Shopping数据关联规则挖掘

1. 数据预处理

导入shopping数据，将购买数据转为逻辑变量，查看变量性质。

变量分为两类：家居用品（主要是食物）以及人口统计特征（如性别、年龄、婚姻状况等）

```
load("shopping.rda")
for (i in 1:10) {
    shopping[, i] = as.logical(shopping[, i])
}
summary(shopping)
```

```
##  Ready_made      Frozen_foods     Alcohol         Fresh_Vegetables
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:399       FALSE:470       FALSE:476       FALSE:721
##  TRUE :387       TRUE :316       TRUE :310       TRUE :65
##  NA's :0         NA's :0         NA's :0         NA's :0
##
##     Milk         Bakery_goods    Fresh_meat      Toiletries
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:638       FALSE:449       FALSE:763       FALSE:708
##  TRUE :148       TRUE :337       TRUE :23        TRUE :78
##  NA's :0         NA's :0         NA's :0         NA's :0
##
##    Snacks        Tinned_goods       GENDER          Age
##  Mode :logical   Mode :logical   Female:423   18 to 30:236
##  FALSE:413       FALSE:428       Male  :363   31 to 40:195
##  TRUE :373       TRUE :358                    41 to 50:134
##  NA's :0         NA's :0                      51 to 60:130
##                                               Over 60 : 91
##       MARITAL    CHILDREN  WORKING
##  Divorced : 98   No :513   No :132
##  Married  :189   Yes:273   Yes:654
##  Separated:150
##  Single   :199
##  Widowed  :150
```

可以看到数据无缺失值，在食物类数据中，有的食物种类的购买量与非购买量是比较均匀的，而有的食物如新鲜蔬菜、鲜肉等购买行为较少，在这种情形下我们实际关注的是人们购买稀有食品的行为；因此，当不购买某种食品的比例过高时（设为0.2），我们忽略这种行为。

```
shopping[, 1:10] = apply(shopping[, 1:10], 2, function(x) {
    x = as.logical(x)
    y = table(x)/length(x)
    if (y[2] <= 0.2)
        x[!x] = NA
    return(x)
})
summary(shopping)
```

```
##  Ready_made      Frozen_foods     Alcohol        Fresh_Vegetables
##  Mode :logical   Mode :logical   Mode :logical   Mode:logical
##  FALSE:399       FALSE:470       FALSE:476       TRUE:65
##  TRUE :387       TRUE :316       TRUE :310       NA's:721
##  NA's :0         NA's :0         NA's :0
##
##    Milk          Bakery_goods    Fresh_meat     Toiletries
##  Mode:logical   Mode :logical   Mode:logical   Mode:logical
##  TRUE:148       FALSE:449       TRUE:23        TRUE:78
##  NA's:638       TRUE :337       NA's:763       NA's:708
##                 NA's :0
##
##    Snacks         Tinned_goods       GENDER          Age
##  Mode :logical   Mode :logical   Female:423   18 to 30:236
##  FALSE:413       FALSE:428       Male  :363   31 to 40:195
##  TRUE :373       TRUE :358                    41 to 50:134
##  NA's :0         NA's :0                      51 to 60:130
##                                               Over 60 : 91
##       MARITAL    CHILDREN  WORKING
##  Divorced : 98   No :513   No :132
##  Married  :189   Yes:273   Yes:654
##  Separated:150
##  Single   :199
##  Widowed  :150
```

2. 关联规则挖掘

用R中的arules包挖掘关联规则。

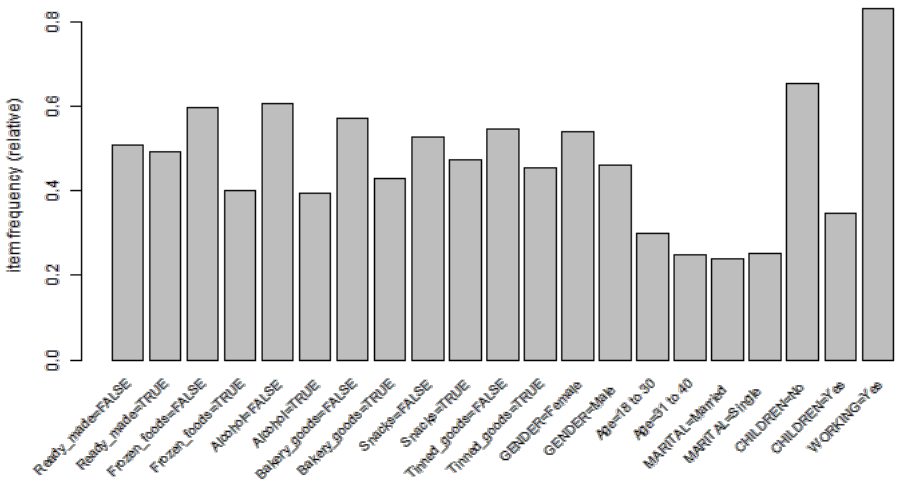
将数据转化成transaction格式。

```
library(arules)
shop_trans = as(shopping, "transactions")
summary(shop_trans)
```

```
## transactions as itemMatrix in sparse format with
##  786 rows (elements/itemsets/transactions) and
##  32 columns (items) and a density of 0.3562
##
## most frequent items:
##        WORKING=Yes       CHILDREN=No       Alcohol=FALSE
##                654                513                 476
## Frozen_foods=FALSE Bakery_goods=FALSE            (Other)
##                470                449                6398
##
## element (itemset/transaction) length distribution:
## sizes
##  11  12  13  14  15
## 557 159  56  13   1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    11.0    11.0    11.0    11.4    12.0    15.0
##
## includes extended item information - examples:
##               labels    variables levels
## 1   Ready_made=FALSE   Ready_made  FALSE
## 2    Ready_made=TRUE   Ready_made   TRUE
## 3 Frozen_foods=FALSE Frozen_foods  FALSE
##
## includes extended transaction information - examples:
##   transactionID
## 1             1
## 2             2
## 3             3
```

