Package ‘bnmonitor’

October 12, 2022

Type Package

Title An Implementation of Sensitivity Analysis in Bayesian Networks

Version 0.1.3

Description An implementation of sensitivity and robustness methods in Bayesian networks in R. It includes methods to perform parameter variations via a variety of co-variation schemes, to compute sensitivity functions and to quantify the dissimilarity of two Bayesian networks via distances and divergences. It further includes diagnostic methods to assess the goodness of fit of a Bayesian networks to data, including global, node and parent-child monitors. References: H. Chan, A. Darwiche (2002) <doi:10.1613/jair.967>; C. Goergen, M. Leonelli (2020) <ArXiv:1809.10794>; M. Leonelli, R. Ramanathan, R.L. Wilkerson (2021) <ArXiv:2107.11785>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

Imports bnlearn, dplyr, ggplot2, gRain, gRbase, graphics, purrr, qgraph, RColorBrewer, reshape2, rlang, tidyr

Suggests testthat, knitr, rmarkdown

URL https://manueleleonelli.github.io/bnmonitor/,
https://github.com/manueleleonelli/bnmonitor

NeedsCompilation no

Author Manuele Leonelli [aut, cre],
Ramsiya Ramanathan [aut],
Rachel Wilkerson [aut]

Maintainer Manuele Leonelli <manuele.leonelli@ie.edu>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2022-04-20 08:22:30 UTC

R topics documented:

bn2 — *Integration with* bn.fit *objects from* bnlearn

Description

Functions that transform an object of class bn.fit and bn.fit.gnet (a Gaussian Bayesian network) to objects of class GBN or CI.

Usage

```
bn2gbn(bnfit)

bn2ci(bnfit)
```

Arguments

bnfit	object of class bn.fit.

Value

The function bn2gbn returns an object of class GBN consisting of a list with entries:

- order: An ordering of the nodes according to the graph.
- mean: The mean vector of the Gaussian distribution.
- covariance: The covariance matrix of the Gaussian distribution.

The function bn2ci returns an object of class CI consisting of the same list as GBN, but with the additional entry cond_ind. cond_ind is a list where each entry consists of A, B and C corresponding to the conditional independence statements A independent of B given C embedded by the network.

bnmonitor — *bnmonitor: A package for sensitivity analysis and robustness in Bayesian networks*

Description

Sensitivity and robustness analysis for Bayesian networks.

Details

bnmonitor provides functions to perform sensitivity analysis for both discrete Bayesian networks (DBNs) and Gaussian Bayesian networks (GBNs).

In the discrete case, it provides three categories of functions: co-variation schemes, dissimilarity measures and sensitivity related functions.

In the continuous case, both standard and model-preserving methods are available for perturbation of the mean vector and the co-variance matrix.

bnmonitor further provides function to perform robustness studies in DBNs to verify how well a network fits a specific dataset.

DBNs - Robustness

The available functions for robustness are:

- *Node monitors* (node_monitor): contribution of each vertex to the overall log-likelihood of the model.
- *Observation's influence* (influential_obs): difference in the log-likelihood of a model learnt with the full dataset and with all but one observation.
- *Final node monitors* (node_monitor): marginal and conditional node monitors to assess the fit of a vertex distribution to the full dataset.
- *Sequential node monitors* (seq_node_monitor): marginal and conditional node monitors for a specific vertex only using sequentially subsets of the dataset.
- *Sequential parent-child monitor* (seq_pa_ch_monitor): parent-child node monitor for a specific vertex and a specific configuration of its parents using sequentially subsets of the dataset.

DBNs - Co-variation schemes

The available co-variation schemes are:

- *Uniform co-variation scheme* (uniform_covar): distributes the probability mass to be co-varied uniformly among the co-varying parameters.
- *Proportional co-variation scheme* (proportional_covar): distributes the probability mass to be co-varied in the same proportion as in the original Bayesian network.
- *Order-preserving co-variation scheme* (orderp_covar):distributes the to be co-varied probability mass among the co-varying parameters so that the original order of parameters is preserved.

DBNs - Dissimilarity measures

The dissimilarity measures quantify the difference between a Bayesian network and its update after parameter variation.
The available dissimilarity measures are:

- *Chan-Darwiche distance* (CD)
- *Kullback-Leibler divergence* (KL)

DBNs - Sensitivity functions

The available functions for sensitivity analysis are:

- *Sensitivity function* (sensitivity): returns a certain probability of interest given a parameter change. Evidence can be considered.
- *Sensitivity query* (sensquery): returns the parameter changes needed to get a certain probability of interest. Evidence can be considered.

GBNs - Model-Preserving matrices

The available functions to construct model-preserving co-variation matrices are:

- *Total co-variation matrix* (`total_covar_matrix`).
- *Partial co-variation matrix* (`partial_covar_matrix`).
- *Row-based co-variation matrix* (`row_covar_matrix`).
- *Column-based co-variation matrix* (`col_covar_matrix`).

GBNs - Mean and Covariance variations

The available functions to perturb the distribution of a GBN are:

- *Mean variations* (`mean_var`).
- *Standard covariance variations* (`covariance_var`).
- *Model-preserving covariance variations* (`model_pres_cov`).

GBNs - Dissimilarity measures

The available dissimilarity measures are:

- Frobenius norm (`Fro`).
- Jeffrey's distance (`Jeffreys`).
- Kullback-Leibler divergence (`KL`).
- Upper bound to the KL divergence (`KL_bounds`).

`cachexia` *Bayesian networks for a cachexia study*

Description

Continuous Bayesian networks comparing the dependence of metabolomics for people who suffer and do not suffer of Cachexia

Usage

```
cachexia_gbn

cachexia_ci

control_gbn

control_ci

cachexia_data
```

Format

Continuous Bayesian networks over six metabolomics: Adipate (A), Betaine (B), Fumarate (F), Glucose (GC), Glutamine (GM) and Valine (V). The networks `cachexia_gbn` and `cachexia_ci` are for people suffering of cachexia and of class `GBN` and `CI` respectively. The networks `control_gbn` and `control_ci` are for people not suffering of cachexia and of class `GBN` and `CI` respectively. The original dataset is stored in `cachexia_data`.

An object of class `CI` of length 4.

An object of class `GBN` of length 3.

An object of class `CI` of length 4.

An object of class `data.table` (inherits from `data.frame`) with 77 rows and 7 columns.

Source

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32. NULL

`CD` *CD-distance*

Description

Chan-Darwiche (CD) distance between a Bayesian network and its update after parameter variation.

Usage

```
CD(
  bnfit,
  node,
  value_node,
  value_parents,
  new_value,
  covariation = "proportional"
)
```

Arguments

`bnfit`	object of class `bn.fit`.
`node`	character string. Node of which the conditional probability distribution is being changed.
`value_node`	character string. Level of `node`.
`value_parents`	character string. Levels of `node`'s parents. The levels should be defined according to the order of the parents in `bnfit[[node]][["parents"]]`. If `node` has no parents, then it should be set to `NULL`.

