Package ‘networktree’

March 18, 2020

Title Recursive Partitioning of Network Models
Version 0.2.2
Date 2020-03-18
Description Methods to create tree models with correlation-based network models (multivariate normal distributions).
Depends R (>= 3.5.0)
License GPL-3
Encoding UTF-8
LazyData true
Imports partykit, qgraph, stats, utils, Matrix, mvtnorm, Formula, grid, graphics, gridBase, reshape2
RoxygenNote 7.0.2
Suggests R.rsp, knitr, rmarkdown, fxregime, zoo
URL https://paytonjjones.github.io/networktree
BugReports http://github.com/paytonjjones/networktree/issues
NeedsCompilation no
Author Payton Jones [aut, cre] (<https://orcid.org/0000-0001-6513-8498>), Thorsten Simon [aut] (<https://orcid.org/0000-0002-3778-7738>), Achim Zeileis [aut] (<https://orcid.org/0000-0003-0918-3766>)
Maintainer Payton Jones <payton_jones@g.harvard.edu>
Repository CRAN
Date/Publication 2020-03-18 19:30:02 UTC

R topics documented:

comparetree .......................................................... 2
dass ................................................................. 3
getnetwork .......................................................... 4
mvnfit ............................................................... 5
Description

Quickly compares two partitions of a networktree object

Usage

```r
comparetree(tree, id1 = 2L, id2 = 3L, transform = "detect",
            highlights = 5, plot = FALSE, plot.type = c("compare", "subtract"),
            layout = "constrained", ...)
```

Arguments

- **tree**: a networktree object
- **id1**: the first partition
- **id2**: the second partition
- **transform**: should stored correlation matrices be transformed to partial correlations or graphical lasso? Can be set to "cor", "pcor", or "glasso". Defaults to automatic detection
- **highlights**: the number of comparisons to highlight
- **plot**: plot a comparison of the two partitions?
- **plot.type**: "compare" or "subtract". "compare" plots the two networks side by side. "subtract" subtracts network 2 from network 1, and plots a network where edge weights indicate the difference
- **layout**: layout for the plots. The default "constrained" uses a FR layout from the full dataset
- **...**: additional arguments passed to qgraph

Examples

```r
set.seed(1)
d <- data.frame(trend = 1:200, foo = runif(200, -1, 1))
d <- cbind(d, rbind(
  mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
                    sigma = matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), ncol = 3)),
```
mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
    sigma = matrix(c(1, 0, 0.5, 0, 1, 0.5, 0.5, 0.5, 1), ncol = 3))
)
colnames(d)[3:5] <- paste0("y", 1:3)

## Generate a networktree
tree1 <- networktree(nodevars=d[,3:5], splitvars=d[,1:2])

## Print out the structure
tree1

## Compare any two partitions
comparetree(tree1, id1=2, id2=3, highlights=3)

## Add a comparison plot
comparetree(tree1, id1=2, id2=3, plot=TRUE)

dass

Depression Anxiety and Stress Scale

Description

This dataset includes a randomly selected subsample of 5000 online participants who participated in a questionnaire available through the Open Source Psychometrics Project (https://openpsychometrics.org/), an organization that maintains an open website for the public to take psychometric tests for educational and entertainment purposes.

Usage
dass

Format

a dataframe. Columns represent questionnaire items and rows represent individuals.

Details

The Depression Anxiety and Stress Scale (DASS) is a self-report instrument for measuring depression, anxiety, and tension or stress. Each of 42 items falls into one of the three corresponding subscales. Labels for DASS items in this dataset are denoted by the prefix "dass" and the suffix "_D", "_A", or "_S", indicating the depression, anxiety, or stress subscale. Also includes demographics such as country, education level, rearing environment (urban/suburban/rural), gender, English as a native language, age, religion, sexual orientation, race, voting status, marriage status, and number of children in one’s family during childhood.

