

Big Data Wrangling With Google Books Ngrams

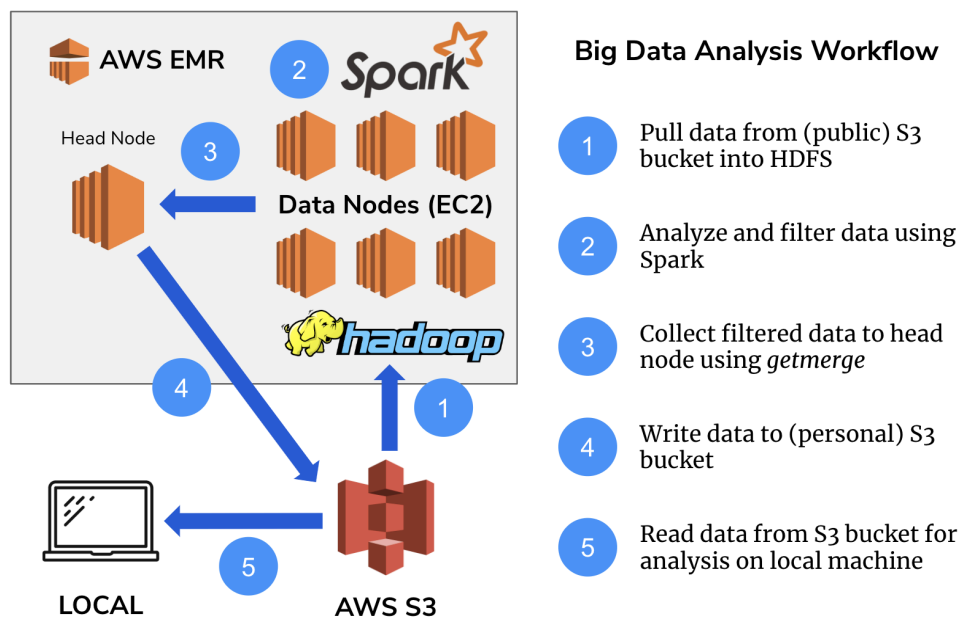
In this assignment, I will apply the skills I've learned in the Big Data Fundamentals unit to load, filter, and visualize a large real-world dataset in a cloud-based distributed computing environment using Hadoop, Spark, Hive, and the S3 filesystem. I will prepare a professional report to summarize my findings and include an appendix with screenshots of the steps completed to setup EMR Cluster

The Google Ngrams dataset was created by Google's research team by analyzing all of the content in Google Books. These digitized texts represent approximately 4% of all books ever printed, spanning a time period from the 1800s into the 2000s.

The dataset is hosted in a public S3 bucket as part of the Amazon S3 Open Data Registry. For this assignment, the data has been converted to CSV and hosted on a public S3 bucket, which can be accessed here: [s3://brainstation-dsft/eng_1M_1gram.csv](https://s3.amazonaws.com/brainstation-dsft/eng_1M_1gram.csv).

As part of this workflow, I will filter and reduce the data down to a manageable size, and then perform some analysis locally on my machine after extracting data from the cloud and processing it using Big Data tools. The workflow and steps in the process are illustrated below:

