Distributed Computing and Architecture Project

VAIBAV PALIWAL, NLEMBA LAURENT

January 12, 2019

**Contents**

1. Installation ……………………………………………………………………... 3
2. Code Description……………………………………………………………… 5
3. IMDB Rating……………………………………………………………………… 5
   1. Task-1………………………………………………………………….5

3.2 Task-2………………………………………………………………….6

4. Retail Online……………………………………………………………………... 6

4.1 Task-3………………………………………………………………….7

4.2 Task-4…………………………………………………………………8

5. Text Similarity…………………………………………………………………….9

5.1 Task-5…………………………………………………………………9

5.2 Task-6………………………………………………………………...10

6. Matrix Multiplication………………………………………………………… 10

6.1 Task-8…………………………………………………………………11

7. Reference……………………………………………………………………………12

8. Contribution………………………………………………………………………..13

Installation:

1. **Python Libraries:** For this project we are going to use some of the famous libraries of Python.
   1. **MRJob:** MRJob lets you write MapReduce jobs in Python 2.7/3.4+ and run them on several platforms.

The following are some features of mrjob that make it easier to run MapReduce.

1. Keep all MapReduce code for one job in a single class

2. Easily upload and install code and data dependencies at runtime

3. Switch input and output formats with a single line of code

4. Write multi-step MapReduce jobs in pure Python

5. Test on your local machine

6. Run on a Hadoop cluster

7. Run in the cloud using [Amazon Elastic MapReduce (EMR)](http://aws.amazon.com/documentation/elasticmapreduce/)

8. Run in the cloud using [Google Cloud Dataproc (Dataproc)](https://cloud.google.com/dataproc/overview)

To install MRJob, you can use following command (pip install mrjob).

1.2 **NLTK:** Another important platform used in this project is “nltk”. It provides suite for text processing

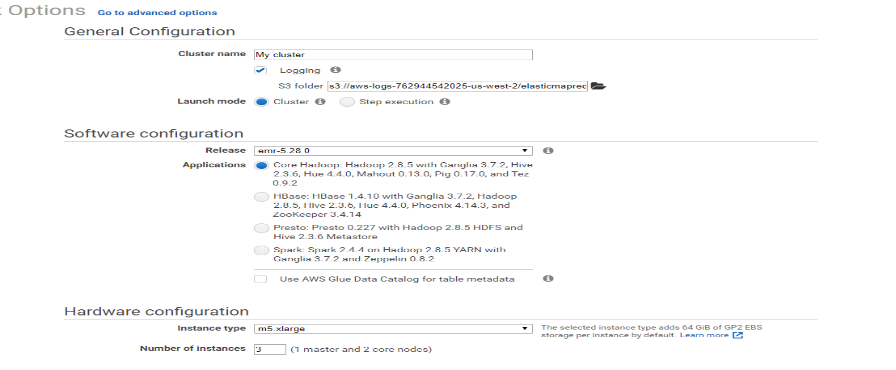
Libraries for classification and tokenization. Main function we used is stop words, count vectorization tf-idf. To install nltk you can used following command (pip install nltk).

* 1. **Heapq:** This module provides an implementation of the heap queue algorithm, also known as the priority queue algorithm.
  2. **Amazon EMR:** Amazon EMR is the industry leading cloud-native big data platform for processing vast amounts of data quickly and cost-effectively at scale. Using open source tools such as [Apache Spark](https://aws.amazon.com/emr/features/spark/), [Apache Hive](https://aws.amazon.com/emr/features/hive/), [Apache HBase](https://aws.amazon.com/emr/features/hbase/), [Apache Flink](https://aws.amazon.com/blogs/big-data/use-apache-flink-on-amazon-emr/), [Apache Hudi (Incubating)](https://aws.amazon.com/emr/features/hudi/), and [Presto](https://aws.amazon.com/emr/features/presto/), coupled with the dynamic scalability of [Amazon EC2](https://aws.amazon.com/ec2/) and scalable storage of [Amazon S3](https://aws.amazon.com/s3/), EMR gives analytical teams the engines and elasticity to run Petabyte-scale analysis for a fraction of the cost of traditional on-premises clusters.

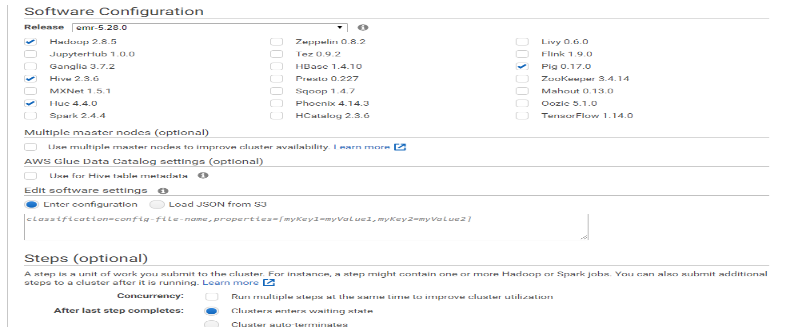
To run map reduce jobs you can either run from your tools using some commands and other way to create cluster in EMR and run mrjob.

