

# –SPAM & HAM Classification–

## –CA675

**Name-** Pranjnay Bhardwaj

**Student No:-** 23265772

**Email:**[-pranjnay.bhardwaj2@mail.dcu.ie](mailto:pranjnay.bhardwaj2@mail.dcu.ie)

**GitLabrepoLink:-**[https://gitlab.computing.dcu.ie/bhardwp2/CA675\\_Cloud\\_Assignments\\_23265772](https://gitlab.computing.dcu.ie/bhardwp2/CA675_Cloud_Assignments_23265772)

**Cloud**

**Platform(AWS):-**<https://awsacademy.instructure.com/courses/58839/modules/items/5186817>

## Dataset:-<https://zenodo.org/records/8339691>

The Dataset Named CEAS\_08 has all the required columns in tabular CSV format needed for SQL-based RDBMS. It has sender, receiver, date, subject, body, label, urls. Rows Count-64824 is enough for preprocessing and Transformations using HQL.

| sender             | receiver  | date       | subject     | body     | label | urls |
|--------------------|-----------|------------|-------------|----------|-------|------|
| Young Esp user4@gv | user4@gv  | Tue, 05 Au | Never agree | Buck up, | 1     | 1    |
| Mok <iplir         | user2.2@{ | Tue, 05 Au | Befriend J  |          | 1     | 1    |
| Daily Top          | user2.9@{ | Tue, 05 Au | CNN.com     | >+==+=+: | 1     | 1    |
| Michael P; Sp      | SpamAssa  | Tue, 05 Au | Re: svn co  | Would    | 0     | 1    |

## Steps Taken:-

### Cloud Infrastructure Setup:-

Cloud Technology Used:- AWS Cloud

Applications on Cloud premises used:- Hadoop, Hive, Spark

-Start Lab on Sandbox Download Pem file.

- Building AWS EMR Cluster:-

1:-Cluster 1

Your cluster "My cluster-007" has been successfully created.

Amazon EMR > EMR on EC2: Clusters > My cluster-007

My cluster-007

Updated 1 minute ago

**Summary**

| Cluster info                             | Applications                                       | Cluster management   | Status and time  |
|--|--|--|--|
| Cluster ID<br>j-2ACIDSTG6KBQX            | Amazon EMR version<br>emr-6.14.0                   | Log destination in Amazon S3<br>mylogsemr1   | Status<br><span>Waiting</span>                           |
| Cluster configuration<br>Instance groups | Installed applications<br>Hadoop 3.3.3, Hive 3.1.3 | Persistent application UIs<br>YARN timeline server <a href="#">[ ]</a><br>Tez UI <a href="#">[ ]</a> | Creation time<br>November 10, 2023, 15:31<br>(UTC+00:00) |
| Capacity<br>1 Primary   2 Core   0 Task  |  | Primary node public DNS<br><a href="#">[ ]</a> ec2-3-239-10-159.compute-1.amazonaws.com              | Elapsed time<br>13 minutes, 22 seconds                   |
|  |  | Connect to the Primary node using  |  |

## 2:-Cluster 2

The screenshot shows the AWS EMR Cluster Overview page. At the top, a green banner says "Your cluster 'My cluster07' has been successfully created." Below the banner, the cluster name "My cluster07" is displayed. The page is divided into four main sections: Cluster info, Applications, Cluster management, and Status and time. Under Cluster info, it shows Cluster ID (j-1EYJX402VZ8U), Cluster configuration (Instance groups), and Capacity (1 Primary, 2 Core, 0 Task). Under Applications, it shows Amazon EMR version (emr-6.14.0) and Installed applications (Hadoop 3.3.3, Spark 3.4.1). Under Cluster management, it shows Log destination in Amazon S3 (mylogsmr7), Persistent application UIs (Spark History Server, YARN timeline server), Primary node public DNS (ec2-35-175-127-223.compute-1.amazonaws.com), and links to Connect to the Primary node using SSH and SSM. The Status and time section indicates the cluster is "Waiting" (Status), created on November 13, 2023, at 18:37 (Creation time), and has been running for 51 minutes, 34 seconds (Elapsed time). At the bottom, there are tabs for Properties, Bootstrap actions, Instances (Hardware), Steps, Applications, Configurations, Monitoring, Events, and Tags (0).

As you can see in the above screenshots the EMR cluster named My cluster07 & My cluster-007 have been created successfully with a log destination in the S3 directory mentioned above.

Specification:-Core:-2- M4large

Service role and EC2 Instance profile roles are used in creating clusters as default roles.

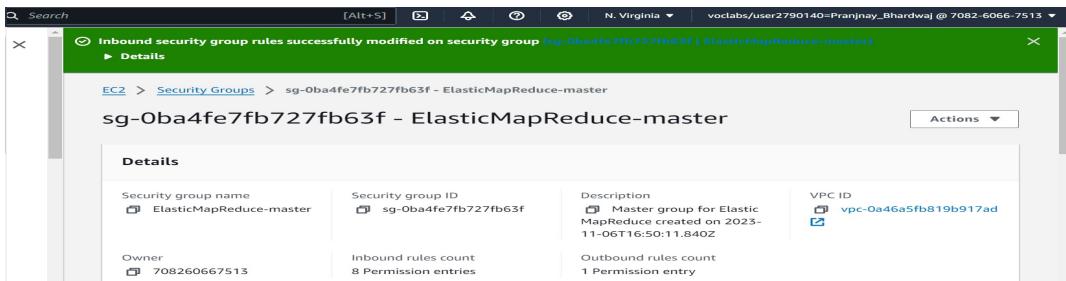
The key used- Vokey in Hive and while running 2nd cluster with spark -mykey007

The screenshot shows the AWS IAM Key Pairs page. It displays a table with two rows of key pairs. The columns are Name, Type, Created, Fingerprint, and ID. The first row, 'vokey', is listed with a checkbox next to it. The second row, 'mykey007', is listed with a checked checkbox next to it. The Fingerprint and ID columns show the unique identifiers for each key pair.

| Name     | Type | Created                | Fingerprint                                 | ID                    |
|----------|------|------------------------|---|-----------------------|
| vokey    | rsa  | 2023/11/14 15:10 GMT+0 | e5:b1:4d:28:41:79:25:bc:95:d5:f5:1f:3d:...  | key-00261fd1af41c0591 |
| mykey007 | rsa  | 2023/11/14 15:18 GMT+0 | fc:2c:54:b3:1e:f4:c4:05:cfcc:09:9a:15:ac... | key-037d684aa87ca68d2 |

The above key is created for Putty.