The full dataset is publicly available at https://openpsychometrics.org/_rawdata/DASS_data_21.02.19.zip and can be cited as:

**Examples**

```r
head(dass)

# Example networktree with DASS

data(dass)

# Select depression subscale

nodeVars <- colnames(dass)[(grep("_D", colnames(dass)))]

splitVars <- c("gender","orientation","race","married","engnat")

myTree<-networktree(dass[,nodeVars], dass[,splitVars])

myTree

plot(myTree)
```

---

**Description**

Easily extract a network from one of the nodes in a networktree object

**Usage**

```r
getnetwork(tree, id = 1L, transform = "detect", ...)
```

**Arguments**

- `tree`: a networktree object
- `id`: the node in the tree to extract. Use `summary(tree)` to see id numbers for each split
- `transform`: should stored correlation matrices be transformed to partial correlations or graphical lasso? Can be set to "cor", "pcor", or "glasso". Defaults to automatic detection
- `...`: arguments passed to `qgraph` (e.g., "tuning", "threshold")

**Examples**

```r
set.seed(1)
d <- data.frame(trend = 1:200, foo = runif(200, -1, 1))
d <- cbind(d, rbind(
    mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
    sigma = matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), ncol = 3)),
    mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
    sigma = matrix(c(1, 0, 0.5, 0, 1, 0.5, 0.5, 0.5, 1), ncol = 3))
))
```
mvnfit

```r
colnames(d)[3:5] <- paste0("y", 1:3)

## Now use the function
tree1 <- networktree(nodevars=d[,3:5], splitvars=d[,1:2])

getnetwork(tree1, id=1)
```

---

**mvnfit**

*Maximum Likelihood Estimation for Multivariate Normal Model*

**Description**

Fit a multivariate normal model without covariates or covariance restrictions. In addition to the (straightforward) parameter estimates the fitted log-likelihood and corresponding score contributions are computed.

**Usage**

```r
mvnfit(y, x = NULL, start = NULL, weights = NULL, offset = NULL,
       model = c("correlation", "mean", "variance"), ..., estfun = FALSE,
       object = FALSE)
```

**Arguments**

- **y**: A matrix or data.frame where each row corresponds to a k-dim observation.
- **x**: Not used yet
- **start**: Not used yet
- **weights**: Not used yet
- **offset**: Not used yet
- **model**: Vector of characters. Specifies which estimated parameters are returned.
- **...**: Not used yet
- **estfun**: Logical. Should the matrix of score contributions (aka estimating functions) be returned?
- **object**: Not used yet

**Details**

Used internally in when method="mob"
networktree

networktree: Partitioning of network models

Description

Computes a tree model with networks at the end of branches. Can use model-based recursive partitioning or conditional inference.

Wraps the mob() and ctree() functions from the partykit package.

Note: this package is in its early stages and the interface may change for future versions.

Usage

networktree(...)

## Default S3 method:
networktree(nodevars, splitvars, method = c("mob", "ctree"), model = "correlation", transform = c("cor", "pcor", "glasso"), na.action = na.omit, weights = NULL, ...)

## S3 method for class 'formula'
networktree(formula, data, transform = c("cor", "pcor", "glasso"), method = c("mob", "ctree"), na.action = na.omit, model = "correlation", ...)

Arguments

... additional arguments passed to mob_control (mob) or ctree_control (ctree)
nodevars the variables with which to compute the network. Can be vector, matrix, or dataframe
splitvars the variables with which to test split the network. Can be vector, matrix, or dataframe
method "mob" or "ctree"
model can be any combination of c("correlation", "mean", "variance") splits are determined based on the specified characteristics
transform should stored correlation matrices be transformed to partial correlations or a graphical lasso for plotting? Can be set to "cor" (default), "pcor", or "glasso"
na.action a function which indicates what should happen when the data contain missing values (NAs).
weights weights
formula A symbolic description of the model to be fit. This should either be of type y1 + y2 + y3 ~ x1 + x2 with node vectors y1, y2, and y3 or y ~ x1 + x2 with a matrix response y; x1 and x2 are used as partitioning variables.
data a data frame containing the variables in the model
plot.networktree

References


Examples

```r
set.seed(1)
d <- data.frame(trend = 1:200, foo = runif(200, -1, 1))
d <- cbind(d, rbind(
  mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
    sigma = matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), ncol = 3)),
  mvtnorm::rmvnorm(100, mean = c(0, 0, 0),
    sigma = matrix(c(1, 0, 0.5, 0, 1, 0.5, 0.5, 0.5, 1), ncol = 3))
))
colnames(d)[3:5] <- paste0("y", 1:3)

## Now use the function
tree1 <- networktree(nodevars=d[,3:5], splitvars=d[,1:2])

## Formula interface
tree2 <- networktree(y1 + y2 + y3 ~ trend + foo, data=d)

## Conditional version
tree3 <- networktree(nodevars=d[,3:5], splitvars=d[,1:2],
  method="ctree")

## Change control arguments
tree4 <- networktree(nodevars=d[,3:5], splitvars=d[,1:2],
  alpha=0.01)
```

plot.networktree  

Plotting 'treenetwork' objects

Description

Wraps plot.party to plot a tree model with networks on the ends. Networks are plotted with qgraph, and additional arguments are passed there.