Copyright  BrainStation

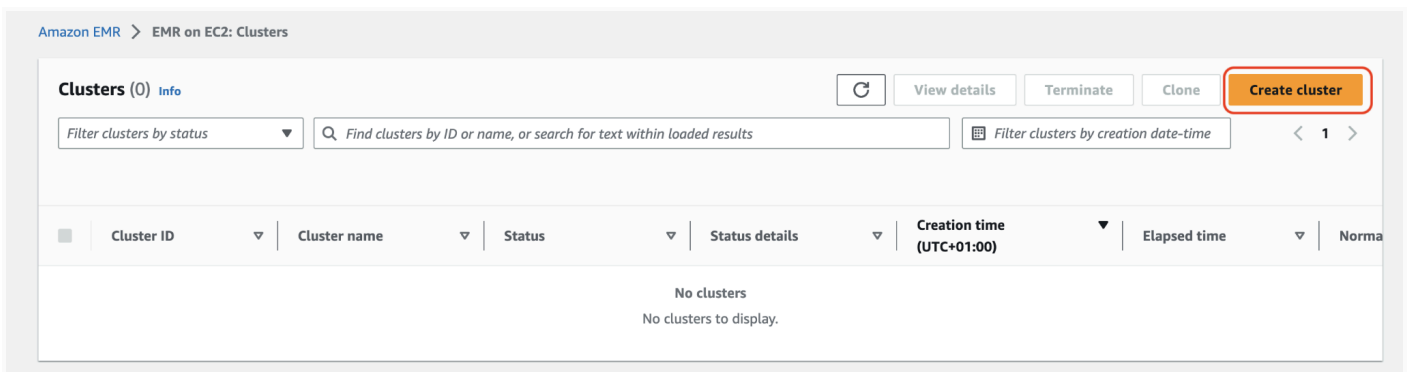
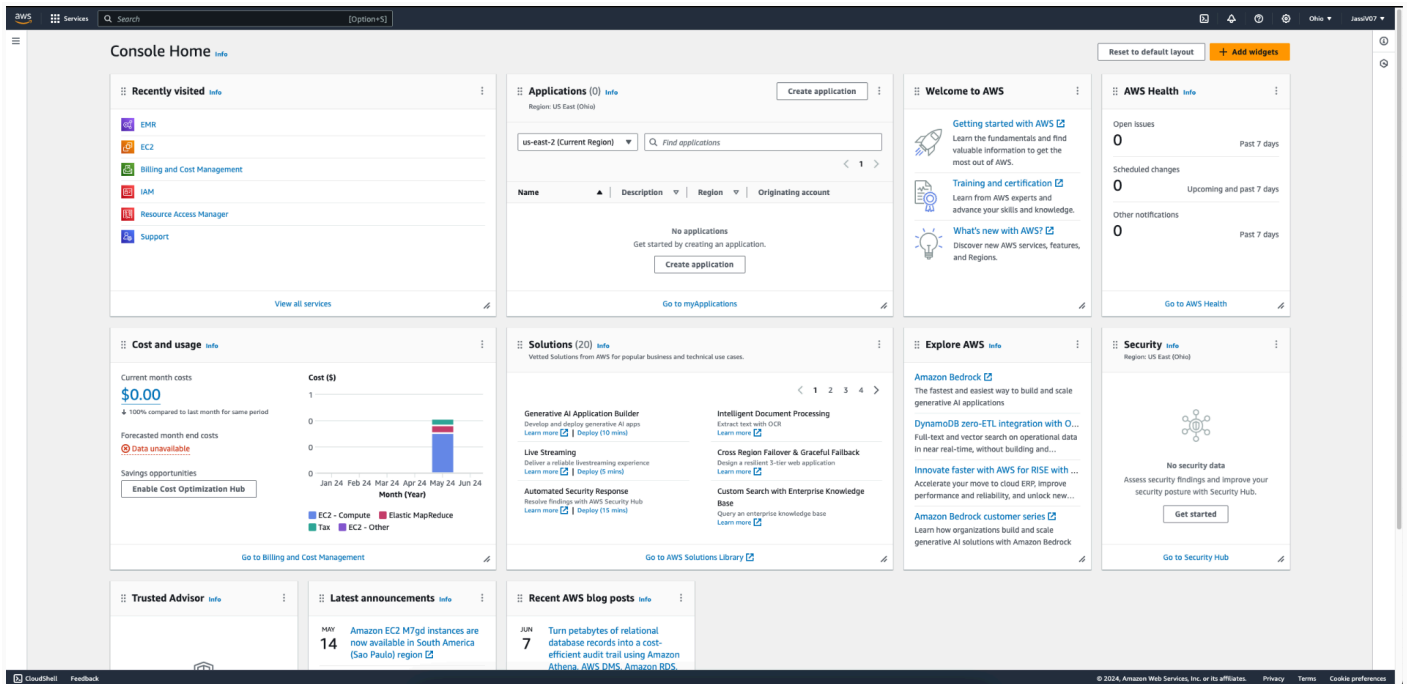


Unit 4 Deliverable 1

Jashkirat Viridi

01. Spin up a new EMR cluster on AWS for using Spark and EMR notebooks

- Go to <https://aws.amazon.com> and sign in to your account.
- Navigate to the EMR panel in AWS and click 'Create Cluster':




c. Name the Cluster and Select Applications.