`new_value` numeric vector with elements between 0 and 1. Values to which the parameter should be updated. It can take a specific value or more than one. In the case of more than one value, these should be defined through a vector with an increasing order of the elements. `new_value` can also be set to the character string `all`: in this case a sequence of possible parameter changes ranging from 0.05 to 0.95 is considered.

`covariation` character string. Co-variation scheme to be used for the updated Bayesian network. Can take values `uniform`, `proportional`, `orderp`, `all`. If equal to `all`, uniform, proportional and order-preserving co-variation schemes are used. Set by default to `proportional`.

Details

The Bayesian network on which parameter variation is being conducted should be expressed as a `bn.fit` object. The name of the node to be varied, its level and its parent's levels should be specified. The parameter variation specified by the function is:

P (node = `value_node` | parents = `value_parents`) = `new_value`

The CD distance between two probability distributions P and P' defined over the same sample space $\mathcal{Y}$ is defined as

$$CD(P, P') = \log \max_{y \in \mathcal{Y}} \left(\frac{P(y)}{P'(y)} \right) - \log \min_{y \in \mathcal{Y}} \left(\frac{P(y)}{P'(y)} \right)$$

Value

The function `CD` returns a dataframe including in the first column the variations performed, and in the following columns the corresponding CD distances for the chosen co-variation schemes.

References

Chan, H., & Darwiche, A. (2005). A distance measure for bounding probabilistic belief change. International Journal of Approximate Reasoning, 38(2), 149-174.

Renooij, S. (2014). Co-variation for sensitivity analysis in Bayesian networks: Properties, consequences and alternatives. International Journal of Approximate Reasoning, 55(4), 1022-1042.

See Also

`KL.bn.fit`

Examples

```
CD(synthetic_bn, "y2", "1", "2", "all", "all")
CD(synthetic_bn, "y1", "2", NULL, 0.3, "all")
```

chds *Christchurch Health and Development Study*

Description

Simulated data and Bayesian networks from the Christchurch Health and Development Study

Usage

```
chds

chds_bn

chds_bn.fit
```

Format

The dataframe `chds` includes 500 observations randomly simulated from the `bn.fit` object `chds_bn.fit`. It has four variables:

- **Social**: family's social background with levels `"High"` and `"Low"`
- **Economic**: family's economic status with levels `"High"` and `"Low"`
- **Events**: number of family life events with levels `"High"`, `"Average"` and `"Low"`
- **Admission**: hospital admission of the child with levels `"yes"` and `"no"`
- **statistics**: mark out of 100 for statistics

`chds_bn` is an object of class `bn` including the MAP Bayesian network from Barclay et al. (2013) and `chds_bn.fit` is an object of class `bn.fit` including the probabilities from the same article.

An object of class `data.frame` with 500 rows and 4 columns.

An object of class `bn` of length 3.

An object of class `bn.fit` (inherits from `bn.fit.dnet`) of length 4.

References

Fergusson, D. M., Horwood, L. J., & Shannon, F. T. (1986). Social and family factors in childhood hospital admission. Journal of Epidemiology & Community Health, 40(1), 50-58.

Barclay, L. M., Hutton, J. L., & Smith, J. Q. (2013). Refining a Bayesian network using a chain event graph. International Journal of Approximate Reasoning, 54(9), 1300-1309.

covariance_var — *Standard variation of the covariance matrix*

Description

Computation of an updated GBN object after a variation of the covariance matrix.

Usage

```
covariance_var(gbn, entry, delta)
```

Arguments

gbn	object of class GBN.
entry	a vector of length 2 specifying the entry of the covariance matrix to vary.
delta	additive variation coefficient for the entry of the co-variation matrix given in entry.

Details

Let the original Bayesian network have a Normal distribution $\mathcal{N}(\mu, \Sigma)$ and let entry be equal to (i, j). For a variation of the covariance matrix by an amount δ, a variation matrix D is constructed as

$$D_{k,l} = \begin{cases} \delta & \text{if } k = i, l = j \\ \delta & \text{if } l = i, k = j \\ 0 & \text{otherwise} \end{cases}$$

Then the resulting distribution after the variation is $\mathcal{N}(\mu, \Sigma + D)$, assuming $\Sigma + D$ is positive semi-definite.

Value

If the resulting covariance is positive semi-definite, covariance_var returns an object of class GBN with an updated covariance matrix. Otherwise it returns an object of class npsd.gbn, which has the same components of GBN but also has a warning entry specifying that the covariance matrix is not positive semi-definite.

References

Gómez-Villegas, M. A., Maín, P., & Susi, R. (2007). Sensitivity analysis in Gaussian Bayesian networks using a divergence measure. Communications in Statistics—Theory and Methods, 36(3), 523-539.

Gómez-Villegas, M. A., Main, P., & Susi, R. (2013). The effect of block parameter perturbations in Gaussian Bayesian networks: Sensitivity and robustness. Information Sciences, 222, 439-458.

See Also

mean_var, model_pres_cov

Examples

```
covariance_var(synthetic_gbn,c(1,1),3)
covariance_var(synthetic_gbn,c(1,2),-0.4)
```

`covariation` *Co-variation schemes*

Description

Functions that return an updated Bayesian network using the proportional, uniform and order-preserving co-variation schemes.

Usage

```
proportional_covar(bnfit, node, value_node, value_parents, new_value)

orderp_covar(bnfit, node, value_node, value_parents, new_value)

uniform_covar(bnfit, node, value_node, value_parents, new_value)
```

Arguments

`bnfit`	object of class `bn.fit`.
`node`	character string. Node of which the conditional probability distribution is being changed.
`value_node`	character string. Level of `node`.
`value_parents`	character string. Levels of `node`'s parents. The levels should be defined according to the order of the parents in `bnfit[[node]][["parents"]]`. If `node` has no parents, then it should be set to `NULL`.
`new_value`	numeric value between 0 and 1. Value to which the parameter should be updated.

Details

The Bayesian network on which parameter variation is being conducted should be expressed as a `bn.fit` object. The name of the node to be varied, its level and its parent's levels should be specified. The parameter variation specified by the function is:

P (`node` = `value_node` | parents = `value_parents`) = `new_value`

For `orderp_covar`, if two or more parameters in a distribution have the same value, the order is given by the one in the respective conditional probability table. Furthermore, the parameter associated to the largest probability of the conditional probability law cannot be varied.

Value

An object of class `bn.fit` with updated probabilities.

References

Laskey, K. B. (1995). Sensitivity analysis for probability assessments in Bayesian networks. IEEE Transactions on Systems, Man, and Cybernetics, 25(6), 901-909.

Renooij, S. (2014). Co-variation for sensitivity analysis in Bayesian networks: Properties, consequences and alternatives. International journal of approximate reasoning, 55(4), 1022-1042.

Leonelli, M., & Riccomagno, E. (2018). A geometric characterisation of sensitivity analysis in monomial models. arXiv preprint arXiv:1901.02058.

Examples

```
proportional_covar(synthetic_bn, "y3", "2", c("2","1"), 0.3)
uniform_covar(synthetic_bn, "y2", "1", "2", 0.3)
orderp_covar(synthetic_bn, "y1", "1", NULL, 0.3)
```

---

covariation_matrix *Co-variation matrices*

Description

Construction of model-preserving co-variation matrices for objects of class `CI`.

