MIS 545 Apriori Algorithm and Assoication Rules

Kirsten Fure – 8/15/2019

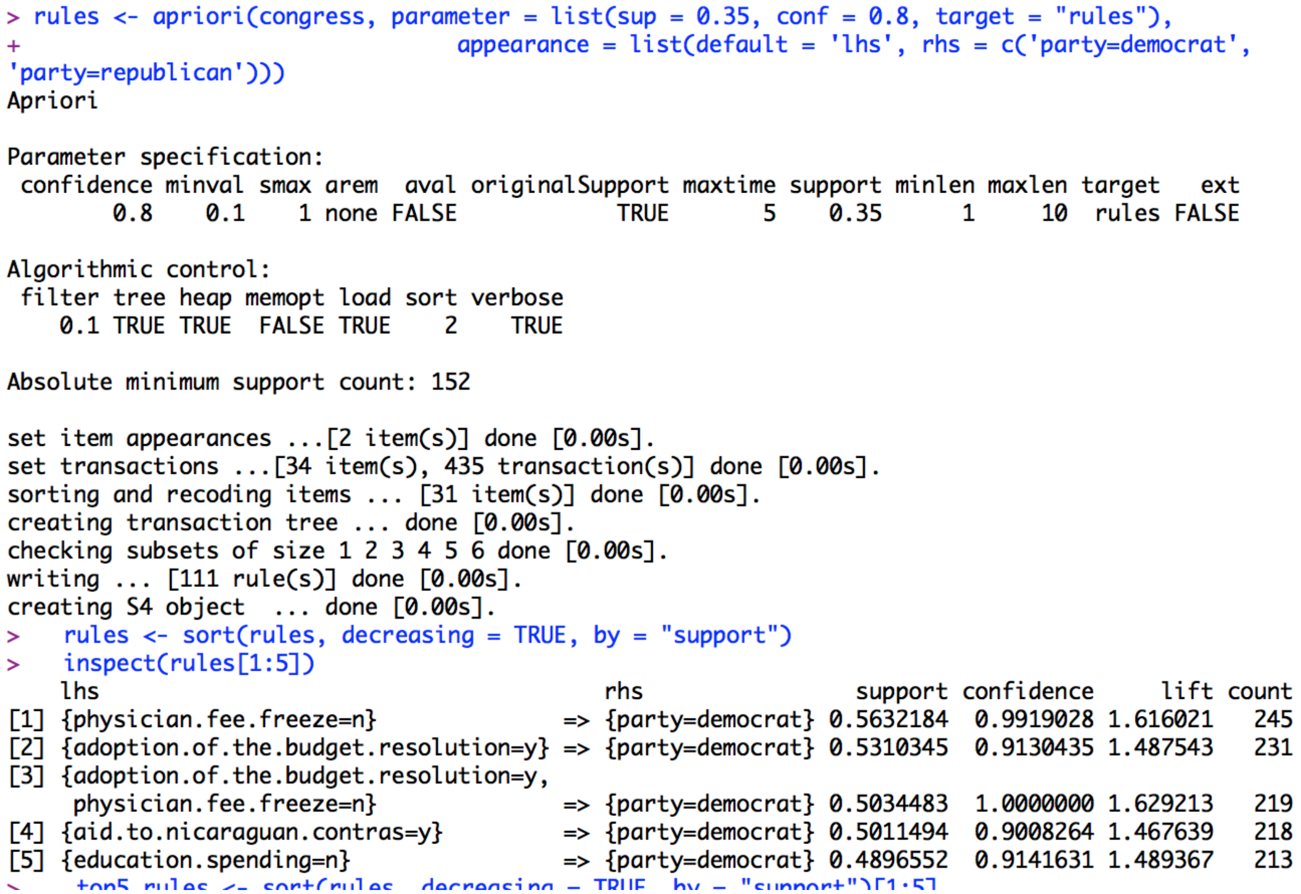
1. R Code

```
setwd(" ")

if(!require(arules)){
	install.packages("arules")
}
library(arules)

if(!require(arulesViz)){
	install.packages("arulesViz")
}
library(arulesViz)

if(!require(igraph)){
	install.packages("igraph")
}
library(igraph)

if(!require(visNetwork)){
	install.packages("visNetwork")
}
library(visNetwork)


if(!require(plyr)){
	install.packages("plyr")
}
library(plyr)


congress <- read.csv("Congressional Voting Records.csv", na.string = '?')

#   check data type
str(congress)
nrow(congress)

# generate association rules
rules <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target =
    "rules"),  appearance = list(default = 'lhs', rhs = c('party=democrat',
    'party=republican')))
```