查看item的频率

```
itemFrequencyPlot(shop_trans, support = 0.2, cex.names = 0.8)
```



用apriori算法挖掘关联规则。查看item的频率分布，可知item freqency的25分位数为0.1838，设置最小support为0.1，confidence为0.7,lift值为1.2

```
itemFreq = itemFrequency(shop_trans, type = "relative")
quantile(itemFreq)
```

```
##      0%     25%     50%     75%    100%
## 0.02926 0.18384 0.37087 0.51209 0.83206
```

```
# apriori
rules <- apriori(shop_trans, parameter = list(support = 0.1, confidence = 0.7))
```

```
##
## parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.7    0.1    1 none FALSE            TRUE     0.1      1     10
##  target   ext
##   rules FALSE
##
## algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## apriori - find association rules with the apriori algorithm
## version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[32 item(s), 786 transaction(s)] done [0.00s].
## sorting and recoding items ... [29 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.00s].
## writing ... [1227 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rule_lif = subset(rules, subset = lift > 1.2)
summary(rule_lif)
```

```
## set of 633 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3   4   5   6
##   8 129 324 167   5
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    4.00    4.00    4.05    5.00    6.00
##
## summary of quality measures:
##     support        confidence         lift
##  Min.   :0.101   Min.   :0.700   Min.   :1.20
##  1st Qu.:0.108   1st Qu.:0.739   1st Qu.:1.27
##  Median :0.122   Median :0.784   Median :1.34
##  Mean   :0.134   Mean   :0.790   Mean   :1.38
##  3rd Qu.:0.148   3rd Qu.:0.829   3rd Qu.:1.44
##  Max.   :0.396   Max.   :0.963   Max.   :2.66
##
## mining info:
##        data ntransactions support confidence
##  shop_trans           786     0.1        0.7
```

分别按照support和confidence排序，查看前20的规则。

```
inspect(head(sort(rule_lif, by = "support"), n = 20))
```

```
##    lhs                       rhs                    support confidence  lift
## 1  {Ready_made=TRUE}      => {CHILDREN=No }         0.3957     0.8036 1.231
## 2  {Ready_made=TRUE,
##     WORKING=Yes}          => {CHILDREN=No }         0.3537     0.8225 1.260
## 3  {Alcohol=TRUE}         => {CHILDREN=No }         0.3270     0.8290 1.270
## 4  {Frozen_foods=FALSE,
##     Bakery_goods=FALSE}   => {Alcohol=FALSE}        0.2990     0.7655 1.264
## 5  {Alcohol=FALSE,
##     Bakery_goods=FALSE}   => {Frozen_foods=FALSE}   0.2990     0.7630 1.276
## 6  {Alcohol=TRUE,
##     WORKING=Yes}          => {CHILDREN=No }         0.2939     0.8493 1.301
## 7  {Frozen_foods=TRUE,
##     WORKING=Yes}          => {CHILDREN=No }         0.2837     0.8168 1.252
## 8  {CHILDREN=Yes}         => {Alcohol=FALSE}        0.2799     0.8059 1.331
## 9  {Bakery_goods=FALSE,
##     Tinned_goods=FALSE}   => {CHILDREN=No }         0.2735     0.7963 1.220
## 10 {Bakery_goods=FALSE,
##     CHILDREN=No }         => {Tinned_goods=FALSE}   0.2735     0.7049 1.295
## 11 {Frozen_foods=FALSE,
##     GENDER=Female}        => {Alcohol=FALSE}        0.2634     0.7695 1.271
## 12 {Alcohol=FALSE,
##     GENDER=Female}        => {Frozen_foods=FALSE}   0.2634     0.7473 1.250
## 13 {CHILDREN=Yes}         => {Frozen_foods=FALSE}   0.2595     0.7473 1.250
## 14 {CHILDREN=Yes}         => {Ready_made=FALSE}     0.2506     0.7216 1.422
## 15 {Frozen_foods=FALSE,
##     Snacks=FALSE}         => {Alcohol=FALSE}        0.2494     0.7396 1.221
## 16 {Snacks=FALSE,
##     Tinned_goods=FALSE}   => {CHILDREN=No }         0.2405     0.8182 1.254
## 17 {Frozen_foods=FALSE,
##     Bakery_goods=FALSE,
##     WORKING=Yes}          => {Alcohol=FALSE}        0.2392     0.7642 1.262
## 18 {Alcohol=FALSE,
##     Bakery_goods=FALSE,
##     WORKING=Yes}          => {Frozen_foods=FALSE}   0.2392     0.7520 1.258
## 19 {Alcohol=FALSE,
##     Tinned_goods=FALSE}   => {Frozen_foods=FALSE}   0.2379     0.7362 1.231
## 20 {Bakery_goods=FALSE,
##     Tinned_goods=FALSE,
##     WORKING=Yes}          => {CHILDREN=No }         0.2379     0.7924 1.214
```

```
inspect(head(sort(rule_lif, by = "confidence"), n = 20))
```

```
##    lhs                      rhs                    support confidence  lift
## 1  {Frozen_foods=TRUE,
##     Alcohol=TRUE,
##     GENDER=Male  ,
##     WORKING=Yes}         => {CHILDREN=No }         0.1005     0.9634 1.476
## 2  {Frozen_foods=TRUE,
##     Tinned_goods=FALSE,
##     GENDER=Male  }       => {CHILDREN=No }         0.1094     0.9451 1.448
## 3  {Alcohol=TRUE,
##     Bakery_goods=FALSE,
##     Tinned_goods=FALSE,
##     WORKING=Yes}         => {CHILDREN=No }         0.1043     0.9425 1.444
## 4  {Frozen_foods=FALSE,
##     Bakery_goods=FALSE,
##     CHILDREN=Yes,
##     WORKING=Yes}         => {Alcohol=FALSE}        0.1132     0.9368 1.547
## 5  {Frozen_foods=FALSE,
##     Alcohol=FALSE,
##     Tinned_goods=FALSE,
##     CHILDREN=No }        => {Ready_made=TRUE}      0.1260     0.9340 1.897
## 6  {Tinned_goods=FALSE,
##     GENDER=Male  ,
##     Age=18 to 30}        => {CHILDREN=No }         0.1247     0.9333 1.430
## 7  {Bakery_goods=FALSE,
##     Snacks=FALSE,
##     Tinned_goods=FALSE,
##     GENDER=Male  }       => {CHILDREN=No }         0.1056     0.9326 1.429
## 8  {Alcohol=TRUE,
##     Bakery_goods=FALSE,
##     Tinned_goods=FALSE} => {CHILDREN=No }         0.1209     0.9314 1.427
## 9  {Bakery_goods=FALSE,
##     GENDER=Male  ,
##     Age=18 to 30}        => {CHILDREN=No }         0.1031     0.9310 1.426
## 10 {GENDER=Male  ,
##     Age=18 to 30,
##     MARITAL=Single   }   => {CHILDREN=No }         0.1005     0.9294 1.424
## 11 {Frozen_foods=TRUE,
##     Alcohol=TRUE,
##     GENDER=Male  }       => {CHILDREN=No }         0.1145     0.9278 1.422
## 12 {Ready_made=TRUE,
##     Tinned_goods=FALSE,
##     GENDER=Male  }       => {CHILDREN=No }         0.1285     0.9266 1.420
## 13 {Alcohol=TRUE,
##     Bakery_goods=FALSE,
##     WORKING=Yes}         => {CHILDREN=No }         0.1425     0.9256 1.418
## 14 {Frozen_foods=FALSE,
##     Alcohol=FALSE,
##     Tinned_goods=FALSE,
##     CHILDREN=No ,
##     WORKING=Yes}         => {Ready_made=TRUE}      0.1107     0.9255 1.880
## 15 {Ready_made=FALSE,
##     Frozen_foods=FALSE,
##     Bakery_goods=FALSE,
##     CHILDREN=Yes}        => {Alcohol=FALSE}        0.1221     0.9231 1.524
## 16 {Ready_made=FALSE,
##     Alcohol=FALSE,
##     Bakery_goods=FALSE,
##     CHILDREN=Yes}        => {Frozen_foods=FALSE}   0.1221     0.9231 1.544
## 17 {GENDER=Male  ,
##     Age=18 to 30}        => {CHILDREN=No }         0.1489     0.9213 1.412
## 18 {Frozen_foods=FALSE,
##     Bakery_goods=FALSE,
##     CHILDREN=Yes}        => {Alcohol=FALSE}        0.1476     0.9206 1.520
## 19 {Ready_made=TRUE,
##     Tinned_goods=FALSE,
##     GENDER=Male  ,
##     WORKING=Yes}         => {CHILDREN=No }         0.1145     0.9184 1.407
## 20 {Ready_made=TRUE,
##     Bakery_goods=FALSE,
##     Tinned_goods=FALSE} => {CHILDREN=No }         0.1527     0.9160 1.404
```