Usage

```
total_covar_matrix(ci, entry, delta)

col_covar_matrix(ci, entry, delta)

partial_covar_matrix(ci, entry, delta)

row_covar_matrix(ci, entry, delta)
```

Arguments

`ci`	object of class `CI`.
`entry`	a vector of length two specifying the entry of the covariance matrix to vary.
`delta`	multiplicative variation coefficient for the entry of the covariance matrix given in `entry`.

Details

Functions to compute total, partial, row-based and column-based co-variation matrices to ensure the conditional independences of the original Bayesian network hold after a variation. If no co-variation is required for model-preservation the functions return a matrix filled with ones (no co-variation).

Value

A co-variation matrix of the same size of the covariance matrix of `CI`.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

`model_pres_cov`

Examples

```
total_covar_matrix(synthetic_ci,c(1,1),0.3)
total_covar_matrix(synthetic_ci,c(1,2),0.3)
partial_covar_matrix(synthetic_ci,c(1,2),0.3)
row_covar_matrix(synthetic_ci,c(1,2),0.3)
col_covar_matrix(synthetic_ci,c(1,2),0.3)
```

`diabetes` *Pima Indian Diabetes Data*

Description

Discretized version of the widely-used Pima Indians Diabetes Database

Format

A dataframe with 392 observations on the following 9 binary variables:

- **PREG**: number of times pregnant (low/high)
- **GLUC**: plasma glucose concentration (low/high)
- **PRES**: diastolic blood pressure (low/high)
- **TRIC**: triceps skin fold thickness (low/high)
- **INS**: 2-hour serum insulin (low/high)
- **MASS**: body mass index (low/high)
- **PED**: diabetes pedigree function (low/high)
- **AGE**: age (low/high)
- **DIAB**: test for diabetes (neg/pos)

Source

These data have been taken from the UCI Repository Of Machine Learning Databases. We chose this dataset because it best showcases the function of our monitors. However, we acknowledge that this data is used here without the consent of or compensation for the original Akimel O'odham participants.

`final_node_monitor` *Final node monitors*

Description

Marginal and conditional node monitors over the last observation of the data for all vertices of a Bayesian network using the full dataset

Usage

```
final_node_monitor(dag, df)
```

Arguments

`dag`	an object of class `bn` from the `bnlearn` package
`df`	a base R style dataframe

Details

Consider a Bayesian network over variables $Y_1, \ldots, Y_m$ and suppose a dataset $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ has been observed, where $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{im})$ and y_{ij} is the i-th observation of the j-th variable. Let p_n denote the marginal density of Y_j after the first $n-1$ observations have been processed. Define

$$E_n = \sum_{k=1}^{K} p_n(d_k) \log(p_n(d_k)),$$

$$V_n = \sum_{k=1}^{K} p_n(d_k) \log^2(p_n(d_k)) - E_n^2,$$

where $(d_1, \ldots, d_K)$ are the possible values of Y_j. The marginal node monitor for the vertex Y_j is defined as

$$Z_j = \frac{-\log(p_n(y_{ij})) - E_n}{\sqrt{V_n}}.$$

Higher values of Z_j can give an indication of a poor model fit for the vertex Y_j.

The conditional node monitor for the vertex Y_j is defined as

$$Z_j = \frac{-\log(p_n(y_{nj}|y_{n1}, \ldots, y_{n(j-1)}, y_{n(j+1)}, \ldots, y_{nm})) - E_n}{\sqrt{V_n}},$$

where E_n and V_n are computed with respect to $p_n(y_{nj}|y_{n1}, \ldots, y_{n(j-1)}, y_{n(j+1)}, \ldots, y_{nm})$. Again, higher values of Z_j can give an indication of a poor model fit for the vertex Y_j.

Value

A dataframe including the names of the vertices, the marginal node monitors and the conditional node monitors. It also return two plots where vertices with a darker color have a higher marginal z-score or conditional z-score, respectively, in absolute value.

References

Cowell, R. G., Dawid, P., Lauritzen, S. L., & Spiegelhalter, D. J. (2006). Probabilistic networks and expert systems: Exact computational methods for Bayesian networks. Springer Science & Business Media.

Cowell, R. G., Verrall, R. J., & Yoon, Y. K. (2007). Modeling operational risk with Bayesian networks. Journal of Risk and Insurance, 74(4), 795-827.

See Also

`influential_obs`, `node_monitor`, `seq_node_monitor`, `seq_pa_ch_monitor`

Examples

```
final_node_monitor(chds_bn, chds[1:100,])
```

`fire_alarm` *Bayesian network on fire alarm system*

Description

`fire_alarm` is a `bn.fit` object including a Bayesian network for a fire alarm system.

Usage

```
fire_alarm
```

Format

The Bayesian network `fire_alarm` includes the following nodes:

- **Fire**: two-level factor with levels `TRUE` and `FALSE`. It indicates presence or absence of a fire.
- **Smoke**: two level-factor with levels `TRUE` and `FALSE`. It indicates presence or absence of smoke.
- **Alarm**: three level-factor with levels `TRUE`, `MALFUNCTION` and `FALSE`. It indicates if the alarm is ringing, malfunctioning or not ringing.
- **Tampering**: two level-factor with levels `TRUE` and `FALSE`. It indicates if the alarm system has been tampered or not.
- **Leaving**: two level-factor with levels `TRUE` and `FALSE`. It indicates if the building is being evacuated or not.
- **Report**: two level-factor with levels `TRUE` and `FALSE`. It indicates if the incident has been reported or not.

Source

Hei Chan, Adnan Darwiche (2002). "When do numbers really matter?". Journal of Artificial Intelligence Research 17 (265-287).

Fro — *Frobenius norm*

Description

`Fro` returns the Frobenius norm between a Bayesian network and its update after parameter variation.

Usage

```
Fro(x, ...)
```

Arguments

`x`	object of class `GBN` or `CI`.
`...`	parameters specific to the class used.

Details

The details depend on the class the method `Fro` is applied to.

Value

A dataframe whose columns depend of the class of the object.

See Also

`KL.GBN`, `KL.CI`, `Fro.CI`, `Fro.GBN`, `Jeffreys.GBN`, `Jeffreys.CI`

Fro.CI — *Frobenius norm for* `CI`

Description

`Fro.CI` returns the Frobenius norm between an object of class `CI` and its update after a model-preserving parameter variation.

Usage

```
## S3 method for class 'CI'
Fro(x, type, entry, delta, log = TRUE, ...)
```

Arguments

`x`	object of class `CI`.
`type`	character string. Type of model-preserving co-variation: either `"total"`, `"partial"`, `row`, `column` or `all`. If `all` the Frobenius norm is computed for every type of co-variation matrix.
`entry`	a vector of length 2 indicating the entry of the covariance matrix to vary.
`delta`	numeric vector with positive elements, including the variation parameters that act multiplicatively.
`log`	boolean value. If `TRUE`, the logarithm of the Frobenius norm is returned. Set by default to `TRUE`.
`...`	additional arguments for compatibility.

Details

Computation of the Frobenius norm between a Bayesian network and its updated version after a model-preserving variation.