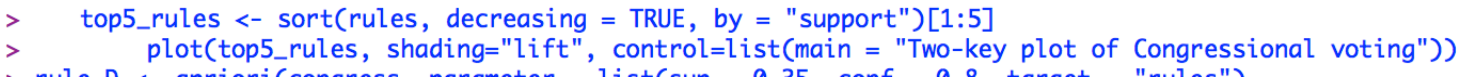
```
rules <- sort(rules, decreasing = TRUE, by = "support")

inspect(rules[1:5])

top5_rules <- sort(rules, decreasing = TRUE, by = "support")[1:5]

#   overview of rules
plot(top5_rules, shading="lift", control=list(main = "Two-key plot of
    Congressional voting"))

#   Targeting Party
rule_D <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target
         = "rules"), appearance = list(default = 'lhs', rhs =
         c('party=democrat')))

rule_D <- sort(rule_D, decreasing = TRUE, by = "confidence")
inspect(rule_D[1:2])

rule_R <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target =
    "rules"), appearance = list(default = 'lhs', rhs = c('party=republican')))

rule_R <- sort(rule_R, decreasing = TRUE, by = "confidence")
inspect(rule_R[1:2])

#   parallel coordinates plot
plot(top5_rules, method = "paracoord", shading = "support")

#   create a basic graph structure
ig <- plot(top5_rules, method = "graph")

#   use igraph
ig_df <- get.data.frame(ig, what = "both")

#   generate nodes
nodes <- data.frame(id = ig_df$vertices$name,
      #   the size of nodes: could change to lift or confidence
      value = ig_df$vertices$support,
      title = ifelse(ig_df$vertices$label == "", ig_df$vertices$name,
      ig_df$vertices$label), ig_df$vertices)

#   generate edges
edges <- ig_df$edges

#   directed network    manipulate network
network <- visNetwork(nodes, edges) %>%
         visOptions(manipulation = TRUE) %>%   #    manipulate network
```

```
        visEdges(arrows = 'to', scaling = list(min = 2, max = 2)) %>%
                                                    #   directed network
        visInteraction(navigationButtons = TRUE)   # navigation buttons
network
```

