Package ‘minet’

March 26, 2024

Title  Mutual Information NETworks
Version 3.60.0
Date 2014-07
Author Patrick E. Meyer, Frederic Lafitte, Gianluca Bontempi
Description This package implements various algorithms for inferring mutual information networks from data.
Maintainer Patrick E. Meyer <software@meyerp.com>
License Artistic-2.0
URL http://minet.meyerp.com
Imports infotheo
biocViews Microarray, GraphAndNetwork, Network, NetworkInference
git_url https://git.bioconductor.org/packages/minet
git_branch RELEASE_3_18
git_last_commit 3cf4b45
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-25

R topics documented:

aracne ................................................................. 2
build.mim ................................................................ 3
clr ...................................................................... 4
minet ................................................................. 5
mrnet ............................................................... 7
mrnetb .............................................................. 8
syn.data ............................................................ 9
syn.net .............................................................. 10
validate ........................................................... 10
vis.res .............................................................. 11

Index 14
Description

This function takes the mutual information matrix as input in order to return the inferred network according to the Aracne algorithm. This algorithm applies the data processing inequality to all triplets of nodes in order to remove the least significant edge in each triplet.

Usage

aracne( mim, eps=0 )

Arguments

mim A square matrix whose i,j th element is the mutual information between variables Xi and Xj - see build.mim.
eps Numeric value indicating the threshold used when removing an edge : for each triplet of nodes (i,j,k), the weakest edge, say (ij), is removed if its weight is below min{(ik),(jk)}-eps - see references.

Details

The Aracne procedure starts by assigning to each pair of nodes a weight equal to their mutual information. Then, the weakest edge of each triplet is interpreted as an indirect interaction and is removed if the difference between the two lowest weights is above a threshold eps.

Value

aracne returns a matrix which is the weighted adjacency matrix of the network. In order to display the network, load the package Rgraphviz and use the following command:
plot( as( returned.matrix , "graphNEL") )

References


See Also

build.mim, clr, mrnet, mrnetb
**build.mim**

**Examples**

```r
data(syn.data)
mim <- build.mim(syn.data, estimator="spearman")
net <- aracne(mim)
```

**build.mim**

*Build Mutual Information Matrix*

**Description**

`build.mim` takes the dataset as input and computes the mutual information between all pairs of variables according to the mutual information estimator `estimator`. The results are saved in the mutual information matrix (MIM), a square matrix whose $(i,j)$ element is the mutual information between variables $X_i$ and $X_j$.

**Usage**

```r
build.mim(dataset, estimator = "spearman", disc = "none", nbins = sqrt(NROW(dataset)))
```

**Arguments**

- **dataset**: data.frame containing gene expression data or any dataset where columns contain variables/features and rows contain outcomes/samples.
- **estimator**: The name of the entropy estimator to be used. The package can use the four mutual information estimators implemented in the package "infotheo": "mi.empirical", "mi.mm", "mi.shrink", "mi.sg" and three estimators based on correlation: "pearson", "spearman", "kendall" (default: "spearman") - see details.
- **disc**: The name of the discretization method to be used with one of the discrete estimators: "none", "equalfreq", "equalwidth" or "globalequalwidth" (default: "none") - see infotheo package.
- **nbins**: Integer specifying the number of bins to be used for the discretization if disc is different from "none". By default the number of bins is set to $\sqrt{n}$ where $n$ is the number of samples.

**Details**

- "mi.empirical": This estimator computes the entropy of the empirical probability distribution.
- "mi.mm": This is the Miller-Madow asymptotic bias corrected empirical estimator.
- "mi.shrink": This is a shrinkage estimate of the entropy of a Dirichlet probability distribution.
- "mi.sg": This is the Schurmann-Grassberger estimate of the entropy of a Dirichlet probability distribution.
- "pearson": This computes mutual information for normally distributed variable.
- "spearman": This computes mutual information for normally distributed variable using Spearman's correlation instead of Pearson's correlation.
- "kendall": This computes mutual information for normally distributed variable using Kendall's correlation instead of Pearson's correlation.
Value

build.mim returns the mutual information matrix.

