Note Mining Association Rule

Contents

ETL

```
library(arules)
grocery = read.transactions('./Data/groceries.csv', sep = ',')
inspect(head(grocery))
```

```
##     items
## [1] {citrus fruit,
##      margarine,
##      ready soups,
##      semi-finished bread}
## [2] {coffee,
##      tropical fruit,
##      yogurt}
## [3] {whole milk}
## [4] {cream cheese,
##      meat spreads,
##      pip fruit,
##      yogurt}
## [5] {condensed milk,
##      long life bakery product,
##      other vegetables,
##      whole milk}
## [6] {abrasive cleaner,
##      butter,
##      rice,
##      whole milk,
##      yogurt}
```

```
LIST(head(grocery))
```

```
## [[1]]
## [1] "citrus fruit"        "margarine"           "ready soups"
## [4] "semi-finished bread"
##
## [[2]]
## [1] "coffee"         "tropical fruit" "yogurt"
##
## [[3]]
## [1] "whole milk"
##
## [[4]]
## [1] "cream cheese" "meat spreads" "pip fruit"    "yogurt"
##
## [[5]]
## [1] "condensed milk"           "long life bakery product"
## [3] "other vegetables"         "whole milk"
##
## [[6]]
## [1] "abrasive cleaner" "butter"           "rice"             "whole milk"
## [5] "yogurt"
```

```
size(head(grocery, 20))
```

```
##  [1] 4 3 1 4 4 5 1 5 1 2 5 9 1 3 2 4 1 1 1 1
```

EDA

```
str(grocery)
```

```
## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:43367] 29 88 118 132 33 157 167 166 38 91 ...
##   .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
##   .. .. ..@ Dim     : int [1:2] 169 9835
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : NULL
##   .. .. ..@ factors : list()
##   ..@ itemInfo   :'data.frame':  169 obs. of  1 variable:
##   .. ..$ labels: chr [1:169] "abrasive cleaner" "artif. sweetener" "baby cosmetics" "baby food" ...
##   ..@ itemsetInfo:'data.frame':  0 obs. of  0 variables
```

```
summary(grocery)
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.02609146
##
## most frequent items:
##       whole milk other vegetables       rolls/buns             soda
##             2513             1903             1809             1715
##           yogurt          (Other)
##             1372            34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
```

```
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.409   6.000  32.000
##
## includes extended item information - examples:
##             labels
## 1 abrasive cleaner
## 2 artif. sweetener
## 3   baby cosmetics
```

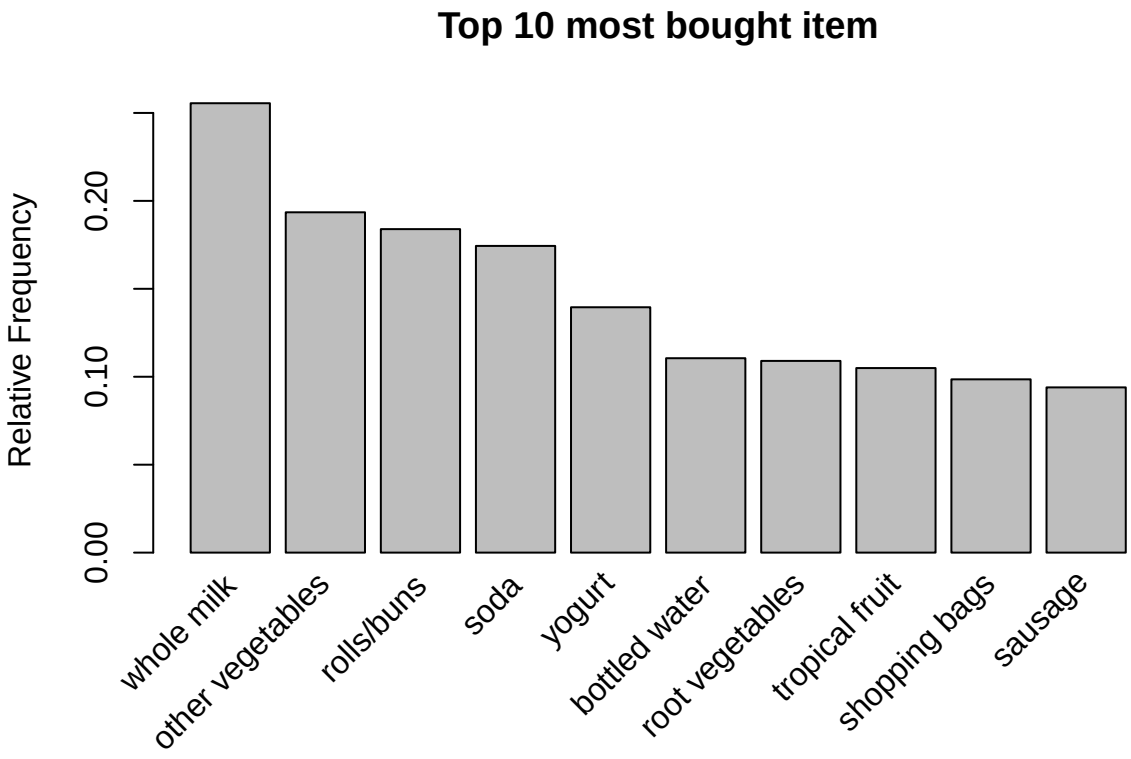
```
class(grocery)
```

```
## [1] "transactions"
## attr(,"package")
## [1] "arules"
```

```
inspect(head(grocery))
```

```
##     items
## [1] {citrus fruit,
##      margarine,
##      ready soups,
##      semi-finished bread}
## [2] {coffee,
##      tropical fruit,
##      yogurt}
## [3] {whole milk}
## [4] {cream cheese,
##      meat spreads,
##      pip fruit,
##      yogurt}
## [5] {condensed milk,
##      long life bakery product,
##      other vegetables,
##      whole milk}
## [6] {abrasive cleaner,
##      butter,
##      rice,
##      whole milk,
##      yogurt}
```

```
itemFrequencyPlot(grocery, topN = 10, main = 'Top 10 most bought item',
                  ylab = 'Relative Frequency')
```