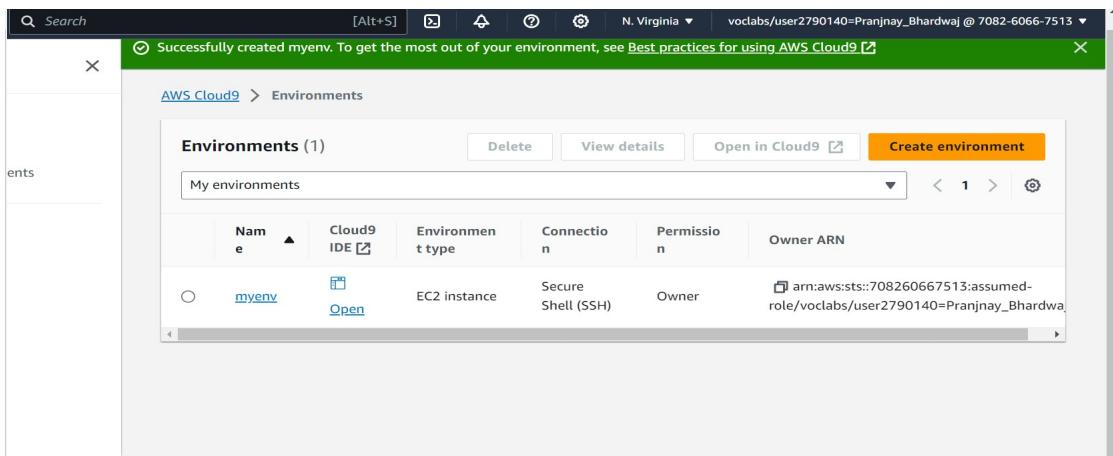
## Editing inbound rules:-



-Added SSH with Ipv4

-And for the 2nd Cluster with My IP.

- Once Public DNS is available with a **waiting**, we can move on to Creation of Cloud 9 Environment.
  - We need to select same subnet as given in cluster during creation of cloud 9 Environment



- Open Cloud 9 IDE and upload labuser.pem file and use chmod 400 labsuser.pem -To read file  
And then ssh -i labsuser.pem hadoop@<PUBLIC-DNS> -To connect  
**Public DNS-** copy from cluster

```

hadoop@ip-172-31-18-25 ~ % Immediate + 
vocabs:/environment $ chmod 400 labsuser.pem
vocabs:/environment $ ssh -i labsuser.pem hadoop@ec2-54-160-174-213.compute-1.amazonaws.com
The authenticity of host 'ec2-54-160-174-213.compute-1.amazonaws.com (172.31.18.253)' can't be established.
ECDSA key fingerprint is SHA256:Apm5cJI5ZCylvtSSBydWJ4y1g7tkb9DEAneobh40GA.
ECDSA key fingerprint is MD5:da:9d:d0:89:12:85:5d:c4:d4:9d:78:e6:1b:fe:80:3b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-160-174-213.compute-1.amazonaws.com,172.31.18.253' (ECDSA) to the list of known hosts.

      #_
      ~\_\ #####
      ~~ \#####
      ~~ \|##|
      ~~ \|#/
      ~~ V~' '-->
      ~~ /_ A newer version of Amazon Linux is available!
      ~~ /_ /_
      /m/` Amazon Linux 2023, GA and supported until 2028-03-15.
      https://aws.amazon.com/linux/amazon-linux-2023/

0 packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.

:EEEEEEEEE MBBBBBBB MBBBBBBB RRRRRRRRRRRRRR
:E:::::::::::E M:::::::M M:::::::M R:::::::R
:E:::::EEEEEEE:::E M:::::::M M:::::::M R:::::R R:::::R
 E:::::E     EEEE M:::::::M M:::::::M RR:::::R R:::::R
 E:::::E     M::::::M:M M:::M:::::M R:::R R:::::R
ECDSA key fingerprint is MD5:da:9d:d0:89:12:85:5d:c4:d4:9d:78:e6:1b:fe:80:3b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-160-174-213.compute-1.amazonaws.com,172.31.18.253' (ECDSA) to the list of known hosts.

      #_
      ~\_\ #####
      ~~ \#####

```

## S3 Bucket Creation:-

I created 3 folders for input and output data in S3.

Using Commands & by using Create folder in S3 Buckets.

```

vocabs:~/environment $ aws s3api put-object --bucket cloudca675data-bucket --key inputdata/
{
    "ETag": "\"d41d8cd98f00b204e9800998ecf8427e\"",
    "ServerSideEncryption": "AES256"
}
vocabs:~/environment $ aws s3api put-object --bucket cloudca675data-bucket --key outputdata/
{
    "ETag": "\"d41d8cd98f00b204e9800998ecf8427e\"",
    "ServerSideEncryption": "AES256"
}

```

CSV file uploaded directly by Upload button in S3 folder.

