

# Big Data Wrangling with Google Books Ngrams

Author: Darius Smith

Date: March 29, 2023

School: BrainStation

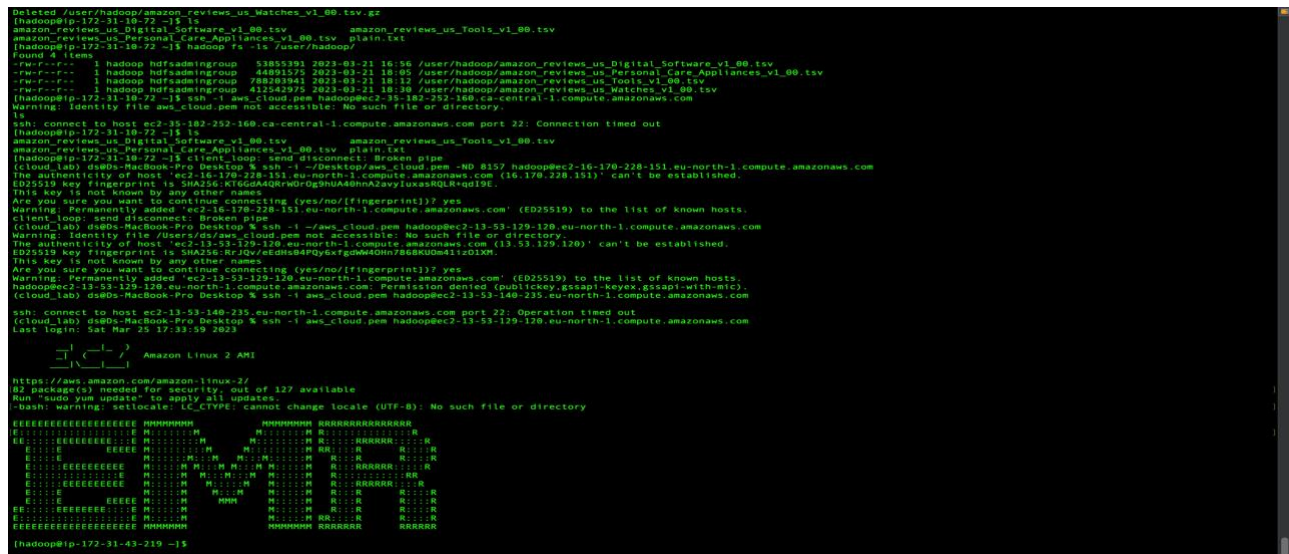
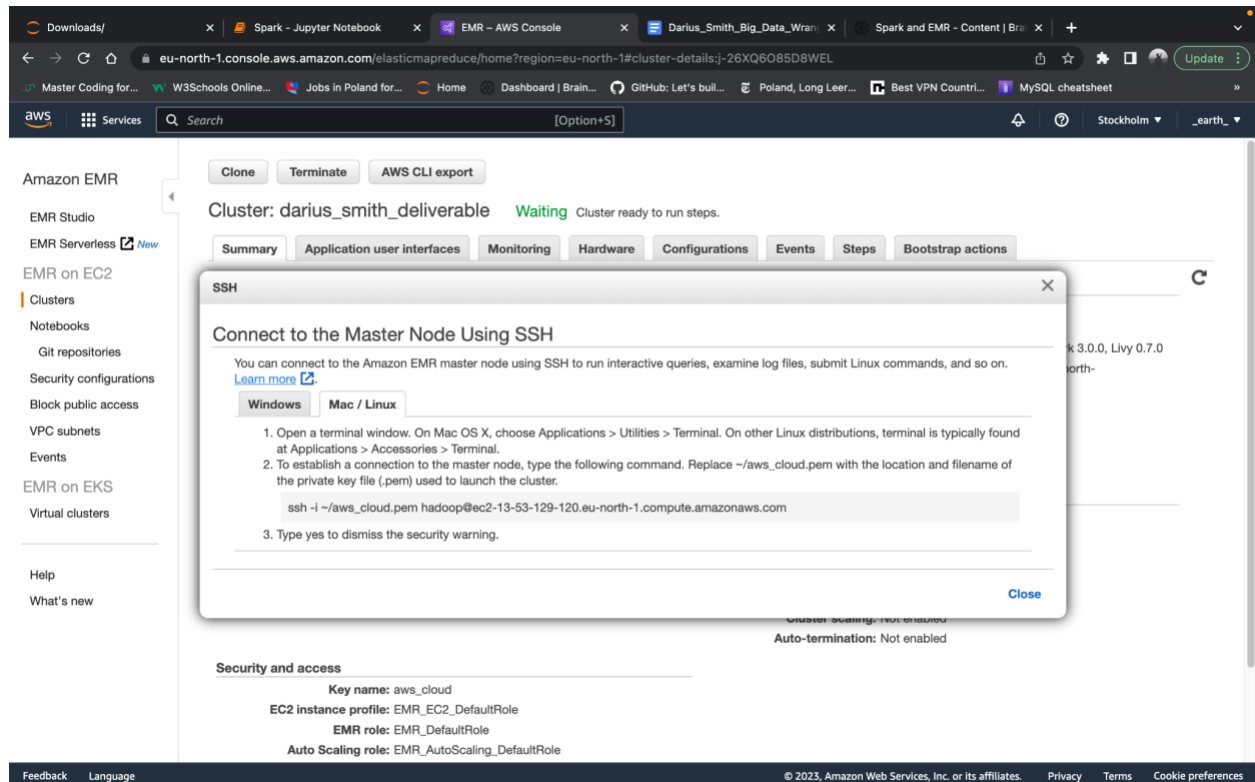
The purpose of this document is to serve as a tutorial for application of Big Data Fundamentals. This document will include a jupyter notebook which will follow a Big Data analysis workflow using a real-world dataset in a cloud-based distributed computing environment. The following were used in this tutorial, Hadoop, Spark, Hive, and the S3 filesystem.