Value

A dataframe including in the first column the variations performed, and in the following columns the corresponding Frobenius norms for the chosen model-preserving co-variations.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

`KL.GBN`, `KL.CI`, `Fro.GBN`, `Jeffreys.GBN`, `Jeffreys.CI`

Examples

```
Fro(synthetic_ci,"total",c(1,1),seq(0.9,1.1,0.01))
Fro(synthetic_ci,"partial",c(1,4),seq(0.9,1.1,0.01))
Fro(synthetic_ci,"column",c(1,2),seq(0.9,1.1,0.01))
Fro(synthetic_ci,"row",c(3,2),seq(0.9,1.1,0.01))
```

Fro.GBN *Frobenius norm for* GBN

Description

Fro.GBN returns the Frobenius norm between between an object of class GBN and its update after a standard parameter variation.

Usage

```
## S3 method for class 'GBN'
Fro(x, entry, delta, log = TRUE, ...)
```

Arguments

x	object of class GBN.
entry	a vector of length 2 indicating the entry of the covariance matrix to vary.
delta	numeric vector, including the variation parameters that act additively.
log	boolean value. If TRUE, the logarithm of the Frobenius norm is returned. Set by default to TRUE.
...	additional arguments for compatibility.

Details

Computation of the Frobenius norm between a Bayesian network and the additively perturbed Bayesian network, where the perturbation is either to the mean vector or to the covariance matrix. The Frobenius norm is not computed for perturbations of the mean since it is always equal to zero.

Value

A dataframe including in the first column the variations performed and in the second column the corresponding Frobenius norm.

See Also

KL.GBN, KL.CI, Fro.CI, Jeffreys.GBN, Jeffreys.CI

Examples

```
Fro(synthetic_gbn,c(3,3),seq(-1,1,0.1))
```

`global_monitor` *Global monitor*

Description

Negative marginal log-likelihood of the model

Usage

```
global_monitor(dag, df, alpha = "default")
```

Arguments

dag	an object of class bn from the bnlearn package
df	a base R style dataframe
alpha	single integer. By default, number of max levels in df

Value

A numerical value

References

Cowell, R. G., Dawid, P., Lauritzen, S. L., & Spiegelhalter, D. J. (2006). Probabilistic networks and expert systems: Exact computational methods for Bayesian networks. Springer Science & Business Media.

Cowell, R. G., Verrall, R. J., & Yoon, Y. K. (2007). Modeling operational risk with Bayesian networks. Journal of Risk and Insurance, 74(4), 795-827.

See Also

node_monitor, influential_obs, final_node_monitor, seq_node_monitor, seq_pa_ch_monitor

Examples

```
global_monitor(chds_bn, chds, 3)
```

`influential_obs` *Influential observations*

Description

Influence of a single observation to the global monitor

Usage

```
influential_obs(dag, data, alpha = "default")
```

Arguments

dag	an object of class bn from the bnlearn package
data	a base R style dataframe
alpha	single integer. By default, the number of max levels in data

Details

Consider a Bayesian network over variables $Y_1, \dots, Y_m$ and suppose a dataset $(\boldsymbol{y}_1, \dots, \boldsymbol{y}_n)$ has been observed, where $\boldsymbol{y}_i = (y_{i1}, \dots, y_{im})$ and y_{ij} is the i-th observation of the j-th variable. Define $\boldsymbol{y}_{-i} = (\boldsymbol{y}_1, \dots, \boldsymbol{y}_{i-1}, \boldsymbol{y}_{i+1}, \dots, \boldsymbol{y}_n)$. The influence of an observation to the global monitor is defined as

$$|\log(p(\boldsymbol{y}_1, \dots, \boldsymbol{y}_n)) - \log(p(\boldsymbol{y}_{-i}))|.$$

High values of this index denote observations that highly contribute to the likelihood of the model.

Value

A vector including the influence of each observation.

See Also

influential_obs, node_monitor, seq_node_monitor, seq_pa_ch_monitor

Examples

```
influential_obs(chds_bn, chds[1:100,], 3)
```

`Jeffreys` *Jeffreys Divergence*

Description

`Jeffreys` returns the Jeffreys divergence between a continuous Bayesian network and its update after parameter variation.

Usage

```
Jeffreys(x, ...)
```

Arguments

`x`	object of class `bn.fit`, `GBN` or `CI`.
`...`	parameters specific to the class used.

Details

The details depend on the class the method `Jefrreys` is applied to.

Value

A dataframe whose columns depend of the class of the object.

See Also

`KL.GBN`, `KL.CI`, `Fro.CI`, `Fro.GBN`, `Jeffreys.GBN`, `Jeffreys.CI`

`Jeffreys.CI` *Jeffreys Divergence for* `CI`

Description

`Jeffreys.CI` returns the Jeffreys divergence between an object of class `CI` and its update after a model-preserving parameter variation.

Usage

```
## S3 method for class 'CI'
Jeffreys(x, type, entry, delta, ...)
```

Arguments

x	object of class CI.
type	character string. Type of model-preserving co-variation: either "total", "partial", row,column or all. If all the Jeffreys divergence is computed for every type of co-variation matrix.
entry	a vector of length 2 indicating the entry of the covariance matrix to vary.
delta	numeric vector with positive elements, including the variation parameters that act multiplicatively.
...	additional arguments for compatibility.

Details

Computation of the Jeffreys divergence between a Bayesian network and its updated version after a model-preserving variation.

Value

A dataframe including in the first column the variations performed, and in the following columns the corresponding Jeffreys divergences for the chosen model-preserving co-variations.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

KL.GBN, KL.CI, Fro.CI, Fro.GBN, Jeffreys.GBN

Examples

```
Jeffreys(synthetic_ci,"total",c(1,1),seq(0.9,1.1,0.01))
Jeffreys(synthetic_ci,"partial",c(1,4),seq(0.9,1.1,0.01))
Jeffreys(synthetic_ci,"column",c(1,2),seq(0.9,1.1,0.01))
Jeffreys(synthetic_ci,"row",c(3,2),seq(0.9,1.1,0.01))
Jeffreys(synthetic_ci,"all",c(3,2),seq(0.9,1.1,0.01))
```

Jeffreys.GBN   *Jeffreys Divergence for* GBN

Description

Jeffreys.GBN returns the Jeffreys divergence between an object of class GBN and its update after a standard parameter variation.

Usage

```
## S3 method for class 'GBN'
Jeffreys(x, where, entry, delta, ...)
```

Arguments

`x`	object of class `GBN`.
`where`	character string: either `mean` or `covariance` for variations of the mean vector and covariance matrix respectively.
`entry`	if `where == "mean"`, `entry` is the index of the entry of the mean vector to vary. If `where == "covariance"`, entry is a vector of length 2 indicating the entry of the covariance matrix to vary.
`delta`	numeric vector, including the variation parameters that act additively.
`...`	additional arguments for compatibility.

Details

Computation of the Jeffreys divergence between a Bayesian network and the additively perturbed Bayesian network, where the perturbation is either to the mean vector or to the covariance matrix.

Value

A dataframe including in the first column the variations performed and in the second column the corresponding Jeffreys divergences.

