Project 2 – X Hashtag Counter using Amazon EMR

Due Date: Monday, March 25 at 11:59 PM

Description:

A political campaign team for a city council seat needs to get a sense of current important issues for the constituents of the city. One way of getting this important information is through analyzing social media feeds. For example, by getting the most frequently hashtag words in X(twitter). In this project, you are going to analyze a file that contains tweets and give the top 20 hashtags in the collection.

Input Data

Data uploaded on an S3 bucket on AWS. The name of the bucket on AWS will be available soon.

Processing

You are allowed to choose any big data processing tool you are familiar with. Hadoop, Hive, Presto, Spark, etc., are some of the tools you can use. You can also write your own parallel processing code (for example mapreduce) using a programming language of your choice.

Your processing tool should be deployed on Amazon EMR and the processing should be done using EMR cluster. Please refer to **Launching EMR on AWS** at the end of this document on how to launch an EMR cluster on AWS.

Restriction

Due to cost issues (EMR is expensive), you **MUST** follow the following rules strictly. **Failure to follow the rules will result in some points deductions from your grade**.

- Only one EMR cluster per group should be created at a time. Name your resources as p2-[your_groupNumber].
- You have to follow the "Launch EMR on AWS" steps strictly.
- The EMR cluster should be transient (take a look at 6.(b) on the EMR launch steps).

Before you deploy your solution on AWS, work on your solution using the tool of your choice on a smaller dataset on your local machine. Once you are sure your solution works for a small data set, we will give you the authorization to create your EMR cluster on AWS. Please make sure you have a working solution before you ask for the AWS privilege.

Dataset:

A sample small dataset is uploaded on Canvas for testing your solution on your local machine. A larger dataset for the EMR will be available on AWS. The name of the S3 bucket will be posted on Piazza soon.

Submission:

Your submission has two parts:

Canvas:

- Submission is via Canvas. One submission per group.
- A zip folder that contains:
    - The pdf file that contains the 20 top hashtags together with the corresponding number of occurrences of the hashtags
    - The original output file of your solution – this is the file that will be outputted by your solution
    - A short description – 2/3 paragraphs that describes your data processing steps. This includes a short description of the tool you used, why you used it, the steps you followed to process data, etc.
    - Your data processing code

AWS

- We will create a folder on a bucket on AWS (we will post the name of the folder for each group on piazza soon) and let your output (the top 20 hashtags together with the number of occurrences for each hashtag) written to the folder.

Launching EMR on AWS

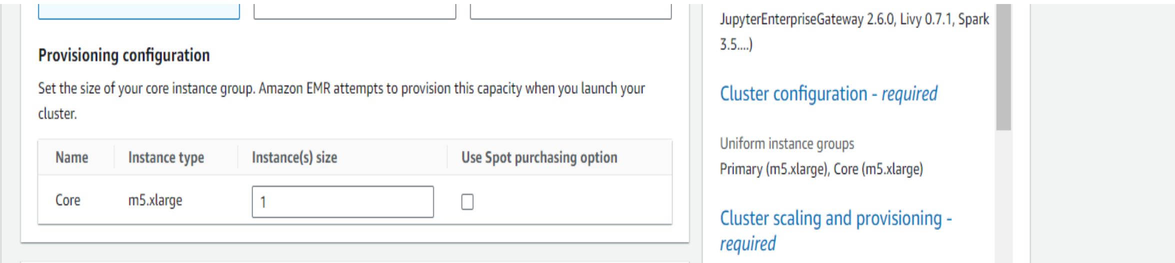
Login to your AWS account management console

1. In the search box to the right of **Services**, search for and choose **EMR**.
2. Launch the process to create an EMR cluster.
    - Choose **Create cluster**.
3. Configure the options for Step 1: Software and Steps.
    - In the **Name and applications** section set Name to `p2-[your_groupNumber]`.
    - For **Amazon EMR release**, choose **emr-7.0.0**.
    - **Ensure that the application(s)/tool(s) for your processing is/are are selected**
        - Clear (*deselect*) all other selected applications

Analysis: The Amazon EMR release that you choose determines the version of Hadoop and other tools that will be installed on the cluster. You can also install many other Hadoop-related projects, such as Hadoop User Experience (Hue), Pig, and Spark.