2. 进一步挖掘

可以看到得到的规则鱼龙混杂，如果要进行有方向的关联规则挖掘，则应该将数据转化为dataframe形式，通过正则表达式做进行进一步处理。对于rules按照support confidence降序排列。

```
### convert to dataframe
Rules = as(rule_lif, "data.frame")
Rules$rules = as.character(Rules$rules)
rule_spl = do.call(rbind, sapply(Rules$rules, strsplit, "=>"))
rownames(rule_spl) = NULL
colnames(rule_spl) = c("lhs", "rhs")
Rules = data.frame(Rules, rule_spl)
Rules = Rules[order(Rules$support, Rules$confidence, decreasing = T), ]
```

(1) 购买行为

当要求right hand rules必须出现日常用品项时，我们可以把视角局限到人们的购买行为。

```
supervise = Rules[grepl("(TRUE)|(FALSE)", Rules$rhs, perl = T), ]
supervise[1:20, 1:4]
```

```
##                                                                      rules
## 356             {Frozen_foods=FALSE,Bakery_goods=FALSE} => {Alcohol=FALSE}
## 357             {Alcohol=FALSE,Bakery_goods=FALSE} => {Frozen_foods=FALSE}
## 21                                        {CHILDREN=Yes} => {Alcohol=FALSE}
## 341              {Bakery_goods=FALSE,CHILDREN=No } => {Tinned_goods=FALSE}
## 351                 {Frozen_foods=FALSE,GENDER=Female} => {Alcohol=FALSE}
## 352                 {Alcohol=FALSE,GENDER=Female} => {Frozen_foods=FALSE}
## 20                                    {CHILDREN=Yes} => {Frozen_foods=FALSE}
## 19                                      {CHILDREN=Yes} => {Ready_made=FALSE}
## 332                   {Frozen_foods=FALSE,Snacks=FALSE} => {Alcohol=FALSE}
## 964 {Frozen_foods=FALSE,Bakery_goods=FALSE,WORKING=Yes} => {Alcohol=FALSE}
## 965 {Alcohol=FALSE,Bakery_goods=FALSE,WORKING=Yes} => {Frozen_foods=FALSE}
## 343             {Alcohol=FALSE,Tinned_goods=FALSE} => {Frozen_foods=FALSE}
## 945  {Bakery_goods=FALSE,CHILDREN=No ,WORKING=Yes} => {Tinned_goods=FALSE}
## 182                   {Frozen_foods=FALSE,CHILDREN=Yes} => {Alcohol=FALSE}
## 183                   {Alcohol=FALSE,CHILDREN=Yes} => {Frozen_foods=FALSE}
## 319          {Ready_made=FALSE,Frozen_foods=FALSE} => {Bakery_goods=FALSE}
## 16                                    {Age=18 to 30} => {Tinned_goods=FALSE}
## 958      {Frozen_foods=FALSE,GENDER=Female,WORKING=Yes} => {Alcohol=FALSE}
## 959      {Alcohol=FALSE,GENDER=Female,WORKING=Yes} => {Frozen_foods=FALSE}
## 349                 {Bakery_goods=FALSE,GENDER=Female} => {Alcohol=FALSE}
##     support confidence  lift
## 356  0.2990     0.7655 1.264
## 357  0.2990     0.7630 1.276
## 21   0.2799     0.8059 1.331
## 341  0.2735     0.7049 1.295
## 351  0.2634     0.7695 1.271
## 352  0.2634     0.7473 1.250
## 20   0.2595     0.7473 1.250
## 19   0.2506     0.7216 1.422
## 332  0.2494     0.7396 1.221
## 964  0.2392     0.7642 1.262
## 965  0.2392     0.7520 1.258
## 343  0.2379     0.7362 1.231
## 945  0.2379     0.7083 1.301
## 182  0.2265     0.8725 1.441
## 183  0.2265     0.8091 1.353
## 319  0.2265     0.7149 1.251
## 16   0.2214     0.7373 1.354
## 958  0.2163     0.7556 1.248
## 959  0.2163     0.7359 1.231
## 349  0.2150     0.7478 1.235
```