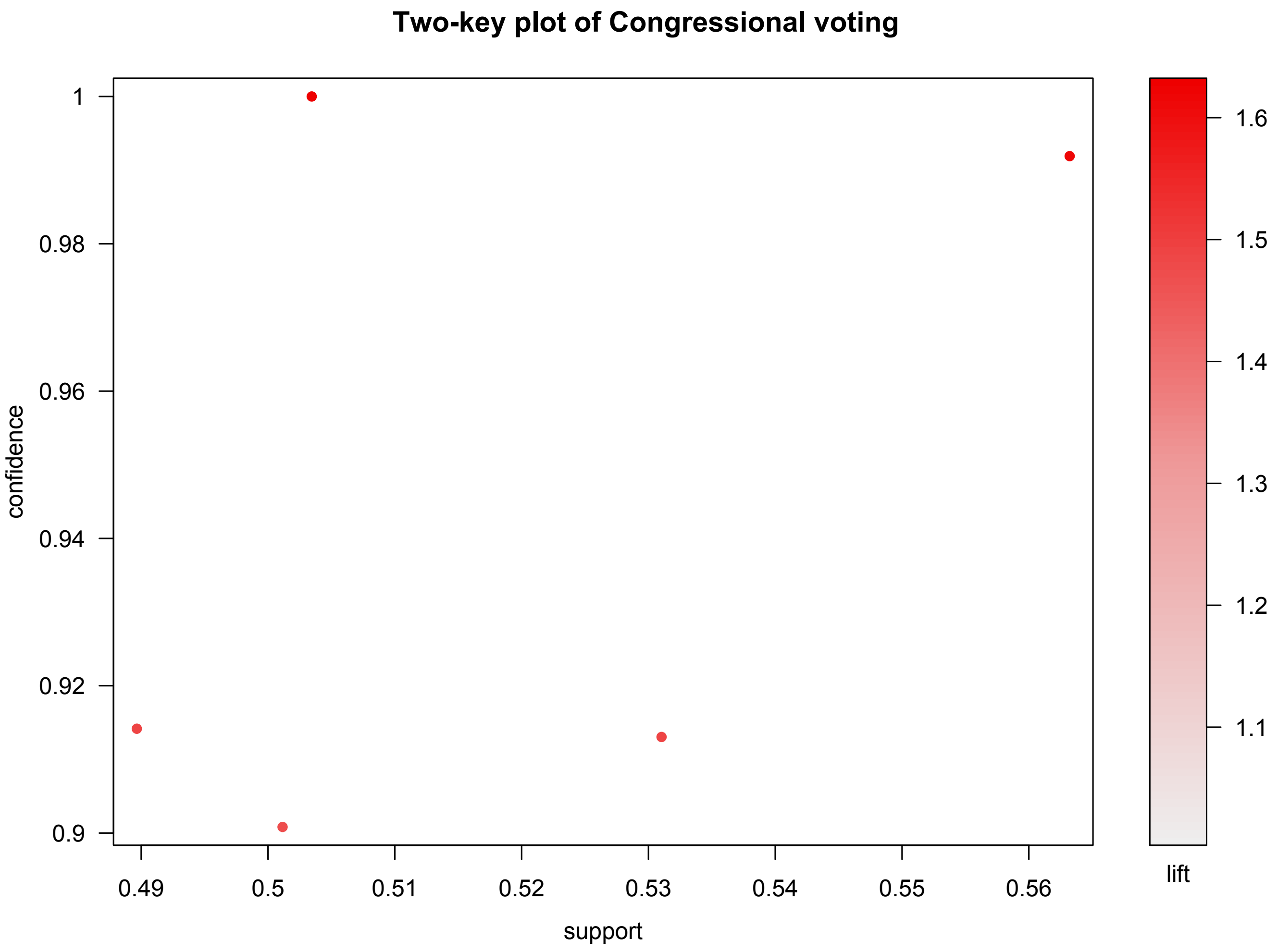
2.



