Data Mining
Homework Assignment #2

Dmytro Fishman, Anna Leontjeva and Jaak Vilo

February 25, 2014

Table 1: Example of the transaction data set

CustomerID	TransactionID	BasketContent
1	1234	{Aspirin, Panadol}
1	4234	{Aspirin, Sudafed}
2	9373	{Tylenol, Cepacol}
2	9843	{Aspirin, Vitamin C, Sudafed}
3	2941	{Tylenol, Cepacol}
3	2753	{Aspirin, Cepacol}
4	9643	{Aspirin, Vitamin C}
4	9691	{Aspirin, Ibuprofen, Panadol}
5	5313	{Panadol, Vitamin C}
5	1003	{Tylenol, Cepacol, Ibuprofen}
6	5636	{Tylenol, Panadol, Cepacol}
6	3478	{Panadol, Sudafed, Ibuprofen}

Task 1

a. For the data from Table 1 compute the support and support count for itemsets {Aspirin}, {Tylenol, Cepacol}, {Aspirin, Ibuprofen, Panadol}.

b. Compute the confidence for the following association rules: {Aspirin, Vitamin C} $\rightarrow$ {Sudafed}, {Aspirin} $\rightarrow$ {Vitamin C}, {Vitamin C} $\rightarrow$ {Aspirin}. Why the results for last two rules are different?

c. List all the frequent itemsets if the support count threshold $s_{min} = 3$.

d. What does the anti-monotonicity property of support imply? Give an example using the above data set.

Task 2

Apply Apriori algorithm on the drug data set example 1 with support count threshold $s_{min} = 4$. Show the candidate and frequent itemsets for each iteration. Enumerate all the final frequent itemsets. Also indicate the association rules that could be generated from these itemsets and highlight the strongest ones.

Task 3

Construct an FP-tree using the same data set 1 (use the same support count threshold $s_{min} = 4$) . Explain all the steps of the tree construction and draw a resulting tree. Based on this tree answer the questions: how many transactions contain {Aspirin} and {Cepacol}? How many transactions were made in total?

Task 4

Simulate frequent pattern enumeration based on the FP-tree constructed in the previous exercise. Report all the frequent patterns.

Task 5

In this task we will get familiar with the statistical computing language R. Install it. We suggest you to download also the IDE that will make your life much easier: R studio. Once you are set up, take a look at the introduction of R from the CRAN page (Manuals → An Introduction to R) or just google any basic tutorial. R is an open source and has a very powerful community with plenty of tutorials and websites. Once you feel more comfortable with it, go through the following tutorial, run it, check and report the results, describe and interpret them:

```
#install necessary packages (run only once)
install.packages("arules")
install.packages("arulesViz")

#load datafrom the url
data_url =
    url("https://courses.cs.ut.ee/MTAT.03.183/2014_spring/uploads/Main/titanic.txt")
titanic = read.table(data_url, sep =',', header = TRUE)

#observe the data
##first 6 observations
head(titanic)
#types of features
str(titanic)
#dimensionality of the data
dim(titanic)
```

```
#load package for frequent set mining
library(arules)
#help with apriori
?apriori
#run apriori algorithm with default settings
rules = apriori(titanic)
#inspection of the result
inspect(rules)

#now let us assume, we want to see only those rules that have
    rhs as survived:
rules = apriori(titanic,appearance = list(rhs=c("Survived=No",
    "Survived=Yes"),default="lhs"))
inspect(rules)

#let us relax the default settings for the rules we are
    looking for
rules = apriori(titanic,parameter = list(minlen=2, supp=0.05,
    conf=0.8),appearance = list(rhs=c("Survived=No",
    "Survived=Yes"),default="lhs"))

#visualization
library(arulesViz)
plot(rules, method="graph", control=list(type="items"))
```

Note, when you paste the code from the pdf, some symbols may be copied incorrectly. We suggest you to type the above commands yourself, but in case you are too lazy, use the R script we uploaded for you on the page. P.S. If you have problems reading the data from the url, consider copying the data file from link to your computer and use command read.table(PATH, header = TRUE, sep = ',').

Task 6 (2pt)

Some rules do not provide extra knowledge as other rules already contain the information. For example, if there is the rule {Class='2nd', Age='Child'}→ {'Survived'='Yes'}, then the rule {Class='2nd', Age='Child',Sex='Female'}→ {'Survived'='Yes'} is not so informative. Such rules are called 'redundant'. Come up with the definition of the redundancy of the rules. Using the script and the data from task 5 tune default parameters so that there are redundant rules in the output. Next, add the "filter" that outputs only non-redundant rules.