进一步的，我们可以分类查看不同类别的商品关联的购买行为。

```
cate_supvise = lapply(colnames(shopping[1:10]), function(x) {
    y = Rules[grepl(x, Rules$rhs, perl = T), 1:4]
    cat(x, "\n")
    show(y[1:min(nrow(y), 10), ])
    cat("\n")
    return(y)
})
```

```
## Ready_made
##                                                                      rules
## 19                                    {CHILDREN=Yes} => {Ready_made=FALSE}
```

```
## 166                    {Alcohol=FALSE,CHILDREN=Yes} => {Ready_made=FALSE}
## 164                {Frozen_foods=FALSE,CHILDREN=Yes} => {Ready_made=FALSE}
## 246                  {Bakery_goods=TRUE,CHILDREN=No } => {Ready_made=TRUE}
## 168                      {CHILDREN=Yes,WORKING=Yes} => {Ready_made=FALSE}
## 632     {Bakery_goods=TRUE,CHILDREN=No ,WORKING=Yes} => {Ready_made=TRUE}
## 459 {Frozen_foods=FALSE,Alcohol=FALSE,CHILDREN=Yes} => {Ready_made=FALSE}
## 853  {Frozen_foods=FALSE,Alcohol=FALSE,CHILDREN=No } => {Ready_made=TRUE}
## 466        {Alcohol=FALSE,CHILDREN=Yes,WORKING=Yes} => {Ready_made=FALSE}
## 161                    {GENDER=Female,CHILDREN=Yes} => {Ready_made=FALSE}
##     support confidence  lift
## 19   0.2506     0.7216 1.422
## 166  0.2112     0.7545 1.486
## 164  0.1947     0.7500 1.477
## 246  0.1947     0.7356 1.494
## 168  0.1845     0.7073 1.393
## 632  0.1756     0.7459 1.515
## 459  0.1743     0.7697 1.516
## 853  0.1603     0.7730 1.570
## 466  0.1552     0.7439 1.465
## 161  0.1552     0.7093 1.397
##
## Frozen_foods
##                                                                    rules
## 357             {Alcohol=FALSE,Bakery_goods=FALSE} => {Frozen_foods=FALSE}
## 352                  {Alcohol=FALSE,GENDER=Female} => {Frozen_foods=FALSE}
## 20                                  {CHILDREN=Yes} => {Frozen_foods=FALSE}
## 965 {Alcohol=FALSE,Bakery_goods=FALSE,WORKING=Yes} => {Frozen_foods=FALSE}
## 343             {Alcohol=FALSE,Tinned_goods=FALSE} => {Frozen_foods=FALSE}
## 183                   {Alcohol=FALSE,CHILDREN=Yes} => {Frozen_foods=FALSE}
## 959      {Alcohol=FALSE,GENDER=Female,WORKING=Yes} => {Frozen_foods=FALSE}
## 348             {Bakery_goods=FALSE,GENDER=Female} => {Frozen_foods=FALSE}
## 309                {Ready_made=TRUE,Alcohol=FALSE} => {Frozen_foods=FALSE}
## 947 {Alcohol=FALSE,Tinned_goods=FALSE,WORKING=Yes} => {Frozen_foods=FALSE}
##     support confidence  lift
## 357  0.2990     0.7630 1.276
## 352  0.2634     0.7473 1.250
## 20   0.2595     0.7473 1.250
## 965  0.2392     0.7520 1.258
## 343  0.2379     0.7362 1.231
## 183  0.2265     0.8091 1.353
## 959  0.2163     0.7359 1.231
## 348  0.2137     0.7434 1.243
## 309  0.2125     0.7591 1.269
## 947  0.2087     0.7421 1.241
##
## Alcohol
##                                                                    rules
## 356             {Frozen_foods=FALSE,Bakery_goods=FALSE} => {Alcohol=FALSE}
## 21                                  {CHILDREN=Yes} => {Alcohol=FALSE}
## 351                  {Frozen_foods=FALSE,GENDER=Female} => {Alcohol=FALSE}
## 332                   {Frozen_foods=FALSE,Snacks=FALSE} => {Alcohol=FALSE}
## 964 {Frozen_foods=FALSE,Bakery_goods=FALSE,WORKING=Yes} => {Alcohol=FALSE}
## 182                    {Frozen_foods=FALSE,CHILDREN=Yes} => {Alcohol=FALSE}
## 958      {Frozen_foods=FALSE,GENDER=Female,WORKING=Yes} => {Alcohol=FALSE}
## 349                  {Bakery_goods=FALSE,GENDER=Female} => {Alcohol=FALSE}
## 308                {Ready_made=TRUE,Frozen_foods=FALSE} => {Alcohol=FALSE}
## 165                    {Ready_made=FALSE,CHILDREN=Yes} => {Alcohol=FALSE}
##     support confidence  lift
## 356  0.2990     0.7655 1.264
## 21   0.2799     0.8059 1.331
## 351  0.2634     0.7695 1.271
## 332  0.2494     0.7396 1.221
## 964  0.2392     0.7642 1.262
## 182  0.2265     0.8725 1.441
## 958  0.2163     0.7556 1.248
## 349  0.2150     0.7478 1.235
## 308  0.2125     0.7557 1.248
## 165  0.2112     0.8426 1.391
##
## Fresh_Vegetables
##    rules support confidence lift
## NA  <NA>      NA          NA   NA
##
## Milk
##    rules support confidence lift
## NA  <NA>      NA          NA   NA
##
## Bakery_goods
##                                                                  rules
```

```
## 319                {Ready_made=FALSE,Frozen_foods=FALSE} => {Bakery_goods=FALSE}
## 320                      {Ready_made=FALSE,CHILDREN=No } => {Bakery_goods=FALSE}
## 886    {Ready_made=FALSE,Frozen_foods=FALSE,WORKING=Yes} => {Bakery_goods=FALSE}
## 893       {Snacks=FALSE,Tinned_goods=FALSE,CHILDREN=No } => {Bakery_goods=FALSE}
## 260                     {Ready_made=FALSE,GENDER=Male  } => {Bakery_goods=FALSE}
## 937 {Frozen_foods=FALSE,Tinned_goods=FALSE,CHILDREN=No } => {Bakery_goods=FALSE}
## 884  {Ready_made=FALSE,Frozen_foods=FALSE,Alcohol=FALSE} => {Bakery_goods=FALSE}
## 889          {Ready_made=FALSE,CHILDREN=No ,WORKING=Yes} => {Bakery_goods=FALSE}
## 941      {Alcohol=FALSE,Tinned_goods=FALSE,CHILDREN=No } => {Bakery_goods=FALSE}
## 962      {Frozen_foods=FALSE,Alcohol=FALSE,CHILDREN=No } => {Bakery_goods=FALSE}
##     support confidence  lift
## 319  0.2265     0.7149 1.251
## 320  0.1870     0.7277 1.274
## 886  0.1756     0.7077 1.239
## 893  0.1718     0.7143 1.250
## 260  0.1692     0.7389 1.293
## 937  0.1628     0.7072 1.238
## 884  0.1616     0.7299 1.278
## 889  0.1578     0.7251 1.269
## 941  0.1527     0.7453 1.305
## 962  0.1514     0.7301 1.278
##
## Fresh_meat
##    rules support confidence lift
## NA  <NA>      NA          NA   NA
##
## Toiletries
##    rules support confidence lift
## NA  <NA>      NA          NA   NA
##
## Snacks
##
rules
## 1076  {Ready_made=TRUE,Frozen_foods=FALSE,Tinned_goods=FALSE,CHILDREN=No } =>
{Snacks=FALSE}
## 708                {Ready_made=FALSE,Bakery_goods=FALSE,Tinned_goods=TRUE} =>
{Snacks=FALSE}
## 1071 {Ready_made=TRUE,Frozen_foods=FALSE,Alcohol=FALSE,Tinned_goods=FALSE} =>
{Snacks=FALSE}
##      support confidence  lift
## 1076  0.1132     0.7120 1.355
## 708   0.1120     0.7097 1.351
## 1071  0.1069     0.7000 1.332
##
## Tinned_goods
##                                                                          rules
## 341                    {Bakery_goods=FALSE,CHILDREN=No } => {Tinned_goods=FALSE}
## 945        {Bakery_goods=FALSE,CHILDREN=No ,WORKING=Yes} => {Tinned_goods=FALSE}
## 16                                          {Age=18 to 30} => {Tinned_goods=FALSE}
## 137                          {Age=18 to 30,CHILDREN=No } => {Tinned_goods=FALSE}
## 894       {Bakery_goods=FALSE,Snacks=FALSE,CHILDREN=No } => {Tinned_goods=FALSE}
## 139                           {Age=18 to 30,WORKING=Yes} => {Tinned_goods=FALSE}
## 826     {Ready_made=TRUE,Frozen_foods=FALSE,WORKING=Yes} => {Tinned_goods=FALSE}
## 300                  {Ready_made=TRUE,Bakery_goods=FALSE} => {Tinned_goods=FALSE}
## 938 {Frozen_foods=FALSE,Bakery_goods=FALSE,CHILDREN=No } => {Tinned_goods=FALSE}
## 676      {Bakery_goods=FALSE,GENDER=Male  ,CHILDREN=No } => {Tinned_goods=FALSE}
##     support confidence  lift
## 341  0.2735     0.7049 1.295
## 945  0.2379     0.7083 1.301
## 16   0.2214     0.7373 1.354
## 137  0.1718     0.7714 1.417
## 894  0.1718     0.7143 1.312
## 139  0.1692     0.7348 1.349
## 826  0.1679     0.7097 1.303
## 300  0.1667     0.7043 1.293
## 938  0.1628     0.7072 1.299
## 676  0.1603     0.7925 1.455
```