Usage

```r
## S3 method for class 'networktree'
plot(x, transform = NULL, layout = "lock",
     partyargs = list(), ...)
```
Arguments

- **x**: an object of type 'networktree'
- **transform**: "cor", "pcor", or "glasso". If set to NULL, transform detected from x
- **layout**: network layout, passed to qgraph. Default "lock" computes spring layout for the full sample and applies this to all graphs
- **partyargs**: additional arguments (list format) passed to partykit::plot.party
- **...**: additional arguments passed qgraph

predict.networktree  Predict 'networktree' objects

Description

Wraps predict.party

Usage

```r
## S3 method for class 'networktree'
predict(object, newdata = NULL, type = c("node", "parameter"), ...)
```

Arguments

- **object**: a fitted 'networktree'
- **newdata**: An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
- **type**: "node", or "parameter". Specifies whether to predict nodes (return value is a vector) or parameters (matrix).
- **...**: not used

print.networktree  Printing 'treenetwork' objects

Description

Wraps print.modelparty to print a tree model with networks on the ends.

Usage

```r
## S3 method for class 'networktree'
print(x, parameters = FALSE, FUN = NULL, ...)
```
Arguments

- **x**: an object of type `networktree`
- **parameters**: print parameters for each partition? See `getnetwork` function for extracting parameters conveniently
- **FUN**: only evaluated if `parameters=TRUE`, passed to `print.modelparty`
- **...**: additional arguments passed to `print.modelparty`

---

**tipi**  
*Ten Item Personality Questionnaire*

Description

This dataset includes 1899 online participants who participated in a questionnaire available through the Open Source Psychometrics Project (https://openpsychometrics.org/), an organization that maintains an open website for the public to take psychometric tests for educational and entertainment purposes.

Usage

```r
tipi
```

Format

A dataframe. Columns represent questionnaire items and rows represent individuals.

Details

The Ten Item Personality Questionnaire (TIPI) is a brief inventory of the Big Five personality domains. Each personality domain is assessed with two items. One item measures the domain normally and the other item measures the domain in reverse (e.g., "reserved, quiet" for reverse extraversion).

Labels for TIPI items in this dataset correspond to the first letter of each Big Five personality domain (Extraversion, Neuroticism, Conscientiousness, Agreeableness, and Openness to experience), with the character "r" indicating items that measure the domain in reverse.

Also includes demographics such as education level, rearing environment (urban/suburban/rural), gender, English as a native language, age, religion, sexual orientation, race, voting status, marriage status, and number of children in one’s family during childhood.

The dataset is publicly available at http://openpsychometrics.org/_rawdata/GCBS.zip and can be cited as:

Examples

head(tipi)

## Example networktree with TIPI
data(tipi)
nodeVars <- c("E","A_r","C","N","O","E_r","A_r","C_r","N_r","O_r")
splitVars <- c("gender","education","engnat")
myTree<-networktree(tipi[,nodeVars], tipi[,splitVars])
myTree
plot(myTree)

workaholic  Workaholism and Psychiatric Symptoms

Description

This dataset includes 16,426 workers who were assessed on symptoms of psychiatric disorders (ADHD, OCD, anxiety, depression) and workaholism.

Usage

workaholic

Format

a dataframe. Columns represent symptoms and rows represent individuals

Details

Scales: Adult ADHD Self-Report Scale, Obsession-Compulsive Inventory-Revised, Hospital Anxiety and Depression Scale, and the Bergen Work Addiction Scale.

Also includes demographics such as age, gender, work status, position, sector, annual income.

The dataset is publicly available at https://doi.org/10.1371/journal.pone.0152978 and can be cited as:

Examples

head(workaholic)

## Example networktree with OCI-R scale
data(workaholic)
nodeVars <- paste("OCIR",1:18,sep="")
splitVars <- c("Workaholism_diagnosis","Gender")
myTree<-networktree(workaholic[,nodeVars]. workaholic[,splitVars])
myTree
plot(myTree)
Index

 Topic datasets
 dass, 3
 tipi, 9
 workaholic, 10

 comparetree, 2
 ctree_control, 6

dass, 3

getnetwork, 4

mob_control, 6

mvnfit, 5

networktree, 6

plot.networktree, 7

predict.networktree, 8

print.networktree, 8

tipi, 9

workaholic, 10