

Mining High-Utility Frequent Patterns in Utility Databases

What is High-Utility Frequent pattern mining?

High utility frequent pattern mining aims to discover all the patterns with **utility** of pattern is no less than user-specified **minimum utility** (*minutil*) and **support** is no less than user-specified **minimum support** (*minSup*).

What is the utility database?

A utility database is a collection of transaction, where each transaction contains a set of items and a positive integer called **internal utility** respectively. And each unique item in database is also associated with another positive number called **external utility**

Transactions	Item	Profit
(a,2) (b,3) (c,1) (g,1)	a	4
(b,3) (c,2) (d,3) (e,2)	b	3
(a,2) (b,1) (c,3) (d,4)	c	6
(a,3) (c,2) (d,1) (f,2)	d	2
(a,3) (b,1) (c,2) (d,1) (g,2)	e	5
(c,2) (d,2) (e,3) (f,1)	f	2
(a,2) (b,1) (c,1) (d,2)	g	3
(a,1) (e,2) (f,2)		
(a,2) (b,2) (c,4) (d,2)		
(b,3) (c,2) (d,2) (e,2)		

Note: Duplicate items must not exist in a transaction.

Acceptable format of utility databases in PAMI

Each row in a utility database must contain only items, total sum of utilities and utility values.

```

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

```

Understanding the statistics 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 length of transaction that existed in database
- Average length of all transactions that exists in database
- Maximum length of transaction that existed in database
- Minimum utility value exists in database
- Average utility exists in database
- Maximum utility exists in database
- Standard deviation of transaction length
- Variance in transaction length
- Sparsity of database

The sample code

```

import PAMI.extras.dbStats.utilityDatabaseStats as stats
obj = stats.utilityDatabaseStats('sampleInputFile.txt', ' ')
obj.run()
obj.printStats()

```

What is the input to high-utility frequent pattern mining algorithms

Algorithms to mine the high-utility patterns requires utility database, minUtil (specified by user).

- Input utility database is accepted following formats:

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

- minUtil should be mentioned in **count**.
- minSup should be mentioned in either **count**(in numbers) or **percentage**(in float, multiplied with length of database)
- separator (delimiter used in input file) default delimiter is `\t`

What is the output of high-utility frequent pattern mining algorithms

The output of these algorithms is in two ways:

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

How to run the high-utility pattern algorithm in terminal

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

And execute the following command on terminal.

```
python3 algorithmName.py path of Sample input file path of output
file $minUtil$ $minSup$ separator
```

Sample command to execute the EFIM algorithm in highUtilityPattern/basic folder

```
python3 HUFIM.py /Users/Downloads/inputFile.txt  
/Users/Downloads/outputFile.txt $20$ $5$ ' '
```

How to implement the HUFIM algorithm 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 minUtil (like 10) in place of minUtil
- Specify the minSup (like 5) in place of minSup
- Specify the separator of input file after minSup. (If no separator is specified the default tab separator is considered for input file)

```
import PAMI.highUtilityFrequentPattern.basic.HUFIM as alg  
obj = alg.HUFIM(iFile, minUtil, minSup, sep)  
obj.startMine()  
obj.savePatterns(oFile) (to store the patterns in file)  
Df = obj.getPatternsAsDataFrame() (to store the patterns in dataframe)  
obj.printStats() (to print the no of patterns, runtime and memory consumption  
details)
```

What is the output of high utility pattern mining algorithms

Returns the pattern and utility respectively with \$minUtil=20\$ and \$minSup=5\$

The output in file format:

The format followed to save in file is: `pattern : utility : support`

c d : 35:8
c d a : 34:5
c d b : 39:6
c a : 27:6
c a b : 30:5
c b : 29:7
d a : 22:5
d b : 25:6

The output in DataFrame format:

	Patterns	Utility	Support
0	c d	35	8
1	c d a	34	5
2	c d b	39	6
3	c a	27	6
4	c a b	30	5
5	c b	29	7
6	d a	22	5
7	d b	25	6