References

Goergen, C., & Leonelli, M. (2018). Model-preserving sensitivity analysis for families of Gaussian distributions. arXiv preprint arXiv:1809.10794.

See Also

`KL.GBNKL.CI`, `Fro.CI`, `Fro.GBN`, `Jeffreys.CI`

Examples

```
Jeffreys(synthetic_gbn,"mean",2,seq(-1,1,0.1))
Jeffreys(synthetic_gbn,"covariance",c(3,3),seq(-1,1,0.1))
```

KL *KL Divergence*

Description

KL returns the Kullback-Leibler (KL) divergence between a Bayesian network and its update after parameter variation.

Usage

```
KL(x, ...)
```

Arguments

x	object of class bn.fit, GBN or CI.
...	parameters specific to the class used.

Details

The details depend on the class the method KL is applied to.

Value

A dataframe whose columns depend of the class of the object.

See Also

KL.GBN, KL.CI, Fro.CI, Fro.GBN, Jeffreys.GBN, Jeffreys.CI

KL.bn.fit *KL Divergence for* bn.fit

Description

KL.bn.fit returns the Kullback-Leibler (KL) divergence between a Bayesian network and its update after parameter variation.

Usage

```
## S3 method for class 'bn.fit'
KL(
  x,
  node,
  value_node,
  value_parents,
  new_value,
  covariation = "proportional",
  ...
)
```

Arguments

`x`	object of class `bn.fit`.
`node`	character string. Node of which the conditional probability distribution is being changed.
`value_node`	character string. Level of `node`.
`value_parents`	character string. Levels of `node`'s parents. The levels should be defined according to the order of the parents in `bnfit[[node]][["parents"]]`. If `node` has no parents, then it should be set to `NULL`.
`new_value`	numeric vector with elements between 0 and 1. Values to which the parameter should be updated. It can take a specific value or more than one. In the case of more than one value, these should be defined through a vector with an increasing order of the elements. `new_value` can also be set to the character string `all`: in this case a sequence of possible parameter changes ranging from 0.05 to 0.95 is considered.
`covariation`	character string. Co-variation scheme to be used for the updated Bayesian network. Can take values `uniform`, `proportional`, `orderp`, `all`. If equal to `all`, uniform, proportional and order-preserving co-variation schemes are used. Set by default to `proportional`.
`...`	additional parameters to be added to the plot.

Details

The Bayesian network on which parameter variation is being conducted should be expressed as a `bn.fit` object. The name of the node to be varied, its level and its parent's levels should be specified. The parameter variation specified by the function is:

P (`node` = `value_node` | parents = `value_parents`) = `new_value`

Value

A dataframe with the varied parameter and the KL divergence for different co-variation schemes. If `plot` = `TRUE` the function returns a plot of the KL divergences.

References

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. The annals of mathematical statistics, 22(1), 79-86.

Leonelli, M., Goergen, C., & Smith, J. Q. (2017). Sensitivity analysis in multilinear probabilistic models. Information Sciences, 411, 84-97.

See Also

CD

Examples

```
KL(synthetic_bn, "y2", "1", "2", "all", "all")
KL(synthetic_bn, "y1", "2", NULL, 0.3, "all")
```

KL.CI *KL Divergence for* CI

Description

`KL.CI` returns the Kullback-Leibler (KL) divergence between an object of class `CI` and its update after a model-preserving parameter variation.

Usage

```
## S3 method for class 'CI'
KL(x, type, entry, delta, ...)
```

Arguments

`x`	object of class `CI`.
`type`	character string. Type of model-preserving co-variation: either `"total"`, `"partial"`, `row`,`column` or `all`. If `all` the KL divergence is computed for every type of co-variation matrix.
`entry`	a vector of length 2 indicating the entry of the covariance matrix to vary.
`delta`	numeric vector with positive elements, including the variation parameters that act multiplicatively.
`...`	additional arguments for compatibility.

Details

Computation of the KL divergence between a Bayesian network and its updated version after a model-preserving variation.

Value

A dataframe including in the first column the variations performed, and in the following columns the corresponding KL divergences for the chosen model-preserving co-variations.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

KL.GBN, Fro.CI, Fro.GBN, Jeffreys.GBN, Jeffreys.CI

Examples

```
KL(synthetic_ci, "total", c(1,1), seq(0.9,1.1,0.01))
KL(synthetic_ci, "partial", c(1,4), seq(0.9,1.1,0.01))
KL(synthetic_ci, "column", c(1,2), seq(0.9,1.1,0.01))
KL(synthetic_ci, "row", c(3,2), seq(0.9,1.1,0.01))
KL(synthetic_ci, "all", c(3,2), seq(0.9,1.1,0.01))
```

KL.GBN *KL Divergence for* GBN

Description

KL.GBN returns the Kullback-Leibler (KL) divergence between an object of class GBN and its update after a standard parameter variation.

Usage

```
## S3 method for class 'GBN'
KL(x, where, entry, delta, ...)
```

Arguments

x	object of class GBN.
where	character string: either mean or covariance for variations of the mean vector and covariance matrix respectively.
entry	if where == "mean", entry is the index of the entry of the mean vector to vary. If where == "covariance", entry is a vector of length 2 indicating the entry of the covariance matrix to vary.
delta	numeric vector, including the variation parameters that act additively.
...	additional arguments for compatibility.

Details

Computation of the KL divergence between a Bayesian network and the additively perturbed Bayesian network, where the perturbation is either to the mean vector or to the covariance matrix.

Value

A dataframe including in the first column the variations performed and in the second column the corresponding KL divergences.

References

Gómez-Villegas, M. A., Maín, P., & Susi, R. (2007). Sensitivity analysis in Gaussian Bayesian networks using a divergence measure. Communications in Statistics—Theory and Methods, 36(3), 523-539.

Gómez-Villegas, M. A., Main, P., & Susi, R. (2013). The effect of block parameter perturbations in Gaussian Bayesian networks: Sensitivity and robustness. Information Sciences, 222, 439-458.

See Also

KL.CI, Fro.CI, Fro.GBN, Jeffreys.GBN, Jeffreys.CI

Examples

```
KL(synthetic_gbn,"mean",2,seq(-1,1,0.1))
KL(synthetic_gbn,"covariance",c(3,3),seq(-1,1,0.1))
```

KL_bounds *Bounds for the KL-divergence*

Description

Computation of the bounds of the KL-divergence for variations of each parameter of a CI object.

Usage

```
KL_bounds(ci, delta)
```

Arguments

ci	object of class CI.
delta	multiplicative variation coefficient for the entry of the covariance matrix given in entry.

Details

Let Σ be the covariance matrix of a Gaussian Bayesian network with n vertices. Let D and Δ be variation matrices acting additively on Σ. Let also $\tilde{\Delta}$ be a model-preserving co-variation matrix. Denote with Y and $\tilde{Y}$ the original and the perturbed random vectors. Then for a standard sensitivity analysis

$$KL(\tilde{Y}||Y) \leq 0.5n \max\left\{f(\lambda_{\max}(D\Sigma^{-1})), f(\lambda_{\min}(D\Sigma^{-1}))\right\}$$

whilst for a model-preserving one

$$KL(\tilde{Y}||Y) \leq 0.5n \max\left\{f(\lambda_{\max}(\tilde{\Delta}\circ\Delta)), f(\lambda_{\min}(\tilde{\Delta}\circ\Delta))\right\}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and the smallest eigenvalues, respectively, $f(x) = \ln(1+x) - x/(1+x)$ and $\circ$ denotes the Schur or element-wise product.

