Truc Huynh

Purdue University Fort Wayne

Association Rules Mining Exercises

PART I.

Reading Read Fournier-Viger et al., "A survey of itemset mining", WIREs Data Mining and Knowledge Discovery, Vol. 7:e1207, 2017. Write a (detail) summary of each section following:

1. Variations of the itemset mining problem:
   The popular problem includes below
   a. A huge amount of item sets (based on minimum support): Too many patterns make data analyzing difficult. Algorithms are designed to extract concise representations of frequent itemsets to solve this matter. A set of frequent itemset is called "A concise representation" when it is summarizing all frequent itemset in that data frame. Set of frequent item-sets:
      - Closed itemset: frequent item-sets that supersets have the same support.
      - Maximal Item-sets: frequent item-sets in which supersets are not frequent.
      - Generator item-sets: frequent item-sets in which subsets do not have the same support (also called key item-sets).
   b. Introducing constraints to filter less-interesting patterns. There are several constraints, the most popular are:
      - Monotone.
      - The Anti-monotone constraint is one of the easiest and most beneficial to integrate into a Frequent Itemset Mining (FIM) mining algorithm.
      - The succinct constraint can be checked by only looking at the single item that it contains.
      - Convertible constraint: neither monotone nor anti-monotone. Besides, if some strategies are applied by the FIM algorithm, they can be converted into antimonotone constraints.
   c. The rare items problem: not every item in real life is equal. However, FIM assumes that all items are equal.
   d. Designed to be applied as batch algorithms: In dynamic environments (where changes happen instantly), updating the transaction database(TD) will lead to applying the FIM algorithm again to update the pattern. Some solution has been implemented, the most popular is a designed algorithm that can update result in real-time. Some algorithms are:
      - Incremental mining algorithms: when new transactions are inserted, deleted, or modified in a TD, frequent itemset patterns will be automatically updated.
      - Stream mining algorithm: is optimized to process transactions as quickly as possible by calculating the approximate set of frequent itemset.
      - Interactive mining algorithms: the idea is focusing on the needed item-sets only instead of mining and updating a large number of itemsets that may not all be useful.
   e. Database Format:
      - Weighted itemset mining is used to find itemsets that have a minimum weight -> mine infrequent weighted itemsets.
      - High-utility itemset mining(HUIM): is an extension of weighted itemset mining where both weights and purchase quantities in transactions are considered.
      - Uncertain itemset mining: designed to considerable uncertainty about the data
      - Fuzzy itemset mining: a well-studied extension of itemset mining.
   f. Traditionally applied to find itemsets in a single set of transactions is also a limit of FIM. However, it is often useful to discover the difference between two or more sets of transactions in real-life applications.

2. Other pattern mining problems related to itemset:

   This session shows some other important pattern mining problems and how to solve them:

   a. An association rule mining algorithm works best with a minimum support threshold mins up and a minimum confidence threshold min conf → discover patterns representing strong associations between items → Limitation: May find frequent itemsets that are weakly correlated.
   b. Sequential pattern mining: include discovering sequences frequently appearing in a set of sequences → Limitation: Similar to FIM's problem, and to discover sequential rules → Application: more effective than sequential patterns for some tasks involving prediction → a frequent partial order can summarize several sequential patterns
   c. Episode mining: → Limitation: mine in a single sequence rather than in a set of sequences → used to analyze web-click streams, telecommunication data, sensor readings, sequences of events on an assembly line, and network traffic data
   d. Periodic pattern mining is an algorithm for discovering periodic frequent patterns → Application: discovering periodic behavior of customers and finding recurring events (stock market analysis, market analysis, and bioinformatics).
   e. Subgraph mining: goal is to discover frequent subgraphs in a database of graphs rather than frequent item-sets in transactions → Limitation: Similar to FIM's problem, search space is generally very large → Application: mine closed and maximal frequent subgraphs.

PART II:

Question 1:

(1) All non-empty subsets of a frequent itemset must also be frequent:

Proof:

Let $S \subseteq I$ be a frequent itemset, i.e. $s$upport $(S) \geq$ minSup

Let $\emptyset \neq S' \subseteq$ S

Thesupport$(S') \geq b)$ support(S)

$\geq^{\text{S is freq.}}$minSup

i.e. S' is a frequent itemset

(Seidl., n.d.)

