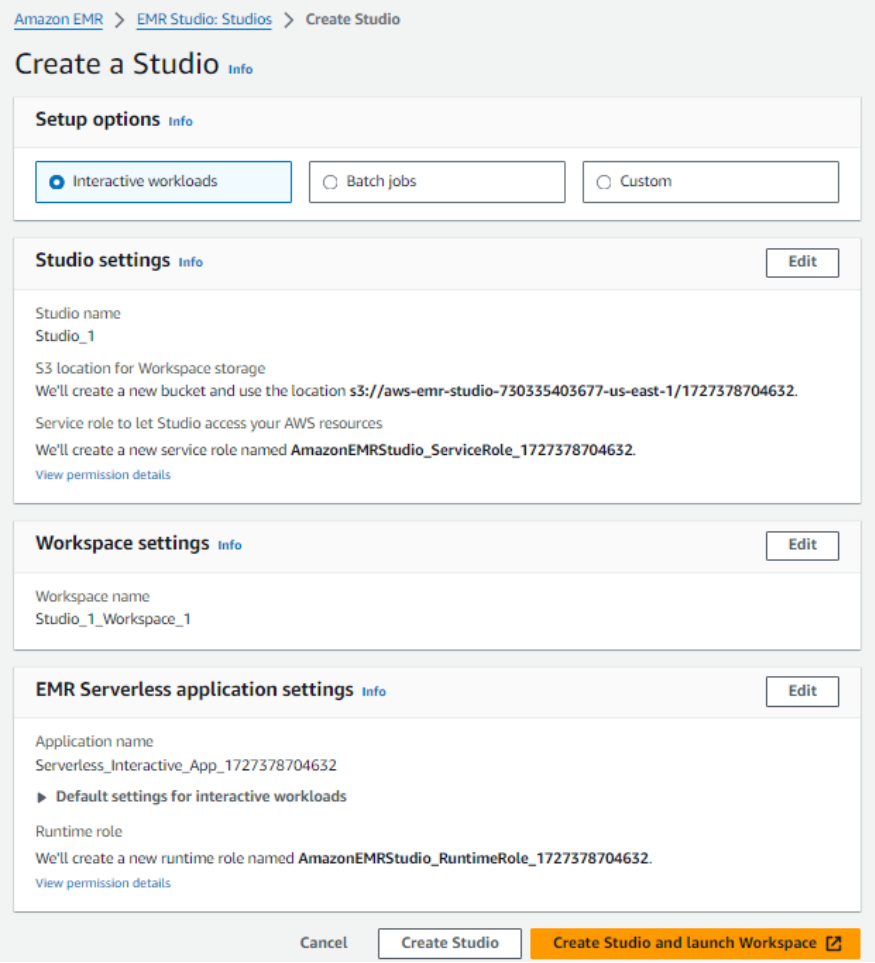
4. Click **the Create Cluster** button. Wait for the status to change from Starting to Running to Waiting. cluster status changes to Waiting when the cluster is up, running, and ready to accept work.
   You may need to choose the refresh icon on the right or refresh your browser to see status updates.

Screenshot of running Cluster

C. Create Workspace (previously called Notebook)

1. Go to your S3 bucket "**hos06emrnotebook-<yourname>**"
2. Go back to Amazon EMR, under EMR on EC2, select Notebooks and Git repos > Go to Workspaces (Notebooks).

Click Create Studio. Leave the configurations as default and click Create studio and launch Workspace.



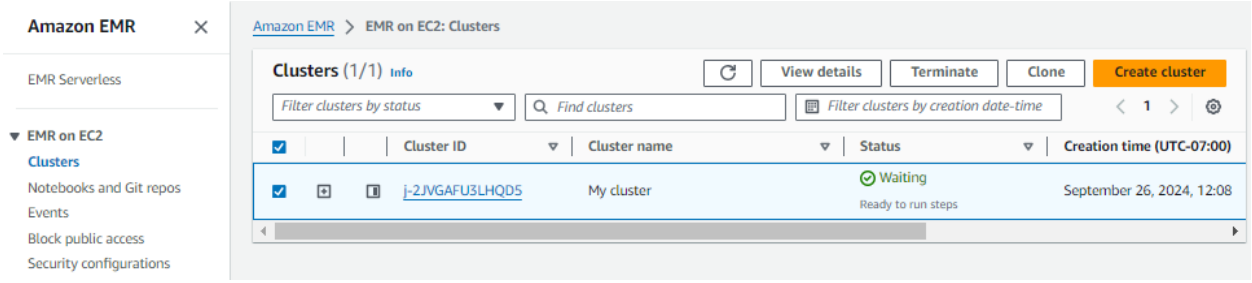
3. Go back to Notebooks and Git repos. You will see that there is a workspace created. Click that workspace > Attach Cluster.

Now, you can use Jupyter Notebook to write Spark code.

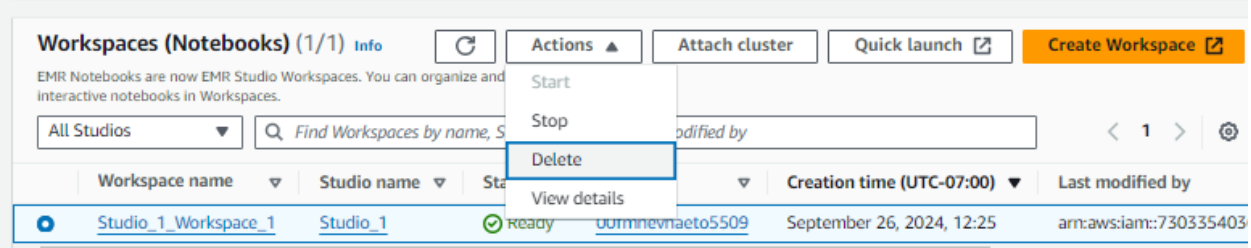
Screenshot of Notebook*

D. Clean up

1. Go to **Clusters**. Choose the clusters you created > Click Terminate to delete the cluster. The status will change from Terminating to Terminated.



2. Go to **Notebooks and Git repos**. Choose the workspace just created. Under the **Action** button, choose Delete. The workspace status will go from Deleting to Deleted, and eventually will be removed from the table.



3. In the search bar, type S3 and go to Amazon S3. You will see that there are several buckets created during this exercise.

- aws-emr-studio*
- Aws-logs*
- Hos06emrnotebook-<yourname>

Empty and delete all these buckets.

HOS submission instructions:

1. Please install the GitHub Desktop: https://cityuseattle.github.io/docs/git/github_desktop/

2. Clone, organize, and submit your work through GitHub Desktop: https://cityuseattle.github.io/docs/hoporhos