Author(s)

Patrick E. Meyer, Frederic Lafitte, Gianluca Bontempi

References


See Also

clr, aracne, mrnet, mrnetb

Examples

data(syn.data)
mim <- build.mim(syn.data,estimator="spearman")

clr

Context Likelihood or Relatedness Network

Description

clr takes the mutual information matrix as input in order to return the inferred network - see details.

Usage

clr( mim, skipDiagonal=1 )

Arguments

mim A square matrix whose i,j th element is the mutual information between variables $X_i$ and $X_j$ - see build.mim.

skipDiagonal Skips the diagonal in the calculation of the mean and sd, default=1.
Details

The CLR algorithm is an extension of relevance network. Instead of considering the mutual information \( I(X_i; X_j) \) between features \( X_i \) and \( X_j \), it takes into account the score \( \sqrt{z_i^2 + z_j^2} \), where

\[
z_i = \max \left\{ 0, \frac{I(X_i; X_j) - \mu_i}{\sigma_i} \right\}
\]

and \( \mu_i \) and \( \sigma_i \) are, respectively, the mean and the standard deviation of the empirical distribution of the mutual information values \( I(X_i; X_k), k=1,\ldots,n \).

Value

\( \text{clr} \) returns a matrix which is the weighted adjacency matrix of the network. In order to display the network, load the package Rgraphviz and use the following command plot( as( returned.matrix , “graphNEL” ) )

Author(s)

Implementation: P. E. Meyer and J.C.J. van Dam

References


See Also

\texttt{build.mim}, \texttt{aracne}, \texttt{mrnet}, \texttt{mrnetb}

Examples

\begin{verbatim}
data(syn.data)
mim <- build.mim(syn.data, estimator="spearman")
net <- clr(mim)
\end{verbatim}

\begin{verbatim}
minet Mutual Information Network
\end{verbatim}

Description

For a given dataset, \texttt{minet} infers the network in two steps. First, the mutual information between all pairs of variables in dataset is computed according to the \texttt{estimator} argument. Then the algorithm given by \texttt{method} considers the estimated mutual informations in order to build the network. This package is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.
Usage

minet(dataset, method="mrnet", estimator="spearman", disc="none", nbins=sqrt(NROW(dataset)))

Arguments

- **dataset**: data.frame where columns contain variables/features and rows contain outcomes/samples.
- **method**: The name of the inference algorithm: "clr", "aracne", "mrnet" or "mrnetb" (default: "mrnet") - see references.
- **estimator**: The name of an entropy estimator (or correlation) to be used for mutual information computation ("pearson", "spearman", "kendall" and from infotheo package: "mi.empirical", "mi.mm", "mi.shrink", "mi.sg"), (default: "spearman") - see build.mim.
- **disc**: The name of the discretization method to be used, if required by the estimator: "none", "equalfreq", "equalwidth" or "globalequalwidth" (default: "none") - see infotheo package.
- **nbins**: Integer specifying the number of bins to be used for the discretization if disc is set properly. By default the number of bins is set to \( \sqrt{N} \) where \( N \) is the number of samples.

Value

minet returns a matrix which is the weighted adjacency matrix of the network. The weights range from 0 to 1 and can be seen as a confidence measure on the presence of the arcs. In order to display the network, load the package Rgraphviz and use the following command:

```r
plot(as(returned.matrix, "graphNEL"))
```

Author(s)

Patrick E. Meyer, Frederic Lafitte, Gianluca Bontempi

References


See Also

build.mim, clr, mrnet, mrnetb, aracne

Examples

```r
data(syn.data)
net1 <- minet(syn.data)
net2 <- minet(syn.data, estimator="pearson")
net3 <- minet(syn.data, method="clr")
```
mrnet

**Maximum Relevance Minimum Redundancy**

**Description**

*mrnet* takes the mutual information matrix as input in order to infer the network using the maximum relevance/minimum redundancy feature selection method - see details.