Steps to create cluster in EMR:

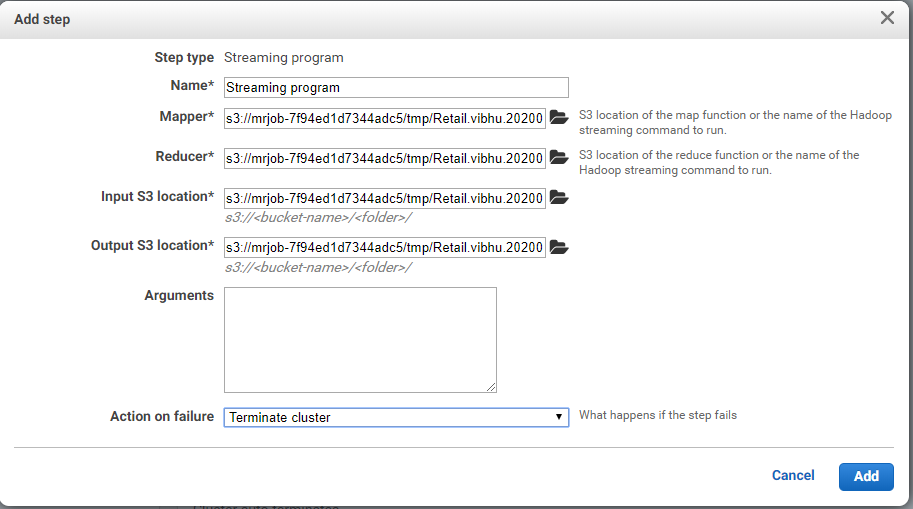
1. To create cluster in EMR, so for that login to the AWS and open EMR, where you will see the option of creating cluster. You need to click it and you will see:



1. Now you need to go to the advanced settings.



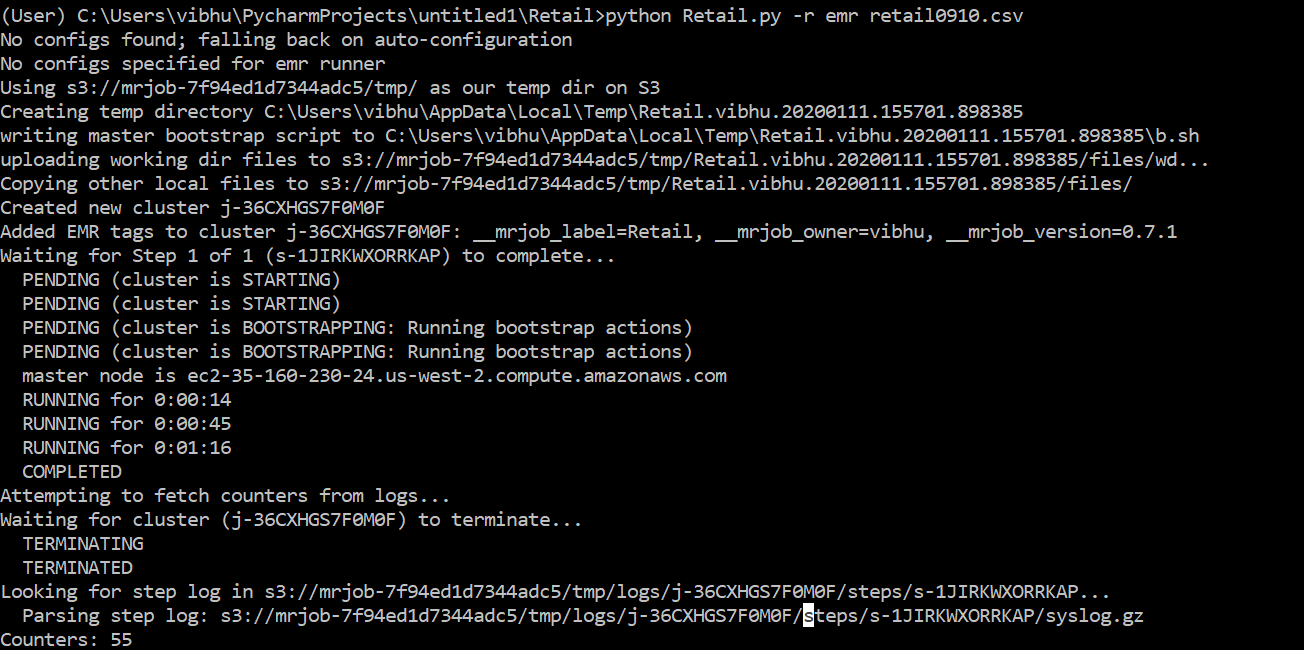
1. Now go to step type option where you need to tell EMR about your mapper, reducer, input, output file location.



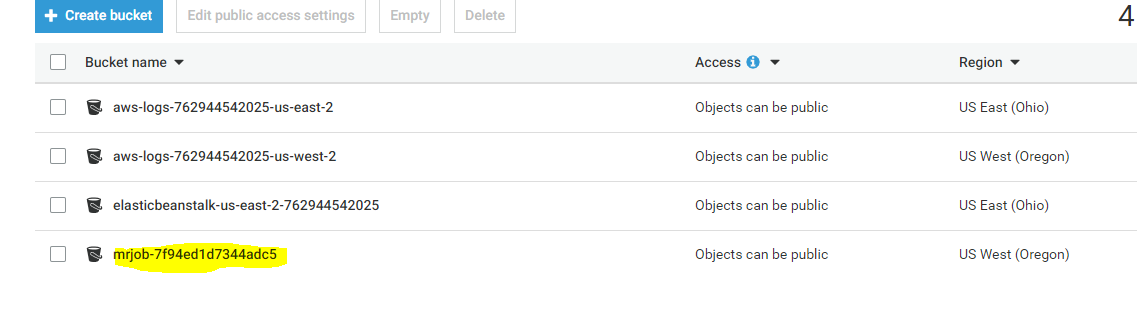
We have used another way to run my mrjob on EMR Hadoop cluster i.e. with commands from the tool. The tool that we used in my project is Canopy. So, for this option firstly you need to save AWS Access key and Secret key in environmental variables. Then you need to open the Canopy command prompt and run the following command:

*Python file\_name.py -r emr file.format*

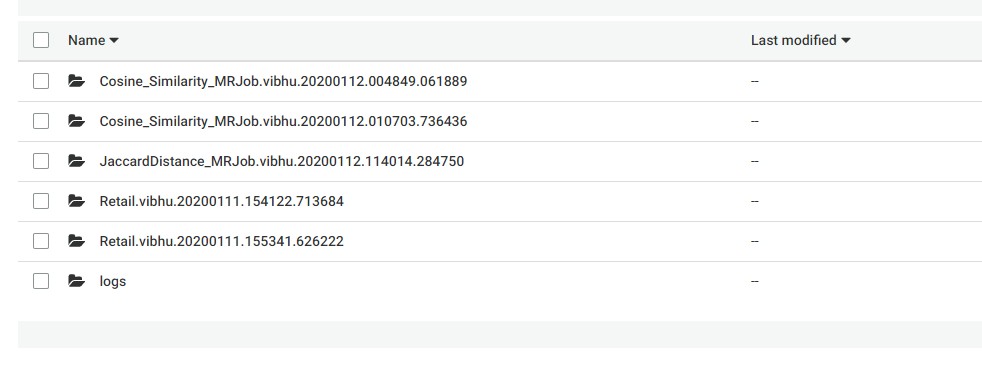
For example:



When you run your mrjob with this option, it will automatically save your files in AWS S3 bucket and process your mrjob on the cluster. The highlighted one is my mrjob bucket in S3.



After running the command mentioned above it will automatically create cluster in EMR, as you can see mine.



**CODE Description:**

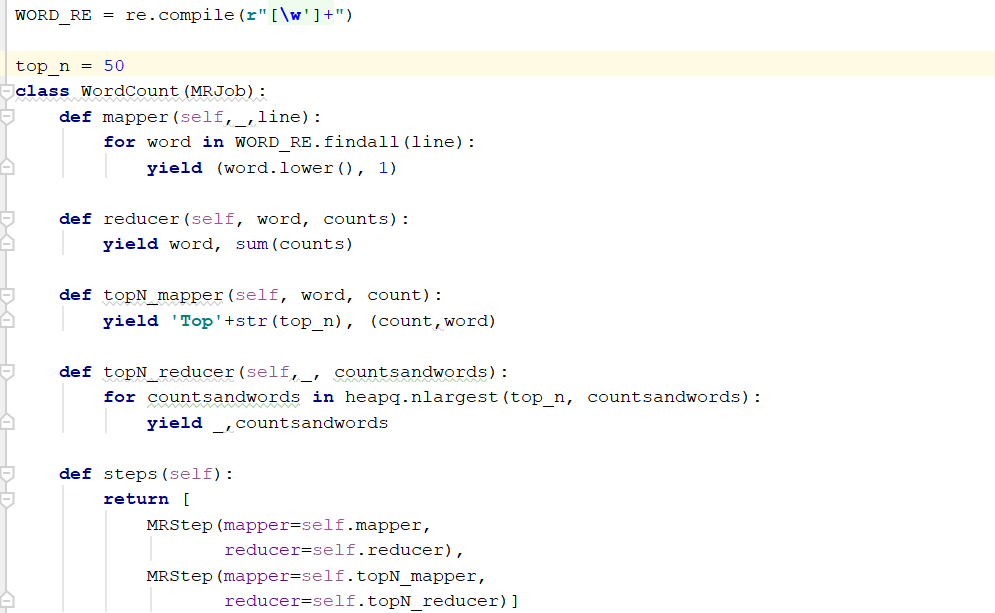
We used Canopy, PyCharm and Jupyter Notebook tools to run and preprocess the mrjob and given files. Command to run scripts in Python console: *!python python\_file.py file.format*

IMDB Rating:

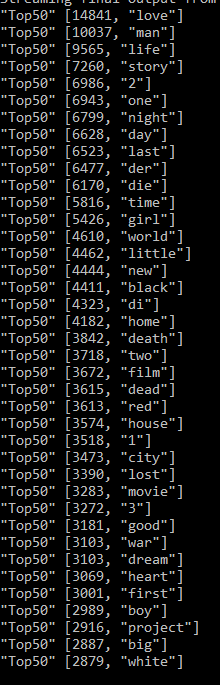
**Task-1:**

For this task we are provided with text file with details about movies, shorts, TV-shows. Our task one is to calculate top 50 most common keywords used in primary titles for the type movie and short using MapReduce. So for this firstly we did some preprocessing and create new files from them. Then we used that files to run on our MRJob.

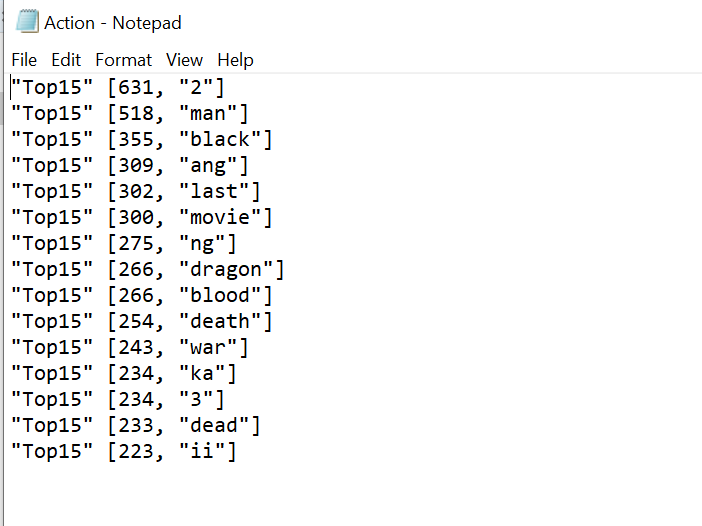
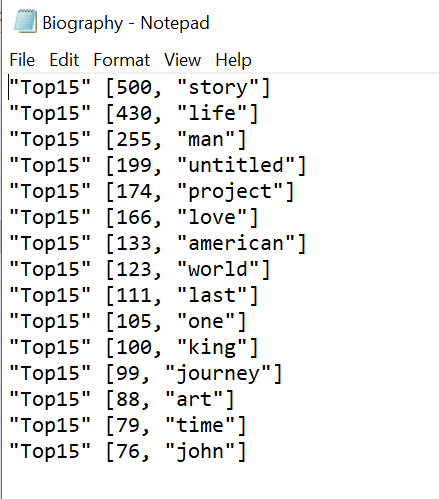
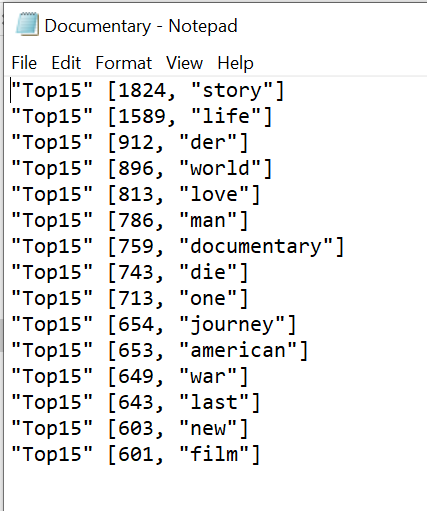