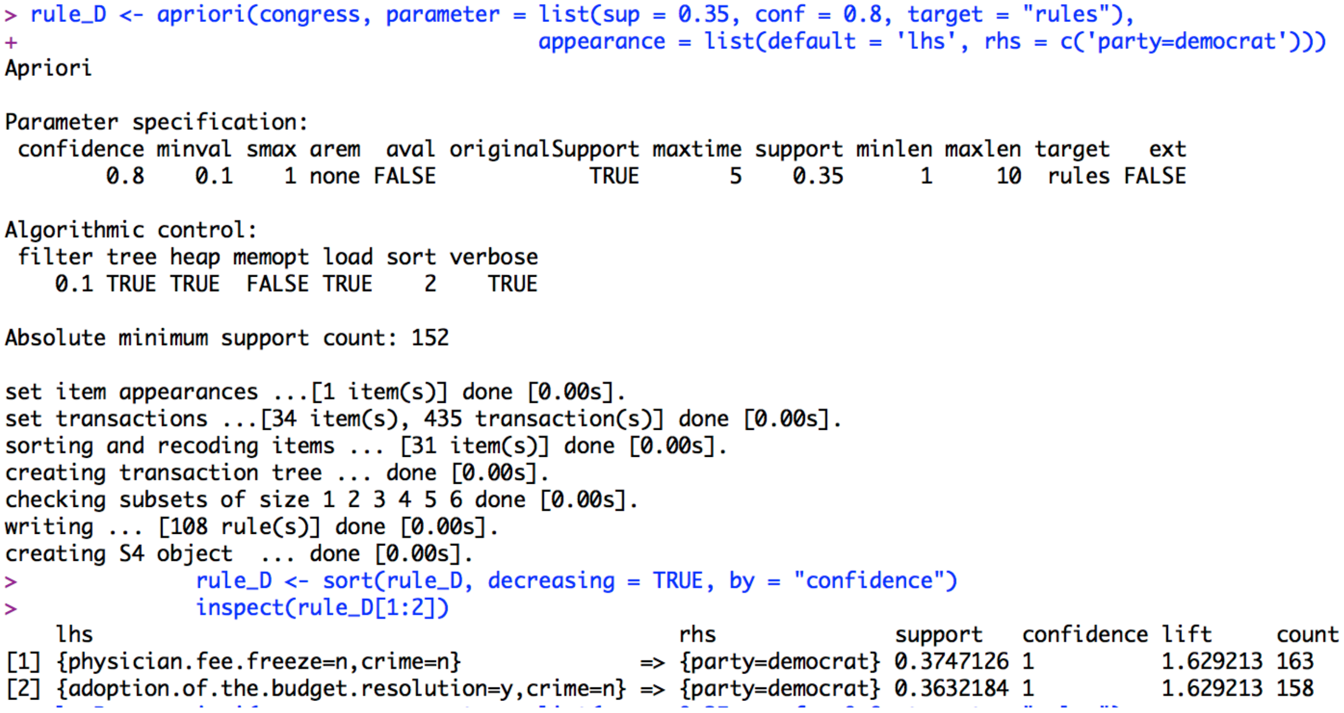
```
> rules <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target = "rules"),
+                                 appearance = list(default = 'lhs', rhs = c('party=democrat',
'party=republican')))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5    0.35      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 152

set item appearances ...[2 item(s)] done [0.00s].
set transactions ...[34 item(s), 435 transaction(s)] done [0.00s].
sorting and recoding items ... [31 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 ... [111 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>   rules <- sort(rules, decreasing = TRUE, by = "support")
>   inspect(rules[1:5])
    lhs                                     rhs                support   confidence     lift count
[1] {physician.fee.freeze=n}             => {party=democrat} 0.5632184  0.9919028 1.616021   245
[2] {adoption.of.the.budget.resolution=y} => {party=democrat} 0.5310345  0.9130435 1.487543   231
[3] {adoption.of.the.budget.resolution=y,
     physician.fee.freeze=n}             => {party=democrat} 0.5034483  1.0000000 1.629213   219
[4] {aid.to.nicaraguan.contras=y}        => {party=democrat} 0.5011494  0.9008264 1.467639   218
[5] {education.spending=n}               => {party=democrat} 0.4896552  0.9141631 1.489367   213
>    top5.rules <- sort(rules, decreasing = TRUE, by = "support")[1:5]
```



```
>    top5_rules <- sort(rules, decreasing = TRUE, by = "support")[1:5]
>         plot(top5_rules, shading="lift", control=list(main = "Two-key plot of Congressional voting"))
> rule.D <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target = "rules"))
```