- i. Give your cluster a name.
- ii. In the 'Release' dropdown, select emr-6.10.0.
- iii. Select the Custom application bundle, and tick the boxes for Hadoop, Hue, JupyterHub, Livy, Hive, and Spark.


Name


Amazon EMR release [Info](#)
A release contains a set of applications which can be installed on your cluster.


Application bundle


Spark


Core Hadoop


HBase


Presto


Trino


Custom


▼ Customise your application bundle

Applications included in bundle

<input type="checkbox"/> Flink 1.16.0	<input type="checkbox"/> Ganglia 3.7.2	<input type="checkbox"/> HBase 2.4.15
<input type="checkbox"/> HCatalog 3.1.3	<input checked="" type="checkbox"/> Hadoop 3.3.3	<input checked="" type="checkbox"/> Hive 3.1.3
<input checked="" type="checkbox"/> Hue 4.10.0	<input type="checkbox"/> JupyterEnterpriseGateway 2.6.0	<input checked="" type="checkbox"/> JupyterHub 1.5.0
<input checked="" type="checkbox"/> Livy 0.7.1	<input type="checkbox"/> MXNet 1.9.1	<input type="checkbox"/> Oozie 5.2.1
<input type="checkbox"/> Phoenix 5.1.2	<input type="checkbox"/> Pig 0.17.0	<input type="checkbox"/> Presto 0.278
<input checked="" type="checkbox"/> Spark 3.3.1	<input type="checkbox"/> Sqoop 1.4.7	<input type="checkbox"/> TensorFlow 2.11.0
<input type="checkbox"/> Tez 0.10.2	<input type="checkbox"/> Trino 403	<input type="checkbox"/> Zeppelin 0.10.1
<input type="checkbox"/> ZooKeeper 3.5.10		

d. Instance Group

i. Remove the Task instance group

☒ **Instance groups**
Choose one instance type per node group

☐ **Instance fleets**
Choose any combination of instance types within each node group

Instance groups

Primary

Choose EC2 instance type

m5.xlarge
4 vCore 16 GiB memory EBS only storage
On-demand price: -
Lowest spot price: \$0.099 (us-east-1d)

Actions ▼

☐ **Use multiple primary nodes**
To improve cluster availability, use three primary nodes with the same configuration and bootstrap actions. You cannot use multiple primary nodes with instance fleets.

► **Node configuration - optional**

Core

Choose EC2 instance type

m5.xlarge
4 vCore 16 GiB memory EBS only storage
On-demand price: -
Lowest spot price: \$0.099 (us-east-1d)

Actions ▼

► **Node configuration - optional**

Task 1 of 1

Remove instance group

Name

Task - 1

Choose EC2 instance type

ii. Allocate 2 Nodes to the core instance group

Cluster scaling and provisioning option [Info](#)
The Amazon EMR console only supports EMR-managed scaling. To create a cluster with auto-scaling, use CLI or SDK.

☒ **Set cluster size manually**
Use this option if you know your workload patterns in advance.

☐ **Use EMR-managed scaling**
Monitor key workload metrics so that EMR can optimise the cluster size and optimise its resource utilisation.

Name	Instance type	Size	Use spot purchasing option
Core	m5.xlarge	<input type="text" value="2"/> ← instance(s)	<input type="checkbox"/>

e. Cluster Termination

- Set cluster termination to 4h idle time.
- Turn termination protection off.

Cluster termination [Info](#)

☐ Manually terminate cluster

☒ **Terminate cluster after idle time (recommended)**

Idle time
Enter the time until your cluster terminates.

▼

Choose a time that is greater than 1 minute (00:01:00) and less than 7 days. The time is in hh:mm:ss (24-hour) format.

☐ **Terminate cluster after last step has been completed**
You cannot edit this selection after you have created your cluster.