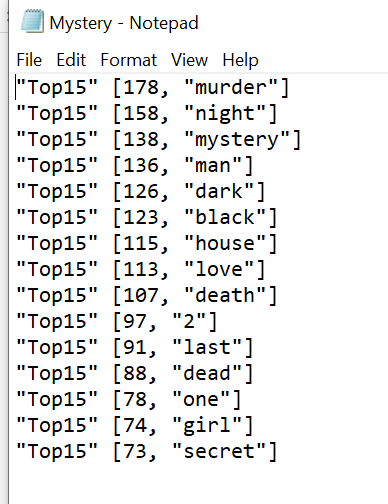


All our new files and output files are attached in submission. The result for this task is below.



**Task-2:** For each possible genre of the type movie, find the 15 most common keywords within their primary titles. For this we are showing only few outputs, as there are many genres. We have attached all preprocessed and output files for submission. Please have a look.

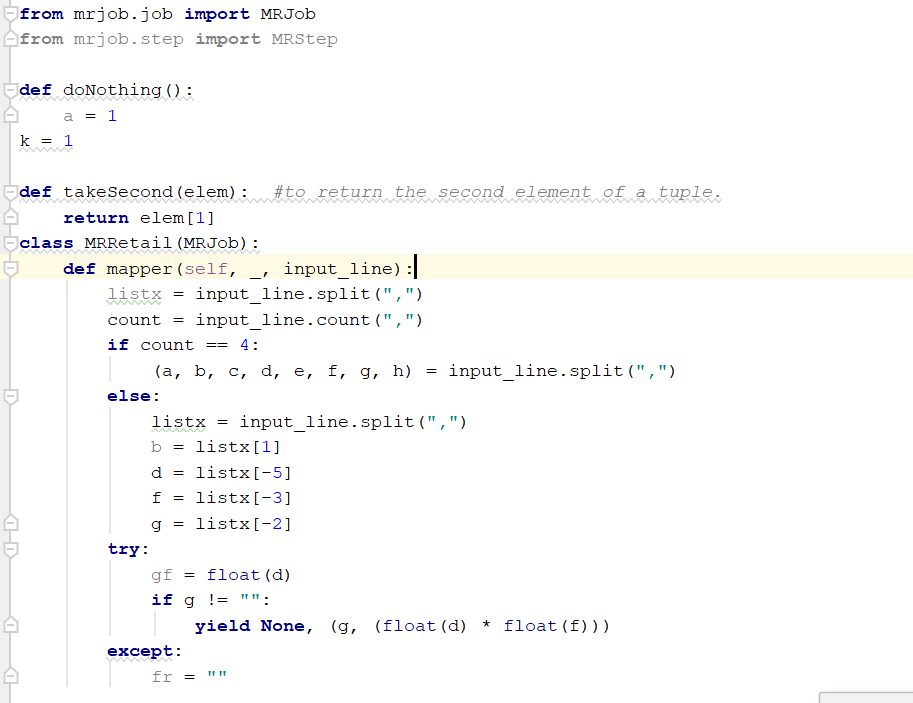
  



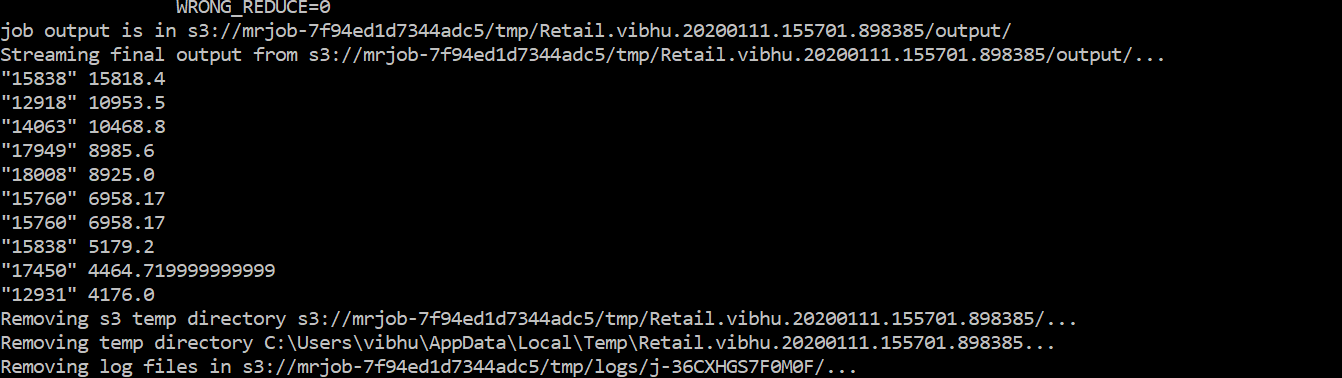
**Online Retail:**

**Task-3:** For this task we need to find top 10 customers that bought the most in terms of total revenue (quantity \* price).