**Step 1:** Log into AWS. When you have access to your account click on the 'EMR' icon. Spin up a new EMR cluster using the AWS Console. Go to 'Create Cluster - Advanced Options.' Be sure to include Hadoop, Spark, Hive, Jupyterhub, and Livy for your cluster. **For the release version, make sure to use EMR 6.1.1.** Click through all steps until 'Step 4.' When you get to this step, be sure to have an EC2 key pair. **Create cluster.** (Reference pictures are below.)

The screenshot shows the AWS Management Console interface for creating an EMR cluster. The browser tabs include 'Spark - Jupyter Notebook', 'EMR - AWS Console', and 'Darius\_Smith\_Big\_Data\_Wran...'. The URL is 'eu-north-1.console.aws.amazon.com/elasticmapreduce/home?region=eu-north-1#create-cluster'. A notification banner at the top states: 'The new EMR console will become the default console starting Mar 27, 2023. Switch to the new console. If you want, you can still switch back. Learn more.' The page title is 'Create Cluster - Advanced Options' with a 'Go to quick options' link. On the left, a sidebar shows the progress: 'Step 1: Software and Steps' (selected), 'Step 2: Hardware', 'Step 3: General Cluster Settings', and 'Step 4: Security'. The main content area is titled 'Software Configuration'. It features a 'Release' dropdown set to 'emr-6.1.1'. Below this is a grid of software options with checkboxes: Hadoop 3.2.1, JupyterHub 1.1.0, Ganglia 3.7.2, Hive 3.1.2, ZooKeeper 3.4.14, Hue 4.7.1, Spark 3.0.0, Zeppelin 0.9.0, Tez 0.9.2, HBase 2.2.5, Presto 0.232, MXNet 1.6.0, Phoenix 5.0.0, HCatalog 3.1.2, Livy 0.7.0, Flink 1.11.0, Pig 0.17.0, PrestoSQL 338, Sqoop 1.4.7, Oozie 5.2.0, and TensorFlow 2.1.0. Below the grid are sections for 'Multiple master nodes (optional)', 'AWS Glue Data Catalog settings (optional)', and 'Edit software settings'. The 'Edit software settings' section has radio buttons for 'Enter configuration' (selected) and 'Load JSON from S3', with a text area containing a placeholder for a configuration file path. At the bottom of the console, there is a footer with 'Feedback', 'Language', and copyright information for Amazon Web Services.