| Objects  | Properties                      | Permissions | Metrics       | Management | Access Points |          |      |      |               |      |               |                          |                                 |        |   |   |   |                          |                            |        |   |   |   |                          |                             |        |   |   |   |
|--|---------------------------------|-------------|---------------|------------|---------------|----------|------|------|---------------|------|---------------|--------------------------|---------------------------------|--------|---|---|---|--------------------------|----------------------------|--------|---|---|---|--------------------------|-----------------------------|--------|---|---|---|
| <b>Objects (3)</b> <p>Objects are the fundamental entities stored in Amazon S3. You can use <a href="#">Amazon S3 inventory</a> to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. <a href="#">Learn more</a></p> <div style="display: flex; justify-content: space-between;"> <span><a href="#"></a></span> <span><a href="#"></a></span> <span><a href="#"></a></span> <span><a href="#"></a></span> <span><a href="#"></a></span> <span><a href="#"></a></span> </div> <div style="display: flex; justify-content: space-between; margin-top: 5px;"> <span><a href="#">Create folder</a></span> <span><a href="#" style="background-color: orange; color: white; padding: 2px 10px;"> Upload</a></span> </div> <div style="margin-top: 10px;"> <input placeholder="Find objects by prefix" type="text"/> <span style="float: right;">&lt; 1 &gt; ⌂</span> </div> <table border="1" style="width: 100%; border-collapse: collapse; font-size: small;"> <thead> <tr> <th style="text-align: left; width: 5%;">checkbox</th> <th style="text-align: left; width: 25%;">Name</th> <th style="text-align: left; width: 10%;">Type</th> <th style="text-align: left; width: 15%;">Last modified</th> <th style="text-align: left; width: 10%;">Size</th> <th style="text-align: left; width: 10%;">Storage class</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td><a href="#">hiveoutputdata/</a></td> <td>Folder</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td><input type="checkbox"/></td> <td><a href="#">inputdata/</a></td> <td>Folder</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td><input type="checkbox"/></td> <td><a href="#">outputdata/</a></td> <td>Folder</td> <td>-</td> <td>-</td> <td>-</td> </tr> </tbody> </table> |                                 |             |               |            |               | checkbox | Name | Type | Last modified | Size | Storage class | <input type="checkbox"/> | <a href="#">hiveoutputdata/</a> | Folder | - | - | - | <input type="checkbox"/> | <a href="#">inputdata/</a> | Folder | - | - | - | <input type="checkbox"/> | <a href="#">outputdata/</a> | Folder | - | - | - |
| checkbox   | Name                            | Type        | Last modified | Size       | Storage class |          |      |      |               |      |               |                          |                                 |        |   |   |   |                          |                            |        |   |   |   |                          |                             |        |   |   |   |
| <input type="checkbox"/>   | <a href="#">hiveoutputdata/</a> | Folder      | -             | -          | -             |          |      |      |               |      |               |                          |                                 |        |   |   |   |                          |                            |        |   |   |   |                          |                             |        |   |   |   |
| <input type="checkbox"/>   | <a href="#">inputdata/</a>      | Folder      | -             | -          | -             |          |      |      |               |      |               |                          |                                 |        |   |   |   |                          |                            |        |   |   |   |                          |                             |        |   |   |   |
| <input type="checkbox"/>   | <a href="#">outputdata/</a>     | Folder      | -             | -          | -             |          |      |      |               |      |               |                          |                                 |        |   |   |   |                          |                            |        |   |   |   |                          |                             |        |   |   |   |

## Trigger Hive:- To connect Hive

-typed hive-Got in Hive

Then I created a table using HQL for all the fields with columns as per CSV file.

```
hadoop@ip-172-31-18-25 ~ % Immediate      x  bash - "ip-172-31-26-93.e" + 
at java.lang.reflect.Method.invoke(Method.java:498)
at org.apache.hadoop.util.RunJar.run(RunJar.java:323)
at org.apache.hadoop.util.RunJar.main(RunJar.java:236)
FAILED: ParseException line 6:12 cannot recognize input near ',' 'label' 'putdata' in column type
hive> CREATE EXTERNAL TABLE spam_ham (
  >   sender STRING,
  >   receiver STRING,
  >   dates STRING,
  >   subject STRING,
  >   body_ STRING,
  >   label BIGINT,
  >   urls BIGINT
  >
  > )
  > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
  > LOCATION 's3://cloudca675data-bucket/inputdata/';
OK
Time taken: 14.588 seconds
hive> SELECT * FROM SPAM_HAM LIMIT 10;
OK
sender receiver      date    subject body      NULL      NULL
Young Esposito <Young@world.de> user4@gvc.ceas-challenge.cc  "Tue 05 Aug 2008 16:31:02 -0700"  Never agree to be a loser      NULL      NULL
Become a lover no woman will be able to resist! NULL      NULL      NULL      NULL      NULL      NULL
http://whitedone.com/  NULL      NULL      NULL      NULL      NULL      NULL
NULL      NULL      NULL      NULL      NULL      NULL
NULL      NULL      NULL      NULL      NULL      NULL
come. Even as Nazi tanks were rolling down the streets the dreamersphilosopher or a journalist. He was still not sure.I do the same." 1      1      NULL      NULL      NULL
Mok <ipline's1983@icicle.ph> user2.2@gvc.ceas-challenge.cc  "Tue 05 Aug 2008 18:31:03 -0500"  Befriend Jenna Jameson NULL      NULL
Upgrade your sex and pleasures with these techniques http://www.brightmade.com NULL      NULL      NULL      NULL      NULL
NULL      NULL      NULL      NULL      NULL      NULL
Time taken: 3.349 seconds, Fetched: 10 row(s)
hive> |
```

The SPAM\_HAM Table consists same data as per the CSV file and the data types are STRING & BIGINT.

See in above Screenshot.

Now for Cleaning & filtering- I created a separate table with the Spam\_ham\_processed filtering command in where clause such as RLIKE with pattern -RLIKE '^([0-9][a-z][A-Z])'; - to get rid all non-relevant text rows and NOT NULL conditions- to filter the body with no messages.

```

hive> CREATE TABLE IF NOT EXISTS spam_ham_processed
> STORED AS PARQUET AS
> SELECT sender, receiver, dates, subject, body_
> FROM spam_ham
> WHERE body_IS NOT NULL
> AND subject RLIKE '([0-9][a-z][A-Z])'
> AND body_RLIKE '([0-9][a-z][A-Z])'
> AND body_NOT LIKE '%>+>%'
> AND sender RLIKE '([0-9][a-z][A-Z])'
> AND receiver RLIKE '([0-9][a-z][A-Z])';
Query ID = hadoop_20231113170755_b2fc6b67-b2cc-434a-886b-8647797c5040
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1699894618882_0001)

-----  

VERTICES      MODE      STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

Map 1 ..... container SUCEEDED 4     4     0     0     0     0  

-----  

VERTICES: 01/01 [=====>>>] 100% ELAPSED TIME: 56.44 s  

-----  

Moving data to directory hdfs://ip-172-31-12-3.ec2.internal:8020/user/hive/warehouse/spam_ham_processed
OK
Time taken: 62.829 seconds
hive> select * from spam_ham_processed limit 10;
OK
FREE medical consultations    FAST shipping    LOM prices and Toll-free customer support!"    1    0
FREE medical consultations    FAST shipping    LOM prices and Toll-free customer support!"    1    0
to overtime. After a stalemate in the overtime period Another energy-related topic in the conclusions is an at the age of 66    Soros was giving away more than $350 million a"    1    1
Park    Sleaty Road Graiguecullen Carlow Ireland Company No.: 370845
FREE medical consultations    FAST shipping    LOM prices and Toll-free customer support!"    1    0
Park    Sleaty Road Graiguecullen Carlow Ireland Company No.: 370845
Park    Sleaty Road Graiguecullen Carlow Ireland Company No.: 370845
FREE medical consultations    FAST shipping    LOM prices and Toll-free customer support!"    1    0
main(){ char h =h*t++    *x=t+2*h    c    i
Park    Sleaty Road Graiguecullen Carlow Ireland Company No.: 370845
Time taken: 0.288 seconds, Fetched: 10 row(s)
hive> select count(*) from spam_ham_processed;
FAILED: SemanticException [Error 10001]: Line 1:21 Table not found 'spam_ham_processed'
hive> select count(*) from spam_ham_processed;
OK
1124