Model Training

```
rule1 = apriori(grocery, parameter=list(supp=0.01, conf=0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [15 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(rule1)
```

```
##      lhs                                rhs                support
## [1]  {curd, yogurt}                  => {whole milk}       0.01006609
## [2]  {butter, other vegetables}      => {whole milk}       0.01148958
## [3]  {domestic eggs, other vegetables} => {whole milk}     0.01230300
```

```
## [4]  {whipped/sour cream, yogurt}           => {whole milk}       0.01087951
## [5]  {other vegetables, whipped/sour cream} => {whole milk}       0.01464159
## [6]  {other vegetables, pip fruit}          => {whole milk}       0.01352313
## [7]  {citrus fruit, root vegetables}        => {other vegetables} 0.01037112
## [8]  {root vegetables, tropical fruit}      => {other vegetables} 0.01230300
## [9]  {root vegetables, tropical fruit}      => {whole milk}       0.01199797
## [10] {tropical fruit, yogurt}               => {whole milk}       0.01514997
## [11] {root vegetables, yogurt}              => {other vegetables} 0.01291307
## [12] {root vegetables, yogurt}              => {whole milk}       0.01453991
## [13] {rolls/buns, root vegetables}          => {other vegetables} 0.01220132
## [14] {rolls/buns, root vegetables}          => {whole milk}       0.01270971
## [15] {other vegetables, yogurt}             => {whole milk}       0.02226741
##      confidence coverage   lift     count
## [1]  0.5823529  0.01728521 2.279125  99
## [2]  0.5736041  0.02003050 2.244885 113
## [3]  0.5525114  0.02226741 2.162336 121
## [4]  0.5245098  0.02074225 2.052747 107
## [5]  0.5070423  0.02887646 1.984385 144
## [6]  0.5175097  0.02613116 2.025351 133
## [7]  0.5862069  0.01769192 3.029608 102
## [8]  0.5845411  0.02104728 3.020999 121
## [9]  0.5700483  0.02104728 2.230969 118
## [10] 0.5173611  0.02928317 2.024770 149
## [11] 0.5000000  0.02582613 2.584078 127
## [12] 0.5629921  0.02582613 2.203354 143
## [13] 0.5020921  0.02430097 2.594890 120
## [14] 0.5230126  0.02430097 2.046888 125
## [15] 0.5128806  0.04341637 2.007235 219
```