This screenshot shows the 'Security Options' section of the 'Create Cluster - Advanced Options' page. The 'EC2 key pair' dropdown is set to 'aws\_cloud'. The checkbox 'Cluster visible to all IAM users in account' is checked. Under 'Permissions', the 'Default' radio button is selected. Below this, there are links for 'EMR role' (EMR\_DefaultRole), 'EC2 instance profile' (EMR\_EC2\_DefaultRole), and 'Auto Scaling role' (EMR\_AutoScaling\_DefaultRole). At the bottom of the section are expandable links for 'Security Configuration' and 'EC2 security groups'. The bottom of the page features 'Cancel', 'Previous', and 'Create cluster' buttons, along with a footer containing 'Feedback', 'Language', and copyright information for Amazon Web Services.

**Step 2:** Connect to the head node of the cluster using SSH. To do this to your created cluster, and then click on ‘Connect to Master Node Using SSH.’ Copy and paste the link into your command line/terminal. If done correctly you will receive an ‘EMR’ visual you’re your command line or terminal. The link is in Mac/Linux, number 2. (Reference pictures are below.)



**Step 3:** Copy the data folder from the S3 bucket *directly* into a directory on the Hadoop File System (HDFS) named `/user/hadoop/eng_1M_1gram`. This would look like `hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/hadoop/eng_1M_1gram`

**Step 4:** Using pyspark, read the data you copied into HDFS in Step 3. To do this, go to your cluster, then select 'Application User Interfaces.' There you will copy and paste the link from the JupyterHub application. Note: Before doing this step be sure to copy the ssh link. (The default user and password are **jovyan** and **jupyter**) or work from pyspark in the terminal if you prefer. Once you have created a pyspark DataFrame, complete the following steps listed below. (Reference pictures below for process.)

[Clone](#) [Terminate](#) [AWS CLI export](#)

Cluster: MyHadoopCluster **Waiting** Cluster ready after last step completed.

[Summary](#) [Application user interfaces](#) [Monitoring](#) [Hardware](#) [Configurations](#) [Events](#) [Steps](#) [Bootstrap actions](#)

### Summary

**ID:** j-3PZX39MQU0PW1

**Creation date:** 2021-05-10 10:08 (UTC-4)

**Elapsed time:** 1 hour, 14 minutes

**After last step completes:** Cluster waits

**Termination protection:** Off [Change](#)

**Tags:** -- [View All](#) / [Edit](#)

**Master public DNS:** ec2-54-175-43-167.compute-1.amazonaws.com [View in console](#)

[Connect to the Master Node Using SSH](#)

### Configuration details

**Release label:** emr-5.33.0

**Hadoop distribution:** Amazon 2.10.1

**Applications:** Hive 2.3.7, Hue 4.9.0, Mahout 0.13.0, Pig 0.17.0, Tez 0.9.2

**Log URI:** s3://aws-logs-102244255867-us-east-1/elasticmapreduce/ [View in console](#)

**EMRFS consistent view:** Disabled

**Custom AMI ID:** --

### Application user interfaces

**Persistent user interfaces** [View in console](#): [YARN timeline server](#), [Tez UI](#)

**On-cluster user interfaces** [View in console](#): Not Enabled [Enable an SSH Connection](#)

### Network and hardware

**Availability zone:** us-east-1b

**Subnet ID:** [subnet-bcdc20f1](#) [View in console](#)

**Master:** **Running** 1 m5.xlarge

**Core:** **Running** 2 m5.xlarge

**Task:** --

**Cluster scaling:** Not enabled

Downloads/

Spark - Jupyter

EMR - AWS Cons

Darius\_Smith\_B

Working In The C

Spark and EMR

Big Data Wrangl

aws ssh tunnel

+

eu-north-1.console.aws.amazon.com/elasticmapreduce/home?region=eu-north-1#cluster-details:j-26XQ6O85D8WEL

