Validação - Somente COVID-19

José Elvano Moraes

4/15/2021

Variáveis selecionadas

```
## Rows: 76,666
## Columns: 9
## $ EVOLUCAO   <fct> 2, 1, 1, 1, 2, 1, 2, 2, 1, 2, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1,~
## $ RENAL      <fct> 2, 2, 2, 1, 2, 1, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,~
## $ DIABETES   <fct> 2, 1, 2, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2,~
## $ OBESIDADE  <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 9, 2,~
## $ PNEUMOPATI <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,~
## $ UTI        <fct> 1, 1, 2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 1, 2, 2, 2,~
## $ CARDIOPATI <fct> 2, 2, 2, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1,~
## $ SUPORT_VEN <fct> 2, 2, 2, 9, 3, 3, 2, 2, 3, 2, 2, 3, 1, 1, 3, 3, 2, 3, 9, 3,~
## $ ANTIVIRAL  <fct> 2, 2, 2, 9, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,~
```

Imposição de estrutura com arcos que fazem sentido clínico (***White list***)

```r
s1 <- sample_frac(ddf, .9, FALSE)
cat(nrow(ddf) - nrow(s1))
```

```
## 7667
```

```r
wl = matrix(c("OBESIDADE", "DIABETES",
              #"IDADE", "DIABETES",
              #"IDADE", "SUPORT_VEN",
              #"IDADE", "NEUROLOGIC",
              "DIABETES", "RENAL",
              "DIABETES", "CARDIOPATI",
              "UTI", "EVOLUCAO"),
        ncol = 2, byrow = TRUE, dimnames = list(NULL, c("from", "to")))

bn1 <- mmhc(s1, whitelist = wl)

#sem WL
bn2 <- mmhc(s1, whitelist = NULL)
```

Descrição DAG sem estrutura imposta

```r
bn2
```

```
##
```

```
##   Bayesian network learned via Hybrid methods
##
##   model:
##    [UTI][SUPORT_VEN|UTI][EVOLUCAO|UTI:SUPORT_VEN][OBESIDADE|SUPORT_VEN]
##    [ANTIVIRAL|UTI:SUPORT_VEN][RENAL|EVOLUCAO:OBESIDADE]
##    [PNEUMOPATI|RENAL:OBESIDADE][DIABETES|RENAL:PNEUMOPATI]
##    [CARDIOPATI|DIABETES:OBESIDADE:PNEUMOPATI]
##   nodes:                                 9
##   arcs:                                  15
##     undirected arcs:                     0
##     directed arcs:                       15
##   average markov blanket size:           3.78
##   average neighbourhood size:            3.33
##   average branching factor:              1.67
##
##   learning algorithm:                    Max-Min Hill-Climbing
##   constraint-based method:               Max-Min Parent Children
##   conditional independence test:         Mutual Information (disc.)
##   score-based method:                    Hill-Climbing
##   score:                                 BIC (disc.)
##   alpha threshold:                       0.05
##   penalization coefficient:              5.570924
##   tests used in the learning procedure:  1374
##   optimized:                             TRUE
```