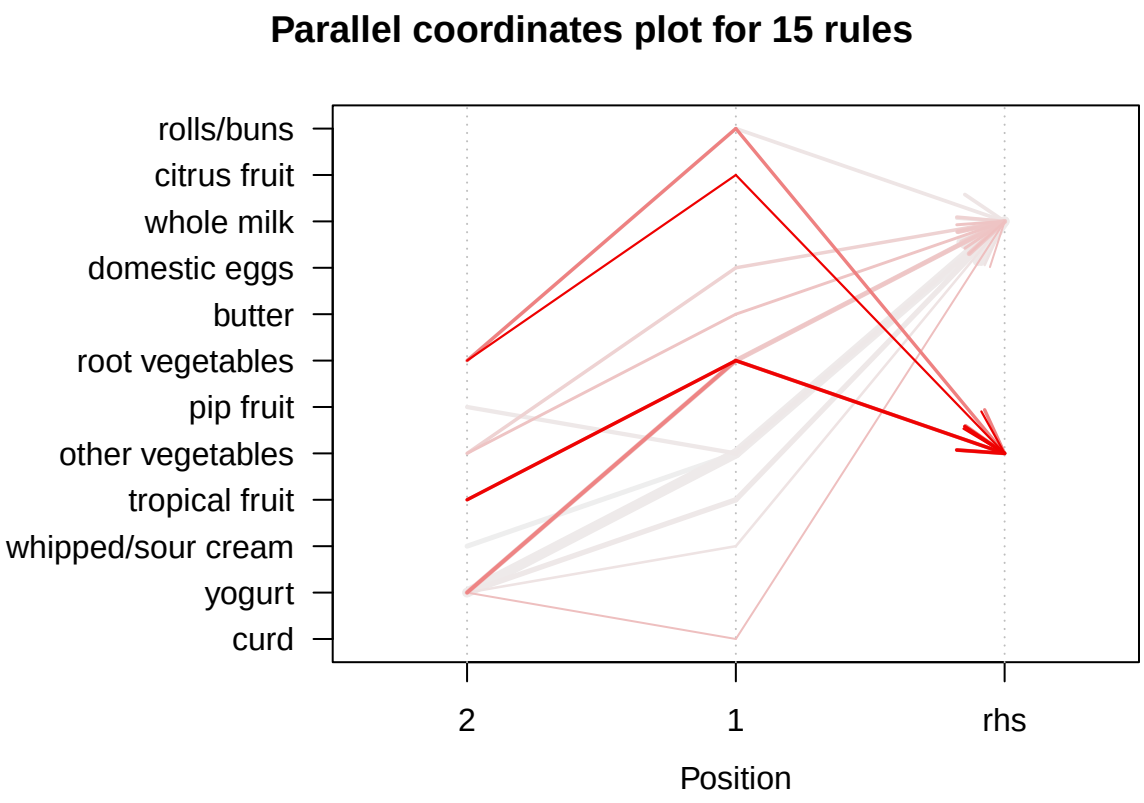
```
sort(rule1, by = 'lift', decreasing = T)
```

```
## set of 15 rules
```

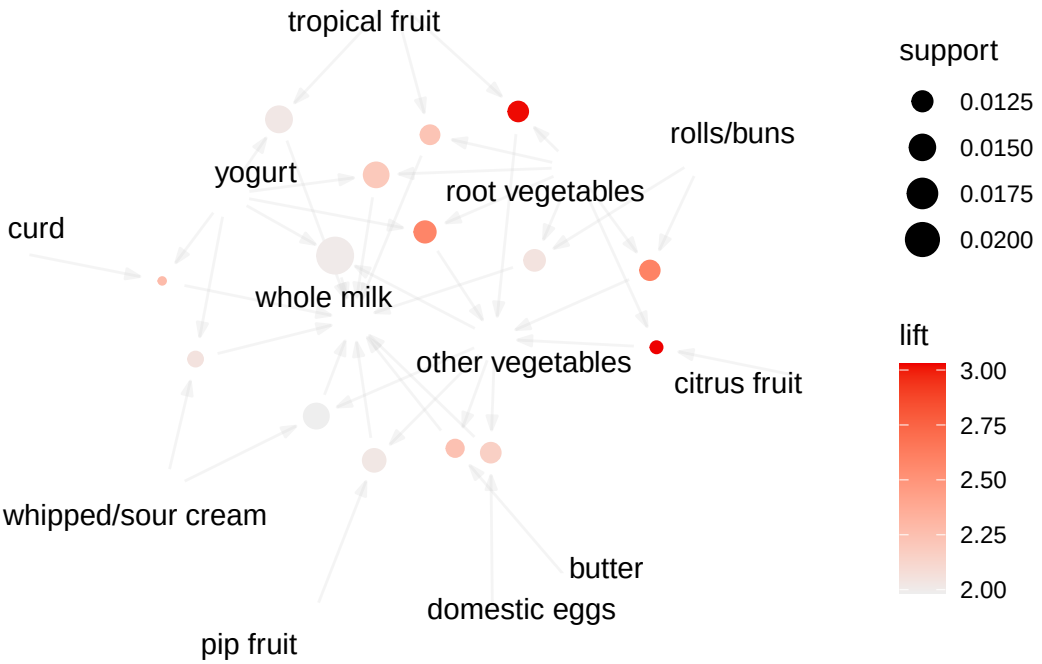
```
library(arulesViz)
```

```
## Warning: package 'arulesViz' was built under R version 4.4.2
```

```
plot(rule1, method = 'paracoord', control=list(reorder=T))
```




```
plot(rule1, method='graph')
```




```
Aturan.S7<- apriori(grocery,
                    parameter=list(supp=0.01,
                                   conf=0.08),
                    appearance=list(default="rhs",
                                    lhs="yogurt"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.08    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [37 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

ETL

```
load('./Data/titanic.raw.rdata')
head(titanic.raw)
```

```
##   Class  Sex   Age Survived
## 1   3rd Male Child       No
## 2   3rd Male Child       No
## 3   3rd Male Child       No
## 4   3rd Male Child       No
## 5   3rd Male Child       No
## 6   3rd Male Child       No
```

EDA