如果我们只关注相关人群特征的购买行为，则可以对规则做限制，使人群特征只出现在左边，而购买行为只出现在右边。

```
string = paste("(", paste(colnames(shopping[11:15]), collapse = ")|("), ")",
    sep = "")
people = Rules[grepl(string, Rules$lhs) & !grepl("(TRUE)|(FALSE)", Rules$lhs) &
    grepl("(TRUE)|(FALSE)", Rules$rhs) & !grepl(string, Rules$rhs), ]
people[1:20, 1:4]
```

```
##                                                                      rules
## 21                                  {CHILDREN=Yes} => {Alcohol=FALSE}
## 20                                {CHILDREN=Yes} => {Frozen_foods=FALSE}
## 19                                  {CHILDREN=Yes} => {Ready_made=FALSE}
## 16                                {Age=18 to 30} => {Tinned_goods=FALSE}
## 187                     {CHILDREN=Yes,WORKING=Yes} => {Alcohol=FALSE}
## 185                 {CHILDREN=Yes,WORKING=Yes} => {Frozen_foods=FALSE}
## 168                   {CHILDREN=Yes,WORKING=Yes} => {Ready_made=FALSE}
## 177                    {GENDER=Female,CHILDREN=Yes} => {Alcohol=FALSE}
## 137                {Age=18 to 30,CHILDREN=No } => {Tinned_goods=FALSE}
## 139                 {Age=18 to 30,WORKING=Yes} => {Tinned_goods=FALSE}
## 176               {GENDER=Female,CHILDREN=Yes} => {Frozen_foods=FALSE}
## 161                 {GENDER=Female,CHILDREN=Yes} => {Ready_made=FALSE}
## 482       {GENDER=Female,CHILDREN=Yes,WORKING=Yes} => {Alcohol=FALSE}
## 110          {MARITAL=Single   ,CHILDREN=No } => {Tinned_goods=FALSE}
## 115          {MARITAL=Single   ,CHILDREN=No } => {Bakery_goods=FALSE}
## 121              {GENDER=Male  ,Age=18 to 30} => {Tinned_goods=FALSE}
## 99           {Age=18 to 30,MARITAL=Single   } => {Tinned_goods=FALSE}
## 480  {GENDER=Female,CHILDREN=Yes,WORKING=Yes} => {Frozen_foods=FALSE}
## 450    {GENDER=Female,CHILDREN=Yes,WORKING=Yes} => {Ready_made=FALSE}
## 389 {GENDER=Male  ,Age=18 to 30,CHILDREN=No } => {Tinned_goods=FALSE}
##     support confidence  lift
## 21   0.2799     0.8059 1.331
## 20   0.2595     0.7473 1.250
## 19   0.2506     0.7216 1.422
## 16   0.2214     0.7373 1.354
## 187  0.2087     0.8000 1.321
## 185  0.1972     0.7561 1.264
## 168  0.1845     0.7073 1.393
## 177  0.1768     0.8081 1.334
## 137  0.1718     0.7714 1.417
## 139  0.1692     0.7348 1.349
## 176  0.1603     0.7326 1.225
## 161  0.1552     0.7093 1.397
## 482  0.1438     0.8014 1.323
## 110  0.1399     0.7383 1.356
## 115  0.1374     0.7248 1.269
## 121  0.1336     0.8268 1.518
## 99   0.1336     0.7955 1.461
## 480  0.1336     0.7447 1.245
## 450  0.1272     0.7092 1.397
## 389  0.1247     0.8376 1.538
```