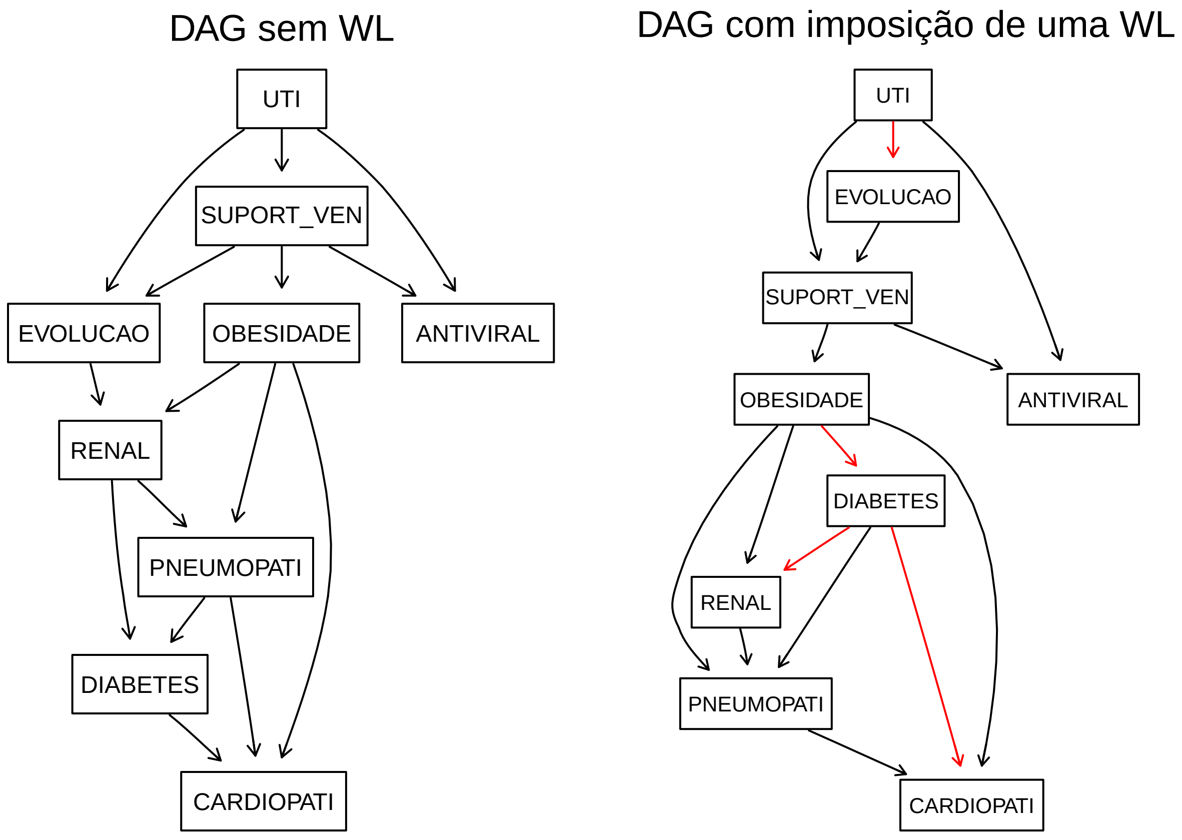
Descrição DAG com estrutura imposta

```
bn1
```