```
str(titanic.raw)
```

```
## 'data.frame':    2201 obs. of  4 variables:
##  $ Class   : Factor w/ 4 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Age     : Factor w/ 2 levels "Adult","Child": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Survived: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
summary(titanic.raw)
```

```
##   Class         Sex          Age       Survived
##  1st :325   Female: 470   Adult:2092   No :1490
##  2nd :285   Male  :1731   Child: 109   Yes: 711
##  3rd :706
##  Crew:885
```

```
class(titanic.raw)
```

```
## [1] "data.frame"
```

Assignment

```
load('./Data/titanic.raw.rdata')
head(titanic.raw)
```

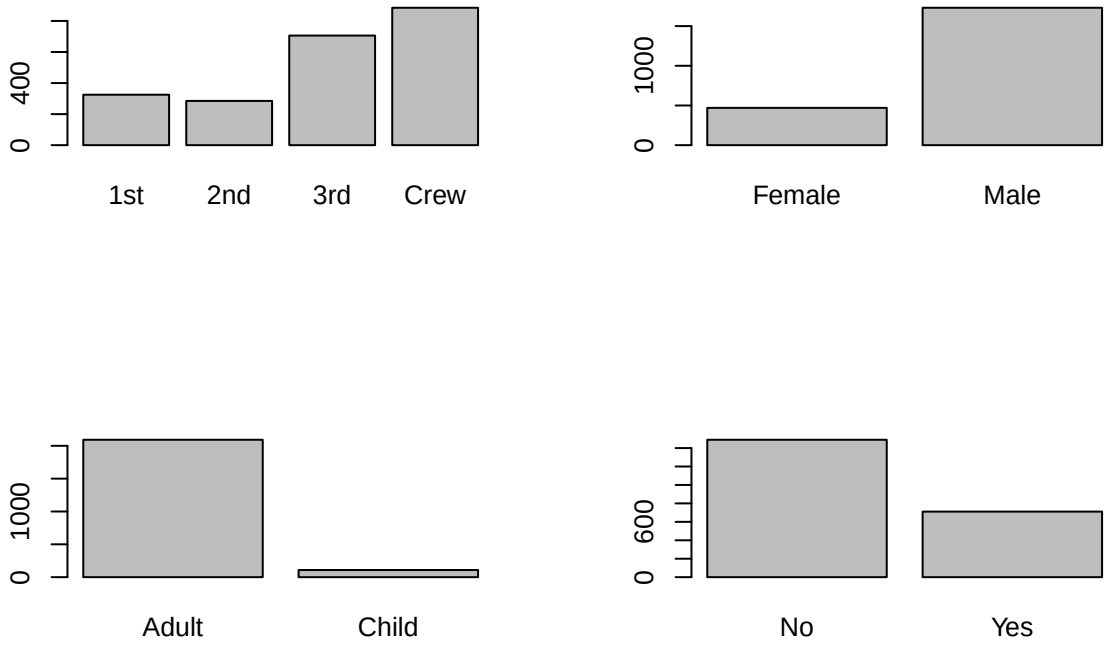
```
##   Class  Sex   Age Survived
## 1   3rd Male Child       No
## 2   3rd Male Child       No
## 3   3rd Male Child       No
## 4   3rd Male Child       No
## 5   3rd Male Child       No
## 6   3rd Male Child       No
```

```
str(titanic.raw)
```

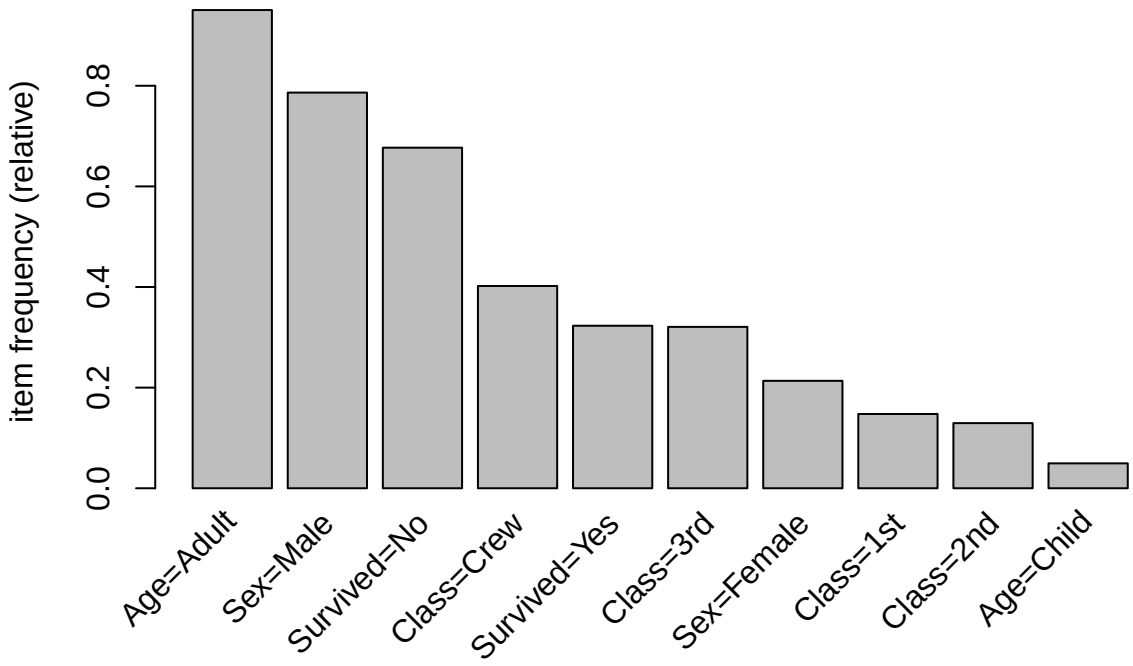
```
## 'data.frame':    2201 obs. of  4 variables:
##  $ Class   : Factor w/ 4 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Age     : Factor w/ 2 levels "Adult","Child": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Survived: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
```

Descriptive Statistics