For example:

If {beer, diaper, nuts} is frequent, so is {beer, diaper}. i.e., every transaction having {beer, diaper, nuts} also contains {beer, diaper} (P. Tan et al., n.d.)

(2) Prove that the support of any nonempty subset X′ of itemset X must be at least as great as the support of X. For the proof, give a brief explanation with example data.

Proof:

Let $\emptyset \neq S' \subseteq S \subseteq I$

For any transaction $T \subseteq I$ in database D, we have:

$S \subseteq T \Rightarrow S' \subseteq$ T

Thus, it holds that

$\{T \in D \mid S \subseteq T\} \subseteq \{T \in D \mid S' \subseteq$ T$\}$

And consequently:

Support(S)= $\{T \in D \mid S \subseteq T\} \leq \{T \in D \mid S' \subseteq$ T$\}$ = support(S')

For example:

Question 2:

(1) Which data set(s) will produce the most frequent itemsets? Explain your answer.

Answer: Data set (e) because it generates the most frequent itemsets over the other data sets.

(2) Which data set(s) will produce the fewest number of frequent itemsets? Explain your answer.

Answer: Data set (d) because it does not contain any frequent itemset in the diagram.

(3) Which data set(s) will produce the longest frequent itemset? Explain your answer.

Answer: Dataset (e) because it generates the longest of frequent itemsets over the other data sets.

(4) Which data set(s) will produce frequent itemsets with the highest maximum support? Explain your answer.

Answer: Data set (b) because it contains the itemsets with the highest maximum support

(5) Which data set(s) will produce frequent itemsets containing items with wide-varying support levels (i.e., items with mixed support, ranging from less than 0.2 to more than 0.7). Explain your answer.

Answer: Data set (e) because it generates the most frequent itemsets with wide-varying support levels over the other data sets.

Question 3: Apply min_support threshold is 0.6 (min_support_count=3)

TID	**Items**
1	{M, O, N, K, E, Y}
2	{D, O, N, K, E, Y}
3	{M, A, K, E}
4	{U, C, K, Y}
5	{C, O, K, I, E}

Scan D

Database D

C1

Itemsets
{A}
{C}
{D}
{E}
{K}
{M}
{N}
{O}
{U}
{Y}

Scan D

C1

Itemsets	**Sup_count**
~~{A}~~	~~20%~~
~~{C}~~	~~40%~~
~~{D}~~	~~20%~~
{E}	80%
{K}	100%
~~{M}~~	~~40%~~
~~{N}~~	~~40%~~
{O}	60%
~~{U}~~	~~20%~~
{Y}	60%

L1

Itemsets	**Sup_count**
{K}	100%
{E}	80%
{O}	60%
{Y}	60%

C2

Itemsets
{KE}
{KO}
{KY}
{EO}
{EY}
{OY}

PRUNE C2

C2

Itemsets
{KE}
{KO}
{KY}
{EO}
{EY}
{OY}

Scan D

C2

Itemsets	**Sup_count**
{KE}	80%
{KO}	60%
{KY}	60%
{EO}	60%
~~{EY}~~	~~40%~~
~~{OY}~~	~~40%~~