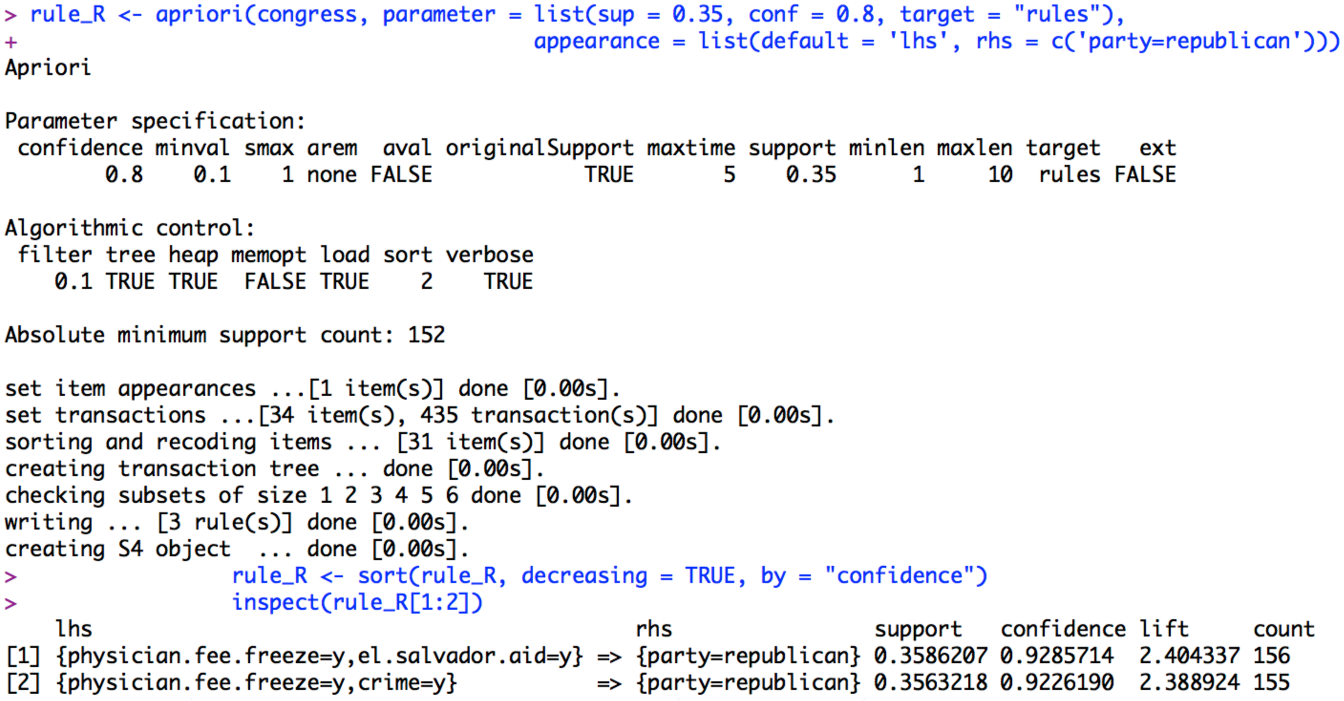


3. Democrat:



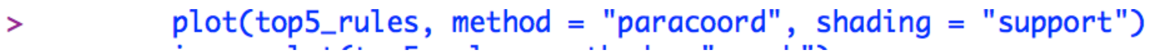
```
> rule_D <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target = "rules"),
+                                      appearance = list(default = 'lhs', rhs = c('party=democrat')))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5    0.35      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 152

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[34 item(s), 435 transaction(s)] done [0.00s].
sorting and recoding items ... [31 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 ... [108 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>             rule_D <- sort(rule_D, decreasing = TRUE, by = "confidence")
>             inspect(rule_D[1:2])
    lhs                                             rhs              support   confidence lift     count
[1] {physician.fee.freeze=n,crime=n}             => {party=democrat} 0.3747126 1          1.629213 163
[2] {adoption.of.the.budget.resolution=y,crime=n} => {party=democrat} 0.3632184 1          1.629213 158
```

Republican:

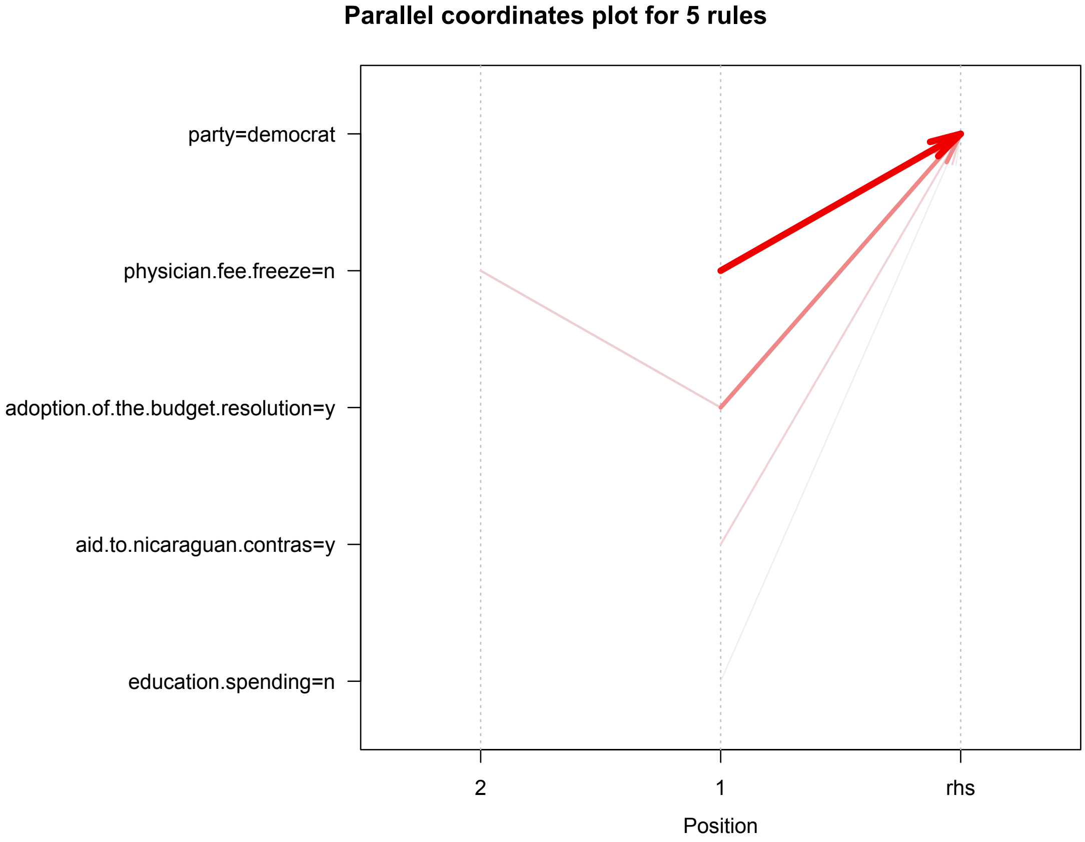


```
> rule_R <- apriori(congress, parameter = list(sup = 0.35, conf = 0.8, target = "rules"),
+                                      appearance = list(default = 'lhs', rhs = c('party=republican')))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5    0.35      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 152

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[34 item(s), 435 transaction(s)] done [0.00s].
sorting and recoding items ... [31 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 ... [3 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>              rule_R <- sort(rule_R, decreasing = TRUE, by = "confidence")
>              inspect(rule_R[1:2])
    lhs                                          rhs                support   confidence lift     count
[1] {physician.fee.freeze=y,el.salvador.aid=y} => {party=republican} 0.3586207 0.9285714  2.404337 156
[2] {physician.fee.freeze=y,crime=y}           => {party=republican} 0.3563218 0.9226190  2.388924 155
```

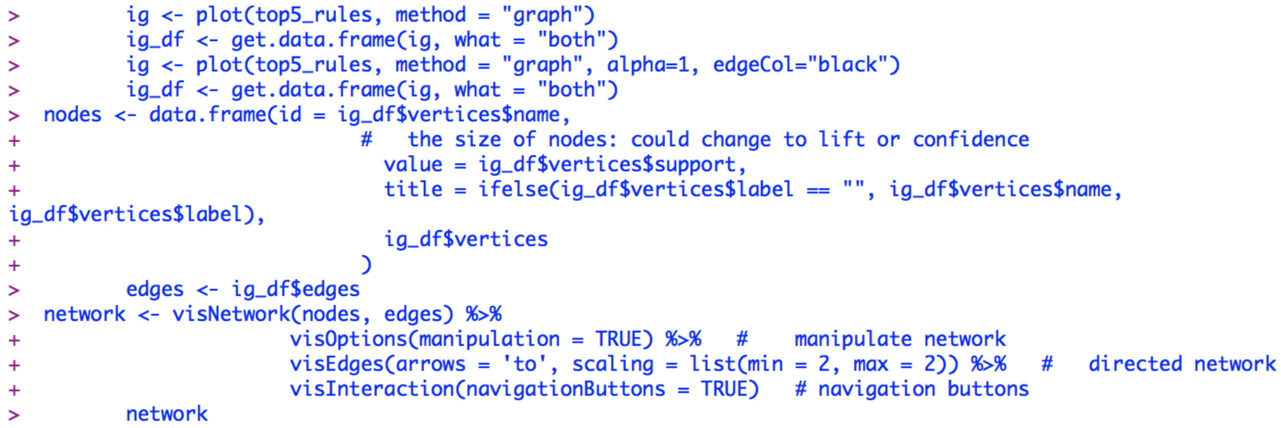
4.



```
>          plot(top5_rules, method = "paracoord", shading = "support")
```

