Validação

José Elvano Moraes

4/15/2021

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.6     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
##
## Attaching package: 'Matrix'
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## Attaching package: 'igraph'
## The following objects are masked from 'package:bnlearn':
##
##     as.igraph, compare, degree, subgraph
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
## The following object is masked from 'package:tidyr':
##
##     crossing
## The following object is masked from 'package:tibble':
##
##     as_data_frame
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:igraph':
##
##     normalize, path, union

## The following object is masked from 'package:bnlearn':
##
##     score

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min

##
## Attaching package: 'graph'

## The following objects are masked from 'package:igraph':
##
##     degree, edges, intersection

## The following objects are masked from 'package:bnlearn':
##
##     degree, nodes, nodes<-

## The following object is masked from 'package:stringr':
##
##     boundary

##
## Attaching package: 'gRbase'

## The following objects are masked from 'package:bnstruct':
##
##     dag, observations, observations<-

## The following objects are masked from 'package:igraph':
##
##     is_dag, topo_sort
```

```
## The following objects are masked from 'package:bnlearn':
##
##     ancestors, children, parents
##
## Attaching package: 'gmp'
## The following objects are masked from 'package:Matrix':
##
##     crossprod, tcrossprod
## The following objects are masked from 'package:base':
##
##     %*%, apply, crossprod, matrix, tcrossprod
## C code of R package 'Rmpfr': GMP using 64 bits per limb
```

Variáveis selecionadas

```
## Rows: 76,666
## Columns: 17
## $ IDADE      <fct> "(37,73]", "(37,73]", "(73,109]", "(37,73]", "(73,109]", "(~
## $ FEBRE      <fct> 2, 2, 2, 1, 2, 1, 1, 1, 1, 2, 1, 2, 1, 2, 2, 1, 1, 2, 2, 2,~
## $ GARGANTA   <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 1, 2, 2, 2, 2,~
## $ DISPNEIA   <fct> 1, 2, 2, 1, 2, 2, 2, 2, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 1, 2,~
## $ SATURACAO  <fct> 1, 1, 2, 2, 2, 2, 2, 1, 2, 2, 1, 1, 1, 1, 2, 2, 2, 1, 2, 2,~
## $ 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,~
## $ PERD_OLFT  <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2,~
## $ PERD_PALA  <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,~
## $ NEUROLOGIC <fct> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 9, 2, 2, 2,~
## $ PNEUMOPATI <fct> 2, 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,~
```

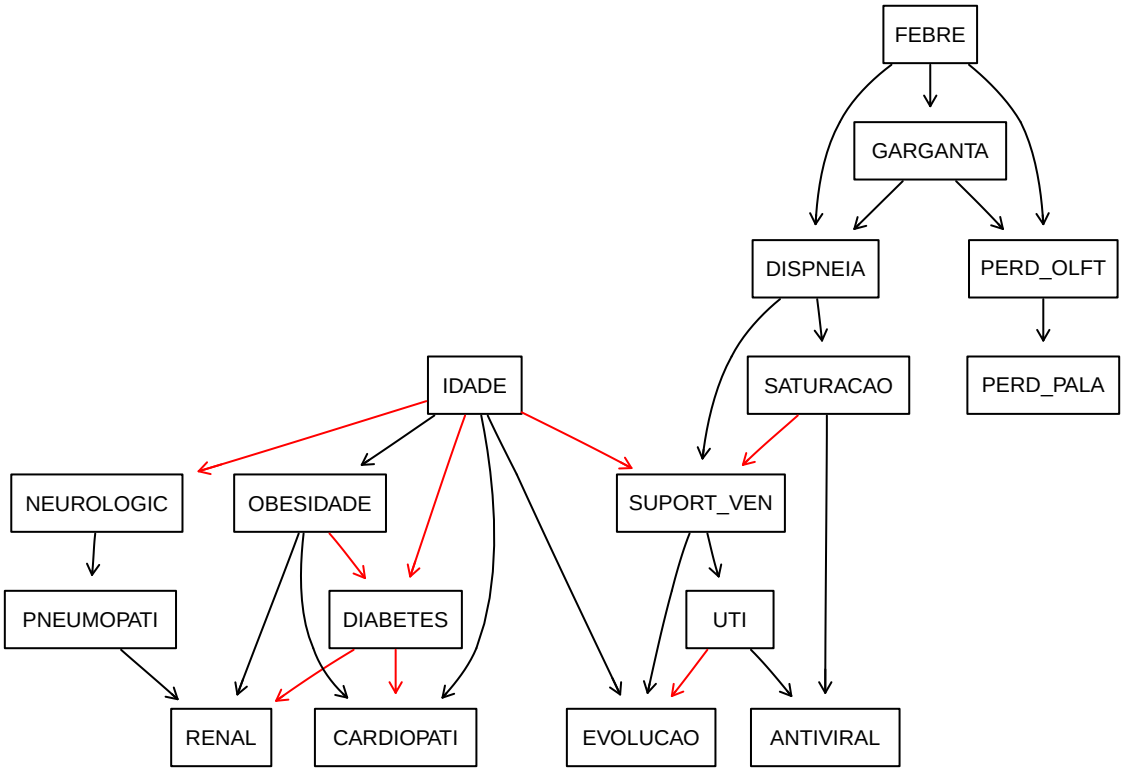
IMposição de estrutura com arcos que fazem sentido clínico (*White list*)

