Package 'meteorits'

July 26, 2019

Type Package

Title Mixtures-of-experts modeling for complex and non-normal distributions ('MEteorits')

Version 0.1.0

Description meteorits is an open source toolbox (available in R and in Matlab) containing several original and flexible mixtures-of-experts models to model, cluster and classify heterogeneous data in many complex situations where the data are distributed according to non-normal, possibly skewed distributions, and when they might be corrupted by atypical observations. The toolbox also contains sparse mixture-of-experts models for high-dimensional data.

URL https://github.com/fchamroukhi/MEteorits

License GPL (>= 3)

Depends R (>= 2.10)

Imports pracma,
methods,
stats,
Rcpp

Suggests knitr,
rmarkdown

LinkingTo Rcpp,
RcppArmadillo

Collate meteorits-package.R
RcppExports.R
logsumexp.R
utils.R
sampleUnivNMoE.R
sampleUnivSNMoE.R
sampleUnivSTMoE.R
sampleUnivTMoE.R
FData.R
ParamSNMoE.R
ParamStMoE.R
ParamTMoE.R
ParamNMoE.R
StatSNMoE.R
StatStMoE.R
StatTMoE.R
StatNMoE.R

ModelSNMoE.R
ModelStMoE.R
ModelTMoE.R
ModelNMoE.R
emSNMoE.R
emStMoE.R
emTMoE.R
emNMoE.R
data-simulatedstructureddata.R

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 6.1.1

R topics documented:

`emNMoE` *emNMoE implements the EM algorithm to fit a NMoE model.*

Description

emNMoE implements the maximum-likelihood parameter estimation of a NMoE model by the Expectation-Maximization (EM) algorithm.

Usage

```
emNMoE(X, Y, K, p = 3, q = 1, n_tries = 1, max_iter = 1500,
  threshold = 1e-06, verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

`X`	Numeric vector of length *n* representing the covariates/inputs $x_1, \dots, x_m$.
`Y`	Numeric vector of length *n* representing the observed response/output $y_1, \dots, y_m$.
`K`	The number of expert components.
`p`	The order of the polynomial regression for the expert regressors network.
`q`	The dimension of the logistic regression for the gating network. For the purpose of segmentation, it must be set to 1.
`n_tries`	Number of times EM algorithm will be launched with different initializations. The solution providing the highest log-likelihood will be returned.
`max_iter`	The maximum number of iterations for the EM algorithm.
`threshold`	A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the EM as stopping criteria.
`verbose`	A logical value indicating whether values of the log-likelihood should be printed during EM iterations.
`verbose_IRLS`	A logical value indicating whether values of the criterion optimized by IRLS should be printed at each step of the EM algorithm.

Details

emNMoE function function implements the EM algorithm for the NMoE model. This functions starts with an initialization of the parameters done by the method `initParam` of the class ParamNMoE, then it alternates between a E-Step (method of the class StatNMoE) and a M-Step (method of the class ParamNMoE) until convergence (until the absolute difference of log-likelihood between two steps of the EM algorithm is less than the `threshold` parameter).

Value

Th EM algorithm returns an object of class ModelNMoE.

See Also

ModelNMoE, ParamNMoE, StatNMoE

---

emSNMoE — *emSNMoE implements the ECM algorithm to fit a SNMoE model.*

Description

emSNMoE implements the maximum-likelihood parameter estimation of a SNMoE model by the Expectation Conditional Maximization (ECM) algorithm.

Usage

```
emSNMoE(X, Y, K, p = 3, q = 1, n_tries = 1, max_iter = 1500,
  threshold = 1e-06, verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

- `X` — Numeric vector of length *n* representing the covariates/inputs $x_1, \ldots, x_m$.
- `Y` — Numeric vector of length *n* representing the observed response/output $y_1, \ldots, y_m$.
- `K` — The number of expert components.
- `p` — The order of the polynomial regression for the expert regressors network.
- `q` — The dimension of the logistic regression for the gating network. For the purpose of segmentation, it must be set to 1.
- `n_tries` — Number of times ECM algorithm will be launched with different initializations. The solution providing the highest log-likelihood will be returned.
- `max_iter` — The maximum number of iterations for the ECM algorithm.
- `threshold` — A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the ECM as stopping criteria.
- `verbose` — A logical value indicating whether values of the log-likelihood should be printed during ECM iterations.
- `verbose_IRLS` — A logical value indicating whether values of the criterion optimized by IRLS should be printed at each step of the ECM algorithm.

Details

emSNMoE function function implements the ECM algorithm for the SNMoE model. This functions starts with an initialization of the parameters done by the method `initParam` of the class ParamSNMoE, then it alternates between a E-Step (method of the class StatSNMoE) and a CM-Step (method of the class ParamSNMoE) until convergence (until the absolute difference of log-likelihood between two steps of the ECM algorithm is less than the `threshold` parameter).

Value

Th ECM algorithm returns an object of class ModelSNMoE.

See Also

ModelSNMoE, ParamSNMoE, StatSNMoE

---

emStMoE — *emStMoE implements the ECM algorithm to fit a StMoE model.*

Description

emStMoE implements the maximum-likelihood parameter estimation of a StMoE model by the Expectation Conditional Maximization (ECM) algorithm.