```

Post filtering & cleaning the count of relevant data is-1124

### spam classification:-

I Created a new table spam classification on top of the spam\_ham\_processed table.

Then I used case statements with a set of spam words(Source:-Google) that are mostly found in spam emails.

If the word(Spam words) matches the body or subject of the sentence then it is spam otherwise it's ham separated in the classification column of table spam\_classification

See below the logic:-

```

hive> CREATE TABLE spam_classification AS
> SELECT
>     sender,
>     receiver,
>     dates,
>     subject,
>     body_,
>     CASE
>         WHEN LOWER(subject) RLIKE '.*earn.*' OR LOWER(body_) RLIKE '.*earn.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*cash.*' OR LOWER(body_) RLIKE '.*cash.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*free.*' OR LOWER(body_) RLIKE '.*free.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*cnn.*' OR LOWER(body_) RLIKE '.*cnn.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*order.*' OR LOWER(body_) RLIKE '.*order.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*credit.*' OR LOWER(body_) RLIKE '.*credit.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*debit.*' OR LOWER(body_) RLIKE '.*debit.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*sold.*' OR LOWER(body_) RLIKE '.*sold.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*sale.*' OR LOWER(body_) RLIKE '.*sale.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*100.*' OR LOWER(body_) RLIKE '.*100.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*money.*' OR LOWER(body_) RLIKE '.*money.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*urgent.*' OR LOWER(body_) RLIKE '.*urgent.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*price.*' OR LOWER(body_) RLIKE '.*price.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*welcome.*' OR LOWER(body_) RLIKE '.*welcome.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*click.*' OR LOWER(body_) RLIKE '.*click.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*amazing.*' OR LOWER(body_) RLIKE '.*amazing.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*guaranteed.*' OR LOWER(body_) RLIKE '.*guaranteed.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*benefit.*' OR LOWER(body_) RLIKE '.*benefit.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*looking.*' OR LOWER(body_) RLIKE '.*looking.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*affordable.*' OR LOWER(body_) RLIKE '.*affordable.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*save.*' OR LOWER(body_) RLIKE '.*save.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*biggest.*' OR LOWER(body_) RLIKE '.*biggest.*' THEN 'spam'
>
>         WHEN LOWER(subject) RLIKE '.*guaranteed.*' OR LOWER(body_) RLIKE '.*guaranteed.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*benefit.*' OR LOWER(body_) RLIKE '.*benefit.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*looking.*' OR LOWER(body_) RLIKE '.*looking.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*affordable.*' OR LOWER(body_) RLIKE '.*affordable.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*save.*' OR LOWER(body_) RLIKE '.*save.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*biggest.*' OR LOWER(body_) RLIKE '.*biggest.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*mega.*' OR LOWER(body_) RLIKE '.*mega.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*get.*' OR LOWER(body_) RLIKE '.*get.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*shop.*' OR LOWER(body_) RLIKE '.*shop.*' THEN 'spam'
>         WHEN LOWER(subject) RLIKE '.*great.*' OR LOWER(body_) RLIKE '.*great.*' THEN 'spam'
>         ELSE 'ham'
>     END AS classification
> FROM spam_ham_processed;

```

```

Query ID = hadoop_20231113134026_a057496b-6147-4ea8-9b0c-5330d8cf8269
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1699874729357_0005)

```

| VERTICES  | MODE      | STATUS    | TOTAL | COMPLETED | RUNNING | PENDING | FAILED | KILLED |
|---|-----------|-----------|-------|-----------|---------|---------|--------|--------|
| Map 1 .....   | container | SUCCEEDED | 2     | 2         | 0       | 0       | 0      | 0      |
| VERTICES: 01/01 [=====>>>] 100% ELAPSED TIME: 31.32 s   |           |           |       |           |         |         |        |        |
| Moving data to directory hdfs://ip-172-31-6-102.ec2.internal:8020/user/hive/warehouse/spam_classification |           |           |       |           |         |         |        |        |
| OK  |           |           |       |           |         |         |        |        |
| Time taken: 40.246 seconds  |           |           |       |           |         |         |        |        |