Master Coding for... W3Schools Online... Jobs in Poland for... Home Dashboard | Brain... GitHub: Let's buil... Poland, Long Leer... Best VPN Countri... MySQL cheatsheet

aws Services Search [Option+S]

Stockholm \_earth\_

Amazon EMR

EMR Studio

EMR Serverless [New](#)

EMR on EC2

Clusters

Notebooks

Git repositories

Security configurations

Block public access

VPC subnets

Events

EMR on EKS

Virtual clusters

Help

What's new

Cluster: darius\_smith\_deliverable **Waiting** Cluster ready to run steps.

[Summary](#) [Application user interfaces](#) [Monitoring](#) [Hardware](#) [Configurations](#) [Events](#) [Steps](#) [Bootstrap actions](#)

### Persistent application user interfaces

Applications installed on the Amazon EMR cluster publish user interfaces (UI) as web sites to monitor cluster activity. Persistent UI logs are available for 30 days after an application ends. Persistent UI don't required SSH tunneling. They are hosted off of the cluster.

**Application user interface** [View in console](#)

[YARN timeline server](#)

[Tez UI](#)

[Spark history server](#)

### On-cluster application user interfaces

On-cluster UI are available only while clusters are running. Because they are hosted on the master node, on-cluster UI require a connection via SSH tunneling. Set up SSH tunneling before accessing these application UI. [Learn more](#)

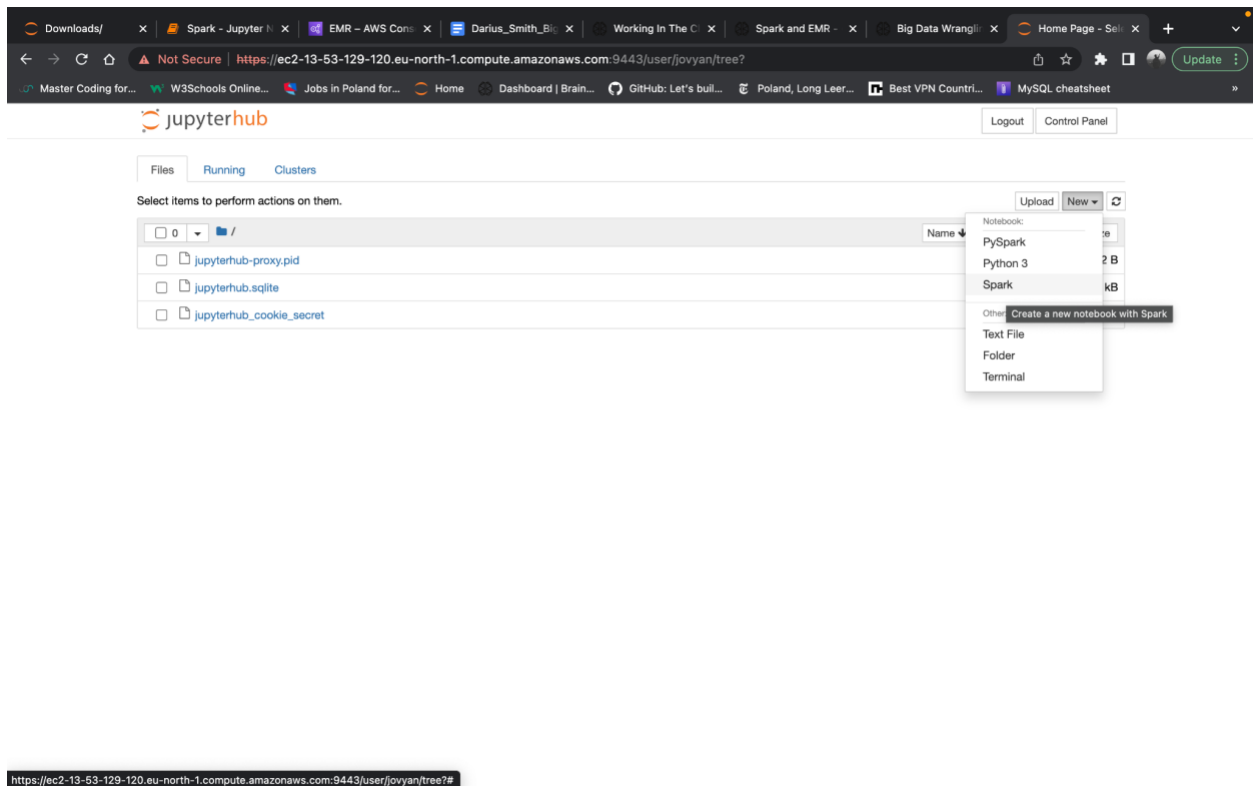
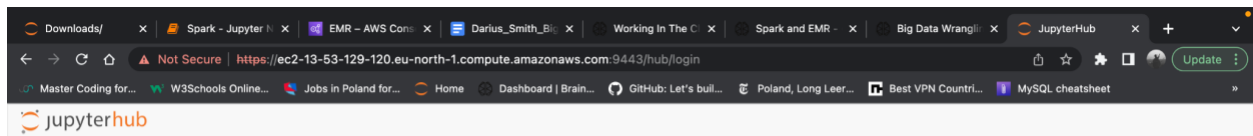
Application	User interface URL <a href="#">View in console</a>	Status
HDFS Name Node	http://ec2-13-53-129-120.eu-north-1.compute.amazonaws.com:9870/	SSH tunnel not enabled
JupyterHub	https://ec2-13-53-129-120.eu-north-1.compute.amazonaws.com:9443/	SSH tunnel not enabled
Spark History Server	http://ec2-13-53-129-120.eu-north-1.compute.amazonaws.com:18080/	SSH tunnel not enabled
Livy	http://ec2-13-53-129-120.eu-north-1.compute.amazonaws.com:8998/	SSH tunnel not enabled
Resource Manager	http://ec2-13-53-129-120.eu-north-1.compute.amazonaws.com:8088/	SSH tunnel not enabled