```
par(mfrow=c(2,2))
barplot(table(titanic.raw$Class))
barplot(table(titanic.raw$Sex))
barplot(table(titanic.raw$Age))
barplot(table(titanic.raw$Survived))
```



```r
library(arules)
titanic = as(titanic.raw, 'transactions')
itemFrequencyPlot(titanic, topN = 10)
```



Association Rule

```r
class(titanic.raw)
```

```
## [1] "data.frame"
```

```r
rule = apriori(titanic, parameter = list(supp=0.1, conf = 0.6))
```

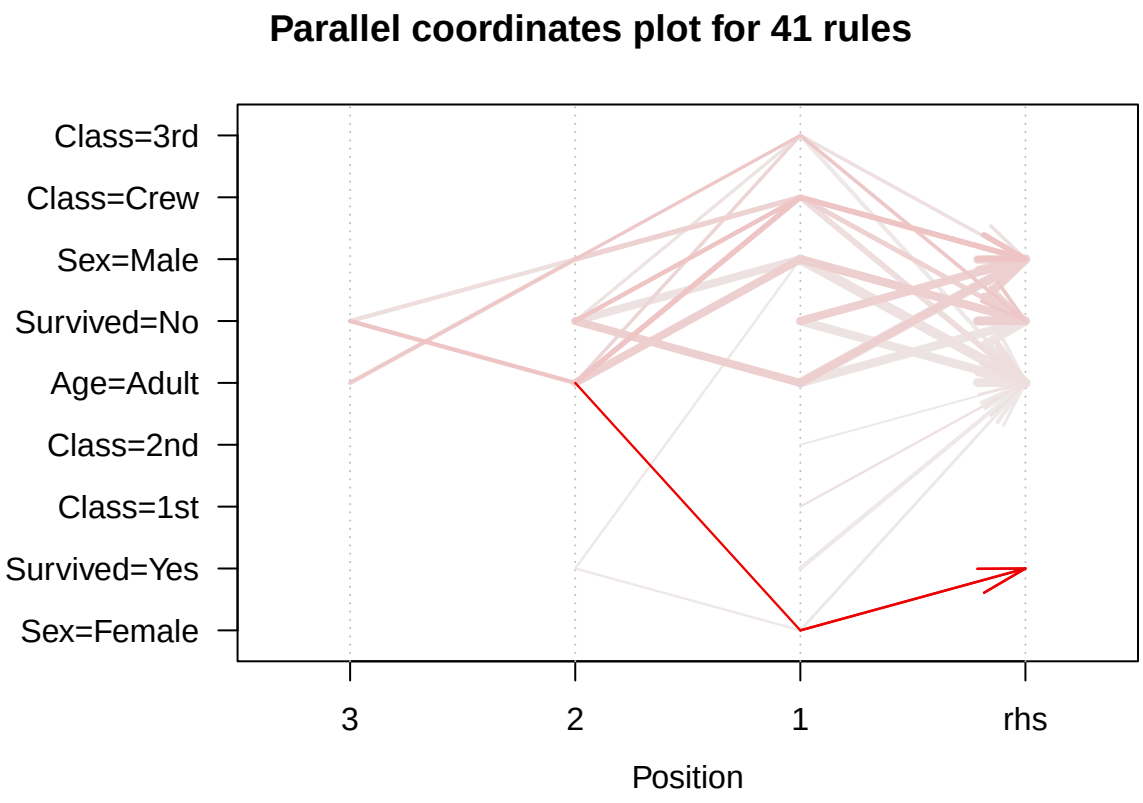
```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 220
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [44 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(head(rule))
```

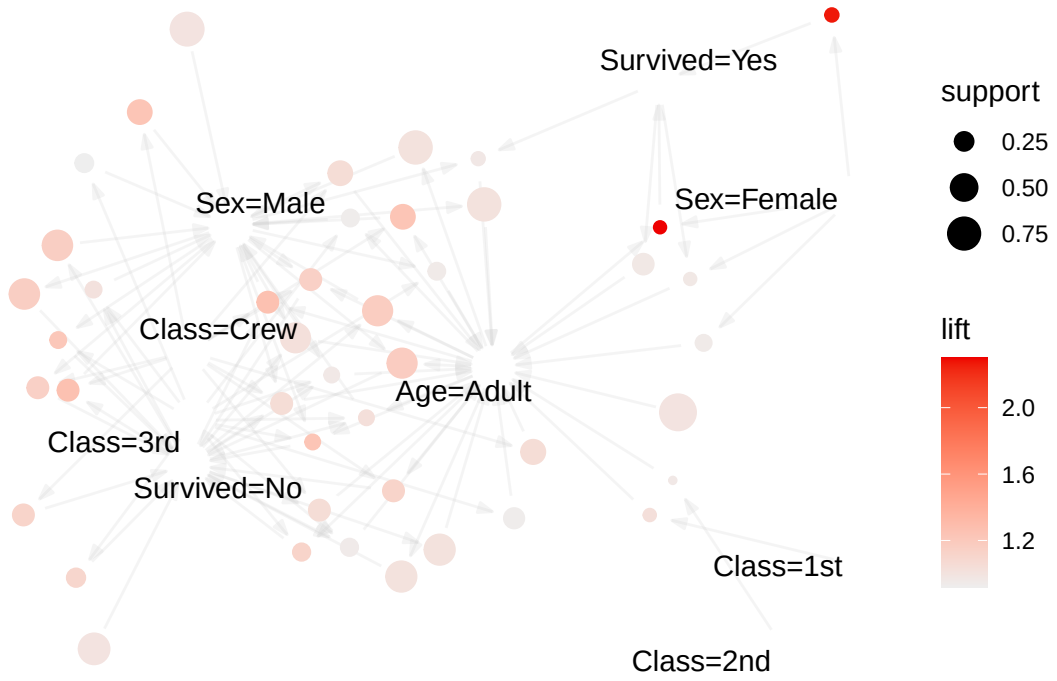
```