## Data Check:-

After Classification

-See below the output for spam ham-

**Observation:-** The accuracy is quite low.

```
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1699874729357_0005)

-----  

 VERTICES    MODE      STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1 ..... container SUCCEEDED 2     2     0     0     0     0     0  

-----  

VERTICES: 01/01 [=====>] 100% ELAPSED TIME: 31.32 s  

-----  

Moving data to directory hdfs://ip-172-31-6-102.ec2.internal:8020/user/hive/warehouse/spam_classification  

OK  

Time taken: 48.246 seconds  

hive> select * from spam_classification limit 10;  

OK  

Josh Jore <cbthqn@gmail.com> John Peacock <cupb.wssyilss@havurah-software.org> "Thu 07 Aug 2008 08:10:23 -0800" Re: [perl #47181] Try OO semantics before throwing fatal error  

r ham  

Matthias Miller <sctm@outofflanwell.com> zi9rpy-fcc0@lists.sourceforge.net "Thu 07 Aug 2008 11:10:52 -0500" [Py2exe-users] patch to fix crash when bundling pysvn in single executable ham  

Aschsoft Inc <xpbald@beerbreasts.com> user2.12@gvc.ceas-challenge.cc "Thu 07 Aug 2008 13:11:15 -0300" Well-Paid Part-Time Job ham  

Derek Hart <Derekboastephens@blackhillsdiscovered.com> user8.2-ext1@gvc.ceas-challenge.cc "Thu 07 Aug 2008 19:07:54 -0200" sims villainous henchman whistable ilyushin ham  

Emerson Melton <chewlingnf43@hartfordlife.com> user8.2@gvc.ceas-challenge.cc "Thu 07 Aug 2008 13:11:33 -0300" Was sind C Produkte? ham  

Camille Newsome <linkludaschmet@kudasch.de> user2.15@gvc.ceas-challenge.cc "Thu 07 Aug 2008 20:49:36 +0200" Try PowerEnlarge to enlarge your penis size and see the difference. s  

patz ka d ham  

Chris Richards <aaeio@giz-works.com> postfix-users <rljdsng-xyqoh@postfix.org> "Thu 07 Aug 2008 10:11:44 -0600" Re: postfix on a laptop: disk spin-up ham  

Irene Garrett <telmanong@clinique.com> user7-ext4@gvc.ceas-challenge.cc "Thu 07 Aug 2008 13:11:45 -0300" Re: ham  

Daily Top 10 <areugidn_1995@soft.com> email2375@gvc.ceas-challenge.cc "Thu 07 Aug 2008 21:40:12 -0000" CNN.com Daily Top 10 spam  

Laurel Goodwin <limmultimodis@multimodis.de> user7@gvc.ceas-challenge.cc "Thu 07 Aug 2008 21:50:07 +0300" Change your sex life with PowerEnlarge! cyeid cabmim tvxwgg ham  

Time taken: 0.112 seconds, Fetched: 10 row(s)  

hive>
```

## Top 10 Spam:-

Once we have Spam and Ham separated then using “spam” in where clause on the classification column of table spam \_calssification with the count, I extracted the top 10 Spam accounts.

```
hive> SELECT sender, receiver, COUNT(*) AS spam_count
> FROM spam_classification
> WHERE classification = 'spam'
> GROUP BY sender, receiver
> ORDER BY spam_count DESC
> LIMIT 10;
Query ID = hadoop_20231113135735_683a85f4-cf1d-48d0-9a77-e0c66321a7dc
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1699874729357_0006)

-----  

 VERTICES    MODE      STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1 ..... container SUCCEEDED 1     1     0     0     0     0     0  

Reducer 2 ..... container SUCCEEDED 2     2     0     0     0     0     0  

Reducer 3 ..... container SUCCEEDED 1     1     0     0     0     0     0  

-----  

VERTICES: 03/03 [=====>] 100% ELAPSED TIME: 8.36 s  

-----  

OK
Per Jessen <uee@computer.org> wkilxloc@opensuse.org 100
Karsten Bräckelmann <qmitxmeh@rudersport.de> zqoqi@spamassassin.apache.org 43
Anders Johansson <jw@rydsbo.net> wkilxloc@opensuse.org 40
Roger Oberholtzer <drtyu@opq.se> wkilxloc@opensuse.org 34
Ralf Hildebrandt <ecfa.ocnmogily@charite.de> rljdsng-xyqoh@postfix.org 30
Georg Brandl <c.xwopma@gmx.net> zvllln-eum@python.org 20
Basil Chupin <o0tzqpzo@tpg.com.au> wkilxloc@opensuse.org 20
Dermott Bolger <phwwwby.czrqex@gmail.com> pq-mism@yahoo-groups.com 19
qydlqcws-iacfym@issues.apache.org xrh@spamassassin.apache.org 18
Christian Heimes <wluhe@cheimes.de> Guido van Rossum <hoauf@python.org> 18
Time taken: 8.823 seconds, Fetched: 10 row(s)
hive>
```

## TOP 10 HAMS:-

Similarly, I extracted Hams using the “Ham” keyword in the where clause on the classification column.

```
hive> SELECT sender, receiver, COUNT(*) AS ham_count
    > FROM spam_classification
    > WHERE classification = 'ham'
    > GROUP BY sender, receiver
    > ORDER BY ham_count DESC
    > LIMIT 10;
Query ID = hadoop_20231113135610_0b6f3e1b-6a3e-40a3-883e-36e022baee7d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1699874729357_0006)

-----  

      VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0
Reducer 2 ..... container  SUCCEEDED   2       2       0       0       0       0
Reducer 3 ..... container  SUCCEEDED   1       1       0       0       0       0
-----  

VERTICES: 03/03  [=====>>>] 100%  ELAPSED TIME: 7.91 s
-----  

OK
qydlqcws-iacfym@issues.apache.org      xrh@spamassassin.apache.org      444
Aaron Kulkis <cmiqlkx91@hotmail.com>      opensuse <wkilxloc@opensuse.org>      179
http://nl.internet.com/ct.html?rtr=on&s=1      3s6f      131
http://nl.internet.com/ct.html?rtr=on&s=1      3tej      128
http://nl.internet.com/ct.html?rtr=on&s=1      3tv8      128
http://nl.internet.com/ct.html?rtr=on&s=1      3snp      125
Perl Jobs <xycn-vtnhz@perl.org> necc@perl.org      86
SpamExperts via Twitter <uiaregi@twitter.com> user2.1@gvc.ceas-challenge.cc      84
http://nl.internet.com/ct.html?rtr=on&s=1      3tzn      69
ysxshwee-tyrxx-liiyxs@lists.tuxonice.net      enpsfhxz-stvnz@lists.tuxonice.net      65
Time taken: 8.469 seconds, Fetched: 10 row(s)
```

## Data Extraction:-

I inserted the data into the S3 bucket folder in the hive\_output\_csv path with the help of the insert overwrite query.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY 's3://cloudca675data-bucket/hive_output_csv/'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > SELECT sender, receiver, dates,subject,body_, COUNT(*) AS ham_count
    > FROM spam_classification
    > WHERE classification = 'ham'
    > GROUP BY sender, receiver, dates,subject,body_
    > ORDER BY ham_count DESC
    > LIMIT 10;
Query ID = hadoop_20231113181503_ec4ae7e8-e227-4daa-835e-57b3d3fed65d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1699894618882_0002)

