# Package 'GroupBN'

January 20, 2025

**Type** Package

**Date** 2021-03-07

**Title** Inferring Group Bayesian Networks using Hierarchical Feature Clustering

**Version** 1.2.0

**Maintainer** Ann-Kristin Becker <annkristinbecker@web.de>

**Description** Group Bayesian Networks: This package implements the inference of group Bayesian networks based on hierarchical feature clustering, and the adaptive refinement of the grouping regarding an outcome of interest, as described in Becker et. al (2021) <doi:10.1371/journal.pcbi.1008735>.

**Depends** R (>= 3.5.0), bnlearn, ClustOfVar, PCAmixdata, arules, zoo

**Imports** PRROC, MLmetrics, rlist, stats, magrittr, visNetwork, plyr, stringr

**License** GPL (>= 2)

**URL** https://www.r-project.org

**Encoding** UTF-8

**LazyData** true

**NeedsCompilation** no

**Author** Ann-Kristin Becker [aut, cre], Lars Kaderali [aut, ths]

**Repository** CRAN

**Date/Publication** 2021-03-07 14:40:02 UTC

## Contents

## `cross.en` *cross.en*

### Description

Calculates the weighted cross entropy / log-loss for a vector of observations and predicted probabilities (weighted by class proportions)

### Usage

```
cross.en(pred, obs, sdpred=NULL, weighted=T)
```

### Arguments

| | |
|---|---|
| `pred` | a numeric vector, the predicted probabilities of the reference class |
| `obs` | the vector of observations, a categorical variable with 2-4 levels |
| `sdpred` | either NULL or a vector containing the standard deviations of every estimate |
| `weighted` | a boolean, if FALSE, the unweighted logloss is calculated. By default, the weighted cross entropy is calculated. |

### Details

if sdpred contains the standard deviations for each estimated probability, then a lower bound of the log loss is returned.

### Value

a numeric value: cross entropy / log loss for comparison of classifiers. The smaller, the better.

### Author(s)

Ann-Kristin Becker

## Examples

```
#observations
obs<-as.factor(c("A","A","B"))
#correct prediction
pred1<-c(1,1,0)
#wrong prediction
pred2<-c(0,0,1)

cross.en(pred=pred1, obs=obs) #small
cross.en(pred=pred2, obs=obs) #large

#prediction of only majority class
pred3<-c(1,1,1)
#prediction of only minority class
pred4<-c(0,0,0)

cross.en(pred=pred3, obs=obs, weighted=TRUE)
cross.en(pred=pred4, obs=obs, weighted=TRUE)
#both equal (as weighted)

cross.en(pred=pred3, obs=obs, weighted=FALSE)
cross.en(pred=pred4, obs=obs, weighted=FALSE)
#unweighted, majority class is favored
```

---

## `discretize.dens` *discretize.dens*

---

### Description

density approximative discretization. Significant peaks in the density are determined and used as starting points for k-means based discretization. If only one peak is present, distribution quartiles are used for binning.

### Usage

```
discretize.dens(data, graph=F, title="Density-approxmative Discretization",
rename.level=F, return.all=T, cluster=F, seed=NULL)
```

### Arguments

| | |
|---|---|
| `data` | a vector containing the data that may be discretized |
| `graph` | a boolean value, if TRUE, the density and the determined binning are plotted |
| `title` | a title for the plot |
| `rename.level` | a boolean value, if TRUE, factor levels are replaced by integers 1:n |
| `return.all` | a boolean value, if FALSE, only the discretized data are returned. |
| `cluster` | a boolean value, if data is a cluster variable and may already be discrete or not |
| `seed` | a random seed number |

## Value

| | |
|---|---|
| `discretized` | the discretized data |
| `levels` | the factor levels |
| `optima` | the x and y coordinates of the determined peaks |

## Author(s)

Ann-Kristin Becker

## Examples

```
testdata = c(rnorm(100,-3,1), rnorm(100,3,1))
d<-discretize.dens(testdata, graph=TRUE)
summary(d$discretized)
```

---

`groupbn` *groupbn*

---

## Description

creates groupbn object (determines an initial clustering based on a hierarchy with target variable and 'separated' variables separated, learns a Bayesian network from grouped data and saves discretization and pca parameters)

## Usage

```
groupbn(hierarchy, k, target, separate=NULL, separate.as.roots=FALSE,
X.quanti=NULL, X.quali=NULL, struct.alg="hc", boot=TRUE,
discretize=TRUE, arc.thresh=NULL,
debug=FALSE, R=100, seed=NULL)
```