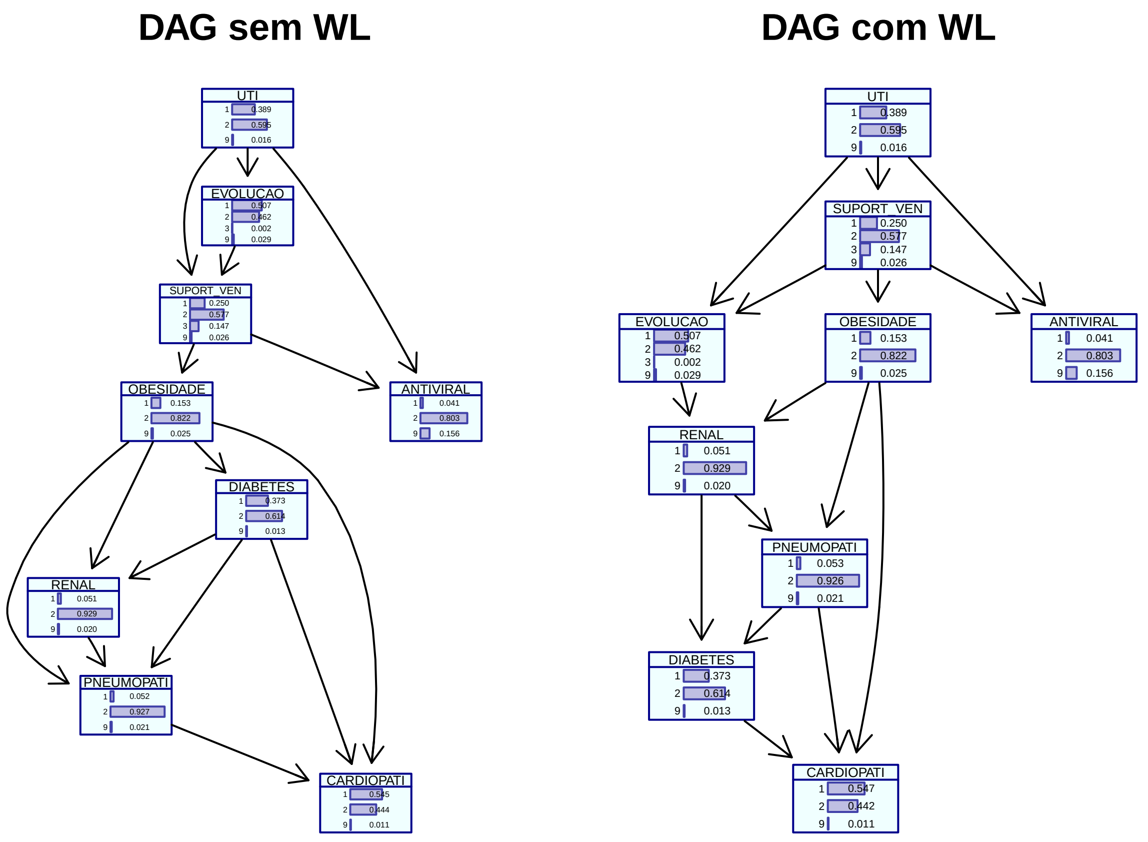
```
##
##   Bayesian network learned via Hybrid methods
##
##   model:
##    [UTI][EVOLUCAO|UTI][SUPORT_VEN|EVOLUCAO:UTI][OBESIDADE|SUPORT_VEN]
##    [ANTIVIRAL|UTI:SUPORT_VEN][DIABETES|OBESIDADE][RENAL|DIABETES:OBESIDADE]
##    [PNEUMOPATI|RENAL:DIABETES:OBESIDADE]
##    [CARDIOPATI|DIABETES:OBESIDADE:PNEUMOPATI]
##   nodes:                                 9
##   arcs:                                  15
##     undirected arcs:                     0
##     directed arcs:                       15
##   average markov blanket size:           3.33
##   average neighbourhood size:            3.33
##   average branching factor:              1.67
##
##   learning algorithm:                    Max-Min Hill-Climbing
##   constraint-based method:               Max-Min Parent Children
##   conditional independence test:         Mutual Information (disc.)
##   score-based method:                    Hill-Climbing
##   score:                                 BIC (disc.)
##   alpha threshold:                       0.05
##   penalization coefficient:              5.570924
##   tests used in the learning procedure:  1136
##   optimized:                             TRUE
```

DAG (*Directed Acyclic Graph*) usando ou não *whitelisting* no algoritmo de aprendizado da estrutura da rede causal

```
par(mfrow = c(1, 2))
graphviz.plot(bn2,
              shape='rectangle',
              highlight = NULL,
              main = 'DAG sem WL')
graphviz.plot(bn1,
              shape='rectangle',
              highlight = list(arcs = wl),
              main = 'DAG com imposição de uma WL')
```