-----  

      VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0
Reducer 2 ..... container  SUCCEEDED   2       2       0       0       0       0
Reducer 3 ..... container  SUCCEEDED   1       1       0       0       0       0
-----  

VERTICES: 03/03  [=====>>>] 100%  ELAPSED TIME: 12.99 s
-----  

OK
Time taken: 13.841 seconds
hive> []
```

```

hive> INSERT OVERWRITE LOCAL DIRECTORY 's3://cloudca675data-bucket/hive_output_csv/'
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ','
> SELECT sender, receiver, dates,subject,body_, COUNT(*) AS spam_count
> FROM spam_classification
> WHERE classification = 'spam'
> GROUP BY sender, receiver,dates,subject,body_
> ORDER BY spam_count DESC
> LIMIT 10;
Query ID = hadoop_20231113181010_e9bac153-df1e-4b25-958f-200b5317000e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1699894618882_0002)

-----  

      VERTICES    MODE     STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0  

Reducer 2 ..... container  SUCCEEDED   2       2       0       0       0       0  

Reducer 3 ..... container  SUCCEEDED   1       1       0       0       0       0  

-----  

VERTICES: 03/03 [=====>>>] 100% ELAPSED TIME: 12.21 s  

-----  

OK  

Time taken: 13.255 seconds
hive> 

```

## Output file:-

The screenshot shows the AWS S3 console interface. The left sidebar has a search bar and navigation links for 'Amazon S3', 'Buckets', 'cloudca675data-bucket', and 'hive\_output\_csv/'. The main area displays a table of objects. The table has columns for Name, Type, Last modified, Size, and Storage class. There are two entries:

- Object: '\_SUCCESS', Type: -, Last modified: November 13, 2023, 18:15:18 (UTC+00:00), Size: 0 B, Storage class: Standard.
- Object: '000000\_0-hadoop\_20231113181503\_ec', Type: -, Last modified: November 13, 2023, 18:15:17 (UTC+00:00), Size: 610.0 B, Storage class: Standard.

Further, I downloaded and changed the file format to CSV manually.

Now this will act as input for tf idf.

## TFIDF For SPAM:-

For calculating TFIDF I used Pyspark

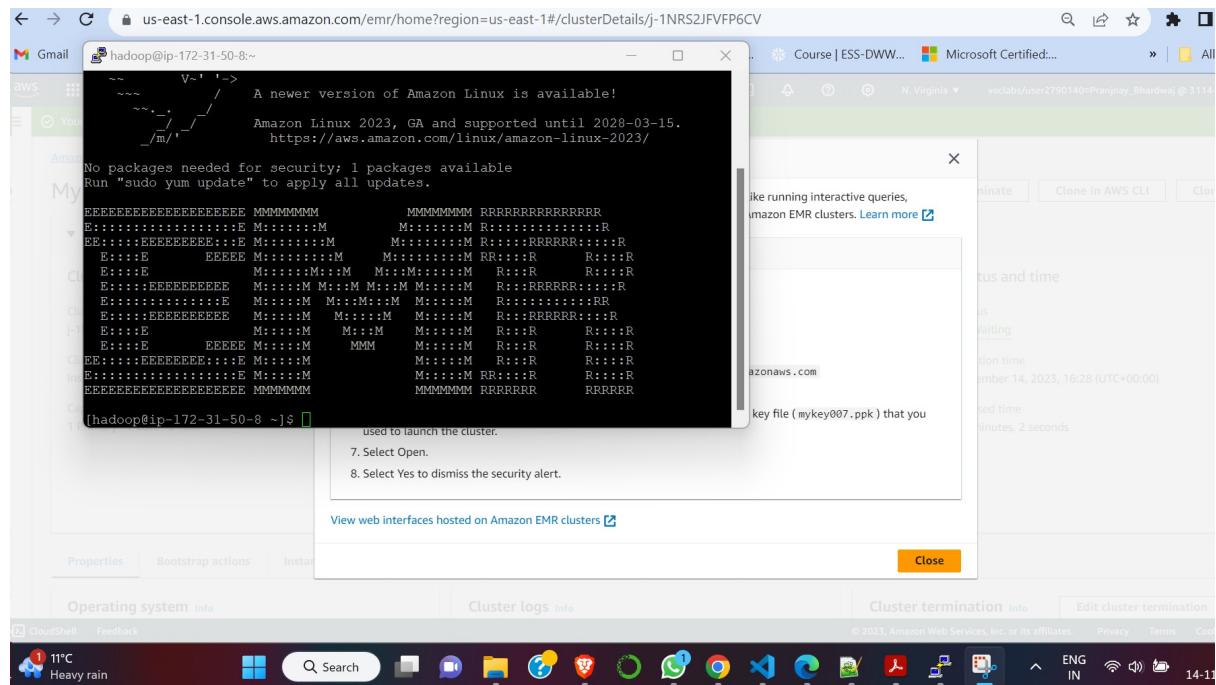
Pyspark has built-in ml features such as Tokenizer, HashingTF and IDF.

The tokenizer converts a text column into a text of words.

HashingTF is a feature that converts text documents to a sparse feature vector, where each word is hashed to a fixed-size index. It represents the term frequencies.

IDF is an estimator. It scales down the term frequencies computed by HashingTF and reflects the importance of words in documents.

For spark, I created a separate cluster as mentioned in page-3(cluster-2)and used putty to connect EMR:-



Now using vi file\_name.py

We can get inside the file and place our program

```
EEEEEEEEEEEEEEEEEE MMMMMMM MBBBBBBB RRRRRRRRRRRRRR
E:::::::::::E M::::::M M:::::::M R:::::::R
EE:::::EEEEEEE:::E M:::::::M M:::::::M R:::::::RRRRR:::R
E:::::E EEEEE M:::::::M M:::::::M RR:::::R R:::::R
E:::::E M:::::::M M:::::::M M:::::::M R:::::R R:::::R
E:::::EEEEEEEEE M::::::M M:::::::M M:::::::M R:::::RRRRR:::R
E:::::EEEEEEEEE M::::::M M:::::::M M:::::::M R:::::RRRRR:::R
E:::::E M::::::M M:::::::M M:::::::M R:::::R R:::::R
E:::::E EEEE M::::::M M:::::::M R:::::R R:::::R
EE:::::EEEEEEE:::E M::::::M M:::::::M R:::::R R:::::R
E:::::EEEEEEEEE M::::::M M:::::::M R:::::R R:::::R
EEEEEEEEEEEEEEEEEE MMMMMMM RRRRRRRR