类似的，如果我们只关注物品之间的关联购买行为，则得到如下结果：

```
item = Rules[!grepl(string, Rules$lhs) & grepl("(TRUE)|(FALSE)", Rules$lhs) &
    grepl("(TRUE)|(FALSE)", Rules$rhs) & !grepl(string, Rules$rhs), ]
item[1:20, 1:4]
```

```
##                                                                                 rules
## 356                  {Frozen_foods=FALSE,Bakery_goods=FALSE} => {Alcohol=FALSE}
## 357                  {Alcohol=FALSE,Bakery_goods=FALSE} => {Frozen_foods=FALSE}
## 332                        {Frozen_foods=FALSE,Snacks=FALSE} => {Alcohol=FALSE}
## 343                  {Alcohol=FALSE,Tinned_goods=FALSE} => {Frozen_foods=FALSE}
## 319               {Ready_made=FALSE,Frozen_foods=FALSE} => {Bakery_goods=FALSE}
## 309                     {Ready_made=TRUE,Alcohol=FALSE} => {Frozen_foods=FALSE}
## 308                     {Ready_made=TRUE,Frozen_foods=FALSE} => {Alcohol=FALSE}
## 283                   {Frozen_foods=FALSE,Tinned_goods=TRUE} => {Alcohol=FALSE}
## 275                     {Ready_made=FALSE,Tinned_goods=TRUE} => {Alcohol=FALSE}
## 294                         {Alcohol=FALSE,Snacks=TRUE} => {Frozen_foods=FALSE}
## 281                   {Bakery_goods=FALSE,Tinned_goods=TRUE} => {Alcohol=FALSE}
## 277                         {Snacks=FALSE,Tinned_goods=TRUE} => {Alcohol=FALSE}
## 914     {Frozen_foods=FALSE,Bakery_goods=FALSE,Snacks=FALSE} => {Alcohol=FALSE}
## 280               {Bakery_goods=FALSE,Tinned_goods=TRUE} => {Frozen_foods=FALSE}
## 915     {Alcohol=FALSE,Bakery_goods=FALSE,Snacks=FALSE} => {Frozen_foods=FALSE}
## 300                 {Ready_made=TRUE,Bakery_goods=FALSE} => {Tinned_goods=FALSE}
## 883 {Ready_made=FALSE,Alcohol=FALSE,Bakery_goods=FALSE} => {Frozen_foods=FALSE}
## 884 {Ready_made=FALSE,Frozen_foods=FALSE,Alcohol=FALSE} => {Bakery_goods=FALSE}
## 821  {Ready_made=TRUE,Alcohol=FALSE,Tinned_goods=FALSE} => {Frozen_foods=FALSE}
## 820  {Ready_made=TRUE,Frozen_foods=FALSE,Tinned_goods=FALSE} => {Alcohol=FALSE}
##     support confidence  lift
## 356  0.2990     0.7655 1.264
## 357  0.2990     0.7630 1.276
## 332  0.2494     0.7396 1.221
## 343  0.2379     0.7362 1.231
## 319  0.2265     0.7149 1.251
## 309  0.2125     0.7591 1.269
## 308  0.2125     0.7557 1.248
## 283  0.1959     0.7897 1.304
## 275  0.1870     0.7819 1.291
## 294  0.1845     0.7214 1.206
## 281  0.1781     0.7821 1.291
## 277  0.1743     0.7527 1.243
## 914  0.1692     0.7472 1.234
## 280  0.1692     0.7430 1.243
## 915  0.1692     0.7308 1.222
## 300  0.1667     0.7043 1.293
## 883  0.1616     0.7299 1.221
## 884  0.1616     0.7299 1.278
## 821  0.1527     0.8276 1.384
## 820  0.1527     0.8000 1.321
```

如果我们关注人群特征之间的关联性质，可以做以下限制，也可以得到有意思的结果：

```
pp = Rules[grepl(string, Rules$lhs) & !grepl("(TRUE)|(FALSE)", Rules$lhs) &
    !grepl("(TRUE)|(FALSE)", Rules$rhs) & grepl(string, Rules$rhs), ]
pp
```

```
##                                                                  rules
## 122                  {GENDER=Male  ,Age=18 to 30} => {CHILDREN=No }
## 104               {GENDER=Male  ,MARITAL=Single   } => {CHILDREN=No }
## 98                {GENDER=Male  ,MARITAL=Single   } => {Age=18 to 30}
## 393       {GENDER=Male  ,Age=18 to 30,WORKING=Yes} => {CHILDREN=No }
## 376 {GENDER=Male  ,Age=18 to 30,MARITAL=Single   } => {CHILDREN=No }
## 377 {Age=18 to 30,MARITAL=Single   ,CHILDREN=No } => {GENDER=Male  }
## 378 {GENDER=Male  ,MARITAL=Single   ,CHILDREN=No } => {Age=18 to 30}
##     support confidence  lift
## 122  0.1489     0.9213 1.412
## 104  0.1260     0.8761 1.342
## 98   0.1081     0.7522 2.505
## 393  0.1056     0.9121 1.397
## 376  0.1005     0.9294 1.424
## 377  0.1005     0.7980 1.728
## 378  0.1005     0.7980 2.658
##                                                lhs              rhs
## 122                  {GENDER=Male  ,Age=18 to 30}    {CHILDREN=No }
## 104               {GENDER=Male  ,MARITAL=Single   }    {CHILDREN=No }
## 98                {GENDER=Male  ,MARITAL=Single   }    {Age=18 to 30}
## 393       {GENDER=Male  ,Age=18 to 30,WORKING=Yes}    {CHILDREN=No }
## 376 {GENDER=Male  ,Age=18 to 30,MARITAL=Single   }    {CHILDREN=No }
## 377  {Age=18 to 30,MARITAL=Single   ,CHILDREN=No }   {GENDER=Male  }
## 378 {GENDER=Male  ,MARITAL=Single   ,CHILDREN=No }    {Age=18 to 30}
```

