Mining Frequent Patterns in Transactional Databases

What is frequent pattern mining?

Frequent pattern mining aims to discover all interesting patterns in a transactional database that have **support** no less than the user-specified **minimum support** (**minSup**) constraint. The **minSup** controls the minimum number of transactions that a pattern must appear in a database.

The first paper of frequent pattern mining : `https://rakesh.agrawal-family.com/papers/sigmod93assoc.pdf`

What is the transactional database?

A transactional database is a collection of transactions, where each transaction contains a transaction-identifier and a set of items.
A hypothetical transactional database containing the items ***a, b, c, d, e, f, and g*** as shown below

tid	Transactions
1	a b c g
2	b c d e
3	a b c d
4	a c d f
5	a b c d g
6	c d e f
7	a b c d
8	a e f
9	a b c d
10	b c d e

Note: Duplicate items must not exist in a transaction.

Acceptable format of transactional databases in PAMI

Each row in a transactional database must contain only items. PAMI algorithms implicitly consider the row number as the transactional-identifier to reduce storage and processing costs.

a b c g
b c d e
a b c d
a c d f
a b c d g
c d e f
a b c d
a e f
a b c d
b c d e

Understanding the statisctics of database

To understand about the database. The below code will give the detail about the transactional database.

- Total number of transactions (Database size)
- Total number of unique items in database
- Minimum lenth of transaction that existed in database
- Average length of all transactions that exists in database
- Maximum length of transaction that existed in database
- Standard deviation of transaction length
- Variance in transaction length
- Sparsity of database

The sample code

```
import PAMI.extras.dbStats.transactionalDatabaseStats as stats

obj = stats.transactionalDatabaseStats('sampleInputFile.txt', ' ')
obj.run()
obj.printStats()
```

What is the input to frequent pattern mining algorithms

Algorithms to mine the frequent patterns requires transactional database and minSup (specified by user).

- Transactional database in following formats:

  > - In string format (`/Users/Likhitha/Downlaods/sampleInputFile.txt')
  > - In URL format (`https://www.u-aizu.ac.jp/~udayrage/datasets/transactionalDatabases/transactional_T10
  > - In DataFrame format (dataframe variable with heading `Transactions')

- minSup should be mentioned in **count (beween 0 to length of database)** or __percentage (multiplied with length of database)

What is the output of frequent pattern mining algorithms

The output of these algorithms is in two ways:

> - Saving the patterns in user specified output file.
> - Returns the patterns in dataframe variable.

How to run the frequent pattern algorithm in terminal

- Download the code from github.
- Navigate to PAMI folder where you downloaded the file.
- Go to frequentPattern folder

You will different types of folders like **basic, closed, maximal, topk, cuda, pyspark** Go to specific folder you are intended to and execute the following command on terminal.

python3 algorithmName.py `path of Sample input file` `path of output file` minSup `seperator`

Sample command to execute the Apriori code in frequentPattern/basic folder

python3 `Apriori.py` `/Users/Donwloads/inputFile.txt` `/Users/Downloads/outputFile.txt` 3 `' '`

How to implement the code by importing PAMI package

Import the PAMI package executing: **pip3 install PAMI**

Run the below sample code by making simple changes

- Replace sampleInputFile name or path in place of iFile and sampleOutputFile name or path in place of oFile
- Specify the minSup (like 10 or 0.1) in place of minSup
- Specify the seperator of input file after minSup. (If no seperator is specified the default tab seperator is considered for input file)

import PAMI.frequentPattern.basic.Apriori as alg
obj = alg.Apriori(iFile, minSup, ' ')
obj.startMine()
obj.savePatterns(oFile) (to store the patterns in file).
Df = obj.getPatternsAsDataFrame() (to store the patterns in dataframe)
obj.printStats()

What is the output of frequent pattern mining algorithms

Returns the pattern and support respectively with $minSup=5$

The output in file format:

b :7
b a :5
b a c :5
b d :6
b d c :6
b c :7
a :7
a d :5
a d c :5
a c :6
d :8
d c :8
c :9

The output in DataFrame format:

	Patterns	Support
0	b	7
1	b a	5
2	b a c	5
3	b d	6
4	b d c	6
5	b c	7
6	a	7
7	a d	5
8	a d c	5
9	a c	6
10	d	8
11	d c	8
12	c	9