Data Science

Programming Assignment #1

컴퓨터전공

2013011695 정태화

1. Goal

Find association rules using the Apriori algorithm.

2. Summary of algorithm

This program goes through following procedure :

1. Read the entire transaction data from the input file.
2. Create a frequent pattern set starts with (L=1) element.
3. Use the Apriori algorithm to create (L=2,3,4, ... , N) elements.
4. Increase N until no more frequent patterns are created.
5. Generate every sets' association rule using this frequent pattern set.
6. Write this association rules into output file.

3. Detailed description of code

A. `vector< set<int> > readTransactions(string p)`

This function reads data from given input file, and parameter 'string p' is for input file path.

Reading each line as transaction, this function parses every line with delimiter '\t' and inserts every items in each transaction into set.

And then it pushes each set into vector so that this function can return the entire transaction as vector format.

B. `void dfs(int start, int length, set<int> &items ,vector< set<int> > &result)`

Executes DFS(Depth-First Search) to generate subsets.

This function generates specified 'length' of subsets from set 'items', and pushes those subsets into vector 'result'.

This dfs function will be used under circumstances such as checking infrequent subset, or making association rules.

C. `map< set<int>, int > makefreqMap(double &min_sup, vector< set<int> > transactions)`

Generates (L=N) length frequent pattern set until there is no more frequent pattern.

At first, make (L=1) frequent set, and using this (L=1) set, make (L=2) candidate set by self-joining. With every (L=2) candidate set, start pruning by checking if any subset of each set is infrequent. To do this, you have to hold (L-1) length candidate set.

After pruning, count how many times this set appears in the entire transaction, and if this count is under minimum support, prune off this set, too.

Repeat these procedures again and again until there is no more frequent pattern, and return completed frequent pattern set in the form of map.

D. `void makeAscRule(map< set<int>, int > &freqMap, string outPath)`

Using previous completed frequent pattern set, generate association rule for every sets.

For example, if a set is {1,2,3}, you have to generate rule for every divided subsets like {1}->{2,3}, {2}->{1,3}, {3}->{1,2}, {1,2}->{3}, {2,3}->{1}, {1,3}->{2}.

Use dfs function to divide each set, and calculate each rule's support and confidence. If set 'A' is divided into 'A1' and 'A2', support of rule [A1->A2] is Support(A), and confidence is support(A) / support(A1).

$$Support[A1 \rightarrow A2] = Support(A)$$
$$Confidence[A1 \rightarrow A2] = \frac{Support(A)}{Support(A1)}$$

After calculation, write each rules into specified output file path in the form of given file format.

4. Instructions for compiling this code

- This project contains ‘apriori.cpp’, ‘Makefile’, and ‘input.txt’.

- In project folder path, just type ‘make’ in terminal, or please type below line. This will generate executable file ‘apriori’ for linux.

```
$ g++ -O2 -o apriori apriori.cpp --std=c++11
```

- Now you will be able to execute this file with specified arguments.

```
$ ./apriori 5 input.txt output.txt
```

5. Any other specifications

- This code is written in C++11.

- Compiler must support C++11 standard.

- This program is compiled with g++ and xcode.

- This program compilation is tested on macOS High Sierra.