```
s1 <- sample_frac(ddf, .95, FALSE)

wl = matrix(c("OBESIDADE", "DIABETES",
              "IDADE", "DIABETES",
              "SATURACAO", "SUPORT_VEN",
              "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)
```

Rede Causal (DAG, *Directed Acyclic Graphic*)

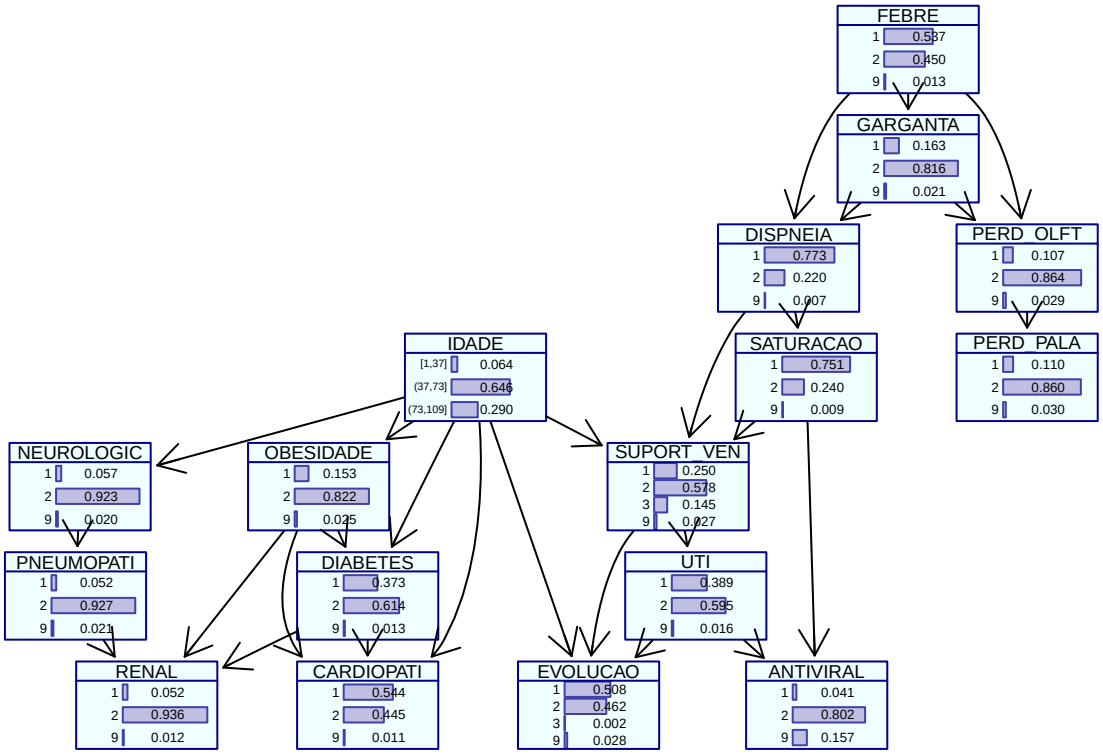
```
#par(mfrow=c(2,2))
graphviz.plot(bn1, shape='rectangle', highlight = list(arcs = wl), main = '...')
```

...




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

#par(mfrow=c(2,2))
graphviz.chart(fitted.1, scale = c(2, 3),  type = "barprob", col = "darkblue", bg = "azure", bar.col =
```