The following table lists web interfaces you can view on the task nodes:

Application	User interface URL
HDFS Data Node	http://ec2-000-000-000-000.compute-1.amazonaws.com:9864/
Node Manager	http://ec2-000-000-000-000.compute-1.amazonaws.com:8042/

Feedback Language

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



**Step 4 (a):** Describe the dataset (examples include size, shape, schema) in pyspark

The dataset has 261,823,226 rows, and 5 columns. The columns are 'token', 'year', 'frequency', 'pages', and 'books.' All the columns are strings. (Jupyter Notebook attached for separate instructions.)

The screenshot shows a Jupyter Notebook titled 'Darius\_Smith\_Big\_Data\_Deliverable' running on a JupyterHub instance. The notebook contains the following code and output:

```
In [7]: df = spark.read.csv('s3://brainstation-dsft/eng_1M_1gram.csv', header=True)

In [8]: df.printSchema()

root
 |-- token: string (nullable = true)
 |-- year: string (nullable = true)
 |-- frequency: string (nullable = true)
 |-- pages: string (nullable = true)
 |-- books: string (nullable = true)

In [9]: df.columns

['token', 'year', 'frequency', 'pages', 'books']

In [10]: df.head(5)

[Row(token='inGermany', year='1927', frequency='2', pages='2', books='2'), Row(token='inGermany', year='1929', frequency='1', pages='1', books='1'), Row(token='inGermany', year='1930', frequency='1', pages='1', books='1'), Row(token='inGermany', year='1933', frequency='1', pages='1', books='1'), Row(token='inGermany', year='1934', frequency='1', pages='1', books='1')]

In [11]: df.show(5)

+-----+-----+-----+-----+-----+
| token | year | frequency | pages | books |
+-----+-----+-----+-----+-----+
| inGermany | 1927 | 2 | 2 | 2 |
| inGermany | 1929 | 1 | 1 | 1 |
| inGermany | 1930 | 1 | 1 | 1 |
| inGermany | 1933 | 1 | 1 | 1 |
| inGermany | 1934 | 1 | 1 | 1 |
+-----+-----+-----+-----+-----+
only showing top 5 rows

In [12]: df.count()

261823225
```