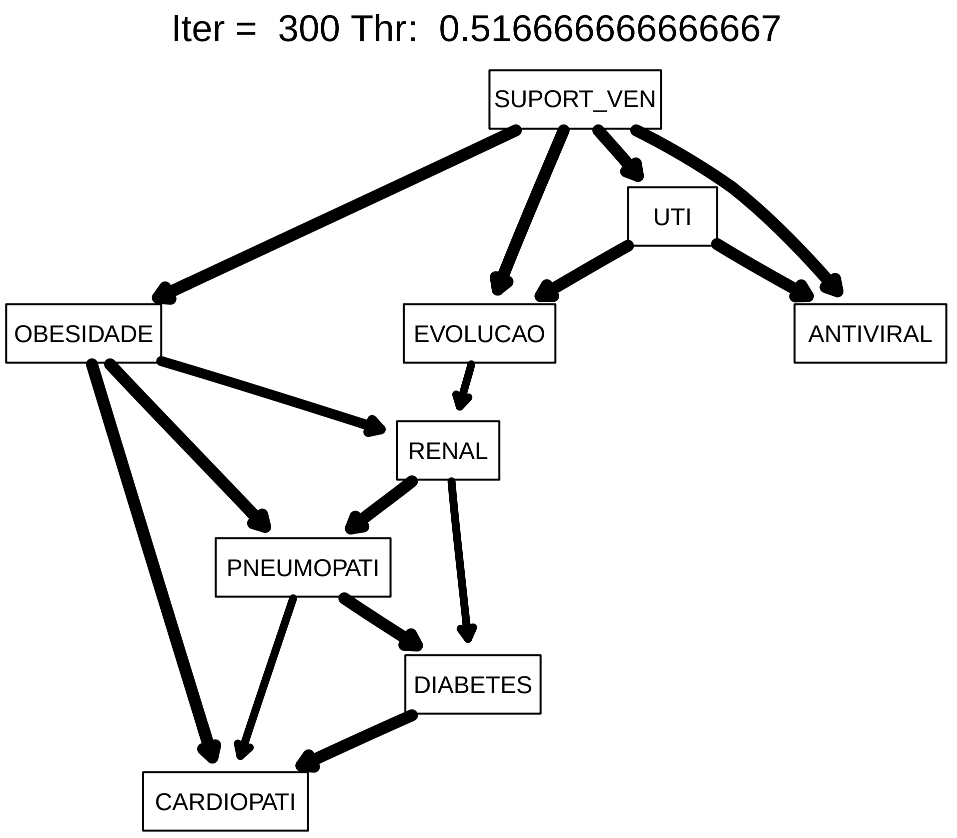


```
fitted.1 <- bn.fit(bn1, s1)
fitted.2 <- bn.fit(bn2, s1)

par(mfrow=c(1,2))
graphviz.chart(fitted.1,
               type = "barprob",
               col = "darkblue",
               bg = "azure",
               bar.col = "darkblue",
               main = "DAG sem WL")

graphviz.chart(fitted.2,
               type = "barprob",
               col = "darkblue",
               bg = "azure",
               bar.col = "darkblue",
               main = "DAG com WL")
```



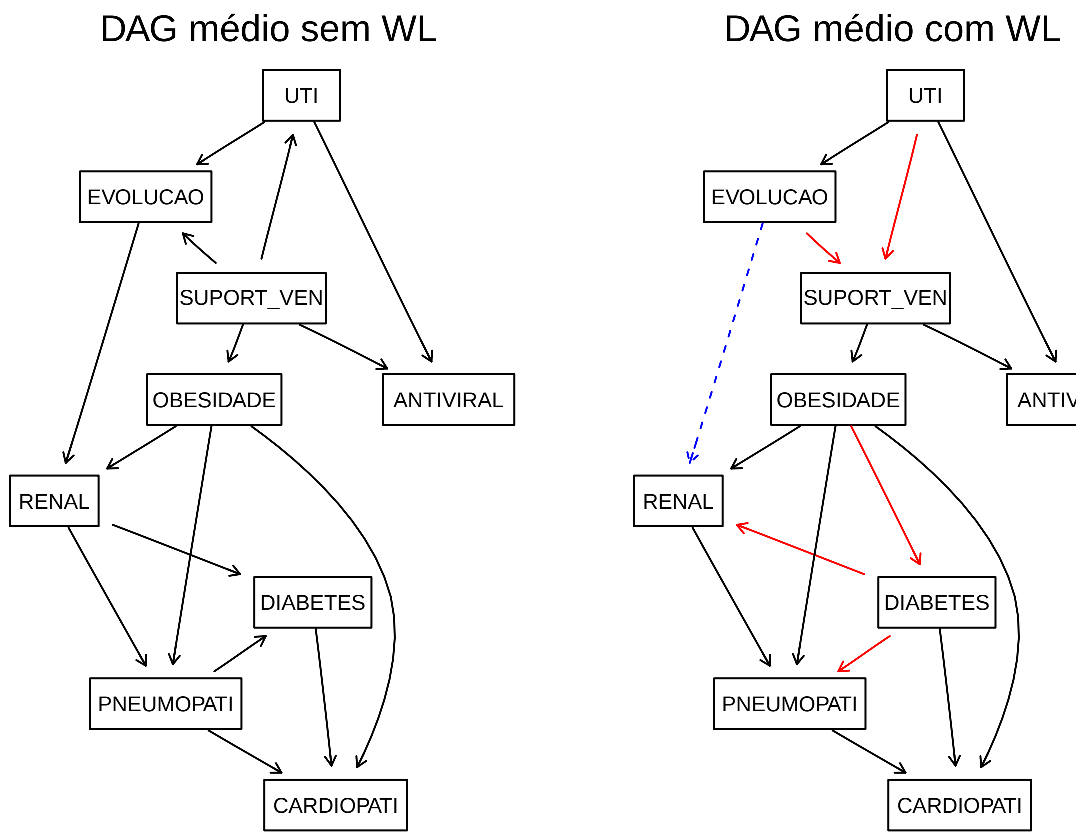
Fase de Bootstrap

```
boots.trap  <- 300

str.diff = boot.strength(s1,
                         R = boots.trap,
                         algorithm = "mmhc")

avg.diff = averaged.network(str.diff)
thr <- paste('Thr: ', attr(str.diff, "threshold"))

strength.plot(avg.diff,
              str.diff,
              shape = "rectangle",
              main = paste("Iter = ", boots.trap, thr))
```