Rede de probabilidades

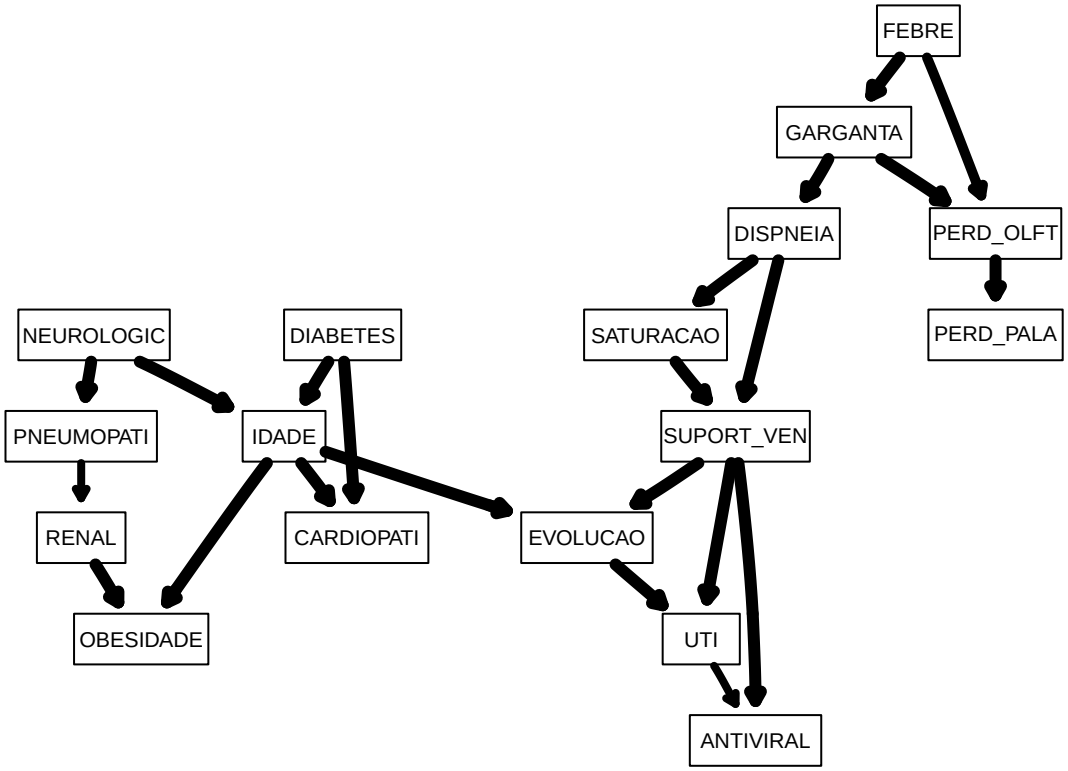


Bootstrapping

```
boots.trap  <- 100
str.diff = suppressMessages(boot.strength(s1, R = boots.trap, algorithm = "mmhc"))
cat(paste('Threshold: ', attr(str.diff, "threshold")))
```