☐ **Use termination protection**
Protect your EC2 instances from accidental termination.

f. Security and Access Management

- i. Select your key pair (keys are associated to geographies so if you switched recently, you might need to create a new key pair).
- ii. In Identity and Access Management, choose the EMR_DefaultRole and EMR_EC2_DefaultRole

▼ **Security configuration and EC2 key pair** [Info](#)

Choose a security configuration or create a new one that you can reuse with other clusters.

Security configuration

Select your cluster encryption, authentication, and instance metadata service settings.

Q Choose a security configuration

↻

Browse [↗](#)

Create security configuration [↗](#)

Amazon EC2 key pair for SSH to the cluster [Info](#)

Q brainstation

×

Browse

Create key pair [↗](#)

▼ **Identity and Access Management (IAM) roles - required** [Info](#)

Choose or create a service role and instance profile for the EC2 instances in your cluster.

Amazon EMR service role [Info](#)

The service role is an IAM role that Amazon EMR assumes to provision resources and perform service-level actions with other AWS services.

☒ Choose an existing service role

Select a default service role or a custom role with IAM policies attached so that your cluster can interact with other AWS services.

☐ Create a service role

Let Amazon EMR create a new service role so that you can grant and restrict access to resources in other AWS services.

Service role

EMR_DefaultRole

▼

↻

EC2 instance profile for Amazon EMR

The instance profile assigns a role to every EC2 instance in a cluster. The instance profile must specify a role that can access the resources for your steps and bootstrap actions.

☒ Choose an existing instance profile

Select a default role or a custom instance profile with IAM policies attached so that your cluster can interact with your resources in Amazon S3.

☐ Create an instance profile

Let Amazon EMR create a new instance profile so that you can specify a custom set of resources for it to access in Amazon S3.

Instance profile

EMR_EC2_DefaultRole

▼

↻

Custom automatic scaling role - optional

When a custom automatic scaling rule triggers, Amazon EMR assumes this role to add and terminate EC2 instances. [Learn more](#) [↗](#)

Custom automatic scaling role

Choose IAM role

▼

↻

Create IAM role [↗](#)

Note Once you create the cluster the process will start but it will take some time for it to get fully loaded and all of the things to get configured, the state will change from Starting to Waiting.

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

- Connect to the Primary Node of the cluster, and access JupyterHub in a browser window, edit the following bash command:
- Replace 'xxxxxxxxxx' with your 'Primary node public DNS' found on the overview page of your cluster.

```
ssh -i mykey.pem -L 9995:localhost:9443  
hadoop@xxxxxxxxxxxxxxxx.compute.amazonaws.com
```

unit_4_del

Updated 7 minutes ago

Terminate

Clone in AWS CLI

Clone

▼ Summary

Cluster info	Applications	Cluster management	Status and time
<div>Cluster ID</div> <div>j-W83ESHBFRN6</div> <div>Cluster configuration</div> <div>Instance groups</div> <div>Capacity</div> <div>1 Primary 1 Core 1 Task</div>	<div>Amazon EMR version</div> <div>emr-6.10.0</div> <div>Installed applications</div> <div>Hadoop 3.3.3, Hive 3.1.3, Hue 4.10.0, JupyterHub 1.5.0, Livy 0.7.1, Spark 3.3.1</div>	<div>Log destination in Amazon S3</div> <div>aws-logs-992382776059-us-east-2/elasticmapreduce</div> <div>Persistent application UIs</div> <div>Spark History Server</div> <div>YARN timeline server</div> <div>Tez UI</div> <div>Primary node public DNS</div> <div>ec2-18-221-130-68.us-east-2.compute.amazonaws.com</div> <div>Connect to the Primary node using SSH</div>	<div>Status</div> <div>⊖ Terminated</div> <div>Creation time</div> <div>June 09, 2024, 17:39 (UTC-04:00)</div> <div>Elapsed time</div> <div>4 hours, 41 minutes</div> <div>End time</div> <div>June 09, 2024, 22:20 (UTC-04:00)</div>

