Robert Kaszubski

CSC 555 – Assignment 6

**1)**

a)

a)

Treating as Boolean as done during the lecture:

Ratings of 1 or 2 = No Rating

Ratings of 3, 4, or 5 = 1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| A | 1 | 1 |  | 1 |  |  | 1 |  |
| B |  | 1 | 1 | 1 |  |  |  |  |
| C |  |  |  | 1 |  | 1 | 1 | 1 |

Jaccard Similarity:

A & B = 2/5

A & C = 2/6

B & C = 1/6

Jaccard Distance: 1 – Jaccard Similarity

A & B = 3/5

A & C = 4/6

B & C = 5/6

e)

Normalize the matrix:

Avg for A: 3.33 or 3

Avg for B: 2.33 or 2

Avg for C: 3

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h |
| A |  | 1 |  | 1 | -2 |  |  | -1 |
| B |  |  | 1 |  | -1 |  | -1 |  |
| C | -1 |  | -2 | 0 |  | 1 | 2 | 0 |

b)

You could cluster users or items into smaller subsets. For instance, in the case of movie ratings, you could combine all ratings for one movie franchise into a single rating that represents the average of the movies of that franchise that the user has seen. The James Bond series has 25 films. If a user has only seen 5 of them, then there is a lot of missing values, and the matrix is sparse. Averaging those ratings, eliminates some of that sparseness as you have a single rating corresponding to all the James Bond films.

**2)**

a)

Spark has RDD or resilient distributed datasets. Resilient meaning that they are fault tolerant. Spark’s cache is fault tolerant. If any part of an RDD is lost, it is automatically recomputed meaning that no data is lost, it can recover the losses.

b)

Storm is fault tolerant. If a worker stops working, it is automatically restarted. If a node fails, the worker is restarted on another node. If a worker restarts its like nothing has happened and no data is lost. There is also replication meaning that each operation is stored multiple times so if one operator fails, a new one is there to take over and continue the stream and real time processing.

c)

The name node or master node is responsible for managing in Hadoop. It’ll allocate resources as needed. The spark tasks will run on the data or worker nodes

d)

I believe there is a mistake with the assignment writeup. The last average should be 4.25 not 4. This should be a 6 not a 5:

Text

Description automatically generated

My Sample Output with ( cat mydata | python storm.py 4 2 ):

Text

Description automatically generated

I printed the values that make up each set to verify, this can be commented out as seen below to give the same output that storm would:

Text

Description automatically generated

Using cat mydata | python storm.py 4 3:

Text

Description automatically generated with medium confidence

When it moves 3 tuples, there isn’t enough data for a third average.

Works with a different window as well:

Text

Description automatically generated with medium confidence

Code:

Text

Description automatically generated

**3)**

a)

The random text file was created using numbers from 1 to 100 and written to a file with the following python code (datagen.py):

A screenshot of a computer

Description automatically generated with medium confidence

The data was written to numdata. This was moved into hdfs then:

Command Line:

$MAHOUT\_HOME/bin/mahout org.apache.mahout.clustering.syntheticcontrol.kmeans.Job --maxIter 10 --numClusters 8 --t1 5000 --t2 3000 --input numdata --output kmeansRes

Last Page of results:

Text

Description automatically generated

A picture containing graphical user interface

Description automatically generated

b)

Split:

A screenshot of a computer

Description automatically generated with medium confidence

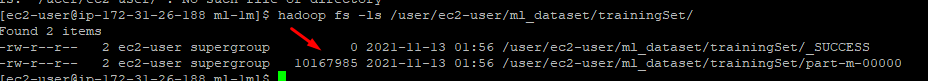
Original CSV file size:

Text

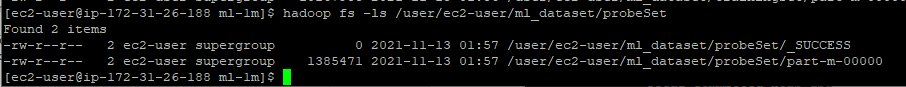
Description automatically generated

11553456 bytes

Sampled Files:



10167985 bytes



1385471 bytes

They add up exactly to 11553456, the size of the csv.

The RMSE is:



0.8860970765612165

Predictions:

Text

Description automatically generated

The top movie recommendation for user:

4 is 3737

6 is 572

**4)**

Installing spark to worker nodes:

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

Code:

text\_file = sc.textFile("hdfs://ec2-3-17-132-17.us-east-2.compute.amazonaws.com/data/bioproject.xml")

counts = text\_file.flatMap(lambda line: line.split(" ")).map(lambda word: (word, 1)).reduceByKey(lambda a, b: a + b)

counts.saveAsTextFile("hdfs://ec2-3-17-132-17.us-east-2.compute.amazonaws.com/data/output")

Output:

Text

Description automatically generated

Output was saved to two files

Sample output:

Text

Description automatically generated