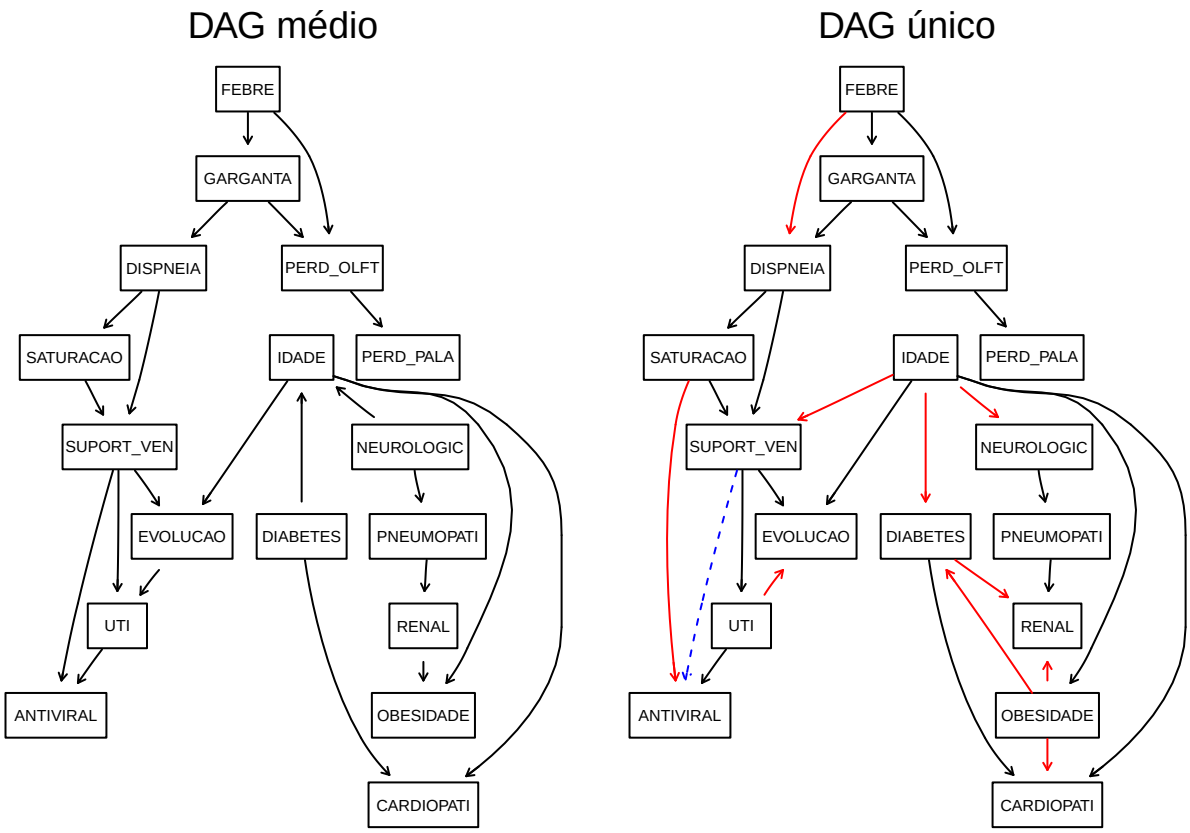
```
## Threshold:  0.39
```

```
avg.diff = averaged.network(str.diff)
strength.plot(avg.diff, str.diff, shape = "rectangle", main = paste("Iterações = ", boots.trap))
```