## Arguments

| | |
|---|---|
| `hierarchy` | a cluster object from ClustOfVar. |
| `k` | a positive integer number, the number of initial clusters. |
| `target` | a string, the name of the target variable. |
| `separate` | a vector of strings, names of variables that should be separated from the groups, such as age, sex,... |
| `separate.as.roots` | a boolean; if TRUE separated variables are used as roots in the network. Can be ignored if separate is empty. |
| `X.quanti` | a numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns). |
| `X.quali` | a categorical matrix of data, or an object that can be coerced to such a matrix (such as a character vector, a factor or a data frame with all factor columns). |

| | |
|---|---|
| `struct.alg` | structure learning algorithm according to bnlearn |
| `arc.thresh` | threshold for bootstrap arcs |
| `discretize` | a boolean, if a network variables should be discretized before network learning |
| `boot` | boolean, if TRUE, a bootstrap based network averaging approach is used |
| `debug` | a boolean, if TRUE, debugging messages are printed |
| `R` | number of bootstrap replicates for model averaging, default is 100 |
| `seed` | a random seed number |

## Value

an object of class groupbn

| | |
|---|---|
| `bn` | a Bayesian Network structure of bn class from bnlearn. |
| `fit` | a Bayesian Network with fitted parameters of bn.fit class from bnlearn. |
| `X.quanti` | a data.frame containing only the quantitative variables. |
| `X.quali` | a data.frame containing only the qualitative variables. |
| `grouping` | a vector of positive integers, giving the cluster assignment. |
| `k` | the number of clusters. |
| `group.data` | a data.frame containing the cluster representants. |
| `target` | a string, the name of the target variable. |
| `separate` | a vector of strings, names of variables that should be separated from the groups. |
| `pca.param` | the PCAmix used to determine the cluster representants. |
| `disc.param` | the cutpoints used to discretize the cluster representants. |
| `score` | Different prediction scores for the target variable using the fitted network. |

## Author(s)

Ann-Kristin Becker

## References

Becker A-K, Dörr M, Felix SB, Frost F, Grabe HJ, Lerch MM, et al. (2021) From heterogeneous healthcare data to disease-specific biomarker networks: A hierarchical Bayesian network approach. PLoS Comput Biol 17(2): e1008735. https://doi.org/10.1371/journal.pcbi.1008735

## See Also

`groupbn_refinement`

## Examples

```
#load example data
data(wine)
wine.test<-wine[wine$Soil%in%c("Reference", "Env1"),1:29]
wine.test$Soil<-factor(wine.test$Soil)
levels(wine.test$Soil)<-c("0", "1")

#cluster data
hierarchy<-hclustvar(X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2])

#Learn group network among 5 clusters with "Soil" as target variable
wine.groupbn<-groupbn(hierarchy, k=5, target="Soil", separate=NULL,
X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2], seed=321)

#Plot network
plot(wine.groupbn)
```

---

## `groupbn.output.table` *groupbn.output.table*

---

### Description

Create an output table with clusters and included variables with similarity scores

### Usage

```
groupbn.output.table(res, with.scores=TRUE)
```

### Arguments

| | |
|---|---|
| `res` | gn object |
| `with.scores` | if TRUE, similarity scores of every cluster member to the cluster center are added to the table |

### Value

a table with one column per group, similarity scores to cluster centers are calculated for each variable

### Author(s)

Ann-Kristin Becker

### See Also

`groupbn` `groupbn_refinement`

## Examples

```
data("wine.groupbn.refined")
df<-groupbn.output.table(wine.groupbn.refined)
```

---

## `groupbn.vis.html.plot` *groupbn.vis.html.plot*

---

### Description

Create an interactive html network object with visNet (displaying similarity scores and number of variables in a score)

### Usage

```
groupbn.vis.html.plot(res, df=NULL, save.file=TRUE, save.name=NULL,
hierarchical=FALSE, nodecolor.all="#E0F3F8",
nodecolor.special="cornflowerblue", main=NULL)
```

### Arguments

| | |
|---|---|
| `res` | a groupbn object |
| `df` | output from output.table if already calculated, otherwise the same table is calculated internally |
| `save.file` | boolean; if TRUE a html file is produced |
| `save.name` | name for saving html object, date is additionally used |
| `hierarchical` | boolean; if TRUE the network is plotted with a hierarchical layout |
| `nodecolor.all` | a color for "normal" nodes |
| `nodecolor.special` | a color for the target variable and all separated nodes, if any. |
| `main` | optionally a title for the plot |