##     lhs             rhs            support   confidence coverage  lift
## [1] {}           => {Survived=No}  0.6769650 0.6769650  1.0000000 1.0000000
## [2] {}           => {Sex=Male}     0.7864607 0.7864607  1.0000000 1.0000000
## [3] {}           => {Age=Adult}    0.9504771 0.9504771  1.0000000 1.0000000
## [4] {Class=2nd}  => {Age=Adult}    0.1185825 0.9157895  0.1294866 0.9635051
## [5] {Class=1st}  => {Age=Adult}    0.1449341 0.9815385  0.1476602 1.0326798
## [6] {Sex=Female} => {Survived=Yes} 0.1562926 0.7319149  0.2135393 2.2657450
##     count
## [1] 1490
## [2] 1731
## [3] 2092
## [4]  261
## [5]  319
## [6]  344
```

Visualization

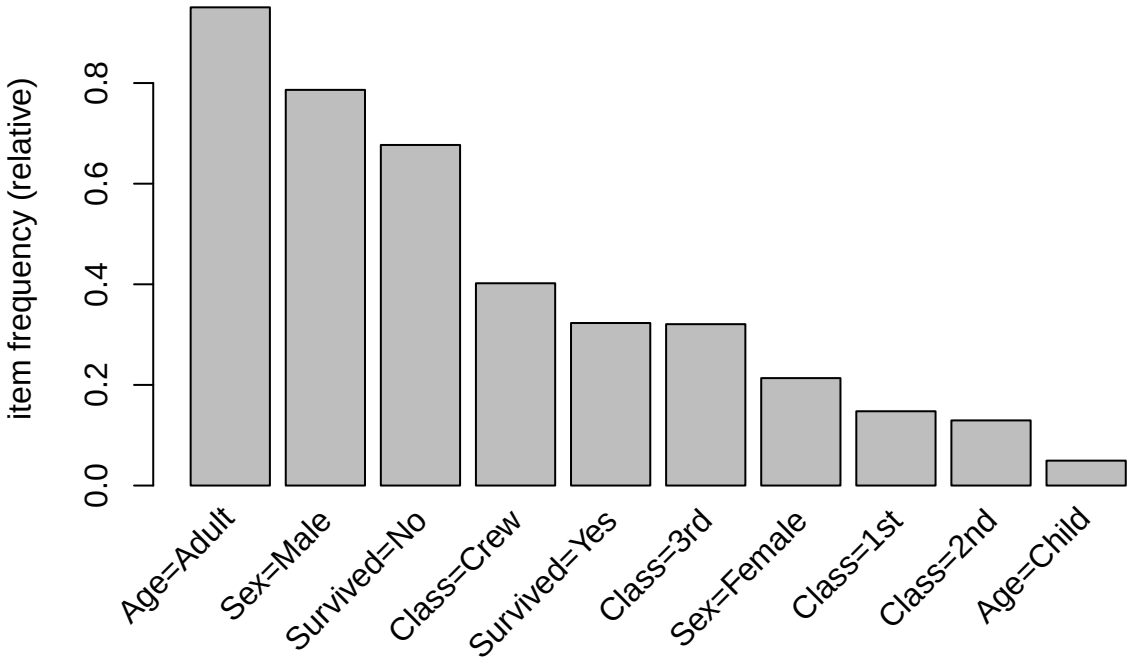
```
library(arulesViz)
par(mfrow=c(2,2))
plot(rule, method = 'paracoord')
```