Value

A dataframe including the KL-divergence bound for each co-variation scheme (model-preserving and standard) and every entry of the covariance matrix. For variations leading to non-positive semidefinite matrix, the dataframe includes a `NA`.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

`KL.CI`, `KL.CI`

Examples

```
KL_bounds(synthetic_ci,1.05)
```

`mathmarks` *Math Marks Data*

Description

Marks out of 100 for 88 students taking examinations in mechanics (C), vectors (C), algebra (0), analysis (O) and statistics (O), where C indicates closed and O indicates open book examination.

Usage

```
data(mathmarks)
```

Format

A dataframe with 88 observations on the following 5 variables

- **mechanics**: mark out of 100 for mechanics
- **vectors**: mark out of 100 for vectors
- **algebra**: mark out of 100 for algebra
- **analysis**: mark out of 100 for analysis
- **statistics**: mark out of 100 for statistics

Source

Mardia, K. V., Kent, J. T. and Bibby, J. M. (1979) Multivariate Analysis. London: Academic Press.

---

mean_var *Standard variation of the mean vector*

Description

Computation of an updated GBN object after a variation of the mean vector.

Usage

```
mean_var(gbn, entry, delta)
```

Arguments

gbn	object of class GBN.
entry	an index specifying the entry of the mean vector to vary.
delta	additive variation coefficient for the entry of the mean vector given in entry.

Details

Let the original Bayesian network have a Normal distribution $\mathcal{N}(\mu, \Sigma)$ and let entry be equal to i. Let μ_i be the i-th entry of μ. For a variation of the mean by an amount δ the resulting distribution is $\mathcal{N}(\mu', \Sigma)$, where μ' is equal to μ except for the i-th entry which is equal to $\mu + \delta$.

Value

An object of class GBN with an updated mean vector.

References

Gómez-Villegas, M. A., Maín, P., & Susi, R. (2007). Sensitivity analysis in Gaussian Bayesian networks using a divergence measure. Communications in Statistics—Theory and Methods, 36(3), 523-539.

Gómez-Villegas, M. A., Main, P., & Susi, R. (2013). The effect of block parameter perturbations in Gaussian Bayesian networks: Sensitivity and robustness. Information Sciences, 222, 439-458.

See Also

covariance_var

Examples

```
mean_var(synthetic_gbn,2,3)
```

---

model_pres_cov *Model-Preserving co-variation*

Description

Model-preserving co-variation for objects of class `CI`.

Usage

```
model_pres_cov(ci, type, entry, delta)
```

Arguments

ci	object of class `CI`.
type	character string. Type of model-preserving co-variation: either `"total"`, `"partial"`, `row` or `column`.
entry	a vector of length two specifying the entry of the covariance matrix to vary.
delta	multiplicative variation coefficient for the entry of the covariance matrix given in `entry`.

Details

Let the original Bayesian network have a Normal distribution $\mathcal{N}(\mu,\Sigma)$ and let `entry` be equal to (i,j). For a multiplicative variation of the covariance matrix by an amount δ, a variation matrix Δ is constructed as

$$\Delta_{k,l} = \begin{cases} \delta & \text{if } k=i, l=j \\ \delta & \text{if } l=i, k=j \\ 0 & \text{otherwise} \end{cases}$$

A co-variation matrix $\tilde{\Delta}$ is then constructed and the resulting distribution after the variation is $\mathcal{N}(\mu,\tilde{\Delta}\circ\Delta\circ\Sigma)$, assuming $\tilde{\Delta}\circ\Delta\circ\Sigma$ is positive semi-definite and where $\circ$ denotes the Schur (or element-wise) product. The matrix $\tilde{\Delta}$ is so constructed to ensure that all conditional independence in the original Bayesian networks are retained after the parameter variation.

Value

If the resulting covariance is positive semi-definite, `model_pres_cov` returns an object of class `CI` with an updated covariance matrix. Otherwise it returns an object of class `npsd.ci`, which has the same components of `CI` but also has a warning entry specifying that the covariance matrix is not positive semi-definite.

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

See Also

covariance_var, covariation_matrix

Examples

```
model_pres_cov(synthetic_ci,"partial",c(1,3),1.1)
model_pres_cov(synthetic_ci,"partial",c(1,3),0.9)
model_pres_cov(synthetic_ci,"total",c(1,2),0.5)
model_pres_cov(synthetic_ci,"row",c(1,3),0.98)
model_pres_cov(synthetic_ci,"column",c(1,3),0.98)
```

---

node_monitor — *Node monitor*

Description

Contribution of each vertex of a Bayesian network to the global monitor

Usage

```
node_monitor(dag, df, alpha = "default")
```

Arguments

dag	an object of class bn from the bnlearn package
df	a base R style dataframe
alpha	single integer. By default, number of max levels in df

Details

Consider a Bayesian network over variables $Y_1, \ldots, Y_m$ and suppose a dataset $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ has been observed, where $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{im})$ and y_{ij} is the i-th observation of the j-th variable. The global monitor is defined as the negative log-likelihood of the model, i.e.

$$-\log(p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)) = -\sum_{j=1}^{m}\sum_{i=1}^{n}\log(p(y_{ij}|\pi_{ij})),$$

where π_{ij} is the value of the parents of Y_j for the i-th observation. The contribution of the j-th vertex to the global monitor is thus

$$-\sum_{i=1}^{n}\log(p(y_{ij}|\pi_{ij})).$$

Value

A dataframe including the name of the vertices and the contribution of the vertices to the global monitor. It also returns a plot where vertices with higher contributions in absolute value are darker.

References

Cowell, R. G., Dawid, P., Lauritzen, S. L., & Spiegelhalter, D. J. (2006). Probabilistic networks and expert systems: Exact computational methods for Bayesian networks. Springer Science & Business Media.

Cowell, R. G., Verrall, R. J., & Yoon, Y. K. (2007). Modeling operational risk with Bayesian networks. Journal of Risk and Insurance, 74(4), 795-827.

See Also

global_monitor, influential_obs, final_node_monitor, seq_node_monitor, seq_pa_ch_monitor

Examples

```
node_monitor(chds_bn, chds, 3)
```

---

plot *Plotting methods*

Description

Plotting methods for outputs of bnmonitor functions

Usage

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

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

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

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

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

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

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

## S3 method for class 'final_node_monitor'
plot(x, which, ...)

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

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

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

Arguments

x	The output of node_monitor.
...	for compatibility
which	select the monitor to plot, either "marginal" or "conditional" (for output of `node_monitor` only).

Value

A plot specific to the object it is applied to.

`print`	*Printing methods*

Description

Printing methods for outputs of `bnmonitor` functions

Usage

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

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

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

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

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

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

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

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

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

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

Arguments

x	an appropriate object
...	for compatibility

Value

Printing specific to the object it is applied to.