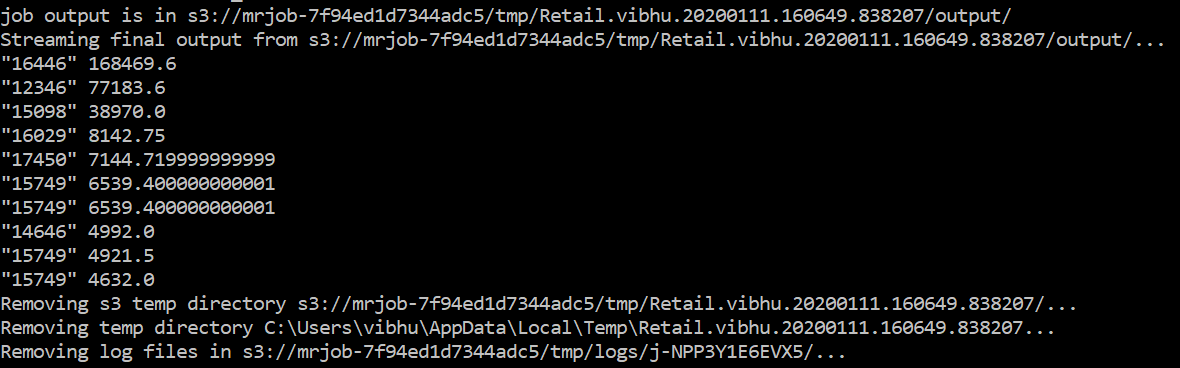
Python Code:



Output result: retail0910.csv



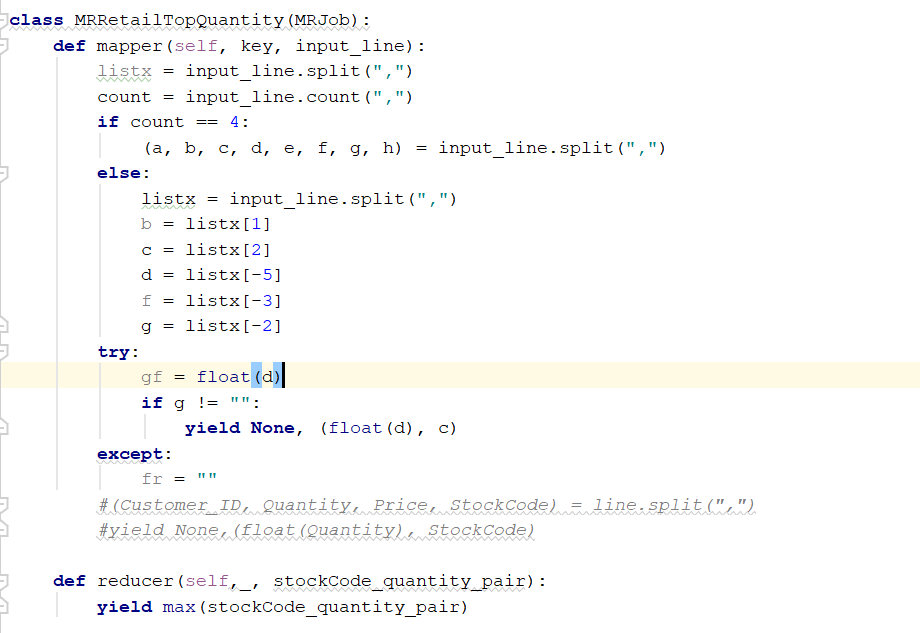
Output result: retail1011.csv



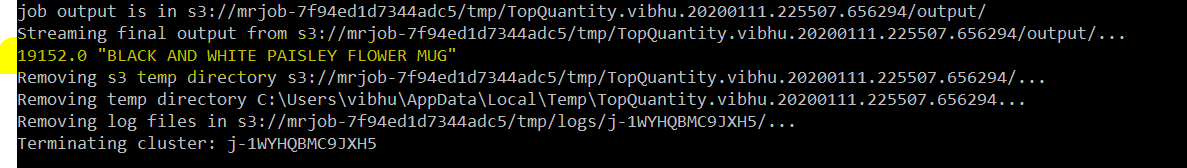
Task-4: Find best-selling product once in terms of total quantity,

and once in terms of total revenue for both retail years.

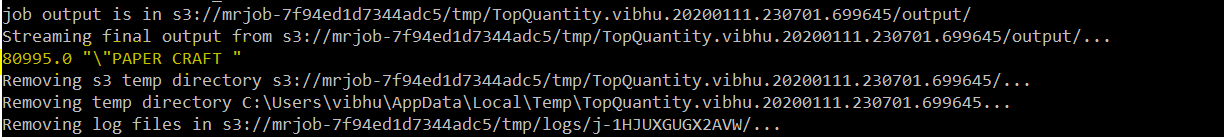
Python Code for Quantity:



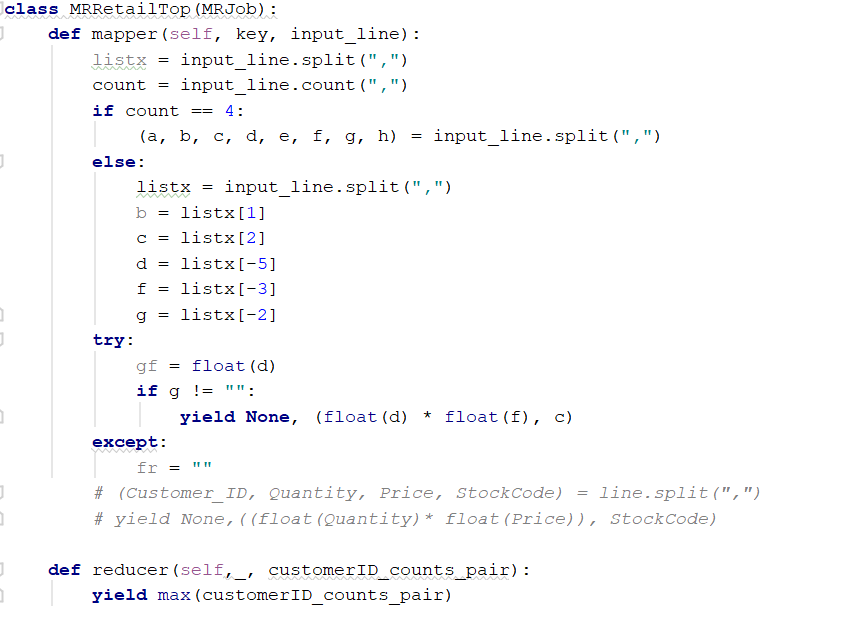
Output result: retail0910.csv



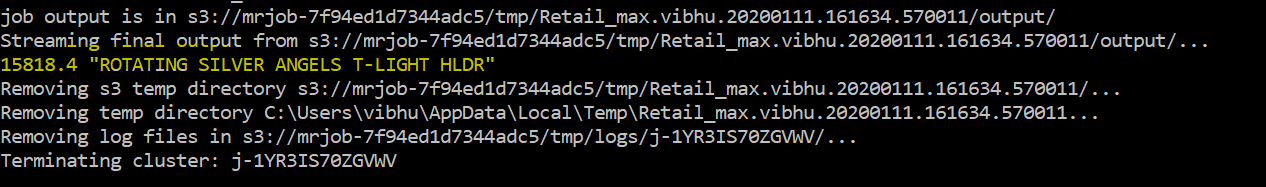
Output result: retail1011.csv



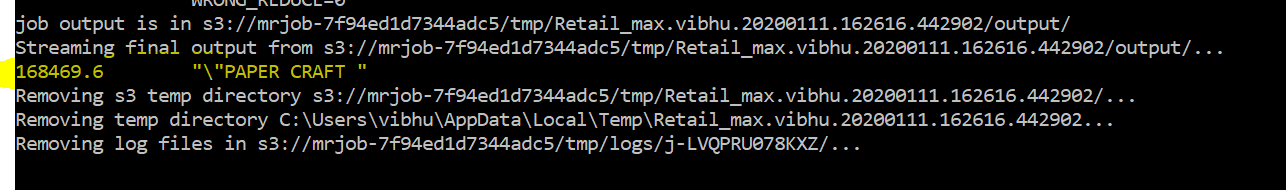
Python code for Revenue:



Output result: retail0910.csv



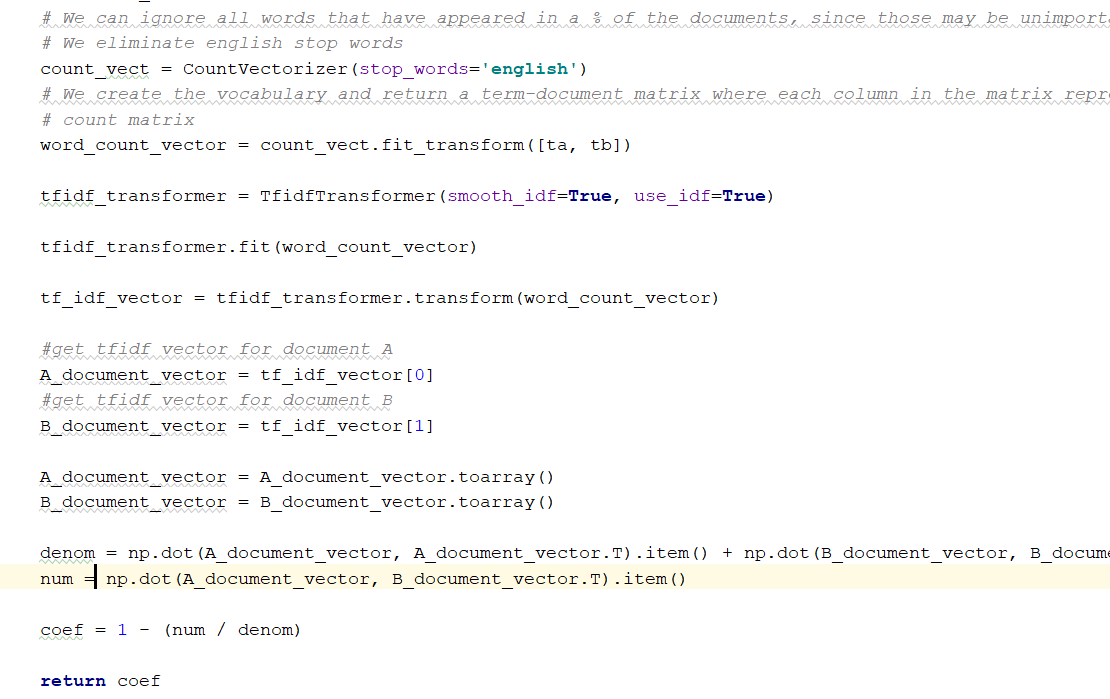
Output result: retail1011.csv

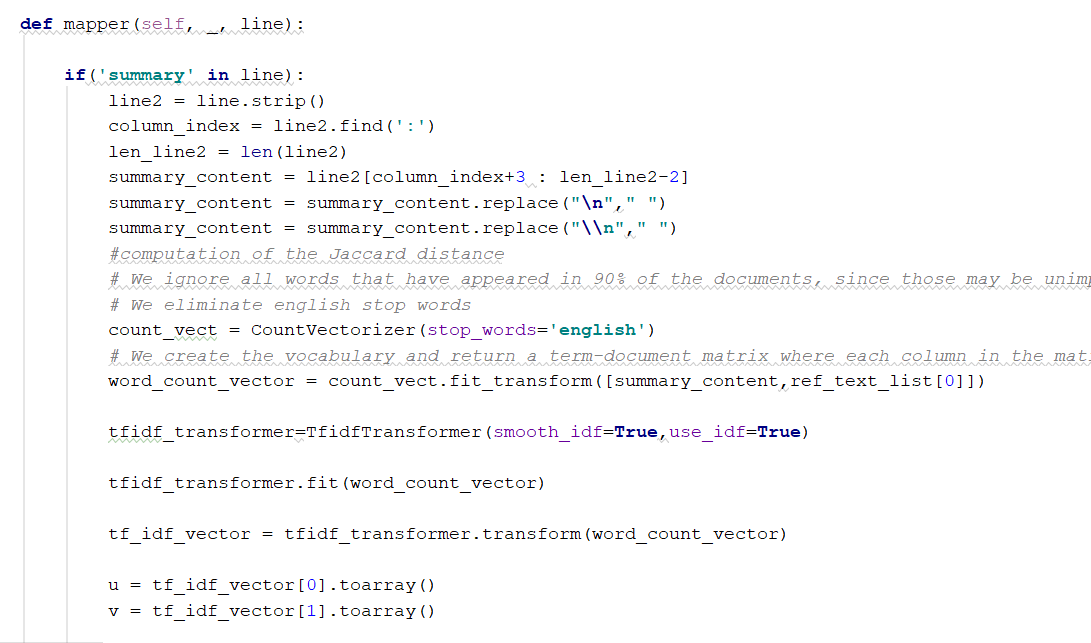


**Similar paper recommendations**

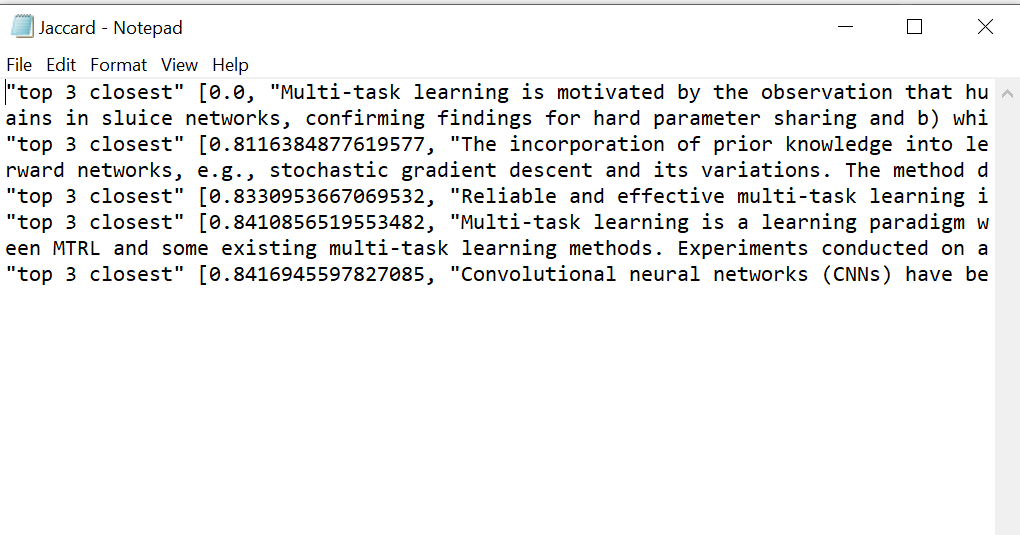
Task-5: Find the paper with the closest Jaccard Distance w.r.t summary.

Python code:



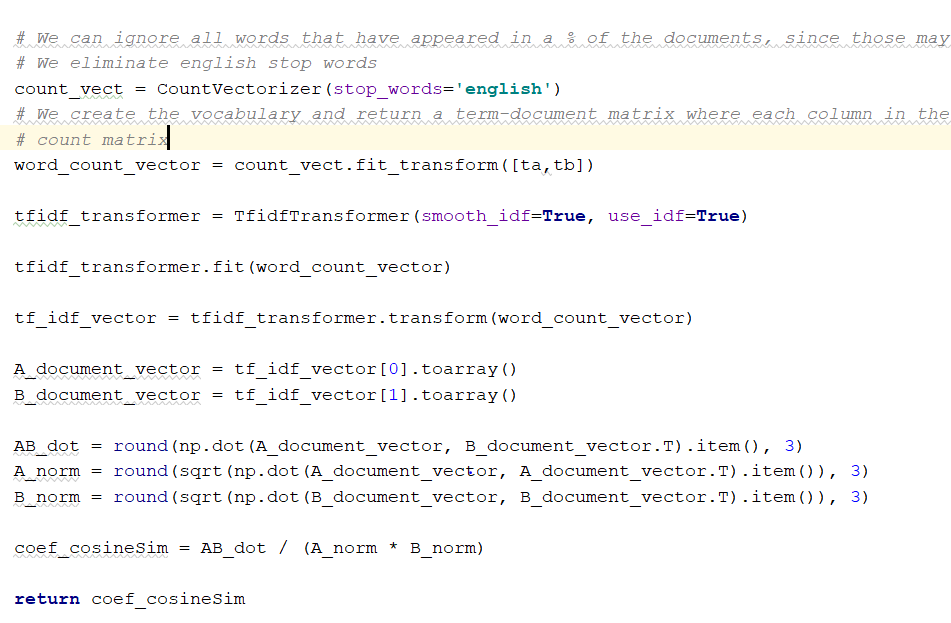


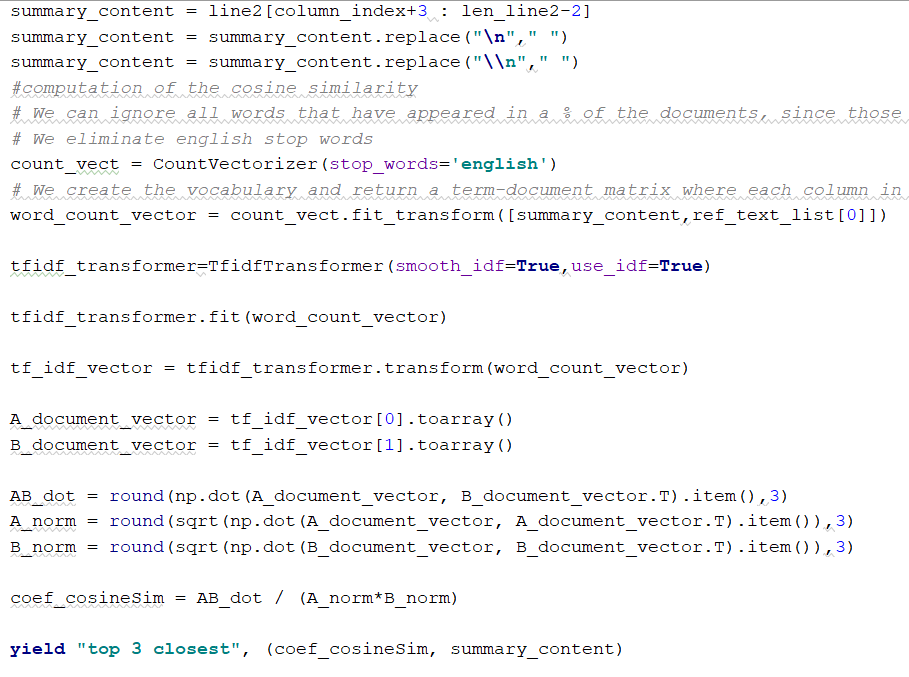
**Output result:** To check the full output please have a look on output text file (Jaccard.txt) attached with submission.



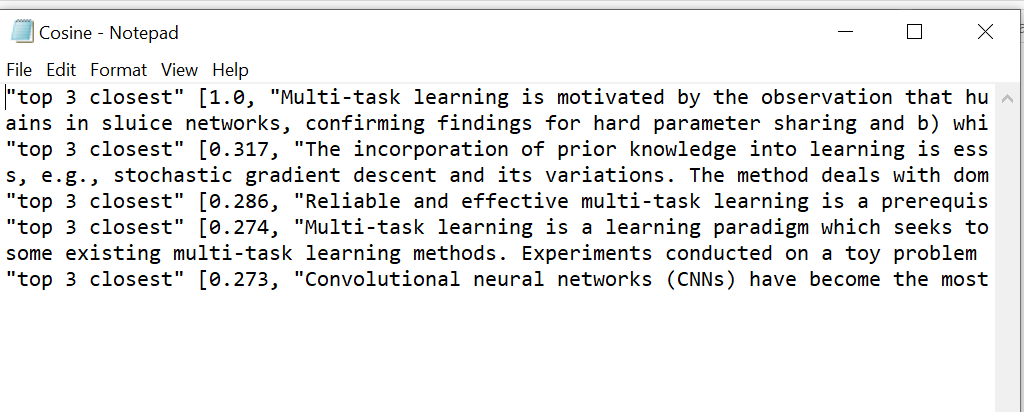
Task-6: Find the paper with the highest cosine similarity score w.r.t summary.

Python Code:





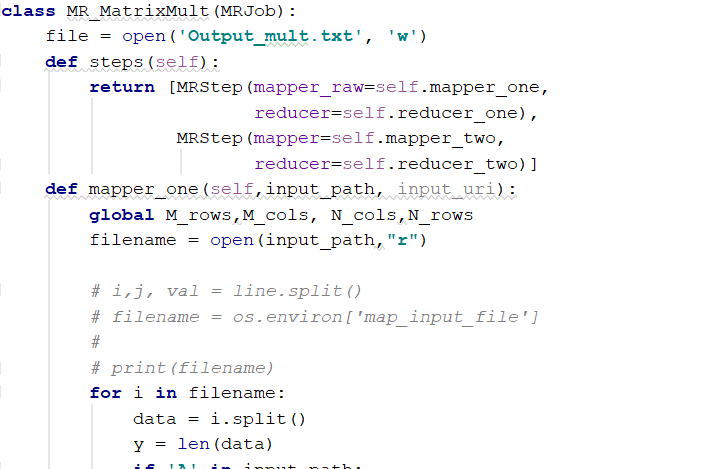
Output result: To check the full output please have a look on output text file (Cosine.txt) attached with submission.



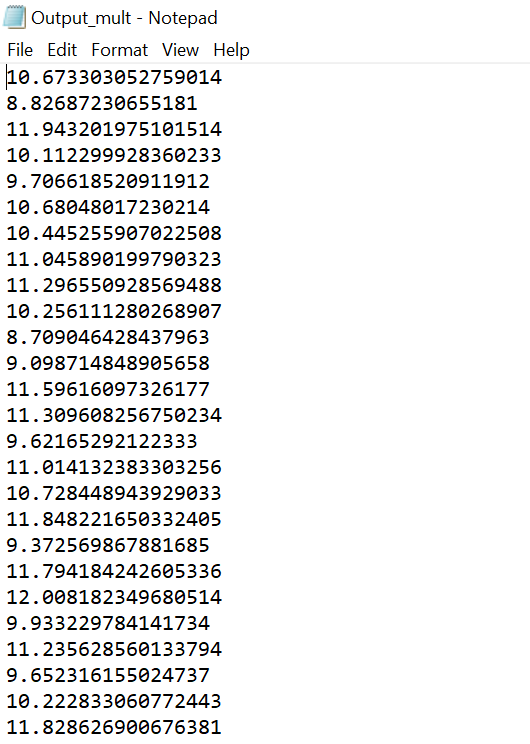
**Matrix Multiplication**

Task-8: Implement the matrix dot product in MapReduce.

Python Code:



Output Result: This matrix result is so big to visualize fully. Please refer to the attached file. The sample picture is:



**Reference:**

<https://docs.python.org/3/library/heapq.html>

<https://aws.amazon.com/emr/>

<https://mrjob.readthedocs.io/en/latest/>

<https://www.statisticshowto.datasciencecentral.com/jaccard-index/>

<https://www.nltk.org/>

<https://en.wikipedia.org/wiki/Cosine_similarity>

**Contribution:**

This project has been done in a teamwork manner that is both participated in almost everything together. We divided the task among us but we both tried to do them individually also. We are happy to say that we invested equal time into this project, we learned from each other mistakes and achievements. This help us to reach the final goal.