fFinding Similar News Article Headlines Using Pyspark

In this problem, we are still going to use the dataset of Australian news from ABC. Similar news may appear in different years. Your task is to find all similar news article headline pairs across different years.

Background: Set similarity self-join

Given a collection of records R, a similarity function **sim**(., .), and a threshold τ, the set similarity self-join on R, is to find all record pairs r and s from R, such that **sim**(r, s) >= τ. In this project, you are required to use the Jaccard similarity function to compute **sim**(r, s). Given the following example, and set τ=0.5,

id	record
0	1 4 5 6
1	2 3 6
2	4 5 6
3	1 4 6
4	2 5 6
5	3 5

The result pairs are:

pair	similarity
(0,2)	0.75
(0,3)	0.75
(1,4)	0.5
(2,3)	0.5
(2,4)	0.5

Input files:

In the file, each line is a headline of a news article, in format of "date,term1 term2 ... ... ". The date and texts are separated by a comma, and the terms are separated by the space character (note that the stop words have been removed already). A sample file is like below:

```
20191124,woman stabbed adelaide shopping centre
20191204,economy continue teetering edge recession
20200401,coronanomics learnt coronavirus economy
20200401,coronavirus home test kits selling chinese community
20201015,coronavirus pacific economy foriegn aid china
20201016,china builds pig apartment blocks guard swine flu
20211216,economy starts bounce unemployment
20211224,online shopping rise due coronavirus
20211229,china close encounters elon musks
```

This sample file “tiny-data.txt” can be downloaded at:
https://webcms3.cse.unsw.edu.au/COMP9313/23T2/resources/88356

Note that it is possible that one term appears multiple times in a headline.

Output:

The output file contains all the similar headlines together with their similarities. In each pair, the headlines must be from different years. Please use the index of the headline in the file as its ID (starting from 0) and use the IDs to represent the headline pairs. Each line is in format of "(Id_1,Id_2)\tSimilarity" (Id_1<Id_2, and there should have no duplicate pairs in the result). The similarities are of double precision. The pairs are sorted in ascending order (by the first and then the second).

Given the example input data with threshold 0.1, the final result should be:

```
(0,7)\t0.1111111111111111
(1,2)\t0.125
(1,4)\t0.1
(1,6)\t0.125
(2,6)\t0.14285714285714285
(2,7)\t0.125
(4,6)\t0.1
(4,7)\t0.1
(4,8)\t0.1
```


Code format:

Please name your python file as "project3.py".

Command of running your codes:

Your program should take three parameters: the input file, the output folder, and the similarity threshold τ.

```
$ spark-submit project3.py input output tau
```

Please ensure that the code you submit can be compiled. Any solution that has compilation errors will receive no more than 6 marks.

Run in Google Dataproc - Cluster configuration:

Create a bucket with name "comp9313-<YOUR_STUDENTID>" in Dataproc. Create a folder "project3" in this bucket for holding the input files.

This project aims to let you see the power of distributed computation. Your code should scale well with the number of nodes used in a cluster. You are required to create three clusters in Dataproc to run the same job:

- Cluster1 - 1 master node and 2 worker nodes;
- Cluster2 - 1 master node and 4 worker nodes;

- Cluster3 - 1 master node and 6 worker nodes.

For both master and worker nodes, select n1-standard-2 (2 vCPU, 7.5GB memory).

Unzip and upload the following data set to your bucket and set τ to 0.85 to run your program:
https://webcms3.cse.unsw.edu.au/COMP9313/23T2/resources/89963.

Record the runtime on each cluster and draw a figure where the x-axis is the number of nodes you used and the y-axis is the time of getting the result, and store this figure in a file "Runtime.jpg". Please also take a screenshot of running your program on Dataproc in each cluster as a proof of the runtime. Compress the three screenshots into a zip file "Screenshots.zip".

Create a project and test everything in your local computer, and finally do it in Google Dataproc.

Marking Criteria

Your source code will be inspected and marked based on readability and ease of understanding. The efficiency and scalability of this project is very important and will be evaluated as well. Below is an indicative marking scheme:

Submission can be compiled and run on Spark: 6
Accuracy: 5 • No unexpected pairs • No missing pairs • Correct order • Correct similarity scores • Correct format
Efficiency: 9 • The rank of runtime (using two local threads): Correct results: 0.9 * (10 – **floor**((rank percentage-1)/10)), e.g., top 10% => 9 Incorrect results: 0.4 * (10 – **floor**((rank percentage-1)/10))
Code format and structure, Readability, and Documentation: 2 • The description of the optimization techniques used

Cautious:

- You need to design an exact approach to finding similar records.
- You cannot compute the pair wise similarities.
- Regular Python programming is not permitted in project3.
- When testing the correctness and efficiency of submissions, all the code will be run with two local threads using the default setting of Spark. Please be careful with your runtime and memory usage.