4. Configure the Hardware.

- In the **Cluster configuration** section, set the instance type and number of nodes to use:
  - Select “uniform instance groups”
  - For the primary node choose **m5.xlarge** from the list
  - Repeat the same process for the Core node type.
  - Remove the Task (Task 1 of 1) instance group
  - Use the default “EBS root volume”
  - Verify that under **Cluster scaling and provisioning** the core **Instance size** is set to 1.
  - Verify that the instance counts are shown as follows in the Summary pane - Core size: **1 instance**.



Provisioning configuration

Set the size of your core instance group. Amazon EMR attempts to provision this capacity when you launch your cluster.

Name	Instance type	Instance(s) size	Use Spot purchasing option
Core	m5.xlarge	1	□

JupyterEnterpriseGateway 2.6.0, Livy 0.7.1, Spark 3.5....)

Cluster configuration - *required*

Uniform instance groups
Primary (m5.xlarge), Core (m5.xlarge)

Cluster scaling and provisioning - *required*

Analysis: The main node coordinates jobs that will run on the cluster. The main node runs the HDFS NameNode as well as the YARN ResourceManager. The core nodes act as HDFS DataNodes and are where HDFS data is replicated and stored on disk. These nodes also run MapReduce tasks as directed by YARN. Task nodes are an option that can support parallelization and Spot Instance types, but you won't need any for this lab. For more information about HDFS, YARN, MapReduce, and other Hadoop topics, see the Apache Hadoop website.

- In the **Networking** section:
  - For **Network**, choose **default VPC**.
  - For **EC2 Subnet**, choose **default subnet** .

5. Configure the options for Step 3: **Cluster termination and node replacement**-

   a) For **Cluster termination** clear (*deselect*) the **Use termination protection** option.

   b) Leave the “Automatically terminate cluster after idle time (Recommended)” selected with 1 hour idle termination

   c) Cluster Logs

   Deselect the **“Publish cluster-specific logs to Amazon S3**” checkbox.

6. Configure the options for Step 4: Security configuration and EC2 key pair - optional.

    - For **Amazon EC2 key pair for SSH to the cluster**, choose your key pair.

      Note: You will download this key pair and use it to connect to the cluster later (if there is a need).

    - Leave **Choose an existing service role** selected.
    - For **Service role**, confirm that **p2-emr-tesse** is chosen. This will allows Elastic MapReduce to call AWS services such as EC2 on your behalf.
    - Under **EC2 instance profile for Amazon EMR** and **Instance profile**, confirm that **p2-emrec2-tesse** is chosen. This allows EC2 instances in an Elastic MapReduce cluster to call AWS services such as S3 on your behalf.

7. To finish creating the cluster, choose **Create cluster**.

   Your cluster will now be provisioned. Don't wait for the provisioning process to complete before continuing to the next step.

   Tip: You can ignore the warning that says *Auto-termination is not available for this account when using this release of EMR*.

8. Configure the security group for the main node to allow SSH connections.

    - While you are on the cluster's **Summary** tab, go to the **Network and security** section below.
    - Expand **EC2 security groups (firewall)** to see the security group for the main node, choose the link for the security group. If you don't see the link yet, refresh the page. It might take a minute or two to appear.

      A new tab opens to the Security Groups page in the Amazon EC2 console.

    - Select the security group for the main node.

      Tip: You might need to expand the **Security group name** column to see the full name.

    - In the bottom pane, choose the **Inbound rules** tab, and then choose **Edit inbound rules**.
    - At the bottom of the page, choose **Add rule**, and then configure SSH access for the AWS Cloud9 instance that has been created for you:
        - **Type:** Choose **SSH**.

        - **Source:** Choose **Anywhere-IPv4**.
    - Choose **Save rules**.

9. Confirm that the cluster is now available.

    - Return to the Amazon EMR console, which should still be open in another browser tab.
    - Refresh the page.
    - In the **Instances (Hardware)** section, verify that the status for the primary and core node types is *Running*.

      Tip: It might take up to 10 minutes since you created the cluster for it to finish provisioning. Refresh the page to update the status.

      Important: Make sure the status of the cluster shows *Waiting* and the status of the nodes show *Running*.