L2

Itemsets	**Sup_count**
{KE}	80%
{KO}	60%
{KY}	60%
{EO}	60%

C3

Itemsets
{KEO}
{KEY}
{KOY}
{EOY}

PRUNE C3

C3

Itemsets
{KEO}
{KEY}
{KOY}
{EOY}}

Scan D

C3

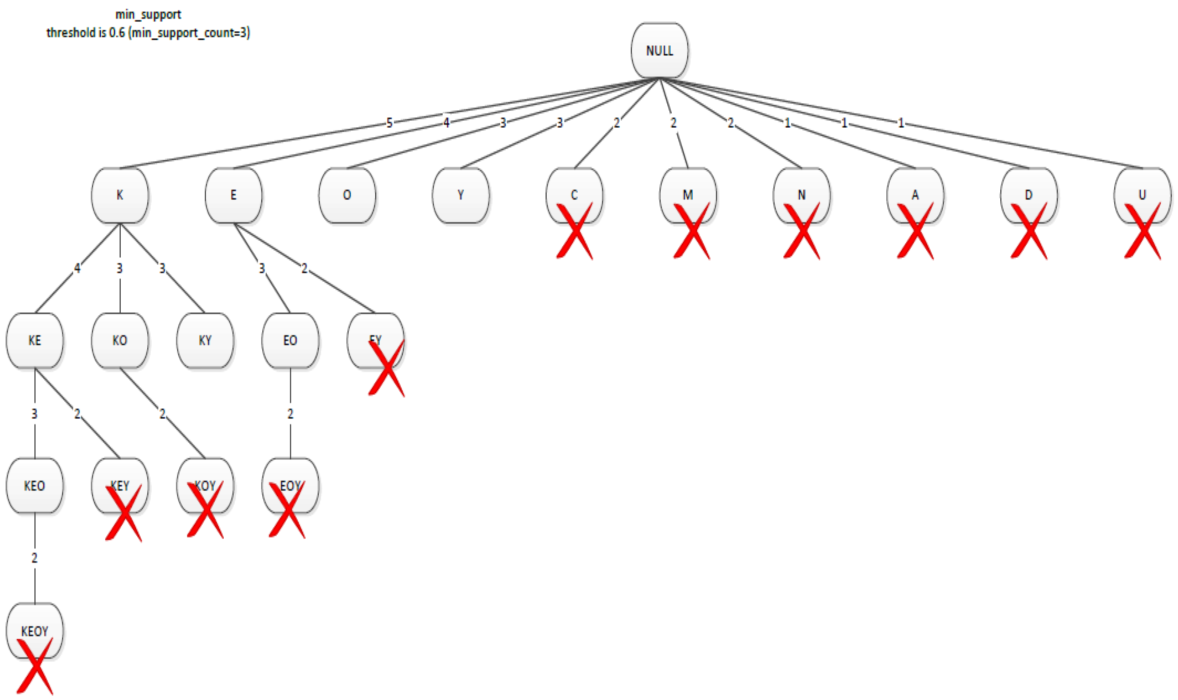
Itemsets	**Sup_count**
{KEO}	60%
~~{KEY}~~	~~40%~~
~~{KOY}~~	~~40%~~
~~{EOY}~~	~~40%~~

L3

Itemsets	**Sup_count**
{KEO}	60%

C4 is Empty

Question 4:


min_support
threshold is 0.6 (min_support_count=3)
NULL
5
4
3
3
2
2
2
1
1
1
K
E
O
Y
C
M
N
A
D
U
4
3
3
3
2
KE
KO
KY
EO
EY
3
2
2
2
KEO
KEY
KOY
EOY
2
KEOY

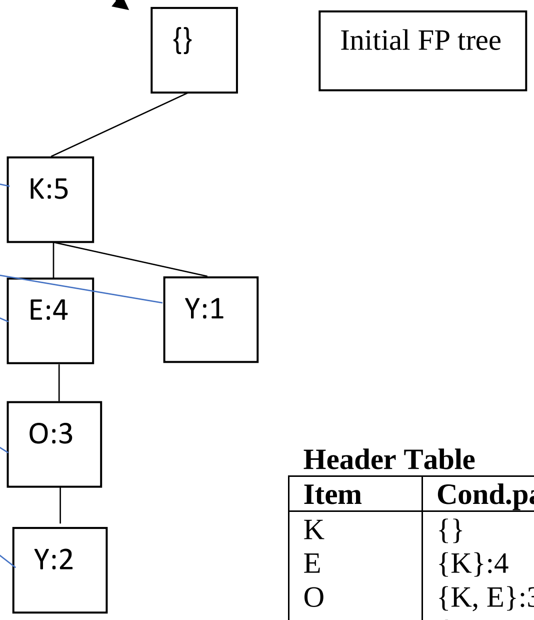
Question 5:
Show the procedure to find all frequent itemsets from the above transaction dataset using FP-growth (with FP-tree) algorithm. Assume the min_support threshold is 0.6 (min_support_count=3). Show (1) Flist, (2) the transaction data with ordered frequent items, (3) FP-tree, (4) Conditional pattern bases, (5) Conditional FP-tree per each pattern base, and (6) the frequent itemsets generated from each conditional FP-tree.