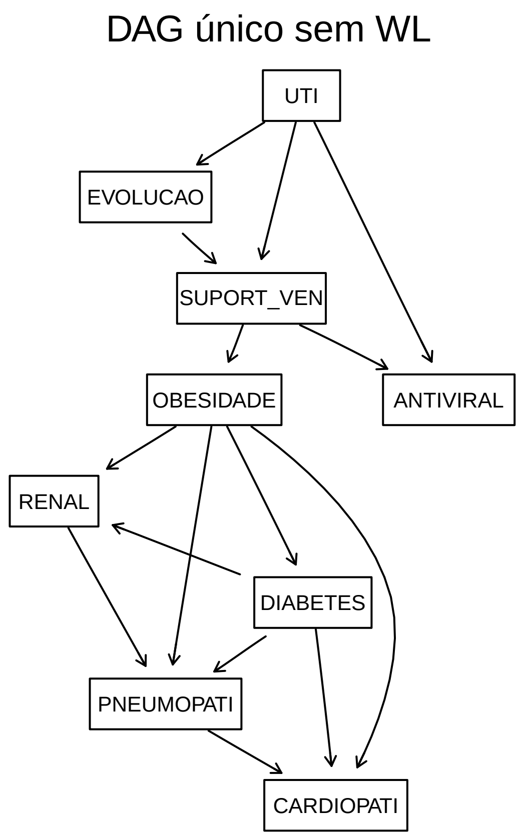


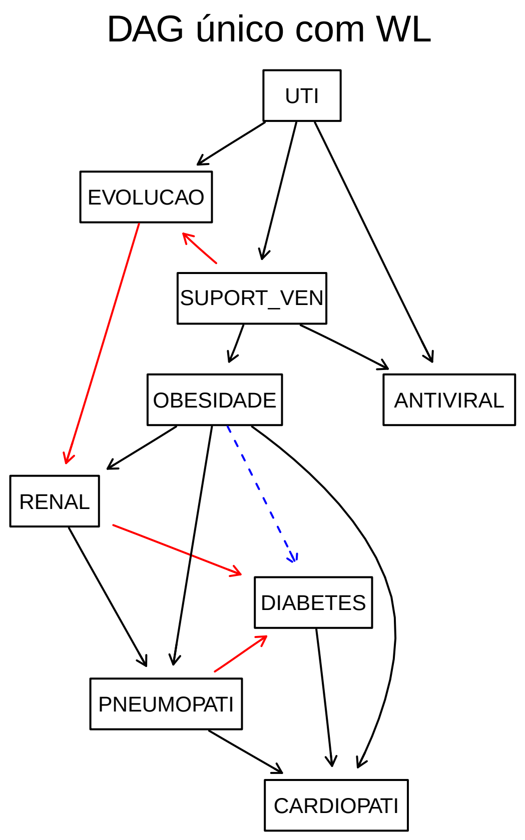
How can we compare the averaged network (avg.diff) with the network we originally learned in from all the data? The most qualitative way is to plot the two networks side by side, with the nodes in the same positions, and highlight the arcs that appear in one network and not in the other, or that appear with different directions.

```
par(mfrow = c(1, 2))
graphviz.compare(avg.diff,
                 bn1,
                 shape = "rectangle",
                 main = c("DAG médio sem WL", "DAG médio com WL"))
```



