Data Mining (EDAMI)

Authors:

Mateusz Hryciów 283 365

April 2021

1. Description of the algorithm

The Generalized Sequential Patterns (GSP) algorithm designed by Srikant and Agrawal in 1996 extended the basis Apriori algorithm with the opportunity to use time constraints, sliding window and taxonomies. For example the shop owner may be interested in knowing whether certain products were bought together (in a given order) in a certain amount of time (e.g a month). Unfortunately such scenario is not possible using basis Apriori algorithm since this algorithm is generating all frequent sets families and not frequent sequential pattern.

First of all it is wise to define the sequence as the ordered list of elements written often in the following format <a(abc)d(cf)>. This syntax means that e.g customer bought firstly product a, then products abc were bought together then product d and so on. In the GSP algorithm there is obviously a support threshold to distinct frequent sequences from not frequent. Although the big amount of potential candidates may be generated thanks to apriori principle ("If a sequence S is not frequent then none of the super-sequences of S is frequent") this number is decreased (search space is decreased).

The general approach in the GSP algorithm may be presented as follows:

1. Initially every item in the database is a candidate of length 1
2. For each searching step (sequence of length k) until no more frequent sequences or no more candidates can be found
    a. Scan the database to compute the support count
    b. Generate the candidates of length k+1 from length k frequent sequences using Apriori

From the SPMF documentation we may find that the threshold is specified as *minsup* argument (in percentage) and there is also one additional argument (in the library implementation) *maximum sequential pattern length* which results the final solution. The input dataset is provided using the text file and the result is output also in the text file (this convention is kept also in the *smpf-py* wrapper). The SPMF library will give us a fairly easy start for implementation.

2. Description of the data set

In out project we decided to use the Bible dataset which is a Bible text converted into the sequence database. The reason for this choice is that there is enough data for sufficient number of patterns detection in a reasonable time. Additionally there is a legend (names given for items) which will help us to interpret the results and find out what "collocation" (in the found pattern context) are frequent in the Bible provided some threshold.

The dataset is present under the http://www.philippe-fournier-viger.com/spmf/datasets/BIBLE.txt link. The part of the file is shown below.

```
-1 10 -1 409 -1 46 -1 10 -1 410 -1 39 -1 -2
411 -1 78 -1 22 -1 121 -1 412 -1 413 -1 46 -1 16 -1 218 -1 -2
```

Each sequence is separated by "-2" and each word in a sequence by "-1". So e.g. first one will be translated into <(10)(409)(46)(10)(410)(39)>.

3. Bibliography

1. C.H. Mooney, J.F. Roddick, “Sequential Pattern Mining: Approaches and Algorithms”, ACM Computing Surveys, March 2013 (https://www.researchgate.net/publication/235246737_Sequential_Pattern_Mining_Approaches_and_Algorithms)
2. “Sequential Pattern Mining” https://www.cc.gatech.edu/~hic/CS7616/pdf/lecture13.pdf
3. R. Srikant, R. Agrawal, „Mining Sequential Patterns: Generalizations and Performance Improvements”, IMB Almaden Research Center, 1996 http://www.philippe-fournier-viger.com/spmf/GSP96.pdf
4. DATA MINING_4 Pattern Discovery in Data Mining_5 2_GSP_Apriori Based Sequential Pattern Mining