

Big Data Wrangling With Google Books Ngrams

Let's start with questions which can be answered with screenshots:

1. Spin up a new EMR cluster using the AWS Console. Be sure to include Hadoop, Hive, Spark, and Jupyterhub for your cluster.

Cluster - Advanced Options [Go to quick options](#)

Cluster Steps

Cluster Settings

Software Configuration

Release emr-5.30.1 ⓘ

<input checked="" type="checkbox"/> Hadoop 2.8.5	<input type="checkbox"/> Zeppelin 0.8.2	<input type="checkbox"/> Livy 0.7.0
<input checked="" type="checkbox"/> JupyterHub 1.1.0	<input type="checkbox"/> Tez 0.9.2	<input type="checkbox"/> Flink 1.10.0
<input type="checkbox"/> Ganglia 3.7.2	<input type="checkbox"/> HBase 1.4.13	<input checked="" type="checkbox"/> Pig 0.17.0
<input checked="" type="checkbox"/> Hive 2.3.6	<input type="checkbox"/> Presto 0.232	<input type="checkbox"/> ZooKeeper 3.4.14
<input type="checkbox"/> MXNet 1.5.1	<input type="checkbox"/> Sqoop 1.4.7	<input type="checkbox"/> Mahout 0.13.0
<input checked="" type="checkbox"/> Hue 4.6.0	<input type="checkbox"/> Phoenix 4.14.3	<input type="checkbox"/> Oozie 5.2.0
<input checked="" type="checkbox"/> Spark 2.4.5	<input type="checkbox"/> HCatalog 2.3.6	<input type="checkbox"/> TensorFlow 1.14.0

Multiple master nodes (optional)

☐ Use multiple master nodes to improve cluster availability. [Learn more](#) ⓘ

AWS Glue Data Catalog settings (optional)

- Here, we have started an EMR cluster with the necessary libraries.

2. Connect to the head node of the cluster using SSH

```
hadoop@ip-172-31-3-218:~$ ssh -i hadoop.pem ec2-user@ip-172-31-3-218
https://aws.amazon.com/amazon-linux-2/
33 package(s) needed for security, out of 89 available

  _I_  _I_  )
 _I_  _I_  /
  _I_  _I_  |
      Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
33 package(s) needed for security, out of 89 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM          MMMMMMMM RRRRRRRRRRRRRRRR
E::::::::::::::::::::E M:::::M          M:::::M R:::::R
EE::::::::::::::::::::E M:::::M          M:::::M R:::::RRRRRR:::::R
E::::E      EEEEE M:::::M          M:::::M RR:::R      R:::R
E::::E      M:::::M:M::M M::M:M::M R:::R      R:::R
E:::::EEEEEEEEEE M:::::M M:::M M:::M M:::::M R:::RRRRRR:::::R
E::::::::::::::::::::E M:::::M M:::M:M::M M:::::M R:::::RR
E:::::EEEEEEEEEE M:::::M M:::::M M:::::M R:::RRRRRR:::::R
E::::E      M:::::M M:::M M:::::M R:::R      R:::R
E::::E      EEEEE M:::::M MMM M:::::M R:::R      R:::R
EE::::::::::::::::::::E M:::::M          M:::::M R:::R      R:::R
E::::::::::::::::::::E M:::::M          M:::::M RR:::R      R:::R
EEEEEEEEEEEEEEEEEEEE MMMMMMMM          MMMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-3-218 ~]$
```

- After initiating the SSH connection we are in the Hadoop directory with EMR.

3. Copy the data folder from the S3 bucket into the /user/hadoop/eng_1M_1gram directory in the Hadoop file system (HDFS) using distcp. The full path for the 1-gram data directory is s3://brainstation-dsft/eng_1M_1gram.csv

