Package: ChoR (via r-universe)

September 15, 2024

Title Chordalysis R Package

Version 0.0-4

Date 2018-05-16

Description Learning the structure of graphical models from datasets with thousands of variables. More information about the research papers detailing the theory behind Chordalysis is available at <http://www.francois-petitjean.com/Research> (KDD 2016, SDM 2015, ICDM 2014, ICDM 2013). The R package development site is <https://github.com/HerrmannM/Monash-ChoR>.

Imports rJava (>= 0.9.9), commonsMath, stats

Suggests graph (>= 1.52.0), Rgraphviz (>= 2.18.0)

SystemRequirements Java (>= 8)

License GPL-3

Encoding UTF-8

RoxygenNote 6.0.1

NeedsCompilation no

LazyData true

Author François Petitjean [aut], Matthieu Herrmann [aut, com, cre], Christoph Bergmeir [ctb]

Maintainer Matthieu Herrmann <matthieu.herrmann@monash.edu>

Date/Publication 2018-05-16 05:04:52 UTC

Repository https://herrmannm.r-universe.dev

RemoteUrl https://github.com/cran/ChoR

RemoteRef HEAD

RemoteSha 2ecb4b2d5de4f649464ec81e802d6295082ef73d

Contents

ChoR *Getting started with the ChoR package*

Description

The chordalysis algorithm allows to learn the structure of graphical models from datasets with thousands of variables. More information about the research papers detailing the theory behind Chordalysis is available at http://www.francois-petitjean.com/Research

If you have problems using ChoR, find a bug, or have suggestions, please contact the package maintainer by email. Do not write to the general R lists or contact the authors of the original chordalysis software.

If you use the package, please cite references in your publications.

Details

Chordalysis allows to learn the structure of graphical models from datasets with thousands of variables. There are 3 differentes algorithms versions: SMT, Budget and MML. SMT, standing for Subfamiliwize Multiple Testing, is generally the method of choice. It superseeds Budget and is always superior to it. Demonstration is in our KDD'16 paper (see CITATION). Both SMT and Budget are based on statistical testing, while MML uses information theory to decide upon a model. The objective of the different techniques is slightly different: SMT controls the familywise error rate (FWER) while MML is a probabilistic method. Our experiments (again in KDD'16) indicate that SMT is superior to MML for most datasets.

References

See citation("ChoR")

Examples

```
# Warning: RJava requires to **copy** your data from R into a JVM.
# If you need extra memory, use this option (here, for 4Gb) **before** loading choR.
# Note: not needed in our case, kept for the example
options( java.parameters = "-Xmx4g" )
library(ChoR)
```

```
# Helper function for graph printing. Require Rgraphviz:
# source("https://bioconductor.org/biocLite.R")
# biocLite("Rgraphviz")
printGraph = function(x){
  if(requireNamespace("Rgraphviz", quietly=TRUE)){
    attrs <- list(node=list(shape="ellipse", fixedsize=FALSE, fontsize=25))
    Rgraphviz::plot(x, attrs=attrs)
  } else { stop("Rgraphviz required for graph printing.") }
}


###### MUSHROOM #####
# We are using a partial UCI mushroom data set (the example should not be too long)
MR.url = system.file("extdata", "mushrooms.csv", package = "ChoR", mustWork = TRUE)

MR.data =
  read.csv(
              MR.url,
              header            = TRUE,             # Here, we have a header
              na.strings        = c("NA","?",""),   # Configure the missing values
              stringsAsFactors  = FALSE,            # Keep strings for now
              check.names       = TRUE              # Replace some special characters
            )

# This file has a special line with types. You can check this with MR.data[1,].
# Let's remove it:
MR.data = MR.data[-1, ]

# Launch the SMT analysis, with:
# ## default pValueThreshold=0.05
# ## computation of attributes cardinality from the data
MR.res = ChoR.SMT(MR.data)

# Access the result:
# ## As a list of cliques:
NR.cl = ChoR.as.cliques(MR.res)
print(NR.cl)
# ## As a formula
NR.fo = ChoR.as.formula(MR.res)
print(NR.fo)
# ## As a graph
if(requireNamespace("graph", quietly=TRUE)){
  NR.gr = ChoR.as.graph(MR.res)
  printGraph(NR.gr)
} else {
  print("'graph' package not installed; Skipping 'as graph' example.")
}



###### Titanic #####
# We are using the titanix data set
```

```
MR.url = system.file("extdata", "titanic.dat.txt", package = "ChoR", mustWork = TRUE)

T.data =
  read.csv( MR.url,
            sep               = "",       # White spaces
            header            = FALSE,
            stringsAsFactors  = FALSE
          )

# Give meaningful names
colnames(T.data) = c(   "Class", "Age", "Sex", "Survived" )
# Chordalysis
T.res = ChoR.SMT(T.data, card = c(4, 2, 2, 2))

if(requireNamespace("graph", quietly=TRUE)){
  T.gr = ChoR.as.graph(T.res)
  printGraph(T.gr)
}
```