---

psd_check *Check for positive semi-definiteness after a perturbation*

Description

psd_check returns a boolean to determine if the covariance matrix after a perturbation is positive semi-definite.

Usage

```
psd_check(x, ...)

## S3 method for class 'GBN'
psd_check(x, entry, delta, ...)

## S3 method for class 'CI'
psd_check(x, type, entry, delta, ...)
```

Arguments

`x`	object of class `GBN` or `CI`.
`...`	additional arguments for compatibility.
`entry`	a vector of length 2 indicating the entry of the covariance matrix to vary.
`delta`	numeric vector, including the variation parameters that act additively.
`type`	character string. Type of model-preserving co-variation: either `total`, `partial`, `row`, `column` or `all`. If `all`, the Frobenius norms are computed for every type of co-variation matrix.

Details

The details depend on the class the method `psd_check` is applied to.

Let Σ be the covariance matrix of a Gaussian Bayesian network and let D be a perturbation matrix acting additively. The perturbed covariance matrix $\Sigma + D$ is positive semi-definite if

$$\rho(D) \leq \lambda_{\min}(\Sigma)$$

where $\lambda_{\min}$ is the smallest eigenvalue end ρ is the spectral radius.

Value

A dataframe including the variations performed and the check for positive semi-definiteness.

Methods (by class)

- `GBN`: `psd_check` for objects `GBN`
- `CI`: `psd_check` for objects `CI`

References

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

Examples

```
psd_check(synthetic_gbn,c(2,4),-3)
psd_check(synthetic_gbn,c(2,3),seq(-1,1,0.1))
psd_check(synthetic_ci,"partial",c(2,4),0.95)
psd_check(synthetic_ci,"all",c(2,3),seq(0.9,1.1,0.01))
```

`sensitivity` *Sensitivity function*

Description

sensitivity returns the sensitivity function for a probabilistic query of interest with respect to a parameter change defined by the user.

Usage

```
sensitivity(
  bnfit,
  interest_node,
  interest_node_value,
  evidence_nodes = NULL,
  evidence_states = NULL,
  node,
  value_node,
  value_parents,
  new_value,
  covariation = "proportional"
)
```

Arguments

- `bnfit` object of class bn.fit.
- `interest_node` character string. Node of the probability query of interest.
- `interest_node_value` character string. Level of interest_node.
- `evidence_nodes` character string. Evidence nodes. If NULL no evidence is considered. Set by default to NULL.
- `evidence_states` character string. Levels of evidence_nodes. If NULL no evidence is considered. If evidence_nodes="NULL", evidence_states should be set to NULL. Set by default to NULL.
- `node` character string. Node of which the conditional probability distribution is being changed.
- `value_node` character string. Level of node.
- `value_parents` character string. Levels of node's parents. The levels should be defined according to the order of the parents in bnfit[[node]][["parents"]]. If node has no parents, then should be set to NULL.
- `new_value` numeric vector with elements between 0 and 1. Values to which the parameter should be updated. It can take a specific value or more than one. For more than one value, these should be defined through a vector with an increasing order of the elements. new_value can also take as value the character string all: in

this case a sequence of possible parameter changes ranging from 0.05 to 0.95 is considered.

covariation
: character string. Co-variation scheme to be used for the updated Bayesian network. Can take values `uniform`, `proportional`, `orderp`, `all`. If equal to `all`, uniform, proportional and order-preserving co-variation schemes are considered. Set by default to `proportional`.

Details

The Bayesian network on which parameter variation is being conducted should be expressed as a bn.fit object. The name of the node to be varied, its level and its parent's level should be specified. The parameter variation specified by the function is:

P (`node` = `value_node` | `parents` = `value_parents`) = `new_value`

and the probabilistic query of interest is:

P (`interest_node` = `interest_node_value` | `evidence_nodes` = `evidence_states`)

Value

A dataframe with the varied parameter values and the output probabilities for the co-variation schemes selected. If `plot = TRUE` the function also returns a plot of the sensitivity function.

References

Coupé, V. M., & Van Der Gaag, L. C. (2002). Properties of sensitivity analysis of Bayesian belief networks. Annals of Mathematics and Artificial Intelligence, 36(4), 323-356.

Leonelli, M., Goergen, C., & Smith, J. Q. (2017). Sensitivity analysis in multilinear probabilistic models. Information Sciences, 411, 84-97.

See Also

`covariation`, `sensquery`

`sensquery` *Sensitivity of probability query*

Description

`sensquery` returns, for a given change in a probability of interest, the parameters' changes to achieve it together with the corresponding CD distances.

Usage

```
sensquery(
  bnfit,
  interest_node,
  interest_node_value,
  new_value,
  evidence_nodes = NULL,
  evidence_states = NULL
)
```

Arguments

- `bnfit` — object of class `bn.fit`.
- `interest_node` — character string. Node of the probability query of interest.
- `interest_node_value` — character string. Level of `interest_node`.
- `new_value` — numeric value between 0 and 1. New value of the probability of interest.
- `evidence_nodes` — character string. Evidence nodes. Set by default to `NULL`.
- `evidence_states` — character string. Levels of `evidence_nodes`. If `NULL` no evidence is considered. If `evidence_nodes="NULL"`, `evidence_states` should be set to `NULL`. Set by default to `NULL`.

Details

The Bayesian network should be expressed as a `bn.fit` object. The name of the node of the probability of interest, its level and the new value should be specified. Evidence could be also indicated. The probability of interest is specified as follows:

P (`interest_node` = `interest_node_value` | `evidence_nodes` = `evidence_states`) = `new_value`

Only the proportional co-variation scheme is used.

Value

A dataframe with the following columns: `node` - the vertex of the proposed change; `Value node` - the level of node to be changed; `Value parents` - the levels of the parent variables of node; `Original value` - the original probability defined by `Node`, `Value node` and `Value parents`; `Suggested change` - the new proposed value for the probability defined by `Node`, `Value node` and `Value parents`; `CD distance` - the CD distance between the original and new network with the `Suggested change`.

References

Chan, H., & Darwiche, A. (2002). When do numbers really matter?. Journal of artificial intelligence research, 17, 265-287.

See Also

`sensitivity`

Examples

```
sensquery(synthetic_bn,"y3", "3", 0.3)
```

`seq_node_monitor` *Sequential node monitors*

Description

Sequential marginal and conditional node monitors for a vertex of a Bayesian network.

Usage

```
seq_marg_monitor(dag, df, node.name)

seq_cond_monitor(dag, df, node.name)
```

Arguments

`dag`	an object of class bn from the `bnlearn` package
`df`	a base R style dataframe
`node.name`	node over which to compute the monitor

Details

Consider a Bayesian network over variables $Y_1, \dots, Y_m$ and suppose a dataset $(\boldsymbol{y}_1, \dots, \boldsymbol{y}_n)$ has been observed, where $\boldsymbol{y}_i = (y_{i1}, \dots, y_{im})$ and y_{ij} is the i-th observation of the j-th variable. Let p_i denote the marginal density of Y_j after the first $i-1$ observations have been processed. Define

$$E_i = \sum_{k=1}^{K} p_i(d_k)\log(p_i(d_k)),$$

$$V_i = \sum_{k=1}^{K} p_i(d_k)\log^2(p_i(d_k)) - E_i^2,$$

where $(d_1, \dots, d_K)$ are the possible values of Y_j. The sequential marginal node monitor for the vertex Y_j is defined as

$$Z_{ij} = \frac{-\sum_{k=1}^{i}\log(p_k(y_{kj})) - \sum_{k=1}^{i} E_k}{\sqrt{\sum_{k=1}^{i} V_k}}.$$

Values of Z_{ij} such that $|Z_{ij}| > 1.96$ can give an indication of a poor model fit for the vertex Y_j after the first i-1 observations have been processed.