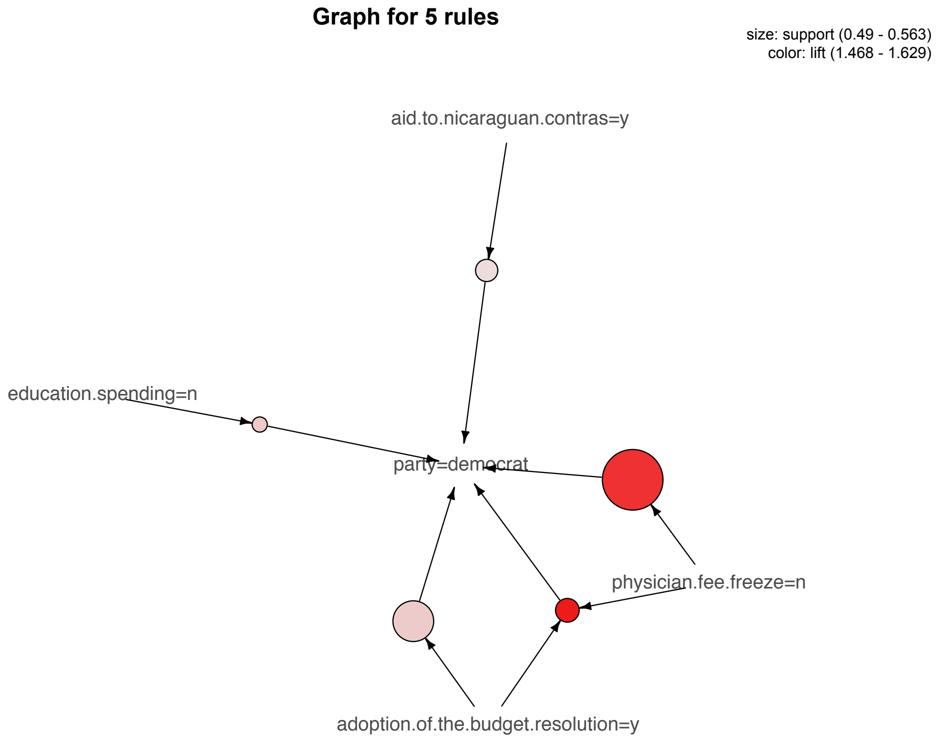


5.