`ChoR.as.cliques` *Get the cliques.*

Description

Get the list of cliques associated to a chordalysis object.

Usage

```
ChoR.as.cliques(x)
```

Arguments

x	A chordalysis object obtained by a call to ChoR.

Value

A list of cliques, a clique being a list of attributes'name, i.e. a list of lists of names.

`ChoR.as.formula` *Get the formula.*

Description

Extract the formula from a Chordalysis object.

Usage

```
ChoR.as.formula(x)
```

Arguments

x	A chordalysis object obtained by a call to ChoR.

Value

a formula representing the model

ChoR.as.graph *Get the graph.*

Description

Get an undirected graph representing the cliques from a Chordalysis object.

Usage

```
ChoR.as.graph(x)
```

Arguments

x	A chordalysis object obtained by a call to ChoR.

Details

The undirected graph use the graph package from Bioconductor.

Value

A graph

ChoR.Budget *Call to the budget chordalysis algorithm.*

Description

Searches a statistically significant decomposable model to explain a dataset using Prioritized Chordalysis.

Usage

```
ChoR.Budget(x, pValueThreshold = 0.05, budgetShare = 0.01, card = NULL)
```

Arguments

x	A dataframe with categorical data; column names are the name of the attributes.
pValueThreshold	A double value, minimum p-value for statistical consistency (commonly 0.05)
budgetShare	A double value, share of the statistical budget to consume at each step (>0 and <=1; 0.01 seems like a reasonable value for most datasets)
card	A vector containing the cardinality of the attributes (position wise).

Details

Call the Budget chordalysis function on the dataframe x. The optionnal card argument can provide a vector of cardinalities for each attribute (i.e. column) of the dataframe. If absent, the cardinalities are computed from the dataframe, but not accurate if some possible values never show up. See papers "Scaling log-linear analysis to high-dimensional data, ICDM 2013", "Scaling log-linear analysis to datasets with thousands of variables, SDM 2015", and "A multiple test correction for streams and cascades of statistical hypothesis tests, KDD 2016" for more details.

Value

A Chordalysis object. Use `ChoR.as.*` functions to access the result.

Examples

```
## Not run:  res = ChoR.Budget(data)
## Not run:  res = ChoR.Budget(data, budgetShare=0.0)
## Not run:  res = ChoR.Budget(data, 0.05, card = c(3, 5, 4, 4, 3, 2, 3, 3))
```

ChoR.loadData *[INTERNAL] Load the data from a dataframe (and with an optionnal cardinality vector)*

Description

Loads the data from x, which should be a dataframe (else, a conversion to a dataframe is attempted).

Usage

```
ChoR.loadData(x, card = NULL)
```

Arguments

x	A dataframe with categorical data; column names are the name of the attributes.
card	A vectore containing the cardinality of the attributes (position wise).

Details

Loads the data from x, which should be a dataframe (else, a conversion to a dataframe is attempted). The data must be categorical, each column being an attribute. The optionnal argument card should be a vector representing the cardinality of each attribute (position wise). If it is provided, its size must be equal to the number of attributes. Else, its values will be computed from the data, and the cardinality for an attribute will be accurate only if all its possible values appear at least once in the data.