3. 不频繁项集挖掘

通过itemFreq可以找出不频繁项集，通过调小support的限制可以得到他们对应的规则。

```
### mining the unfrequency terms
rules <- apriori(shop_trans, parameter = list(support = 0.01, confidence = 0.6))
```

```
##
## parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.6    0.1    1 none FALSE            TRUE    0.01      1     10
##  target   ext
##   rules FALSE
##
## algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## apriori - find association rules with the apriori algorithm
## version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[32 item(s), 786 transaction(s)] done [0.00s].
## sorting and recoding items ... [32 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.04s].
## writing ... [165938 rule(s)] done [0.05s].
## creating S4 object  ... done [0.21s].
```

```
rule = subset(rules, subset = lift > 1.2)
summary(rule)
```

```
## set of 127326 rules
##
## rule length distribution (lhs + rhs):sizes
##     2     3     4     5     6     7     8     9    10
##    34  1047  8725 29444 43452 31218 11335  1950   121
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    5.00    6.00    6.08    7.00   10.00
##
## summary of quality measures:
##     support         confidence         lift
##  Min.   :0.0102   Min.   :0.600   Min.   :1.20
##  1st Qu.:0.0115   1st Qu.:0.714   1st Qu.:1.34
##  Median :0.0153   Median :0.800   Median :1.52
##  Mean   :0.0203   Mean   :0.806   Mean   :1.66
##  3rd Qu.:0.0216   3rd Qu.:0.900   3rd Qu.:1.80
##  Max.   :0.3957   Max.   :1.000   Max.   :8.02
##
## mining info:
##        data ntransactions support confidence
##  shop_trans           786    0.01        0.6
```

```
Rules = as(rule, "data.frame")
Rules$rules = as.character(Rules$rules)
rule_spl = do.call(rbind, sapply(Rules$rules, strsplit, "=>"))
rownames(rule_spl) = NULL
colnames(rule_spl) = c("lhs", "rhs")
Rules = data.frame(Rules, rule_spl)
Rules = Rules[order(Rules$support, Rules$confidence, decreasing = T), ]
infreq_item = itemFreq[itemFreq < quantile(itemFreq, 0.25)]
items = sapply(names(infreq_item), function(x) {
    res = grepl(x, Rules$rhs)
    return(res)
})
```

```
# min_freq=apply(items,1,function(x) { return(min(infreq_item[x])) })

# min_freq[!is.finite(min_freq)]=NA save(min_freq,file='min_freq.rda')
load("min_freq.rda")
Rules[which(!is.na(min_freq))[1:20], ]
```

```
##
rules
## 4393                                    {Ready_made=FALSE,Age=Over 60 ,CHILDREN=No }
=> {MARITAL=Divorced }
## 4329                                       {Alcohol=TRUE,GENDER=Male  ,Age=Over 60 }
=> {MARITAL=Divorced }
## 20503             {Ready_made=FALSE,Bakery_goods=FALSE,MARITAL=Divorced
,CHILDREN=No } => {Age=Over 60 }
## 4368                                   {Ready_made=FALSE,GENDER=Male  ,Age=Over 60 }
=> {MARITAL=Divorced }
## 20464                  {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60 ,WORKING=Yes}
=> {MARITAL=Divorced }
## 20502             {Ready_made=FALSE,Bakery_goods=FALSE,Age=Over 60 ,CHILDREN=No }
=> {MARITAL=Divorced }
## 4349                                  {Frozen_foods=TRUE,GENDER=Male  ,Age=Over 60 }
=> {MARITAL=Divorced }
## 20459                 {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60 ,CHILDREN=No }
=> {MARITAL=Divorced }
## 4341
{Alcohol=TRUE,Bakery_goods=FALSE,MARITAL=Divorced } => {Age=Over 60 }
## 20460                 {Bakery_goods=FALSE,GENDER=Male  ,MARITAL=Divorced
,CHILDREN=No } => {Age=Over 60 }
## 20510                     {Ready_made=FALSE,Age=Over 60 ,CHILDREN=No ,WORKING=Yes}
=> {MARITAL=Divorced }
## 20422                   {Ready_made=FALSE,GENDER=Male  ,Age=Over 60 ,CHILDREN=No }
=> {MARITAL=Divorced }
## 20247                       {Alcohol=TRUE,GENDER=Male  ,Age=Over 60 ,CHILDREN=No }
=> {MARITAL=Divorced }
## 4373                                  {Tinned_goods=FALSE,GENDER=Male  ,Age=Over 60 }
=> {MARITAL=Divorced }
## 62447    {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60 ,CHILDREN=No ,WORKING=Yes}
=> {MARITAL=Divorced }
## 4333                                    {Ready_made=FALSE,Alcohol=TRUE,Age=Over 60 }
=> {MARITAL=Divorced }
## 20423                  {Ready_made=FALSE,GENDER=Male  ,MARITAL=Divorced
,CHILDREN=No } => {Age=Over 60 }
## 62504 {Ready_made=FALSE,Bakery_goods=FALSE,MARITAL=Divorced ,CHILDREN=No
,WORKING=Yes} => {Age=Over 60 }
## 20417            {Ready_made=FALSE,Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60 }
=> {MARITAL=Divorced }
## 62448    {Bakery_goods=FALSE,GENDER=Male  ,MARITAL=Divorced ,CHILDREN=No
,WORKING=Yes} => {Age=Over 60 }
##       support confidence  lift
## 4393  0.02417     0.6129 4.916
## 4329  0.02163     0.8095 6.493
## 20503 0.02163     0.7083 6.118
## 4368  0.02163     0.6296 5.050
## 20464 0.02163     0.6296 5.050
## 20502 0.02163     0.6296 5.050
## 4349  0.02036     0.6957 5.579
## 20459 0.02036     0.6957 5.579
## 4341  0.02036     0.6667 5.758
## 20460 0.02036     0.6400 5.528
## 20510 0.02036     0.6154 4.936
## 20422 0.01908     0.9375 7.519
## 20247 0.01908     0.7895 6.332
## 4373  0.01908     0.7143 5.729
## 62447 0.01908     0.7143 5.729
## 4333  0.01908     0.6818 5.468
## 20423 0.01908     0.6818 5.889
## 62504 0.01908     0.6818 5.889
## 20417 0.01908     0.6250 5.013
## 62448 0.01908     0.6250 5.398
##
lhs
## 4393                                          {Ready_made=FALSE,Age=Over 60
,CHILDREN=No }
## 4329                                             {Alcohol=TRUE,GENDER=Male  ,Age=Over
60 }
## 20503             {Ready_made=FALSE,Bakery_goods=FALSE,MARITAL=Divorced
,CHILDREN=No }
```

```
## 4368                                        {Ready_made=FALSE,GENDER=Male  ,Age=Over
60 }
## 20464                       {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60
,WORKING=Yes}
## 20502                    {Ready_made=FALSE,Bakery_goods=FALSE,Age=Over 60
,CHILDREN=No }
## 4349                                       {Frozen_foods=TRUE,GENDER=Male  ,Age=Over
60 }
## 20459                       {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60
,CHILDREN=No }
## 4341
{Alcohol=TRUE,Bakery_goods=FALSE,MARITAL=Divorced }
## 20460                 {Bakery_goods=FALSE,GENDER=Male  ,MARITAL=Divorced
,CHILDREN=No }
## 20510                           {Ready_made=FALSE,Age=Over 60 ,CHILDREN=No
,WORKING=Yes}
## 20422                         {Ready_made=FALSE,GENDER=Male  ,Age=Over 60
,CHILDREN=No }
## 20247                             {Alcohol=TRUE,GENDER=Male  ,Age=Over 60
,CHILDREN=No }
## 4373                                      {Tinned_goods=FALSE,GENDER=Male  ,Age=Over
60 }
## 62447          {Bakery_goods=FALSE,GENDER=Male  ,Age=Over 60 ,CHILDREN=No
,WORKING=Yes}
## 4333                                        {Ready_made=FALSE,Alcohol=TRUE,Age=Over
60 }
## 20423                    {Ready_made=FALSE,GENDER=Male  ,MARITAL=Divorced
,CHILDREN=No }
## 62504 {Ready_made=FALSE,Bakery_goods=FALSE,MARITAL=Divorced ,CHILDREN=No
,WORKING=Yes}
## 20417                   {Ready_made=FALSE,Bakery_goods=FALSE,GENDER=Male  ,Age=Over
60 }
## 62448    {Bakery_goods=FALSE,GENDER=Male  ,MARITAL=Divorced ,CHILDREN=No
,WORKING=Yes}
##                          rhs
## 4393   {MARITAL=Divorced }
## 4329   {MARITAL=Divorced }
## 20503       {Age=Over 60 }
## 4368   {MARITAL=Divorced }
## 20464  {MARITAL=Divorced }
## 20502  {MARITAL=Divorced }
## 4349   {MARITAL=Divorced }
## 20459  {MARITAL=Divorced }
## 4341        {Age=Over 60 }
## 20460       {Age=Over 60 }
## 20510  {MARITAL=Divorced }
## 20422  {MARITAL=Divorced }
## 20247  {MARITAL=Divorced }
## 4373   {MARITAL=Divorced }
## 62447  {MARITAL=Divorced }
## 4333   {MARITAL=Divorced }
## 20423       {Age=Over 60 }
## 62504       {Age=Over 60 }
## 20417  {MARITAL=Divorced }
## 62448       {Age=Over 60 }
```