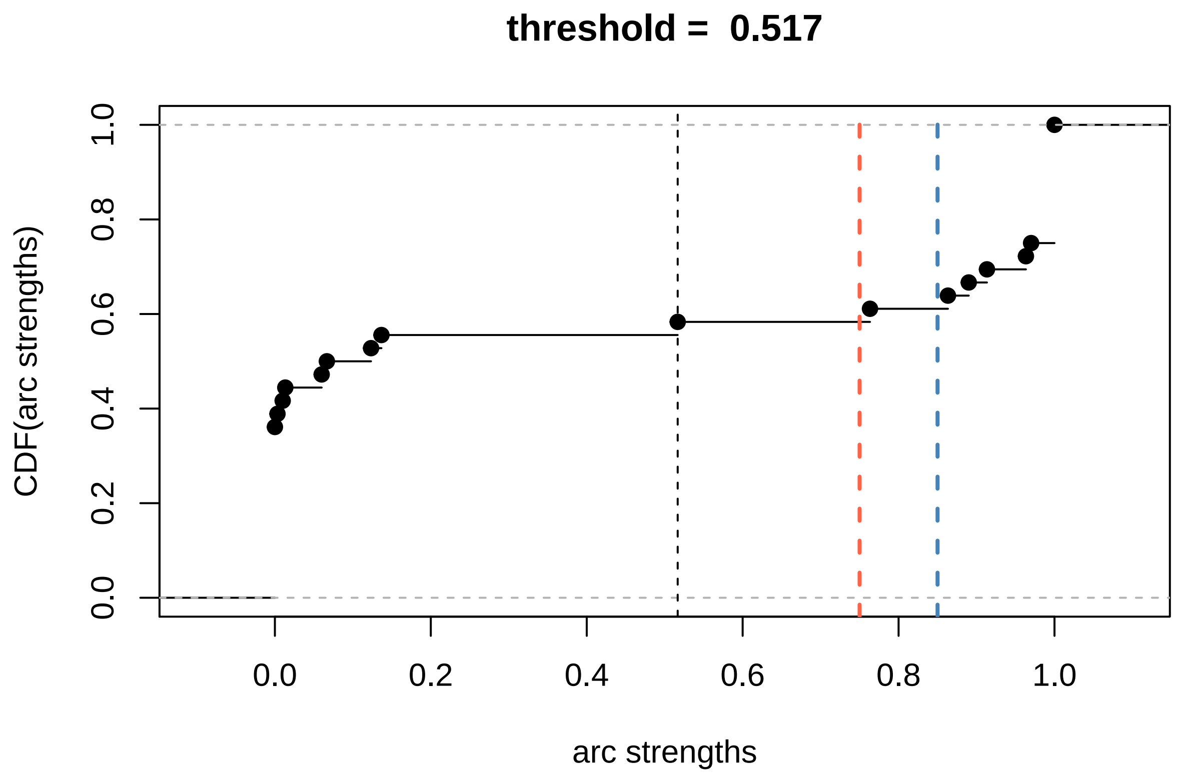
```r
par(mfrow = c(1, 2))
graphviz.compare(bn1,
                 bn2,
                 shape = "rectangle",
                 main = c("DAG único sem WL", "DAG único com WL"))
```