- By using hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/hadoop/eng_1M_1gram we copied the data from S3 bucket to HDFS.

```
The general command line syntax is
command [genericOptions] [commandOptions]

[hadoop@ip-172-31-3-218 ~]$ hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/hadoop/eng_1M_1gram
20/09/02 01:27:10 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissi
lse, ignoreFailures=false, overwrite=false, skipCRC=false, blocking=true, numListStatusThreads=0, maxMaps=20, mapBa
th=100, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=false, atomic
ath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://brainstation-dsft/eng_1M_1gram.csv], targetPath=/
hadoop/eng_1M_1gram, targetPathExists=false, filtersFile='null'}
20/09/02 01:27:10 INFO client.RMPProxy: Connecting to ResourceManager at ip-172-31-3-218.ca-central-1.compute.intern
2.31.3.218:8032
20/09/02 01:27:13 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirCnt = 0
20/09/02 01:27:13 INFO tools.SimpleCopyListing: Build file listing completed.
20/09/02 01:27:13 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
20/09/02 01:27:13 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort
or
20/09/02 01:27:13 INFO tools.DistCp: Number of paths in the copy list: 1
20/09/02 01:27:13 INFO tools.DistCp: Number of paths in the copy list: 1
20/09/02 01:27:14 INFO client.RMPProxy: Connecting to ResourceManager at ip-172-31-3-218.ca-central-1.compute.intern
2.31.3.218:8032
20/09/02 01:27:14 INFO mapreduce.JobSubmitter: number of splits:1
20/09/02 01:27:14 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1599009777952_0001
20/09/02 01:27:14 INFO impl.YarnClientImpl: Submitted application application_1599009777952_0001
20/09/02 01:27:14 INFO mapreduce.Job: The url to track the job: http://ip-172-31-3-218.ca-central-1.compute.interna
88/proxy/application_1599009777952_0001/
20/09/02 01:27:14 INFO tools.DistCp: DistCp job-id: job_1599009777952_0001
20/09/02 01:27:14 INFO mapreduce.Job: Running job: job_1599009777952_0001
20/09/02 01:27:20 INFO mapreduce.Job: Job job_1599009777952_0001 running in uber mode : false
20/09/02 01:27:20 INFO mapreduce.Job: map 0% reduce 0%
```

The above screenshot shows that the command is running a job and the following screenshot shows that the file is now in our HDFS.

```
Total megabyte-milliseconds taken by all map tasks=59817984
Map-Reduce Framework
  Map input records=1
  Map output records=0
  Input split bytes=136
  Spilled Records=0
  Failed Shuffles=0
  Merged Map outputs=0
  GC time elapsed (ms)=175
  CPU time spent (ms)=63150
  Physical memory (bytes) snapshot=1036853248
  Virtual memory (bytes) snapshot=4632616960
  Total committed heap usage (bytes)=885522432
File Input Format Counters
  Bytes Read=224
File Output Format Counters
  Bytes Written=0
DistCp Counters
  Bytes Copied=5292105197
  Bytes Expected=5292105197
  Files Copied=1
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls
Found 1 items
-rw-r--r--  1 hadoop hadoop 5292105197 2020-09-02 01:28 eng_1M_1gram
[hadoop@ip-172-31-3-218 ~]$
```

4. Write the filtered data back to a directory in the Hadoop filesystem from Spark using `df.write.csv()` - e.g. `/user/hadoop/eng_data_1gram`. Be sure to pass the `header=True` parameter. Examine the contents of what you've written using `hadoop fs -ls`.

- Using

`filtered.write.option("header","true").csv("/user/hadoop/eng_data_1gram")`

We wrote the filtered data in the HDFS with a file name 'eng_data_1gram'

We can check the contents with `hdfs fs -ls`:

```
Bytes Copied=5292105197
Bytes Expected=5292105197
Files Copied=1
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls
Found 1 items
-rw-r--r--  1 hadoop hadoop 5292105197 2020-09-02 01:28 eng_1M_1gram
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls
Found 2 items
-rw-r--r--  1 hadoop hadoop 5292105197 2020-09-02 01:28 eng_1M_1gram
drwxr-xr-x  - livy hadoop 0 2020-09-02 02:15 eng_data_1gram
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -rm -r eng_data_1gram
Deleted eng_data_1gram
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls
Found 1 items
-rw-r--r--  1 hadoop hadoop 5292105197 2020-09-02 01:28 eng_1M_1gram
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls
Found 2 items
-rw-r--r--  1 hadoop hadoop 5292105197 2020-09-02 01:28 eng_1M_1gram
drwxr-xr-x  - livy hadoop 0 2020-09-02 02:15 eng_data_1gram
[hadoop@ip-172-31-3-218 ~]$ hadoop fs -ls eng_data_1gram
Found 3 items
-rw-r--r--  1 livy hadoop 0 2020-09-02 02:15 eng_data_1gram/_SUCCESS
-rw-r--r--  1 livy hadoop 0 2020-09-02 02:14 eng_data_1gram/part-00000-0cf66a54-9604-46c7-8a45-b80538ca8300-c000.csv
-rw-r--r--  1 livy hadoop 7305 2020-09-02 02:14 eng_data_1gram/part-00023-0cf66a54-9604-46c7-8a45-b80538ca8300-c000.csv
[hadoop@ip-172-31-3-218 ~]$
```

