MDLE: Assignment 2

– Due date: May 7th, 2025 –

For each of the following exercises, you should implement the solutions using pyspark. Use small samples of the dataset for developing and initial testing, then run on the full data.

What to submit

For each exercise, submit a documented Google Colab notebook or a python script to run through spark-submit, and the results of the algorithm. If the results are too large, submit a download link instead.

The comments should explain the main steps of the solution with sufficient detail.

A. Write a program that implements a simple "People You Might Know" social network friendship recommendation algorithm. The key idea is that if two people have a lot of mutual friends, then the system should recommend that they connect with each other.

   Data:

   • Associated data file is *soc-LiveJournal1Adj.txt* found in the shared folder.
   • The file contains the adjacency list and has multiple lines in the following format:

   `<User><TAB><Friends>`

   `<User>` is a unique integer `ID` corresponding to a unique user and `<Friends>` is a comma separated list of unique `IDs` corresponding to the friends of the user with the unique `ID <User>`. Note that the friendships are mutual: if A is friend with B then B is also friend with A.

   Implement a simple algorithm such that, for each user U, the algorithm recommends N = 10 users who are not already friends with U, but have the most number of mutual friends in common with U.

   The output should contain one line per user in the following format:

   `<User><TAB><Recommendations>`

   where `<User>` is a unique `ID` corresponding to a user and

<Recommendations> is a comma separated list of unique IDs corresponding to the algorithm's recommendation of people that <User> might know, ordered in decreasing number of mutual friends.

Even if a user has less than 10 second-degree friends, output all of them in decreasing order of the number of mutual friends. If a user has no friends, you can provide an empty list of recommendations. If there are recommended users with the same number of mutual friends, then output those user IDs in numerically ascending order.

B. The file 'conditions.csv.gz' (available on the shared folder) lists conditions for a large set of patients. Our purpose is to find associations between conditions.

The file contains the following fields, with multiple non-consecutive entries for each patient: START,STOP,PATIENT,ENCOUNTER,CODE,DESCRIPTION

PATIENT is the patient identifier

CODE is a condition identifier

DESCRIPTION is the name of the condition

You may prefer to reorganize the data before applying the algorithms.

Implement the A-Priori algorithm in pyspark.

1. Apply the implemented algorithm with a support threshold of 1000 to obtain the frequent itemsets for sizes $k = 2$ and $k = 3$. Include in your results the lists of the 10 most frequent itemsets for $k = 2$ and $k = 3$.

2. Obtain associations between conditions by extracting rules of the forms $(X) \rightarrow Y$ and $(X, Y) \rightarrow Z$, with minimum standardized lift of 0.2. Write the rules to a text file, showing the standardized lift, lift, confidence and interest values, sorted by standardized lift.

C. Implement the BFR algorithm and apply it to the FMA dataset.

The FMA dataset consists of 106,574 music tracks represented by 518 features, corresponding to 7 statistics (mean, standard deviation, skew, kurtosis, median, minimum, maximum) calculated from 74 time-based acoustic characteristics. See the dataset description for more details.

The first three lines in the features.csv file identify the feature in each column: the first line indicates the feature group, the second line the

statistics, and the third line the feature number within each feature group.

Some examples of feature identifiers are then: `tonnetz_skew_06`, `chroma_cqt_kurtosis_09`,`mfcc_max_08`,`spectral_contrast_median_07`.

1. Apply an in-memory agglomerative hierarchical clustering algorithm to the small subset (8000 songs) and calculate the radius, diameter and density (using r2 and d2) of all clusters for values of k between 8 and 16.
2. From the results of 1. Select the best number of clusters k and cluster the complete dataset using the BFR algorithm.
3. Using the songs metadata (tracks.csv) create a representation of the cluster in terms of the most common music genres found in the cluster.

Notes: You can integrate any existing implementation of hierarchical clustering. To select the small dataset, filter by the field subset in the tracks.csv file