### Details

Plots an interactive network plot using visNetwork package

### Value

an html widget of class visNetwork

### Author(s)

Ann-Kristin Becker

### See Also

`groupbn` `groupbn_refinement`

## Examples

```
data("wine.groupbn.refined")
groupbn.vis.html.plot(wine.groupbn.refined, hierarchical=TRUE, save.file=FALSE)
```

---

## `groupbn_refinement` *groupbn_refinement*

---

### Description

Adaptive Refinement of a group Bayesian Network using hierarchical Clustering

### Usage

```
groupbn_refinement(res, hierarchy, refinement.part="mb", restart=0, perturb=1,
max.step=10, max.min=Inf, R=100,
return.all=FALSE, arc.thresh=NULL, debug=FALSE, seed=NULL)
```

### Arguments

| | |
|---|---|
| `res` | an object of class groupbn |
| `hierarchy` | a cluster object from ClustOfVar |
| `refinement.part` | "mb", "mb2", "arc.confid" or "all", selects if the refinement steps should be done only within the markov blanket of the target variable (mb), within the second-order markov blanket (mb2), in all clusters with an arcconfidence to target >0 (arc.confid) or within all clusters (all). Default: "mb" |
| `restart` | a positive integer number, the number of restarts |
| `perturb` | a positive integer number, the number of perturbations (splits) in each restart |
| `max.step` | a positive integer number, the maximal number of refinement steps, default is 10 |
| `max.min` | a positive integer number, the maximal run time in minutes, default is unlimited |
| `R` | number of bootstrap replicates for model averaging, default is 100 |
| `return.all` | a boolean, if TRUE, the output is a whole list of group models, if FALSE, the output is only the best-scoring model. |
| `arc.thresh` | threshold for bootstrap arcs |
| `debug` | a boolean, if TRUE, debugging messages are printed |
| `seed` | a random seed number |

### Details

Based on a variable grouping, data are aggregated and a Bayesian network is learned. The target variable is kept separated during this procedure, so that the resulting network model can be used for risk prediction and classification. Starting from a coarse group network, groups are iteratively refined to smaller groups. The heuristic refinement happens downwards along the dendrogram, and stops, if it no longer improves the predictive performance of the model. The refinement part is implemented using a hill-climbing procedure.

## Value

returns an object of class groupbn

## Author(s)

Ann-Kristin Becker

## References

Becker A-K, Dörr M, Felix SB, Frost F, Grabe HJ, Lerch MM, et al. (2021) From heterogeneous healthcare data to disease-specific biomarker networks: A hierarchical Bayesian network approach. PLoS Comput Biol 17(2): e1008735. https://doi.org/10.1371/journal.pcbi.1008735

## See Also

`groupbn` `groupbn.output.table`

## Examples

```
#load example data
data(wine)
wine.test<-wine[wine$Soil%in%c("Reference", "Env1"),1:29]
wine.test$Soil<-factor(wine.test$Soil)
levels(wine.test$Soil)<-c("0", "1")

#cluster data
hierarchy<-hclustvar(X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2])

#Learn group network among 5 clusters with "Soil" as target variable
wine.groupbn<-groupbn(hierarchy, k=5, target="Soil", separate=NULL,
X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2], seed=321)

#Do one refinement step
#Set max.step higher to optimize completely
wine.groupbn.refined<-groupbn_refinement(wine.groupbn, hierarchy,
refinement.part="mb", max.step = 1, seed=321)

#Plot refined network
plot(wine.groupbn.refined)
```

---

`groupbn_refine_manually` *groupbn_refine_manually*

---

## Description

Based on a GroupBN, a cluster can be selected manually, that is split and the refined model is learned.

## Usage

```
groupbn_refine_manually(res, hierarchy, refine, arc.thresh=NULL,
R=100, debug=FALSE, seed=NULL)
```

## Arguments

| | |
|---|---|
| `res` | an object of class groupbn |
| `hierarchy` | a cluster object from ClustOfVar |
| `refine` | name of group to be refined |
| `arc.thresh` | threshold for bootstrap arcs |
| `R` | number of bootstrap replicates for model averaging, default is 100 |
| `debug` | a boolean, if TRUE, debugging messages are printed |
| `seed` | a random seed number |

## Value

returns an object of class groupbn

## Author(s)

Ann-Kristin Becker

## Examples

```
#load example data
data(wine)
wine.test<-wine[wine$Soil%in%c("Reference", "Env1"),1:29]
wine.test$Soil<-factor(wine.test$Soil)
levels(wine.test$Soil)<-c("0", "1")

#cluster data
hierarchy<-hclustvar(X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2])

#Learn group network among 5 clusters with "Soil" as target variable
wine.groupbn<-groupbn(hierarchy, k=5, target="Soil", separate=NULL,
X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2], seed=321)

#Refine cluster 2
wine.groupbn.refined<-groupbn_refine_manually(wine.groupbn, hierarchy,
refine = "cl2", seed=321)

#Plot refined network
plot(wine.groupbn.refined)
```

## is.groupbn *is.groupbn*

### Description

Generic function for groupbn objects

### Usage

```
is.groupbn(x)
```

### Arguments

| | |
|---|---|
| x | an object of class groupbn |

### Value

A boolean; TRUE if x is of class groupbn, FALSE otherwise.