The sequential conditional node monitor for the vertex Y_j is defined as

$$Z_{ij} = \frac{-\sum_{k=1}^{i}\log(p_k(y_{kj}|y_{k1}, \dots, y_{k(j-1)}, y_{k(j+1)}, \dots, y_{km})) - \sum_{k=1}^{i} E_k}{\sqrt{\sum_{k=1}^{i} V_k}},$$

where E_k and V_k are computed with respect to $p_k(y_{kj}|y_{k1}, \dots, y_{k(j-1)}, y_{k(j+1)}, \dots, y_{km})$. Again, values of Z_{ij} such that $|Z_{ij}| > 1.96$ can give an indication of a poor model fit for the vertex Y_j.

Value

A vector including the scores Z_{ij}, either marginal or conditional.

References

Cowell, R. G., Dawid, P., Lauritzen, S. L., & Spiegelhalter, D. J. (2006). Probabilistic networks and expert systems: Exact computational methods for Bayesian networks. Springer Science & Business Media.

Cowell, R. G., Verrall, R. J., & Yoon, Y. K. (2007). Modeling operational risk with Bayesian networks. Journal of Risk and Insurance, 74(4), 795-827.

See Also

`influential_obs`, `node_monitor`, `seq_node_monitor`, `seq_pa_ch_monitor`

Examples

```
seq_marg_monitor(chds_bn, chds[1:100,], "Events")
seq_marg_monitor(chds_bn, chds[1:100,], "Admission")
```

`seq_pa_ch_monitor` *Sequential parent-child node monitors*

Description

Sequential node monitor for a vertex of a Bayesian network for a specific configuration of its parents

Usage

```
seq_pa_ch_monitor(dag, df, node.name, pa.names, pa.val, alpha = "default")
```

Arguments

`dag`	an object of class `bn` from the `bnlearn` package
`df`	a base R style dataframe
`node.name`	node over which to compute the monitor
`pa.names`	vector including the names of the parents of `node.name`
`pa.val`	vector including the levels of `pa.names` considered
`alpha`	single integer. By default, the number of max levels in `df`

Details

Consider a Bayesian network over variables $Y_1, \dots, Y_m$ and suppose a dataset $(\boldsymbol{y}_1, \dots, \boldsymbol{y}_n)$ has been observed, where $\boldsymbol{y}_i = (y_{i1}, \dots, y_{im})$ and y_{ij} is the i-th observation of the j-th variable. Consider a configuration π_j of the parents and consider the sub-vector $\boldsymbol{y}' = (\boldsymbol{y}'_1, \dots, \boldsymbol{y}'_{N'})$ of $(\boldsymbol{y}_1, \dots, \boldsymbol{y}_n)$ including observations where the parents of Y_j take value π_j only. Let $p_i(\cdot|\pi_j)$ be the conditional distribution of Y_j given that its parents take value π_j after the first i-1 observations have been processed. Define

$$E_i = \sum_{k=1}^{K} p_i(d_k|\pi_j) \log(p_i(d_k|\pi_j)),$$

$$V_i = \sum_{k=1}^{K} p_i(d_k|\pi_j) \log^2(p_i(d_k|\pi_j)) - E_i^2,$$

where $(d_1, \dots, d_K)$ are the possible values of Y_j. The sequential parent-child node monitor for the vertex Y_j and parent configuration π_j is defined as

$$Z_{ij} = \frac{-\sum_{k=1}^{i} \log(p_k(y'_{kj}|\pi_j)) - \sum_{k=1}^{i} E_k}{\sqrt{\sum_{k=1}^{i} V_k}}.$$

Values of Z_{ij} such that $|Z_{ij}| > 1.96$ can give an indication of a poor model fit for the vertex Y_j after the first i-1 observations have been processed.

Value

A vector including the scores Z_{ij}.

References

Cowell, R. G., Dawid, P., Lauritzen, S. L., & Spiegelhalter, D. J. (2006). Probabilistic networks and expert systems: Exact computational methods for Bayesian networks. Springer Science & Business Media.

Cowell, R. G., Verrall, R. J., & Yoon, Y. K. (2007). Modeling operational risk with Bayesian networks. Journal of Risk and Insurance, 74(4), 795-827.

See Also

influential_obs, node_monitor, seq_node_monitor, seq_pa_ch_monitor

Examples

```
seq_pa_ch_monitor(chds_bn, chds, "Events", "Social", "High", 3)
```

synthetic_bn *A synthetic Bayesian network*

Description

synthetic_bn is a bn.fit object for a simple Bayesian network involving three variables.

Usage

```
synthetic_bn
```

Format

The Bayesian network bnsens_example comprehends the following nodes:

- **y1**: three-level factor with levels *1*, *2*, *3*.
- **y2**: three-level factor with levels *1*, *2*, *3*.
- **y3**: three-level factor with levels *1*, *2*, *3*.

Source

Manuele Leonelli, Eva Riccomagno (2018). "A geometric characterisation of sensitivity analysis in monomial models". https://arxiv.org/abs/1901.02058

synthetic_cbn *A synthetic continuous Bayesian network*

Description

A synthetic continuous Bayesian network

Usage

```
synthetic_gbn

synthetic_ci
```

Format

A continuous Bayesian networks over four variables ("y1", "y2", "y3", "y4"), embedding the statement "y1" independent of "y3" given "y2". The Bayesian network is available both as an object of class GBN and as an object of class CI.

An object of class GBN of length 3.

An object of class CI of length 4.

Source

C. Görgen & M. Leonelli (2020), Model-preserving sensitivity analysis for families of Gaussian distributions. Journal of Machine Learning Research, 21: 1-32.

`travel` *Bayesian network on travel survey*

Description

`travel` is a `bn.fit` object for the Bayesian network on a traveling preferences survey.

Usage

```
travel
```

Format

The Bayesian network `travel` includes the following nodes:

- **A**: three-level factor with levels `young`, `adult`, `old`. It indicates the age of an individual.
- **S**: two level-factor with levels `M` (male) and `F` (female). It indicates the gender of an individual.
- **E**: two level-factor with levels `high` and `uni`. It indicates the education level of an individual.
- **O**: two level-factor with levels `emp` (employed) and `self` (self-employed). It indicates the occupation of an individual.
- **R**: two level-factor with levels `small` and `big`. It indicates the size of the residence of an individual.
- **T**: three level-factor with levels `car`, `train` and `other`. It indicates the preferred mean of transportation by an individual.

Source

Scutari, M., & Denis, J. B. (2014). Bayesian networks: with examples in R. Chapman and Hall/CRC.

Index