**Usage**

```
mrnet(mim)
```

**Arguments**

- **mim** A square matrix whose i,j th element is the mutual information between variables $X_i$ and $X_j$ - see *build.mim*.

**Details**

The MRNET approach consists in repeating a MRMR feature selection procedure for each variable of the dataset. The MRMR method starts by selecting the variable $X_i$ having the highest mutual information with the target $Y$. In the following steps, given a set $S$ of selected variables, the criterion updates $S$ by choosing the variable $X_k$ that maximizes $I(X_k; Y) - \frac{1}{|S|} \sum_{X_i \in S} I(X_k; X_i)$

The weight of each pair $X_i, X_j$ will be the maximum score between the one computed when $X_i$ is the target and the one computed when $X_j$ is the target.

**Value**

*mrnet* returns a matrix which is the weighted adjacency matrix of the network. In order to display the network, load the package Rgraphviz and use the following command:

```
plot( as( returned.matrix ,"graphNEL") )
```

**Author(s)**

Patrick E. Meyer, Frederic Lafitte, Gianluca Bontempi

**References**


See Also

build.mim, clr, aracne, mrnetb

Examples

data(syn.data)
mim <- build.mim(syn.data, estimator="spearman")
net <- mrnet(mim)

Description

mrnetb takes the mutual information matrix as input in order to infer the network using the maximum relevance/minimum redundancy criterion combined with a backward elimination and a sequential replacement - see references. This method is a variant of mrnet.

Usage

mrnetb(mim)

Arguments

mim A square matrix whose i,j th element is the mutual information between variables $X_i$ and $X_j$ - see build.mim.

Value

mrnetb returns a matrix which is the weighted adjacency matrix of the network. In order to display the network, load the package Rgraphviz and use the following command:
plot( as( returned.matrix ,"graphNEL") )

References


See Also

build.mim, clr, mrnet, aracne
**Examples**

```r
data(syn.data)
mim <- build.mim(syn.data, estimator="spearman")
ext <- mrnetb(mim)
```

---

**syn.data**  
*Simulated Gene Expression Data*

---

**Description**

Dataset containing 100 samples and 50 genes generated by the publicly available SynTReN generator using a yeast source network - see `syn.net`.

**Usage**

```r
data(syn.data)
```

**Format**

`syn.data` is a data frame containing 100 rows and 50 columns. Each row contains a microarray experiment and each column contains a gene.

**Source**

SynTReN 1.1.3 with source network : `yeast.nn.sif`

**References**


**Examples**

```r
data(syn.data)
data(syn.net)
mim <- build.mim(syn.data, estimator="spearman")
inferred.net <- mrnet(mim)
max(fscores(validate( inferred.net, syn.net )))
```
### syn.net  
*SynTReN Source Network*

**Description**

This is the true underlying network used to generate the dataset loaded by `data(syn.data)` - see `syn.data`.

**Usage**

`data(syn.net)`

**Format**

`syn.net` is a boolean adjacency matrix representing an undirected graph of 50 nodes.

**Source**

`syn.net` is the "yeast\_nn.sif" source network from the SynTReN generator where all the variables/nodes not in `syn.data` were removed.

**References**


**Examples**

```r
data(syn.data)
data(syn.net)
mim <- build.mim(syn.data,estimator="spearman")
inferred.net <- mrnet(mim)
max(fscores(validate( inferred.net, syn.net )))
```

### validate  
*Inference Validation*

**Description**

`validate` compares the inferred network to the true underlying network for several threshold values and appends the resulting confusion matrices to the returned object.

**Usage**

`validate(inet,tnet)`
Arguments

inet
This is the inferred network, a data.frame or matrix obtained by one of the functions minet, aracne, clr or mnet.

tnet
The true underlying network. This network must have the same size and variable names as inet.