```
plot(rule, method = 'graph')
```




```
itemFrequencyPlot(titanic, topN = 10)
```



Rule Associated with Survival

```
rule1 = apriori(titanic, parameter = list(supp = 0.1, conf = 0.6),
                appearance = list(default = 'lhs', rhs = 'Survived=Yes'))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 220
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [2 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(head(rule1))
```

```
##     lhs                         rhs            support   confidence coverage
## [1] {Sex=Female}            => {Survived=Yes} 0.1562926 0.7319149  0.2135393
## [2] {Sex=Female, Age=Adult} => {Survived=Yes} 0.1435711 0.7435294  0.1930940
##     lift     count
## [1] 2.265745 344
## [2] 2.301699 316
```

Rule Associated with

```
rule2 = apriori(titanic, parameter = list(supp = 0.1, conf = 0.05),
                appearance = list(lhs = c('Class=1st', 'Class=2nd', 'Class=3rd', 'Age=Child',
                                          'Age=Adult'), rhs = 'Survived=Yes'))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 220
##
## set item appearances ...[6 item(s)] done [0.00s].
## set transactions ...[6 item(s), 2201 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
```

```
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [2 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(head(rule2))
```

```
##     lhs            rhs            support   confidence coverage  lift
## [1] {}          => {Survived=Yes} 0.3230350 0.3230350  1.0000000 1.0000000
## [2] {Age=Adult} => {Survived=Yes} 0.2971377 0.3126195  0.9504771 0.9677574
##     count
## [1] 711
## [2] 654
```

Appendix

#Perlombongan Aturan Sekutuan

#Data: groceries.csv

library(arules)

install.packages(“arulesViz”)

library(arulesViz)

#import data transaksi

tdata<- read.transactions(file.choose(), sep=“,”)

#deskriptif data

inspect(head(tdata, 20))

LIST(head(tdata, 20))

#bilangan item bagi setiap traksaksi/pembelian

size(head(tdata, 20))

#i)Bagaimana untuk melihat item yang paling kerap #dibeli?

itemFrequencyPlot(tdata, topN=10, main=“Item paling kerap dibeli”)

#ii) Bagaimana untuk mendapatkan Aturan Sekutuan yang #signifikan bagi cadangan produk?

#set nilai minimum ambang sokongan & keyakinan #support=0.01, confidence=0.3

Aturan.S1<- apriori(tdata, parameter=list(supp=0.01, conf=0.3))

inspect(Aturan.S1)

#lihat aturan dengan tertib lif menurun

Aturan.S2<- sort(Aturan.S1, by=“lift”, decreasing=T)

inspect(Aturan.S2)

#lihat aturan dengan tertib keyakinan menurun

Aturan.S3<- sort(Aturan.S1, by=“confidence”, decreasing=T)

inspect(Aturan.S3)

lihat aturan dengan tertib support menurun

Aturan.S4<- sort(Aturan.S1, by=“support”, decreasing=T)

inspect(Aturan.S4)

#Pengvisualan aturan sekutuan

#plot rangkaian

plot(Aturan.S4, method=“graph”)

#plot koordinat selari plot(Aturan.S4, method=“paracoord”, control=list(reorder=T))

#jika bilangan aturan sekutuan yg diperolehi adalah #terlalu banyak #tinggikan nilam ambang ukuran untuk kekalkan aturan #sekutuan yg lebih bermakna sahaja

#support=0.01, confidence=0.5) Aturan.S5<- apriori(tdata, parameter=list(supp=0.01, conf=0.5)) inspect(Aturan.S5)

#plot rangkaian plot(Aturan.S5, method="graph")

#plot koordinat selari windows(10,10) plot(Aturan.S5, method="paracoord", control=list(reorder=T))

#iii) Bagaimana untuk mendapatkan Aturan Sekutuan #yang berkait dengan item tertentu?

#kes 1: dapatkan aturan yang mempengaruhi pembelian #item Y (RHS)

#Contoh: Barangan apa yang akan dibeli terlebih dahulu #sebelum pelanggan membeli item "yogurt"

Aturan.S6<- apriori(tdata, parameter=list(supp=0.01, conf=0.08), appearance=list(default="lhs", rhs="yogurt")) inspect(Aturan.S6)

#plot rangkaian plot(Aturan.S6, method="graph")