TID	Items
1	{M, O, N, K, E, Y}
2	{D, O, N, K, E, Y}
3	{M, A, K, E}
4	{U, C, K, Y}
5	{C, O, K, I, E}

(Ordered) frequent items
{K, E, O, Y}
{K, E, O, Y}
{K, E}
{K, Y}
{K, E, O}

Header Table

Item	frequency
K	5
E	4
O	3
Y	3
~~C~~	~~2~~
~~M~~	~~2~~
~~N~~	~~2~~
~~A~~	~~1~~
~~D~~	~~1~~
~~U~~	~~1~~



1. Scan DB, find frequent itemset: L1={E,K,O,Y}
2. Sort frequent item in frequent descending order, F-list. F-list=K-E-O-Y

Header Table

Item	Cond.pattern base
K	{}
E	{K}:4
O	{K, E}:3
Y	{K, E, O}:2, {K}:1

Conditional Pattern Base

Item	Cond.pattern base	Conditional FP-Tree	Frequent Pattern Generated
Y	{K, E, O:2}, {K:1}	<K:2, ~~E:2, O:2~~> <K:1>	{K, Y: 3}
O	{K, E:3}	<K:3, E:3>	{K, O: 3}, {E, O: 3}, {K, E, O: 3}
E	{K:4}	<K:4>	{K, E: 4}

Question 6:

Find (1) all the maximal frequent itemsets and (2) all the closed frequent itemsets when the min_support threshold is 0.6 (min_support_count=3).

Frequent Itemsets	**Support**
{K}	1
{E}	0.8
{O}	0.6
{Y}	0.6
{K, E}	0.8
{K, O}	0.6
{K, Y}	0.6
{E, O}	0.6
{K, E, O}	0.6

Closed: {K}, {E}, {KE}, {K, E, O}

Maximal: {K, E, O}

Question 7: when the confidence threshold is 0.7

Frequent Itemsets	**Support**
{K, E, O}	0.6

c ({K, E} → {O}) =σ{K, E, O}/ σ{K, E} = 3/4= 0.75

c ({K, O} → {E}) =σ{K, E, O}/ σ{K, O} = 3/3= 1

c ({E, O} → {K}) =σ{K, E, O}/ σ{E, O} = 3/3= 1

~~c ({K} → {E,O}) =σ{K, E, O}/σ{K} = 3/5 = 0.6~~ (< confidence threshold)

c ({E} → {K,O}) =σ{K, E, O}/ σ{E}= 3/4= 0.75

c ({O}→ {K,E}) =σ{K, E, O}/ σ{O}= 3/3=1

Strong rules:

Frequent Itemsets	**CONFIDENCE**
K, E → O	0.75
K, O → E	1
E, O → K	1
~~K → E, O~~	~~0.6~~
E → K, O	0.75
O → K, E	1

Question 8:

TID	**Items**
1	{M, O, N, K, E, Y}
2	{D, O, N, K, E, Y}
3	{M, A, K, E}
4	{U, C, K, Y}
5	{C, O, K, I, E}

Contingency table K → Y:

	Y	$\bar{Y}$	sum
K	$f_{11} = 3$	$= 5$	f_{1+} 8
$\bar{K}$	$f_{01} = 3$	$f_{00} = 2$	$f_{0+} = 5$
sum	$f_{+1} = 6$	$f_{+0} = 7$	N= 13

Contingency table Y → K:

	K	$\bar{K}$	sum
Y	$f_{11} = 3$	$f_{10} = 3$	$f_{1+} = 6$
$\bar{Y}$	$f_{01} = 5$	$f_{00} = 2$	$f_{0+} = 7$
sum	$f_{+1} = 8$	$f_{+0} = 5$	N= 13