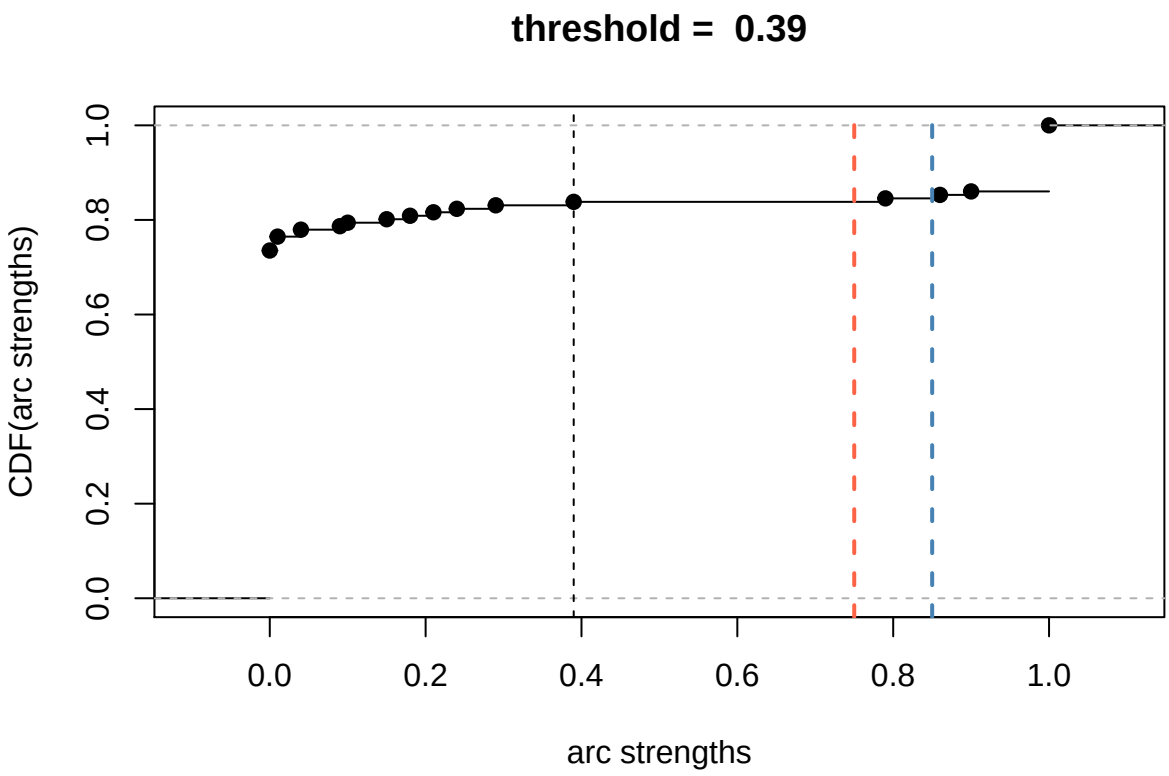
How can we compare the averaged network (avg.diff) with the network we originally learned in from all the data? The most qualitative way is to plot the two networks side by side, with the nodes in the same positions, and highlight the arcs that appear in one network and not in the other, or that appear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

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



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

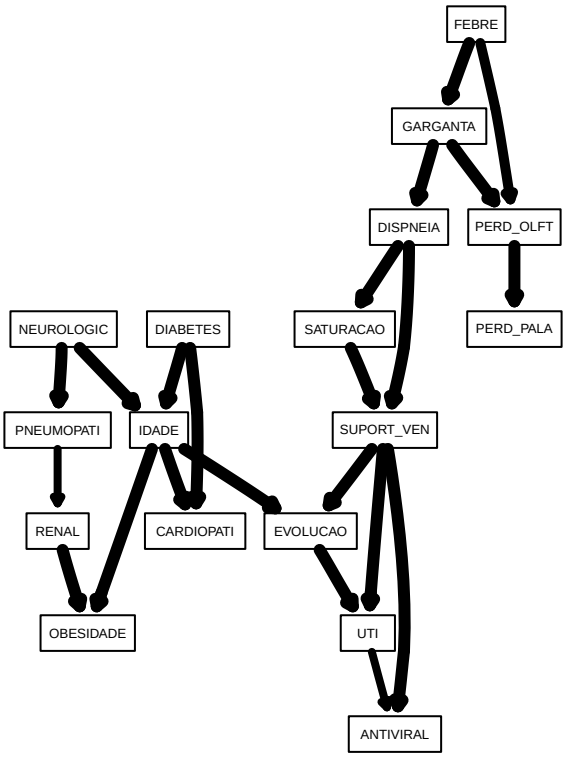
```
plot(str.diff)
abline(v = 0.75, col = "tomato", lty = 2, lwd = 2)
abline(v = 0.85, col = "steelblue", lty = 2, lwd = 2)
```



The simpler network we obtain by setting threshold = 0.8 in averaged.network() is shown below; it is certainly easier to reason with from a qualitative point of view.

```
par(mfrow = c(1, 2))
avg.simpler = averaged.network(str.diff, threshold = 0.75)
strength.plot(avg.diff, str.diff, shape = "rectangle", main = paste("Iterações = ", boots.trap, " Thr =
strength.plot(avg.simpler, str.diff, shape = "rectangle", main = 'Iterações = 100 Thr = 0.75')
```

Iterações = 100 Thr = 0.39



Iterações = 100 Thr = 0.75