**Step 4(b):** Create a new DataFrame from a query using Spark SQL, filtering to include only the rows where the token is "data" and describe the new dataset.

After creating a new DataFrame from a query using Spark SQL, when you filter to include only rows where token is “data” you get a dataset with only 24,642 rows. In the first (10) displays you get the words that have “data” in them.

**Step 4 © and Step 5:**Write the filtered data back to a directory in the HDFS from Spark using `df.write.csv()`. Be sure to pass the `header=True` parameter and examine the contents of what you've written. The code below allows you to write the filtered data back to a directory in the HDFS. Collect the contents of the directory into a single file on the local drive of the head node using `getmerge` and move this file into a S3 bucket in your account.

**Note:** For steps 4© and 5, there was an issue with my hadoop server. When trying to use `df.write.csv`, I was not able to see it in my hadoop file. As a result, I was not able to use `getmerge` to move the file due to ‘space issues.’ However, for steps 4© and 5, I wrote a ‘csv’ directly to my s3 bucket and then downloaded the file to use on a Jupyter notebook.

**Step 6 and Step 7:** On your local machine (or on AWS outside of Spark) in python, read the CSV data from the S3 folder into a pandas DataFrame (You will have to research how to read data into pandas from S3 buckets). **Note:** You must have first authenticated on your machine using `aws configure` on the command line to complete this step). Plot the number of occurrences of the token (the `frequency` column) of `data` over the years using `matplotlib`.

- Navigate to the Jupyter notebook ‘Darius\_Smith\_Big\_Data\_Deliverable JPN II’
- The notebook contains a step by step visual as far as reading the data.
- It also has a plot highlighting the number of occurrences of the token of data.

**Step 8:** Compare Hadoop and Spark as distributed file systems.

There are many advantages/differences between Hadoop and Spark. Listed below are a few advantages with differences:

**Performance:** When it comes to performance, Spark is faster because it uses random access memory (RAM) instead of reading and writing intermediate data to disks. Hadoop stores data on multiple sources and processes it in batches via MapReduce.

**Cost:** When it comes to cost, Hadoop runs at a lower cost since it relies on any disk storage type for data processing. Spark runs at a higher cost because it relies on in-memory computations for real-time data processing, which requires it to use high quantities of RAM to spin up nodes.

**Scalability:** When it comes to scalability, when data volume rapidly grows, Hadoop quickly scales to accommodate the demand via Hadoop Distributed File System (HDFS). In turn, Spark relies on the fault tolerant HDFS for large volumes of data.

**Machine Learning:** Spark is the superior platform in this category because it includes MLlib, which performs iterative in-memory ML computations. It also includes tools that perform regression, classification, persistence, pipeline construction, evaluation

As for storing data:

**Hadoop** – HDFS data is stored in something called blocks. These blocks are the smallest unit of data that the file system can store. Blocks are split across many machines at load time. Blocks are replicated across multiple machines. NameNode keeps track of which blocks make up a file and where they are stored.

**Spark** – Spark does not have its system to organize files in a distributed way (the file system). For this reason, it is usually installed on top of Hadoop so that Spark's advanced analytics applications can make use of the data stored using the (HDFS) – Hadoop distributed file system.

In conclusion, the purpose of this document is to serve as a tutorial for application of Big Data Fundamentals. Using AWS, and its many features such as Hadoop and Spark, we can access huge datasets without having to store them in our servers. These possess several distinct advantages that cannot be overlooked: scalability, flexibility, and affordability—all of which can help the success and growth of businesses.