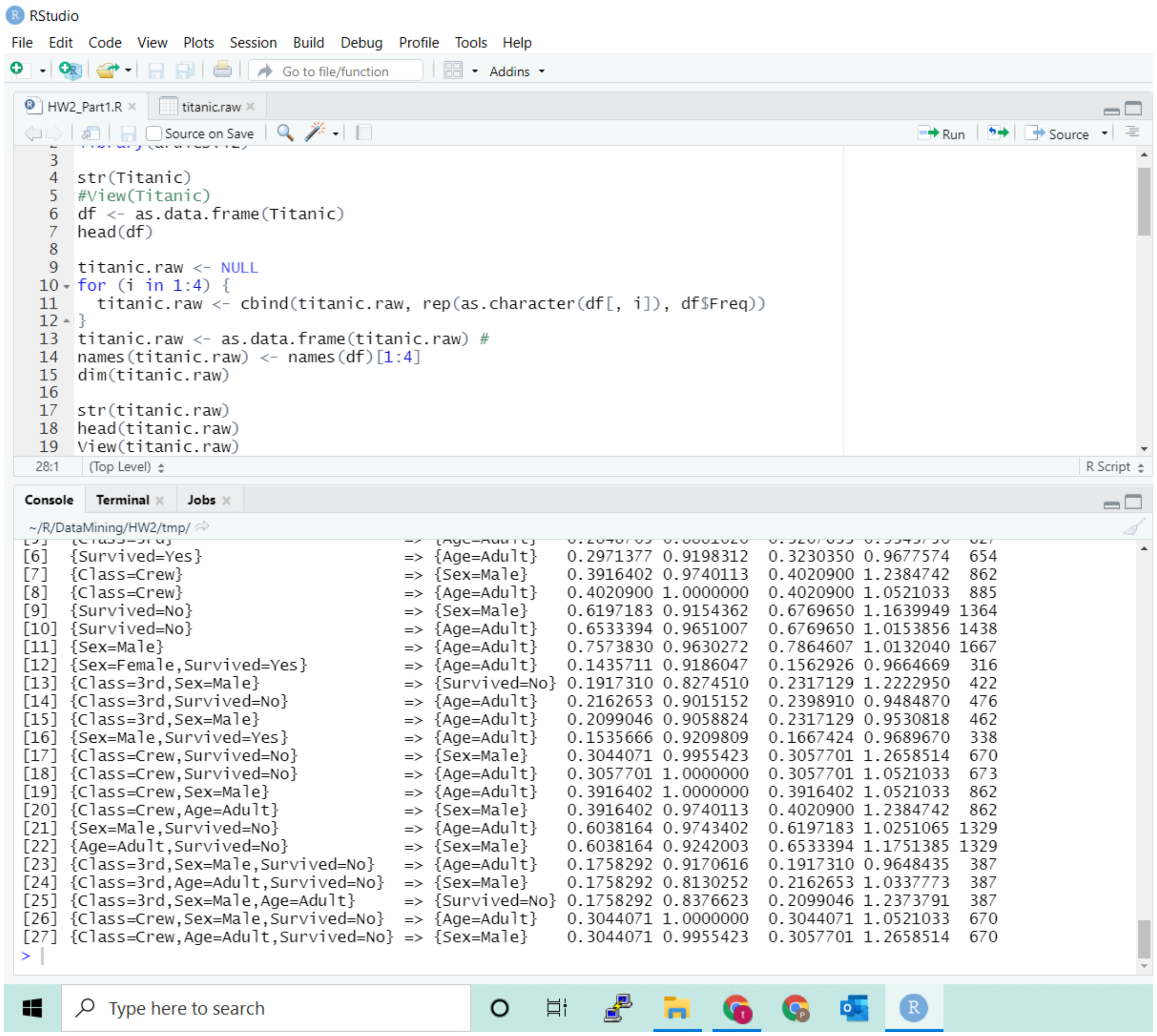
	K → Y	**Y → K**
Support s(K, Y)	s(K ,Y) = f_{11}/N= 3/13= 0.23	s(Y, K) = = f_{11}/N= 3/13= 0.23
Confidence c (K → Y)	c (K → Y) = f_{11}/f_{1+}= 3/8=0.375	c (Y → K) =f_{11}/f_{1+}=3/6=0.5
Lift	Lift(K,Y)= s(K ,Y)/(s(K)*s(Y)) = $(f_{11} * N)/(f_{1+} * f_{+1})$= 3*13/(8*6)=0.8125 < 1: Negative associated	Lift(Y,K)= s(Y ,K)/(s(Y)*s(K)) = $(f_{11} * N)/(f_{1+} * f_{+1})$= 3*13/(6*8)=0.8125 < 1: Negative associated
Leverage	Leverage (K ,Y)= s(K ,Y) - (s(K)*s(Y))=0.23-(8/13*6/13)=0.23-0.28=-0.05 < 0	Leverage (Y ,K)= s(Y ,K) - (s(Y)*s(K))= 0.23-(6/13*8/13) =0.23-0.28=-0.05 < 0
Conviction	$V(K,Y) = max(\frac{\mathrm{P(K).P(}\bar{Y}\mathrm{)}}{\mathrm{P(K}\bar{Y}\mathrm{)}}, \frac{\mathrm{P(Y).P(}\bar{K}\mathrm{)}}{\mathrm{P(Y}\bar{K}\mathrm{)}}) = max(\frac{\mathrm{s(K).s(}\bar{Y}\mathrm{)}}{\mathrm{s(K}\bar{Y}\mathrm{)}}, \frac{\mathrm{s(Y).s(}\bar{K}\mathrm{)}}{\mathrm{s(Y}\bar{K}\mathrm{)}})$ $= max(\frac{8/13 * 7/13}{5/13}, \frac{6/13 * 5/13}{3/13}$ $= max\left(\frac{8 * 7/13 * 13}{5/13}, \frac{6 * 5/13 * 13}{3/13}\right)$ = *max* (0.862, 0.769) = ½ (0.862+ 0.769+|0.862- 0.769|) = 0.862	$V(Y,K) = max(\frac{\mathrm{P(Y).P(}\bar{K}\mathrm{)}}{\mathrm{P(Y}\bar{K}\mathrm{)}}, \frac{\mathrm{P(K).P(}\bar{Y}\mathrm{)}}{\mathrm{P(K}\bar{Y}\mathrm{)}}) = max(\frac{\mathrm{s(Y).s(}\bar{K}\mathrm{)}}{\mathrm{s(Y}\bar{K}\mathrm{)}}, \frac{\mathrm{s(K).s(}\bar{Y}\mathrm{)}}{\mathrm{s(K}\bar{Y}\mathrm{)}})$ $= max\left(\frac{6/13 * 5/13}{3/13}, \frac{8/13 * 7/13}{5/13}\right)$ = *max* (0.769,0.862) = 0.862