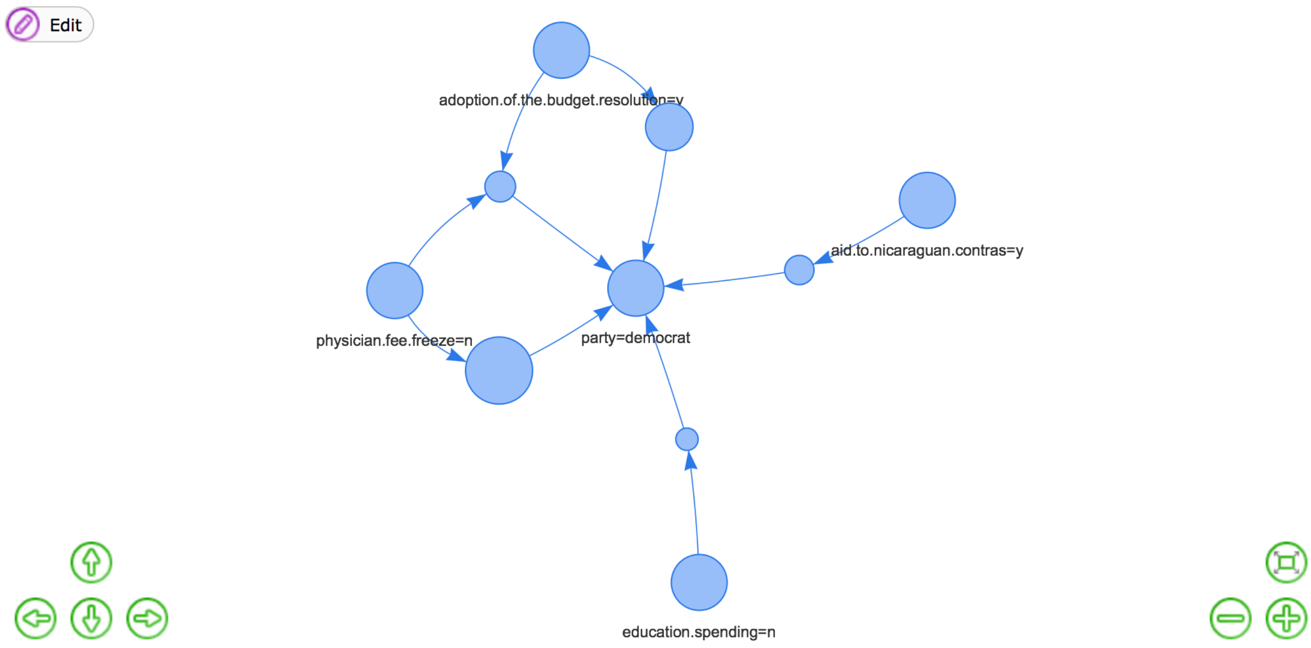


```
>         ig <- plot(top5_rules, method = "graph")
>         ig_df <- get.data.frame(ig, what = "both")
>         ig <- plot(top5_rules, method = "graph", alpha=1, edgeCol="black")
>         ig_df <- get.data.frame(ig, what = "both")
>  nodes <- data.frame(id = ig_df$vertices$name,
+                           #   the size of nodes: could change to lift or confidence
+                             value = ig_df$vertices$support,
+                             title = ifelse(ig_df$vertices$label == "", ig_df$vertices$name,
ig_df$vertices$label),
+                             ig_df$vertices
+                           )
>         edges <- ig_df$edges
>  network <- visNetwork(nodes, edges) %>%
+                     visOptions(manipulation = TRUE) %>%   #    manipulate network
+                     visEdges(arrows = 'to', scaling = list(min = 2, max = 2)) %>%   #   directed network
+                     visInteraction(navigationButtons = TRUE)   # navigation buttons
>         network
```

Basic igraph


Graph for 5 rules
size: support (0.49 - 0.563)
color: lift (1.468 - 1.629)
aid.to.nicaraguan.contras=y
education.spending=n
party=democrat
physician.fee.freeze=n
adoption.of.the.budget.resolution=y

Interactive visNetwork Graph


Edit
adoption.of.the.budget.resolution=y
aid.to.nicaraguan.contras=y
physician.fee.freeze=n
party=democrat
education.spending=n

6. Association rules can be evaluated with 3 different measurements: support, confidence and lift. Usually, lift gives the best results, then confidence, followed by support. Lift is a ratio showing a rule's performance, and it takes into account the frequency of both the antecedent and the consequent. If the lift is higher than one, it indicates a good rule. The confidence value indicates the probability of how often the rule is true but relies on the frequency of the antecedent (and does not account for the frequency of the consequent alone). The measure of support indicates the frequency of the items relevant to the rule within the entire dataset. The higher these values, the better quality the rule will be at having good predictions/associations.

7. High confidence alone can sometimes be misleading. Take the example where you are evaluating whether the purchase of a toothbrush indicates the purchase of milk. The confidence value will be high because a milk purchase is so frequent. In this case, it wouldn't matter too much what items you are evaluating as the antecedent because the consequent (milk purchase) is so frequent. The chance that someone happens to be buying milk at the same time is high, because it is purchased very often. This is when the lift ratio can be helpful, because it takes the frequency of the antecedent alone and the consequent alone into account giving a better total picture.