### Author(s)

Ann-Kristin Becker

### See Also

groupbn

### Examples

```
data("wine.groupbn.refined")
is.groupbn(wine.groupbn.refined)
```

## plot.groupbn *plot.groupbn*

### Description

generic plot function for class groupbn

### Usage

```
## S3 method for class 'groupbn'
plot(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | an object of class groupbn |
| `...` | further arguments |

### Details

Plot the group bayesian network structure

### Value

No return value, called for plotting

### Author(s)

Ann-Kristin Becker

### See Also

`groupbn`

### Examples

```
data("wine.groupbn.refined")
plot(wine.groupbn.refined)
```

---

## `predict.groupbn` *predict.groupbn*

---

### Description

Predict the target variable from a group Bayesian network

### Usage

```
## S3 method for class 'groupbn'
predict(object, X.quanti, X.quali, rename.level=FALSE, return.data=FALSE,
new.fit=FALSE, debug=FALSE, ...)
```

### Arguments

| | |
|---|---|
| `object` | An object of class groupbn generated by the functions groupbn or groupbn_refinement |
| `X.quanti` | quantitative variables |
| `X.quali` | qualitative variables |
| `rename.level` | a boolean; if TRUE, all levels of categorical variables are renamed by integers. Default is FALSE. |
| `return.data` | a boolean; if TRUE, a list with predictions and group.data is returned instead of only predicitions. Default is FALSE. |

| | |
|---|---|
| `new.fit` | a boolean; if TRUE, the parameters are newly fit using the test data. |
| `debug` | a boolean, if TRUE, debugging messages are printed |
| `...` | further arguments |

### Value

Returns a dataframe with a column of predictions and a column of the target data. If the target is discrete, class probabilities are returned. Otherwise continuous scores are returned. If return.data is TRUE, additionally the transformed group data are returned.

### Author(s)

Ann-Kristin Becker

### Examples

```
#load example data
data(wine)
wine.test<-wine[wine$Soil%in%c("Reference", "Env1"),1:29]
wine.test$Soil<-factor(wine.test$Soil)
levels(wine.test$Soil)<-c("0", "1")

data(wine.groupbn.refined)
predict(wine.groupbn.refined, X.quanti=wine.test[,3:29], X.quali=wine.test[,1:2])
```

## `print.groupbn` *print.groupbn*

### Description

This is a method for the function print for objects of the class groupbn.

### Usage

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

### Arguments

| | |
|---|---|
| `x` | An object of class groupbn generated by the functions groupbn or groupbn_refinement |
| `...` | further arguments |

### Value

No return value, prints a description of the object

### Author(s)

Ann-Kristin Becker

## See Also

groupbn

## Examples

```
data("wine.groupbn.refined")
print(wine.groupbn.refined)
```

---

`wine.groupbn.refined` *wine.groupbn.refined*

---

## Description

A refined group Bayesian network with 8 groups learned from dataset 'wine'.

## Usage

```
data("wine.groupbn.refined")
```

## Format

group Bayesian network (class 'groupbn')

name of target variable: Soil number of groups: 8 achieved scoring: F1: 0.92 ; Precision: 1 ; Recall: 0.86 ; AUC-PR: 1 ; AUC-ROC: 1 ; cross-entr.: 1.43; BIC (netw.): -77.21

name description "$bn" "Bayesian network structure" "$fit" "fitted Bayesian network (multinomial)" "$arc.confid" "arc confidence" "$X.quali" "qualitative variables in a data.frame" "$X.quanti" "quantitative variables in a data.frame" "$grouping" "group memberships" "$k" "number of groups of initial grouping" "$group.data" "group representatives used for network inference" "$target" "name of target variable" "$separate" "name of any other separated variables" "$pca.param" "pca parameters of each group" "$disc.param" "discretization intervals of each group" "$score" "cross entropy and additional scoring information"

## Examples

```
data(wine.groupbn.refined)
```

# Index