c. Paste the command in the terminal

```
sh -i brinstation.pem -L 9995:localhost:9443 hadoop@ec2-18-221-130-68.us-east-2.compute.amazonaws.com
t login: Sun Jun 9 22:51:49 2024

#_
#### Amazon Linux 2
#####\
\###| AL2 End of Life is 2025-06-30.
\#/
V~' '->

A newer version of Amazon Linux is available!

Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/

EEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRR
:::EEEEEEEEEE:E M:::::M M:::::M R::::::::::R
:::EEEEEEEEEE:E M:::::M M:::::M R::::RRRRRR::::R
:::E EEEE M:::::M M:::::M RR:::R R:::R
:::E M:::::M:M M::M:M:M R::R R:::R
:::EEEEEEEEEE M::::M M::M M::M M::::M R::RRRRRR::::R
:::EEEEEEEEEE M::::M M::M:M M::::M R::::::::::RR
:::EEEEEEEEEE M::::M M::::M M::::M R::RRRRRR::::R
:::E M::::M M::M M::::M R::R R:::R
:::E EEEE M::::M MMM M::::M R::R R:::R
:::EEEEEEEEEE:E M::::M M::::M R::R R:::R
:::EEEEEEEEEE:E M::::M M::::M RR:::R R:::R
EEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRR RRRRRR

hadoop@ip-172-31-42-62 ~]$
```

03.Copy the data folder from the S3 bucket directly into a directory on the Hadoop File System (HDFS) named /user/hadoop/eng_1M_1gram.

a. Paste the following command in the terminal

```
hadoop distcp s3://brainstation-dsft/eng_1M_1gram.csv /user/hadoop/eng_1M_1gram
```

b. Check `/user/hadoop/eng_1M_1gram`

```
[hadoop@ip-172-31-42-62 ~]$ hadoop fs -ls
Found 3 items
drwxr-xr-x  - hadoop hdfsadmin  group 0 2024-06-09 23:32 .sparkStaging
-rw-r--r--  1 hadoop hdfsadmin  group 5292105197 2024-06-09 22:24 eng_1M_1gram
```


Unit 4 Deliverable 1
Jashkirat Virdi

- c. Access JupyterHub in a browser window at <https://localhost:9995>
- d. Login using username `jovyan` and password `jupyter`

Sign in

Username:

jovyan

Password:

Sign in

- e. Open a New PySpark Notebook

jupyterhub

Logout Control Panel

Files Running Clusters

Select items to perform actions on them.

0 /

Untitled.ipynb

jupyterhub-proxy.pid

jupyterhub.sqlite

jupyterhub_cookie_secret

Run

Upload New

Notebook:

PySpark

Python 3

Spark

Other:

Text File

Folder

Terminal

04. Check Step_4_Big_Data.ipynb Notebook for the Below.

- a. Describe the dataset (examples include size, shape, schema) in pyspark
- 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
- c. Write the filtered data back to a directory in the HDFS from Spark using `df.write.csv()`.

05. 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.

- a. Check hadoop directory using `hadoop fs -ls`

```
[hadoop@ip-172-31-42-62 ~]$ hadoop fs -ls /user/hadoop/
Found 3 items
drwxr-xr-x  - hadoop hdfsadmin  group 0 2024-06-09 23:32 /user/hadoop/.sparkStaging
-rw-r--r--  1 hadoop hdfsadmin  group 5292105197 2024-06-09 22:24 /user/hadoop/eng_1M_1gram
drwxr-xr-x  - livy hdfsadmin  group 0 2024-06-09 23:25 /user/hadoop/filtered_data
```

- b. Use `hadoop fs -getmerge` to merge the contents into a single file on the local drive of the head node.
- c. Move the file into an S3 bucket in your account using `aws s3 cp`.

```
[hadoop@ip-172-31-42-62 ~]$ aws s3 cp filtered_data.csv s3://aws-emr-studio-992382776059-us-east-2/1717965722053/e-9BBQW3CFDNRZC7JG8NQ3YHG9/
upload: ./filtered_data.csv to s3://aws-emr-studio-992382776059-us-east-2/1717965722053/e-9BBQW3CFDNRZC7JG8NQ3YHG9/filtered_data.csv
```