Question 9:

Present the transaction data with the vertical data layout:

TID	**Items**
1	{M, O, N, K, E, Y}
2	{D, O, N, K, E, Y}
3	{M, A, K, E}
4	{U, C, K, Y}
5	{C, O, K, I, E}

Vertical Data Layout									
K	**E**	**O**	**Y**	**C**	**M**	**N**	**A**	**D**	**U**
1	1	1	1	4	1	1	3	2	4
2	2	2	2	5	3	2			
3	3	5	4						
4	5								
5									

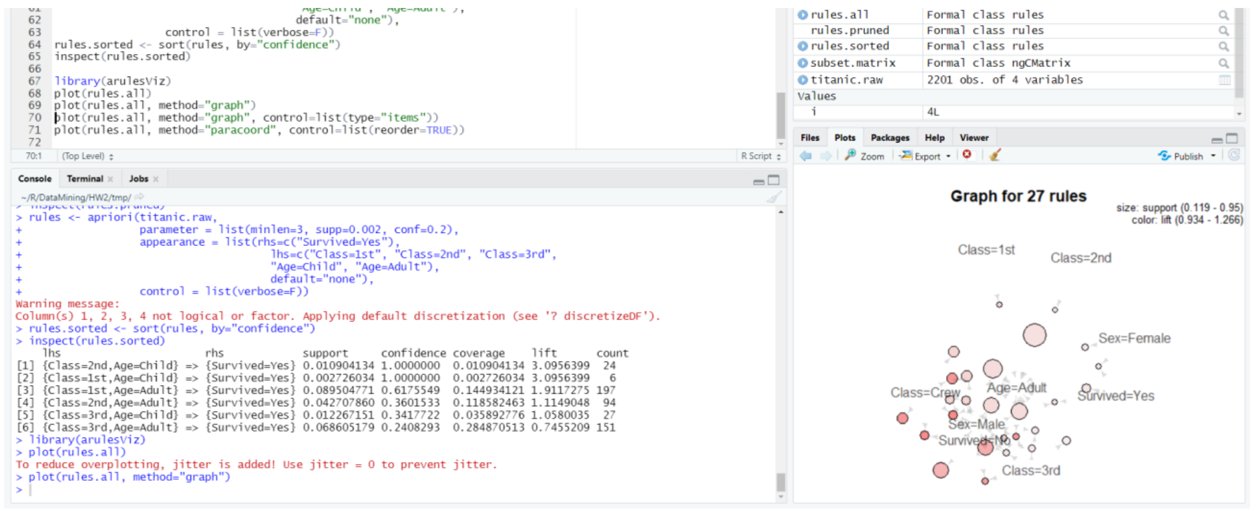


```r
str(Titanic)
#View(Titanic)
df <- as.data.frame(Titanic)
head(df)

titanic.raw <- NULL
for (i in 1:4) {
  titanic.raw <- cbind(titanic.raw, rep(as.character(df[, i]), df$Freq))
}
titanic.raw <- as.data.frame(titanic.raw) #
names(titanic.raw) <- names(df)[1:4]
dim(titanic.raw)

str(titanic.raw)
head(titanic.raw)
View(titanic.raw)
```

```
~/R/DataMining/HW2/tmp/
[6]  {Survived=Yes}                    => {Age=Adult}    0.2971377 0.9198312  0.3230350 0.9677574  654
[7]  {Class=Crew}                      => {Sex=Male}     0.3916402 0.9740113  0.4020900 1.2384742  862
[8]  {Class=Crew}                      => {Age=Adult}    0.4020900 1.0000000  0.4020900 1.0521033  885
[9]  {Survived=No}                     => {Sex=Male}     0.6197183 0.9154362  0.6769650 1.1639949 1364
[10] {Survived=No}                     => {Age=Adult}    0.6533394 0.9651007  0.6769650 1.0153856 1438
[11] {Sex=Male}                        => {Age=Adult}    0.7573830 0.9630272  0.7864607 1.0132040 1667
[12] {Sex=Female,Survived=Yes}         => {Age=Adult}    0.1435711 0.9186047  0.1562926 0.9664669  316
[13] {Class=3rd,Sex=Male}              => {Survived=No}  0.1917310 0.8274510  0.2317129 1.2222950  422
[14] {Class=3rd,Survived=No}           => {Age=Adult}    0.2162653 0.9015152  0.2398910 0.9484870  476
[15] {Class=3rd,Sex=Male}              => {Age=Adult}    0.2099046 0.9058824  0.2317129 0.9530818  462
[16] {Sex=Male,Survived=Yes}           => {Age=Adult}    0.1535666 0.9209809  0.1667424 0.9689670  338
[17] {Class=Crew,Survived=No}          => {Sex=Male}     0.3044071 0.9955423  0.3057701 1.2658514  670
[18] {Class=Crew,Survived=No}          => {Age=Adult}    0.3057701 1.0000000  0.3057701 1.0521033  673
[19] {Class=Crew,Sex=Male}             => {Age=Adult}    0.3916402 1.0000000  0.3916402 1.0521033  862
[20] {Class=Crew,Age=Adult}            => {Sex=Male}     0.3916402 0.9740113  0.4020900 1.2384742  862
[21] {Sex=Male,Survived=No}            => {Age=Adult}    0.6038164 0.9743402  0.6197183 1.0251065 1329
[22] {Age=Adult,Survived=No}           => {Sex=Male}     0.6038164 0.9242003  0.6533394 1.1751385 1329
[23] {Class=3rd,Sex=Male,Survived=No}  => {Age=Adult}    0.1758292 0.9170616  0.1917310 0.9648435  387
[24] {Class=3rd,Age=Adult,Survived=No} => {Sex=Male}     0.1758292 0.8130252  0.2162653 1.0337773  387
[25] {Class=3rd,Sex=Male,Age=Adult}    => {Survived=No}  0.1758292 0.8376623  0.2099046 1.2373791  387
[26] {Class=Crew,Sex=Male,Survived=No} => {Age=Adult}    0.3044071 1.0000000  0.3044071 1.0521033  670
[27] {Class=Crew,Age=Adult,Survived=No} => {Sex=Male}    0.3044071 0.9955423  0.3057701 1.2658514  670
> 
```



```r
                         default="none"),
          control = list(verbose=F))
rules.sorted <- sort(rules, by="confidence")
inspect(rules.sorted)

library(arulesViz)
plot(rules.all)
plot(rules.all, method="graph")
plot(rules.all, method="graph", control=list(type="items"))
plot(rules.all, method="paracoord", control=list(reorder=TRUE))
```

```
~/R/DataMining/HW2/tmp/
> rules <- apriori(titanic.raw,
+                  parameter = list(minlen=3, supp=0.002, conf=0.2),
+                  appearance = list(rhs=c("Survived=Yes"),