After running `hdfs fs -ls` again we see that the 'filtered' file with the token as 'data' was written in our HDFS under the name 'eng_data_1gram.'

5. Collect the contents of the directory into a single file on the local drive of the head node using `getmerge`: e.g. `hadoop fs -getmerge /user/hadoop/eng_data_1gram eng_data_1gram.csv`. Move this file into a S3 bucket in your account using `aws s3 cp`.
 - The code- `hadoop fs -getmerge /user/hadoop/eng_data_1gram eng_data_1gram.csv` merges the dataset from the HDFS into the local directory.

```
hadoop@ip-172-31-7-235 ~]$ hadoop fs -ls eng
ls: `eng': No such file or directory
hadoop@ip-172-31-7-235 ~]$ hadoop fs -ls eng_data_1gram
Found 1 items
drwxr-xr-x  - livy hadoop          0 2020-09-02 19:32 eng_data_1gram/_temporary
hadoop@ip-172-31-7-235 ~]$ hadoop fs -ls eng_data_1gram/temporary
ls: `eng_data_1gram/temporary': No such file or directory
hadoop@ip-172-31-7-235 ~]$ hadoop fs -ls eng_data_1gram/_temporary
ls: `eng_data_1gram/_temporary': No such file or directory
hadoop@ip-172-31-7-235 ~]$ hadoop fs -ls eng_data_1gram/
Found 3 items
-rw-r--r--  1 livy hadoop          0 2020-09-02 19:34 eng_data_1gram/_SUCCESS
-rw-r--r--  1 livy hadoop          0 2020-09-02 19:33 eng_data_1gram/part-00000-08dfdf00-7de3-47dc-a72d-c5d4c1a9a82b-c000.
-rw-r--r--  1 livy hadoop      7305 2020-09-02 19:33 eng_data_1gram/part-00023-08dfdf00-7de3-47dc-a72d-c5d4c1a9a82b-c000.
hadoop@ip-172-31-7-235 ~]$ ls
hadoop@ip-172-31-7-235 ~]$ hadoop fs -getmerge /user/hadoop/eng_data_1gram eng_data_1gram.csv
hadoop@ip-172-31-7-235 ~]$ ls
eng_data_1gram.csv
hadoop@ip-172-31-7-235 ~]$ aws s3 ls
2020-08-28 14:35:39 aws-emr-resources-523306340043-ca-central-1
2020-08-27 15:20:33 aws-logs-523306340043-ca-central-1
2020-08-26 19:14:36 royalknight
hadoop@ip-172-31-7-235 ~]$ aws s3 cp eng_data_1gram.csv s3://royalknight
upload: ./eng_data_1gram.csv to s3://royalknight/eng_data_1gram.csv
hadoop@ip-172-31-7-235 ~]$ aws s3 ls royalknight
2020-09-02 19:36:36      7305 eng_data_1gram.csv
hadoop@ip-172-31-7-235 ~]$
```

The `ls` command shows the contents in the local directory which can be seen that the data file is now in the local directory.

Also using `aws s3 cp eng_data_1gram.csv s3://royalknight` copied the data file into my s3 bucket 'royalknight' which can be confirmed with `aws s3 ls royalknight` which shows the contents of the data in the s3 bucket.

The rest of the sub questions of the 4th is in the 1gram.ipynb notebook.

And the 6th question is in the Plot of 1gram.ipynb.