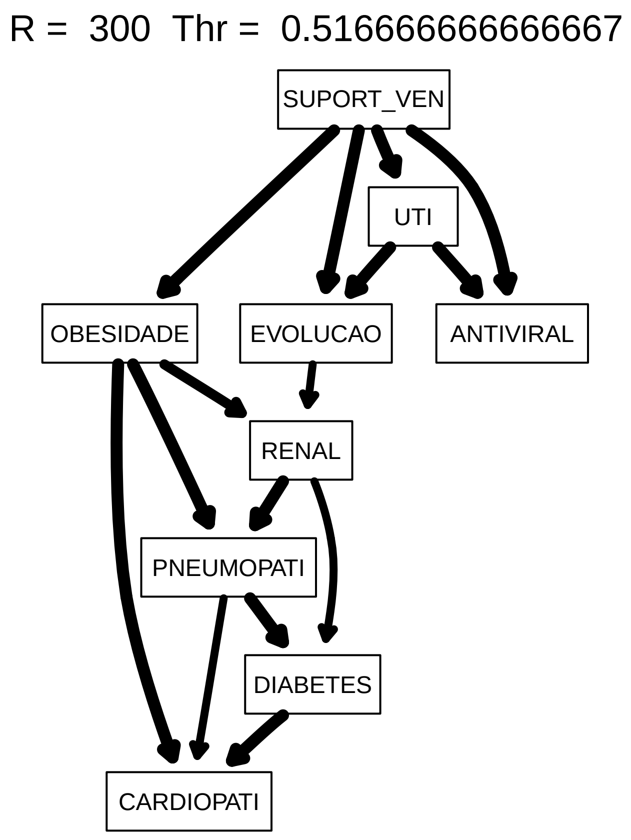
It is also a good idea to look at the threshold with respect to the distribution of the arc strengths

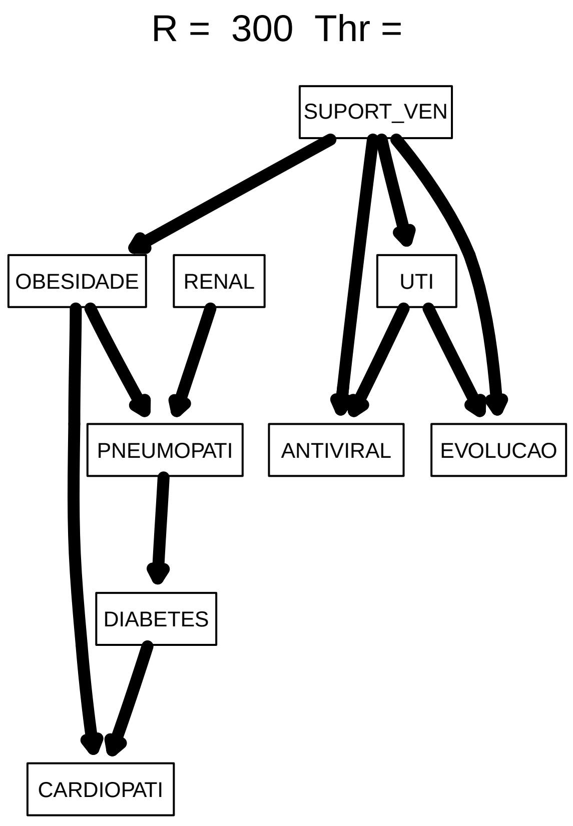
```r
plot(str.diff)
abline(v = 0.75, col = "tomato", lty = 2, lwd = 2)
abline(v = 0.85, col = "steelblue", lty = 2, lwd = 2)
```



The simpler network we obtain by setting **threshold = 0.95** in averaged.network() is shown below; it is certainly easier to reason with from a qualitative point of view. Na figura abaixo R é o número de iterações usadas na fase de *bootstrapping*

```
avg.simpler = averaged.network(str.diff, threshold = 0.95)


par(mfrow = c(1, 2))

strength.plot(avg.diff,
              highlight = list((mb(avg.diff, 'EVOLUCAO'))),
              str.diff,
              shape = "rectangle",
              main = paste("R = ",
                           boots.trap,
                           " Thr = ",
                           attr(str.diff, "threshold")))

strength.plot(avg.simpler,
              highlight = list((mb(avg.simpler, 'EVOLUCAO'))),
              str.diff,
              shape = "rectangle",
              main = paste("R = ",
                           boots.trap,
                           " Thr = ",
                           attr(avg.simpler, "threshold")))
```





```
                                    #0.95))
```

Markov Blanket da variável EVOLUCAO

```
cat (mb(x = avg.diff, node = 'EVOLUCAO'))

## RENAL OBESIDADE UTI SUPORT_VEN
##
cat (mb(x = avg.simpler, node = 'EVOLUCAO'))

## UTI SUPORT_VEN
#par(mfrow = c(1, 2))
#save the currente workspace
save(list = ls(all.names = TRUE),
     file = "kk_09_sem_neuro_sem_idade_90pc_300it.RData",
     envir = .GlobalEnv)
```