如果限制规则的右边必须为物品的集合，则得到如下规则。

```
#
#
infreq_item=itemFreq[itemFreq<quantile(itemFreq,0.25)&grepl('(TRUE|FALSE)',names(itemFreq),

# items=sapply(names(infreq_item),function(x) { res=grepl(x,Rules$rhs)
# return(res) }) min_freq_item=apply(items,1,function(x) {
# return(min(infreq_item[x])) })
# min_freq_item[!is.finite(min_freq_item)]=NA
save(min_freq_item, file = "min_freq_item.rda")
```

```
## Error: object 'min_freq_item' not found
```

```
load("min_freq_item.rda")

Rules[which(!is.na(min_freq_item)), ]
```

```
##
rules
## 58185
{Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=No } =>
{Fresh_Vegetables=TRUE}
## 114124
{Frozen_foods=TRUE,Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=N
 } => {Fresh_Vegetables=TRUE}
## 114145
{Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=No
,WORKING=Yes} => {Fresh_Vegetables=TRUE}
## 151103
{Frozen_foods=TRUE,Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=N
 ,WORKING=Yes} => {Fresh_Vegetables=TRUE}
##        support confidence  lift
## 58185  0.01018     0.6154 7.441
## 114124 0.01018     0.6154 7.441
## 114145 0.01018     0.6154 7.441
## 151103 0.01018     0.6154 7.441
##
lhs
## 58185
{Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=No }
## 114124
{Frozen_foods=TRUE,Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=N
 }
## 114145
{Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=No
,WORKING=Yes}
## 151103
{Frozen_foods=TRUE,Milk=TRUE,Bakery_goods=TRUE,Toiletries=TRUE,Tinned_goods=TRUE,CHILDREN=N
 ,WORKING=Yes}
##                              rhs
## 58185   {Fresh_Vegetables=TRUE}
## 114124  {Fresh_Vegetables=TRUE}
## 114145  {Fresh_Vegetables=TRUE}
## 151103  {Fresh_Vegetables=TRUE}
```