Details

The first network inet is compared to the true underlying network, tnet, in order to compute a confusion (adjacency) matrix. All the confusion matrices, obtained with different threshold values, are appended to the returned object. In the end the validate function returns a data.frame containing steps+1 confusion matrices.

Value

validate returns a data.frame with four columns named thrsh, tp, fp, fn. These values are computed for each of the steps thresholds. Thus each row of the returned object contains the confusion matrix for a different threshold.

See Also

minet, vis.res

Examples

data(syn.data)
data(syn.net)
inf.net <- mrnet(build.mim(syn.data, estimator="spearman"))
table <- validate( inf.net, syn.net )
table <- validate( inf.net, syn.net )
Arguments

**table**
This is the data.frame returned by the `validate` function where columns contain TP,FP,TN,FN values (confusion matrix) as well as the threshold value used - see `validate`.

**beta**
Numeric used as the weight of the recall in the f-score formula - see details. The default value of this argument is 1, meaning precision as important as recall.

**device**
The device to be used. This parameter allows the user to plot precision-recall and receiver operating characteristic curves for various inference algorithms on the same plotting window - see examples.

... arguments passed to `plot`

Details

A confusion matrix contains FP,TP,FN,FP values.

- "true positive rate" \( tpr = \frac{TP}{TN+TP} \)
- "false positive rate" \( fpr = \frac{FP}{FN+FP} \)
- "precision" \( p = \frac{TP}{TP+FP} \)
- "recall" \( r = \frac{TP}{TP+FN} \)
- "f-beta-score" \( F_\beta = (1 + \beta)^\frac{pr}{r+\beta p} \)

Value

The function `show.roc` (`show.pr`) plots the ROC-curve (PR-curve) and returns the device associated with the plotting window.

The function `auc.roc` (`auc.pr`) computes the area under the ROC-curve (PR-curve) using the trapezoidal approximation.

The function `pr` returns a data.frame where steps is the number of thresholds used in the validation process. The first column contains precisions and the second recalls - see details.

The function `rates` also returns a data.frame where the first column contains true positive rates and the second column false positive rates - see details.

The function `fscores` returns fscores according to the confusion matrices contained in the 'table' argument - see details.

References


See Also

`validate`, `plot`
Examples

data(syn.data)
data(syn.net)
# Inference
mr <- minet( syn.data, method="mrnet", estimator="spearman" )
ar <- minet( syn.data, method="aracne", estimator="spearman" )
clr<- minet( syn.data, method="clr", estimator="spearman" )
# Validation
mr.tbl <- validate(mr,syn.net)
ar.tbl <- validate(ar,syn.net)
clr.tbl<- validate(clr,syn.net)
# Plot PR-Curves
max(fscores(mr.tbl))
device <- show.pr(mr.tbl, col="green", type="b")
device <- show.pr(ar.tbl, device=device, col="blue", type="b")
show.pr(clr.tbl, device=device, col="red", type="b")
auc.pr(clr.tbl)
Index

* datasets
  syn.data, 9
  syn.net, 10

* misc
  aracne, 2
  build.mim, 3
  clr, 4
  minet, 5
  mrnet, 7
  mrnetb, 8
  validate, 10
  vis.res, 11
  auc.pr (vis.res), 11
  auc.roc (vis.res), 11
  build.mim, 2, 3, 4–8
  clr, 2, 4, 4, 6, 8, 11
  fscores (vis.res), 11
  minet, 5, 11
  mrnet, 2, 4–6, 7, 8, 11
  mrnetb, 2, 4–6, 8, 8
  plot, 12
  pr (vis.res), 11
  rates (vis.res), 11
  show.pr (vis.res), 11
  show.roc (vis.res), 11
  syn.data, 9, 10
  syn.net, 9, 10
  validate, 10, 11, 12
  vis.res, 11, 11