#plot koordinat selari windows(10,10) plot(Aturan.S6, method="paracoord", control=list(reorder=T))

#kes 2: dapatkan aturan bagi produk apa yang biasa #dibeli selepas pembelian item X (LHS)

#Contoh: Barangan apa yang akan akan dibeli #selepas pelanggan membeli item "yogurt"

Aturan.S7<- apriori(tdata, parameter=list(supp=0.01, conf=0.08), appearance=list(default="rhs", lhs="yogurt")) inspect(Aturan.S7)

#plot rangkaian plot(Aturan.S7, method="graph")

#plot koordinat selari windows(10,10) plot(Aturan.S7, method="paracoord", control=list(reorder=T))

Aturan.S8<- sort(Aturan.S7, by="lift", decreasing=T) inspect(Aturan.S8)

##Latihan #set nilai minimum ambang sokongan & keyakinan #support=0.1, confidence=0.6

Aturan.Lat<- apriori(titanic.raw, parameter=list(supp=0.1, conf=0.6)) inspect(Aturan.Lat)

#Dapatkan deskriptif statistik bagi data. summary(titanic.raw)

#Plotkan aturan sekutuan yg diperoleh
#menggunakan plot-plot yang sesuai. #plot rangkaian plot(Aturan.Lat, method="graph")

#Dapatkan aturan sekutuan yang menunjukkan ciri-ciri #individu yang terselamat dari tragedi titanic #(rhs: p/ubah survival).

Aturan.Lat2<- apriori(titanic.raw, parameter=list(supp=0.1, conf=0.6), appearance=list(default="lhs", rhs="Survived=Yes")) inspect(Aturan.Lat2)

#Pengvisualan aturan sekutuan #plot rangkaian plot(Aturan.Lat2, method="graph")

#plot koordinat selari plot(Aturan.Lat2, method="paracoord", control=list(reorder=T))

#Dapatkan aturan sekutuan bagi orang yang terselamat #daripada kelas 1, 2 & 3 (rhs ialah "Survived=Yes" #dan lhs mengandungi info Class=1st, 2nd & #3rd; Age=Child & Adult)

Aturan.Lat3<- apriori(titanic.raw, parameter=list(supp=0.01, conf=0.6), appearance=list(lhs=c("Class=1st", "Class=2nd", "Class=3rd", "Age=Child", "Age=Adult"), rhs="Survived=Yes")) inspect(Aturan.Lat3)

#Pengvisualan aturan sekutuan #plot rangkaian plot(Aturan.Lat3, method="graph")

#plot koordinat selari plot(Aturan.Lat3, method="paracoord", control=list(reorder=T))

#Perlombongan Data Jujukan install.packages("TraMineR") library(TraMineR)

data(mvad) head(mvad)

#lajur 1 hingga 14 adalah maklumat demografi #lajur 1 hingga 14 bukan data jujukan

#data jujukan bermula lajur 15 hingga 86 #perlu takrifkan lajur 15 hingga 86 kepada format data #jujukan

#takrifkan label & kod bagi setiap keadaan

mvad.labels<- c("employment", "further education", "higher education", "joblessness", "school", "training")

mvad.scode<- c("EM", "FE", "HE", "JL", "SC", "TR")

#bina data kelas jujukan mvad.seq<- seqdef(mvad, 15:86, states=mvad.scode, labels=mvad.labels, xtstep=6)

#Penunjuk ringkasan statistik #i)Min masa proses berada dalam setiap keadaan seqmeant(mvad.seq)

#ii)Min masa proses berada dalam setiap keadaan #bagi kumpulan tertentu.

#bagi kumpulan jantina by(mvad.seq, mvad$male, seqmeant)

#bagi kumpulan status pekerjaan bapa by(mvad.seq, mvad$funemp, seqmeant)

#pengvisulan seqmtplot(mvad.seq, group=mvad$male, main = "Lelaki")seqmtplot(mvad.seq, group = mvad$funemp, main="Bapa Bekerja")

#iii)Bilangan transisi (peralihan). dat1<- seqtransn(mvad.seq) hist(dat1, main="Bilangan transisi") mean(dat1) sd(dat1) table(dat1)

#iv) Kadar peralihan mvad.trate<- seqtrate(mvad.seq)

#v) Keadaan peralihan yang bergantung terhadap masa mvad.trate2<- seqtrate(mvad.seq, time.varying=T)

#Pengvisualan #i) Plot indeks jujukan #20 individu pertama seqiplot(mvad.seq, main="Plot indeks jujukan", idxs=1:20)

#100 individu pertama windows(10,10) seqiplot(mvad.seq, main="Plot indeks jujukan", idxs=1:100)

#individu ke 20 hingga 30 seqiplot(mvad.seq, main="Plot indeks jujukan", idxs=20:30)

#pilih individu yang khusus #(1, 2, 15, 90, 200, 267, 456, 666, 700) seqiplot(mvad.seq, main="Plot indeks jujukan", idxs=c(1,2,15,90,200,267,456,666,700))