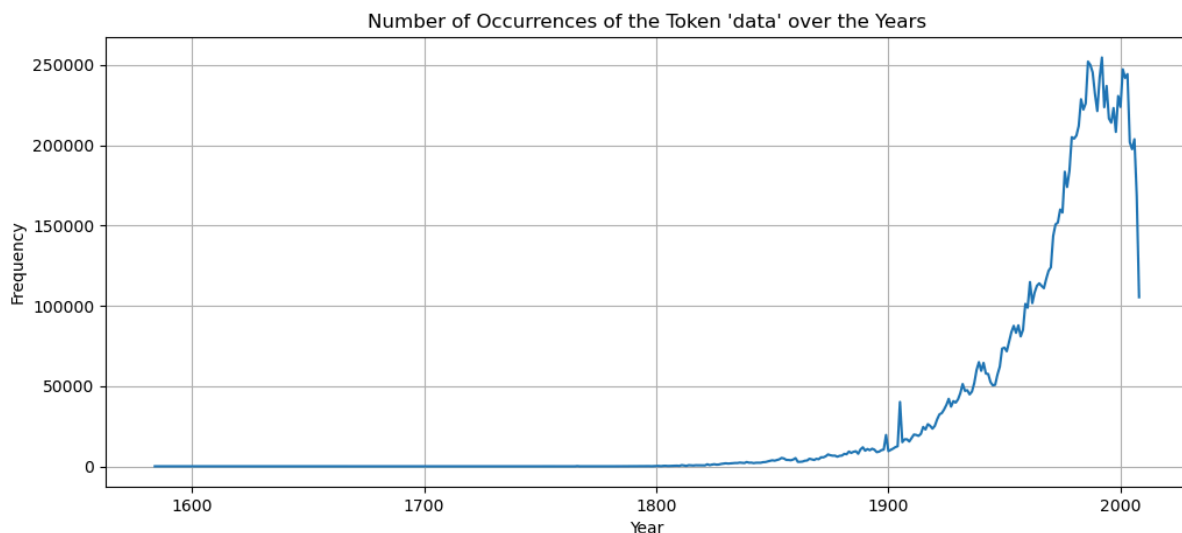
Check Step_6_Big_Data.ipynb Notebook for the Below.

06. On your local machine in python, read the CSV data from the S3 folder into a pandas DataFrame Note you must have first authenticated on your machine using aws configure on the command line to complete this step.

- a. Open Jupyter Notebook and import necessary libraries
- b. Read Data into dataframe

07. Plot the number of occurrences of the token (the frequency column) of data over the years using matplotlib.

- a. Data Wrangling
- b. Plot number of occurrences of the token ``data`` over the years



The heavy usage of the word "data" in books began in the mid-1800s and increased exponentially, reaching a peak between the 1990s and 2000s, likely due to the absence of more recent data.

O8.Compare Hadoop and Spark as distributed file systems.

- a. What are the advantages/ differences between Hadoop and Spark? List two advantages for each.

i. **Advantages of Hadoop:**

1. **Storage Capability (HDFS):** Hadoop Distributed File System (HDFS) allows for the storage and processing of large data sets across distributed clusters. It ensures high fault tolerance and reliability, making it suitable for storing vast amounts of unstructured data.
2. **Cost-Effective:** Hadoop is often more cost-effective for large-scale batch processing due to its open-source nature and compatibility with commodity hardware, reducing the need for expensive proprietary solutions

ii. **Advantages of Spark:**

1. **Speed:** Spark processes data much faster than Hadoop MapReduce due to its in-memory computation capabilities. It can cache data in memory, which significantly speeds up iterative algorithms and interactive data analysis.
2. **Ease of Use:** Spark provides high-level APIs in Java, Scala, Python, and R, making it more accessible for developers. It also includes built-in libraries for streaming, machine learning, and graph processing, simplifying complex data processing tasks.

- b. Explain how the HDFS stores the data.

HDFS (Hadoop Distributed File System) stores data by splitting large files into smaller blocks, typically 128MB or 256MB each. These blocks are distributed across multiple nodes in a cluster for parallel processing. Each block is replicated across multiple nodes (usually three) to ensure fault tolerance and reliability. The NameNode manages the metadata and directory structure, while DataNodes store the actual data blocks. This design allows HDFS to handle large-scale data storage with high availability and resilience.