[hadoop@ip-172-31-50-8 ~]$ vi tfidf.py
```

Press the I -keyword to insert the Python program

program:-[https://gitlab.computing.dcu.ie/bhardwp2/CA675\\_Cloud\\_Assignments\\_23\\_265772/-/blob/main/Source%20Code/pysparktfidf.py?ref\\_type=heads](https://gitlab.computing.dcu.ie/bhardwp2/CA675_Cloud_Assignments_23_265772/-/blob/main/Source%20Code/pysparktfidf.py?ref_type=heads)

Now press escape and :wg then enter.

We are ready to submit the spark job-

```
[hadoop@ip-172-31-92-160 ~]$ vi tfidf5.py
[hadoop@ip-172-31-92-160 ~]$ submit-spark tfidf5.py
-bash: submit-spark: command not found
[hadoop@ip-172-31-92-160 ~]$ spark-submit tfidf5.py
```

**Note:** The file names in the screenshots are different because I tried multiple filenames with different codes

## **Running:-**

### **Results:-**

```
[hadoop@ip-172-31-92-160:~] |  
|ABC NewsMail - morning edition - text only | (10000, [1152,1169,1934,3160,7462,7976,8899], [2.772588722239781, 1.3862943611198906, 1.3862943611198906, 1.3862943611198906, 1.3862943611198906])  
|  
|[Update March 08: How could KiwiSaver stack up for you? | (10000, [2080,2509,2731,5230,6997,7751,8344,8643,9001,9850], [1.791759469228055, 1.791759469228055, 1.791759469228055, 1.791759469228055, 1.791759469228055])  
5,1.791759469228055,1.791759469228055,1.791759469228055,1.791759469228055,1.791759469228055,1.791759469228055])  
|  
|[Re: [Python-Dev] Proxy form not getting through | (10000, [145,2385,2540,5524,5621,7405,8909], [0.8754687373538999, 1.791759469228055, 1.791759469228055])  
8055,1.791759469228055,1.791759469228055,1.791759469228055,1.791759469228055])  
|  
|[UAI] Commonsense'07 Call For Papers | (10000, [2172,2248,5146,6357,8344], [1.791759469228055, 1.791759469228055, 1.791759469228055])  
759469228055,1.0986122886681098])  
|  
|Re: Scalar::Util::refaddr | (10000, [145,7093], [0.8754687373538999, 1.791759469228055])  
|  
|  
|[Re: PathTools beta for testing | (10000, [145,1708,8344,8609,9501], [0.8754687373538999, 1.791759469228055, 1.098612288668109])  
1759469228055,1.791759469228055))  
|  
|[Love package at a low prices | (10000, [2067,6240,6467,8297,8411,9162], [1.791759469228055, 1.791759469228055, 1.3862943611198906])  
,1.791759469228055,1.791759469228055,1.791759469228055])  
|  
+-----+  
  
23/11/15 12:14:47 INFO SparkContext: SparkContext is stopping with exitCode 0.  
23/11/15 12:14:47 INFO SparkUI: Stopped Spark web UI at http://ip-172-31-92-160.ec2.internal:4040  
23/11/15 12:14:47 INFO YarnClientSchedulerBackend: Interrupting monitor thread  
23/11/15 12:14:47 INFO YarnClientSchedulerBackend: Shutting down all executors  
23/11/15 12:14:47 INFO YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down  
23/11/15 12:14:47 INFO YarnClientSchedulerBackend: YARN client scheduler backend Stopped  
23/11/15 12:14:47 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!  
23/11/15 12:14:48 INFO MemoryStore: MemoryStore cleared  
23/11/15 12:14:48 INFO BlockManager: BlockManager stopped  
23/11/15 12:14:48 INFO BlockManagerMaster: BlockManagerMaster stopped  
23/11/15 12:14:48 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!  
23/11/15 12:14:48 INFO SparkContext: Successfully stopped SparkContext  
23/11/15 12:14:48 INFO ShutdownHookManager: Shutdown hook called  
23/11/15 12:14:48 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-dc65c360-6419-4526-8649-7c90d4b89b5a  
23/11/15 12:14:48 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-dc65c360-6419-4526-8649-7c90d4b89b5a/pyspark-83d17a33-d6e5-412f-a9ee-0fa4b7  
23/11/15 12:14:48 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-d9d4aef5-f2ce-45d4-a7c2-41f0f5189591  
[hadoop@ip-172-31-92-160 ~]$
```

### **TFIDF for Hams:-**

Similarly for Hams TFIDF, I used the same program just changed the file location.

## Spark job submit & Running:-

```
[hadoop@ip-172-31-92-160 ~]$ vi tfidfham.py
[hadoop@ip-172-31-92-160 ~]$ spark-submit tfidfham.py
23/11/15 12:38:11 INFO SparkContext: Running Spark version 3.4.1-amzn-2
23/11/15 12:38:11 INFO ResourceUtils: -----
23/11/15 12:38:11 INFO ResourceUtils: No custom resources configured for spark.driver.
23/11/15 12:38:11 INFO ResourceUtils: -----
23/11/15 12:38:11 INFO SparkContext: Submitted application: TFIDF
23/11/15 12:38:11 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: cores, amount: 4, script: , vendor: , memory -> name: memory, amount: 4269, script: , vendor: , offHeap -> name: offHeap, amount: 0, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount: 1,0)
23/11/15 12:38:11 INFO ResourceProfile: Limiting resource is cpus at 4 tasks per executor
23/11/15 12:38:11 INFO ResourceProfileManager: Added ResourceProfile id: 0
23/11/15 12:38:11 INFO SecurityManager: Changing view acls to: hadoop
23/11/15 12:38:11 INFO SecurityManager: Changing modify acls to: hadoop
23/11/15 12:38:11 INFO SecurityManager: Changing view acls groups to:
23/11/15 12:38:11 INFO SecurityManager: Changing modify acls groups to:
23/11/15 12:38:11 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: hadoop; groups with view permissions: EMPTY; users with modify permissions: hadoop; groups with modify permissions: EMPTY
23/11/15 12:38:12 INFO Utils: Successfully started service 'sparkDriver' on port 45199.
23/11/15 12:38:12 INFO SparkEnv: Registering MapOutputTracker
23/11/15 12:38:12 INFO SparkEnv: Registering BlockManagerMaster
23/11/15 12:38:12 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
23/11/15 12:38:12 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
23/11/15 12:38:13 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/11/15 12:38:13 INFO DiskBlockManager: Created local directory at /mnt/tmp/blockmgr-d49e9fd1-6b17-49e6-b351-2c9c935d4f81
23/11/15 12:38:13 INFO MemoryStore: MemoryStore started with capacity 912.3 MiB
23/11/15 12:38:13 INFO SparkEnv: Registering OutputCommitCoordinator
23/11/15 12:38:13 INFO SubResultCacheManager: Sub-result caches are disabled.
23/11/15 12:38:13 INFO JettyUtils: Start Jetty 0.0.0.0:4040 for SparkUI
23/11/15 12:38:14 INFO Utils: Successfully started service 'SparkUI' on port 4040.
23/11/15 12:38:14 INFO Utils: Using 50 preallocated executors (minExecutors: 0). Set spark.dynamicAllocation.preallocateExecutors to 'false' disable executor preallocation.
23/11/15 12:38:15 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at ip-172-31-92-160.ec2.internal/172.31.92.160:8032
[
```

## Output:-

