Mohammad Shafique

CIS 9760

Project 2 – Analyzing 10Gb of Yelp Reviews Data

The goal of this project is to provision a Spark cluster on AWS EMR, connect it to a Jupyter notebook, and then run a series of queries on it as well as generate visualizations. This will help us analyze and understand the data as well as demonstrate our ability to work with the AWS EMR ecosystem, as well as our ability to leverage transformations and actions via PySpark.

The following are required to build and run this project:

1. An AWS EMR cluster on which we will run the Jupyter notebook.
2. An S3 bucker where we will store our Yelp data.
3. Uploading the Jupyter notebook.

Create the AWS EMR cluster as follows:

1. Go to the AWS console and search for EMR.
2. Click create cluster.
3. Click Go to advanced options.
4. For release version select emr-5.31.0.
5. Select Livy and Spark. Unselect Pig.
6. Click next.
7. Double check the EC2 subnet to make sure it is correct.
8. Double check to make sure the instance types are m5.xlarge.
9. Click next.
10. Name the cluster.
11. Click next.
12. For EC2 key pair keep the default option.
13. Click Create cluster.

Create the S3 bucket and upload your data as follows:

1. Go to the AWS console and search for S3.
2. Click Create bucket.
3. Give your bucket a name and click Create bucket.
4. Click on your bucket and click Upload.
5. Add your files and click Upload.

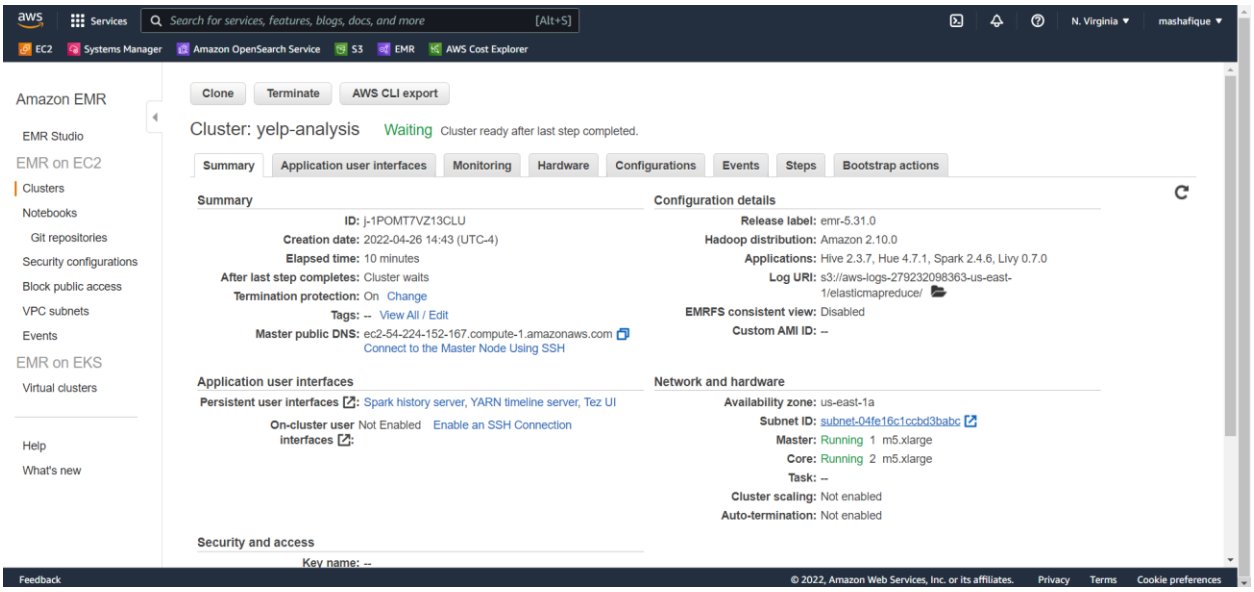
Upload Jupyter notebook as follows:

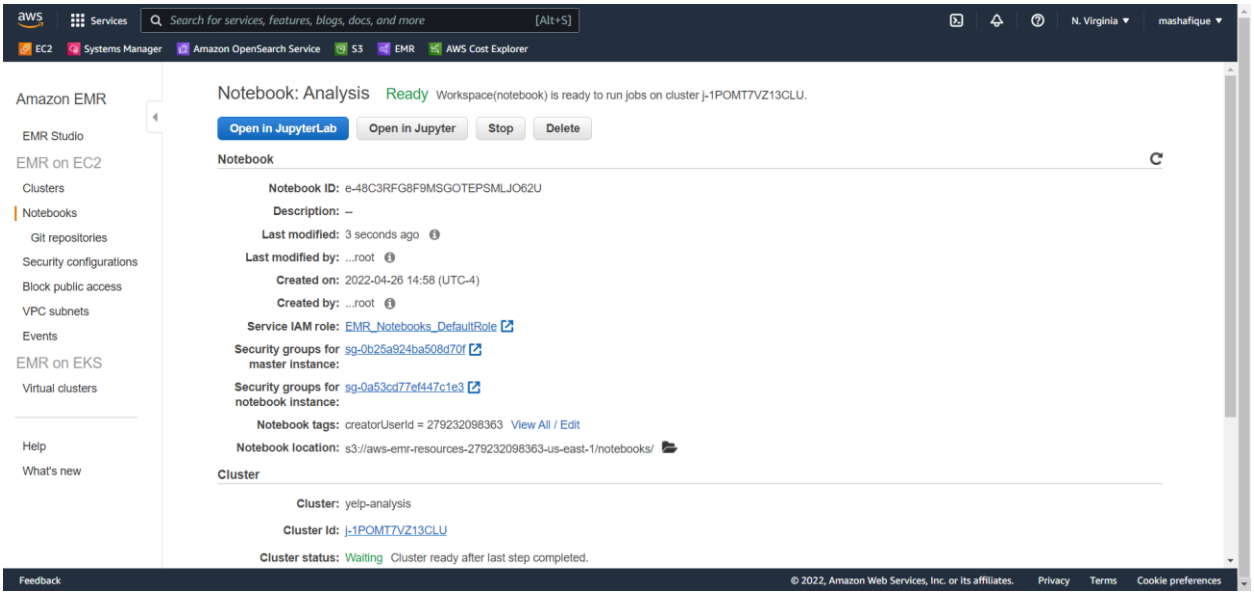
Reference: https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-managed-notebooks-working-with.html

1. Before proceeding, close the notebook editor for any notebooks that you will work with, and then stop the notebook if it's an EMR notebook.

2. Create an EMR notebook and enter a name for it. The name that you enter for the notebook will be the name of the file you need to replace. The new file name must match this file name exactly.
3. Make a note of the location in Amazon S3 that you choose for the notebook. The file that you replace is in a folder with a path and file name like the following pattern: s3://MyNotebookLocation/NotebookID/MyNotebookName.ipynb.
4. Stop the notebook.
5. Replace the old notebook file in the Amazon S3 location with the new one, using the same name.

Below are images of my cluster and notebook configuration.





In this project we begin by installing and importing matplotlib, pandas, pip, and seaborn as well as importing col, explode, split, mean, and length from pyspark.sql.functions.

We then read in our data from our S3 bucket using spark.read.json().

We then using printSchema() to get an understanding of our business data and then move on to cleaning the data by filtering for nulls and adding a column named business_stars which is the same as stars but will be helpful later when we join tables.

We then explode and split the categories of our businesses and take a count of the top 20 categories. The top three categories were restaurants, food, and shopping.

We then create a bar plot to visualize the top 20 categories.

We load in our review data and then clean it by filtering for nulls as well as adding a column called review_stars which is the same as the stars column but will be helpful later when we join tables.

We create a column for the average stars per business for the reviewers that took the time to write a written review.

We join this with our clean business data and calculate the skew. We then plot the skew on a Seaborn distplot and see that we have slightly more reviews with negative skew.

We then want to study if elite users should be trusted. We clean and filter our users table for elite users.

We find the elite users whose review usefulness count is more than 650 since that was the average number of usefulness count for the reviews.

We join our business, review, and user tables.

We create two tables, one for all elite users and one for elite users with more than 650 usefulness count on their reviews. In each of the tables we take the difference between the business rating and the user rating. We then create a Seaborn countplot for each of these tables and find that most of the reviews have a rating difference between -1.0 and 0.5.

This indicates that the elite users can be trusted provided they have a sufficient review usefulness count.