+                                    lhs=c("Class=1st", "Class=2nd", "Class=3rd",
+                                    "Age=Child", "Age=Adult"),
+                                    default="none"),
+                  control = list(verbose=F))
Warning message:
Column(s) 1, 2, 3, 4 not logical or factor. Applying default discretization (see '? discretizeDF').
> rules.sorted <- sort(rules, by="confidence")
> inspect(rules.sorted)
    lhs                    rhs            support     confidence coverage    lift      count
[1] {Class=2nd,Age=Child} => {Survived=Yes} 0.010904134 1.0000000  0.010904134 3.0956399  24
[2] {Class=1st,Age=Child} => {Survived=Yes} 0.002726034 1.0000000  0.002726034 3.0956399   6
[3] {Class=1st,Age=Adult} => {Survived=Yes} 0.089504771 0.6175549  0.144934121 1.9117275 197
[4] {Class=2nd,Age=Adult} => {Survived=Yes} 0.042707860 0.3601533  0.118582463 1.1149048  94
[5] {Class=3rd,Age=Child} => {Survived=Yes} 0.012267151 0.3417722  0.035892776 1.0580035  27
[6] {Class=3rd,Age=Adult} => {Survived=Yes} 0.068605179 0.2408293  0.284870513 0.7455209 151
> library(arulesViz)
> plot(rules.all)
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
> plot(rules.all, method="graph")
> 
```

```
rules.all       Formal class rules
rules.pruned    Formal class rules
rules.sorted    Formal class rules
subset.matrix   Formal class ngCMatrix
titanic.raw     2201 obs. of 4 variables
Values
  i             4L
```

```
library(arules)

library(arulesViz)


str(Titanic)

#View(Titanic)

df <- as.data.frame(Titanic)

head(df)


titanic.raw <- NULL

for (i in 1:4) {

  titanic.raw <- cbind(titanic.raw, rep(as.character(df[, i]), df$Freq))

}

titanic.raw <- as.data.frame(titanic.raw) #

names(titanic.raw) <- names(df)[1:4]

dim(titanic.raw)


str(titanic.raw)

head(titanic.raw)

View(titanic.raw)

summary(titanic.raw)



# find association rules with default settings

rules.all <- apriori(titanic.raw)

rules.all

inspect(rules.all)


# rules with rhs containing "Survived" only
```

```
rules <- apriori(
  titanic.raw,
  control = list(verbose = F),
  parameter = list(
    minlen = 2,
    supp = 0.005,
    conf = 0.8
  ),
  appearance = list(
    rhs = c("Survived=No", "Survived=Yes"),
    default = "lhs"
  )
)
quality(rules) <- round(quality(rules), digits=3)
rules.sorted <- sort(rules, by="lift")
inspect(rules.sorted)

# find redundant rules
subset.matrix <- is.subset(rules.sorted, rules.sorted)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1
which(redundant)

# remove redundant rules
rules.pruned <- rules.sorted[!redundant]
inspect(rules.pruned)



rules <- apriori(titanic.raw,
        parameter = list(minlen=3, supp=0.002, conf=0.2),
        appearance = list(rhs=c("Survived=Yes"),
                 lhs=c("Class=1st", "Class=2nd", "Class=3rd",
                 "Age=Child", "Age=Adult"),
                 default="none"),
        control = list(verbose=F))
rules.sorted <- sort(rules, by="confidence")
inspect(rules.sorted)


library(arulesViz)
plot(rules.all)
plot(rules.all, method="graph")
plot(rules.all, method="graph", control=list(type="items"))
plot(rules.all, method="paracoord", control=list(reorder=TRUE))
```

References:

P. Tan et al., (n.d.) “Introduction to Data Mining”, Chapter 2. Retrieved from https://purdue.brightspace.com/d2l/le/content/127433/viewContent/3920610/View

Seidl., T. (n.d.) Exercise 3: Frequent Itemset Mining, Knowledge Discovery in Databases I SS 2016. Retrieved from: https://www.dbs.ifi.lmu.de/Lehre/KDD/SS16/uebung/blatt03_sol.pdf