Value

A list how two .jarray references (one for the dimension, one for the data) and the dataframe

---

ChoR.MML *Call to the MML chordalysis algorithm.*

Description

Searches a statistically significant decomposable model to explain a dataset.

Usage

```
ChoR.MML(x, card = NULL)
```

Arguments

x	A dataframe with categorical data; column names are the name of the attributes.
card	A vector containing the cardinality of the attributes (position wise).

Details

Call the MML chordalysis function on the dataframe x. The optionnal card argument can provide a vector of cardinalities for each attribute (i.e. column) of the dataframe. If absent, the cardinalities are computed from the dataframe, but may not be accurate if some possible values never show up. See papers "A statistically efficient and scalable method for log-linear analysis of high-dimensional data, ICDM 2014" and "Scaling log-linear analysis to datasets with thousands of variables, SDM 2015" for more details.

Value

A Chordalysis object. Use `ChoR.as.*` functions to access the result.

Examples

```
## Not run:  res = ChoR.MML(data)
## Not run:  res = ChoR.MML(data, c(3, 5, 4, 4, 3, 2, 3, 3))
```

ChoR.processResult *[INTERNAL] Process the result of a java Chordalysis algorithm.*

Description

Convert the result in a 'chordalysis object'.

Usage

```
ChoR.processResult(x, modelStr)
```

Arguments

x	The dataframe used to loadData; column names are the name of the attributes.
modelStr	The result of a java Chordalysis algorithm

Details

Process the result of a call to the java Chordalysis algorithm. The result is a String of the forme "~0*1*2+...+3*4*5". The numbers (+1 for indice correction) are replaced with the corresponding column name in x, and the string is split in a list of cliques, a cliques being a list of name. For example, "~ 0*1*2 + 3*4*5" gives the two cliques [[[[0,1,2]], [[3,4,5]]]]

Value

A Chordalysis object. Use ChoR.as.* functions to access the result.

ChoR.SMT *Call to the SMT chordalysis algorithm.*

Description

Searches a statistically significant decomposable model to explain a dataset using Prioritized Chordalysis.

Usage

```
ChoR.SMT(x, pValueThreshold = 0.05, card = NULL)
```

Arguments

x	A dataframe with categorical data; column names are the name of the attributes.
pValueThreshold	A double value, minimum p-value for statistical consistency (commonly 0.05)
card	A vector containing the cardinality of the attributes (position wise).

Details

Call the SMT chordalysis function on the dataframe x. The optionnal card argument can provide a vector of cardinalities for each attribute (i.e. column) of the dataframe. If absent, the cardinalities are computed from the dataframe, but may not be accurate if some possible values never show up. See papers "A multiple test correction for streams and cascades of statistical hypothesis tests, KDD 2016", "Scaling log-linear analysis to high-dimensional data, ICDM 2013", and "Scaling log-linear analysis to datasets with thousands of variables, SDM 2015" for more details.

Value

A Chordalysis object. Use `ChoR.as.*` functions to access the result.

Examples

```
## Not run:  res = ChoR.SMT(data, 0.05, c(3, 5, 4, 4, 3, 2, 3, 3))
## Not run:  res = ChoR.SMT(data, card = c(3, 5, 4, 4, 3, 2, 3, 3))
```

`print.chordalysis` *Gives a string representation of the model.*

Description

Create a String representation of a model, compatible with the formula interface, e.g. "~a*b*c+...+e*f*g".

Usage

```
## S3 method for class 'chordalysis'
print(x, ...)
```

Arguments

`x`	A "Chordalysis" model, obtained by a call to a ChoR function.
`...`	Unused argument, here for S3 consistency

Value

A String representation of the model.

`toString` *[INTERNAL] Gives a string representation of the model.*

Description

Create a String representation of a model, compatible with the formula interface, e.g. "~a*b*c+...+e*f*g".

Usage

```
toString(x)
```

Arguments

x	A "Chordalysis" model, obtained by a call to a ChoR function.

Value

A String representation of the model.

Index