```
[hadoop@ip-172-31-92-160 ~]
arents
23/11/15 12:39:16 INFO MemoryStore: Block broadcast_7 stored as values in memory (estimated size 199.4 KiB, free 911.1 MiB)
23/11/15 12:39:16 INFO MemoryStore: Block broadcast_7_piece0 stored as bytes in memory (estimated size 21.6 KiB, free 911.1 MiB)
23/11/15 12:39:16 INFO BlockManagerInfo: Added broadcast_7_piece0 in memory on ip-172-31-92-160.ec2.internal:36819 (size: 21.6 KiB, free: 912.2 MiB)
23/11/15 12:39:16 INFO SparkContext: Created broadcast 7 from broadcast at DAGScheduler.scala:1592
23/11/15 12:39:16 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 3 (MapPartitionsRDD[21] at showString at NativeMethodAccessorImpl.java:0) (first 15 tasks are for partitions Vector(0))
23/11/15 12:39:16 INFO YarnScheduler: Adding task set 3.0 with 1 tasks resource profile 0
23/11/15 12:39:16 INFO TaskSetManager: Starting task 0.0 in stage 3.0 (TID 3) (ip-172-31-88-65.ec2.internal, executor 1, partition 0, RACK_LOCAL, 8067 bytes)

23/11/15 12:39:16 INFO BlockManagerInfo: Added broadcast_7_piece0 in memory on ip-172-31-88-65.ec2.internal:33187 (size: 21.6 KiB, free: 2.1 GiB)
23/11/15 12:39:17 INFO BlockManagerInfo: Added broadcast_6_piece0 in memory on ip-172-31-88-65.ec2.internal:33187 (size: 42.3 KiB, free: 2.1 GiB)
23/11/15 12:39:18 INFO TaskSetManager: Finished task 0.0 in stage 3.0 (TID 3) in 2994 ms on ip-172-31-88-65.ec2.internal (executor 1) (1/1)
23/11/15 12:39:18 INFO DAGScheduler: ResultStage 3 (showString at NativeMethodAccessorImpl.java:0) finished in 2.320 s
23/11/15 12:39:18 INFO DAGScheduler: Job 3 is finished. Cancelling potential speculative or zombie tasks for this job
23/11/15 12:39:18 INFO YarnScheduler: Removed TaskSet 3.0, whose tasks have all completed, from pool
23/11/15 12:39:18 INFO YarnScheduler: Killing all running tasks in stage 3: Stage finished
23/11/15 12:39:18 INFO DAGScheduler: Job 3 finished: showString at NativeMethodAccessorImpl.java:0, took 2.326102 s
23/11/15 12:39:18 INFO CodeGenerator: Code generated in 40.559035 ms
+-----+
|text_column          |features
|-----+
|main char h          |(10000,[1637,2486,9033],[1.7047480922384253,1.7047480922384253,1.7047480922384253])
|Park with slides and More |(10000,[2891,6629,7650,9234,9346],[1.7047480922384253,1.2992829841302609,1.2992829841302609,1.7047480922384253,1.047480922384253])
|FREE medical consultations |(10000,[1073,2660,8268],[1.7047480922384253,1.7047480922384253,1.7047480922384253])
|http://nl.internet.com/ct.html?rtr=on&s=1|(10000,[1331],[0.6061358035703155])
|More than 10 features |(10000,[3261,6629,7953,9755],[1.7047480922384253,1.2992829841302609,1.7047480922384253,1.7047480922384253])
|http://nl.internet.com/ct.html?rtr=on&s=1|(10000,[1331],[0.6061358035703155])
|Duke University with |(10000,[3229,5357,7650],[1.7047480922384253,1.7047480922384253,1.2992829841302609])
|http://nl.internet.com/ct.html?rtr=on&s=1|(10000,[1331],[0.6061358035703155])
|http://nl.internet.com/ct.html?rtr=on&s=1|(10000,[1331],[0.6061358035703155])
|http://nl.internet.com/ct.html?rtr=on&s=1|(10000,[1331],[0.6061358035703155])
```

All other relevant screenshots can be found here:-[https://gitlab.computing.dcu.ie/bhardwp2/CA675\\_Cloud\\_Assignments\\_23265772/-/tree/main/Screensnips?ref\\_type=heads](https://gitlab.computing.dcu.ie/bhardwp2/CA675_Cloud_Assignments_23265772/-/tree/main/Screensnips?ref_type=heads)

## **Appendix:-**

### **AWS**

All the tasks are done on the AWS Cloud instance running EMR. All the configurations are done manually.

### **References**

1. TF-IDF:

<https://towardsdatascience.com/tf-idf-calculation-using-map-reduce-algorithm-in-pyspark-e89b5758e64c>

2. Apache Hive: <https://www.tutorialspoint.com/hive/index.htm>

3. Stack Overflow: <https://stackoverflow.com/>

4:-Chatgpt:-chat.openai.com

## **Acknowledgement:-**

I thank prof. Micheal Scriney for guidance.