Usage

```
emStMoE(X, Y, K, p = 3, q = 1, n_tries = 1, max_iter = 1500,
  threshold = 1e-06, verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

`X`	Numeric vector of length *n* representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Numeric vector of length *n* representing the observed response/output $y_1, \ldots, y_m$.
`K`	The number of expert components.
`p`	The order of the polynomial regression for the expert regressors network.
`q`	The dimension of the logistic regression for the gating network. For the purpose of segmentation, it must be set to 1.
`n_tries`	Number of times ECM algorithm will be launched with different initializations. The solution providing the highest log-likelihood will be returned.
`max_iter`	The maximum number of iterations for the ECM algorithm.
`threshold`	A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the ECM as stopping criteria.
`verbose`	A logical value indicating whether values of the log-likelihood should be printed during ECM iterations.
`verbose_IRLS`	A logical value indicating whether values of the criterion optimized by IRLS should be printed at each step of the ECM algorithm.

Details

emStMoE function function implements the ECM algorithm for the StMoE model. This functions starts with an initialization of the parameters done by the method `initParam` of the class ParamStMoE, then it alternates between a E-Step (method of the class StatStMoE) and a CM-Step (method of the class ParamStMoE) until convergence (until the absolute difference of log-likelihood between two steps of the ECM algorithm is less than the `threshold` parameter).

Value

Th ECM algorithm returns an object of class ModelStMoE.

See Also

ModelStMoE, ParamStMoE, StatStMoE

`emTMoE` *emTMoE implements the ECM algorithm to fit a tMoE model.*

Description

emTMoE implements the maximum-likelihood parameter estimation of a tMoE model by the Expectation Conditional Maximization (ECM) algorithm.

Usage

```
emTMoE(X, Y, K, p = 3, q = 1, n_tries = 1, max_iter = 1500,
  threshold = 1e-06, verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

- `X` Numeric vector of length *n* representing the covariates/inputs $x_1, \ldots, x_m$.
- `Y` Numeric vector of length *n* representing the observed response/output $y_1, \ldots, y_m$.
- `K` The number of expert components.
- `p` The order of the polynomial regression for the expert regressors network.
- `q` The dimension of the logistic regression for the gating network. For the purpose of segmentation, it must be set to 1.
- `n_tries` Number of times ECM algorithm will be launched with different initializations. The solution providing the highest log-likelihood will be returned.
- `max_iter` The maximum number of iterations for the ECM algorithm.
- `threshold` A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the ECM as stopping criteria.
- `verbose` A logical value indicating whether values of the log-likelihood should be printed during ECM iterations.
- `verbose_IRLS` A logical value indicating whether values of the criterion optimized by IRLS should be printed at each step of the ECM algorithm.

Details

emTMoE function function implements the ECM algorithm for the tMoE model. This functions starts with an initialization of the parameters done by the method `initParam` of the class ParamTMoE, then it alternates between a E-Step (method of the class StatTMoE) and a CM-Step (method of the class ParamTMoE) until convergence (until the absolute difference of log-likelihood between two steps of the ECM algorithm is less than the `threshold` parameter).

Value

Th ECM algorithm returns an object of class ModelTMoE.

See Also

ModelTMoE, ParamTMoE, StatTMoE

`FData-class` *A Reference Class to represent a functional data set.*

Description

FData is a reference class which represents general independent and identically distributed (i.i.d.) functional objects. The data can be ordered by time (functional time series). In the last case, the field `X` represents the time.

Fields

- `X` Numeric vector of length *m*.
- `Y` Matrix of size (n, m) representing *n* functions of `X` observed at points $1, \ldots, m$.

ModelNMoE-class — *A Reference Class which represents a fitted NMoE model.*

Description

ModelNMoE represents a NMoE model for which parameters have been estimated.

Fields

`param` A ParamNMoE object. It contains the estimated values of the parameters.

`stat` A StatNMoE object. It contains all the statistics associated to the NMoE model.

See Also

ParamNMoE, StatNMoE

ModelSNMoE-class — *A Reference Class which represents a fitted SNMoE model.*

Description

ModelSNMoE represents a SNMoE model for which parameters have been estimated.

Fields

`param` A ParamSNMoE object. It contains the estimated values of the parameters.

`stat` A StatSNMoE object. It contains all the statistics associated to the SNMoE model.

See Also

ParamSNMoE, StatSNMoE

ModelStMoE-class — *A Reference Class which represents a fitted StMoE model.*

Description

ModelStMoE represents a StMoE model for which parameters have been estimated.

Fields

`param` A ParamStMoE object. It contains the estimated values of the parameters.

`stat` A StatStMoE object. It contains all the statistics associated to the StMoE model.

See Also

ParamStMoE, StatStMoE

ModelTMoE-class *A Reference Class which represents a fitted TMoE model.*

Description

ModelTMoE represents a TMoE model for which parameters have been estimated.

Fields

`param` A ParamTMoE object. It contains the estimated values of the parameters.

`stat` A StatTMoE object. It contains all the statistics associated to the TMoE model.

See Also

ParamTMoE, StatTMoE

ParamNMoE-class *A Reference Class which contains parameters of a NMoE model.*

Description

ParamNMoE contains all the parameters of a NMoE model.

Fields

`fData` FData object representing the sample.

`K` The number of mixture components.

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`nu` degree of freedom

`alpha` is the parameter vector of the logistic model with $alpha_K$ being the null vector.

`beta` is the vector of regression coefficients of component k, the updates for each of the expert component parameters consist in analytically solving a weighted Gaussian linear regression problem.

`sigma` The variances for the K mixture components.

`delta` the skewness parameter lambda (by equivalence delta)

See Also

FData

`ParamSNMoE-class` *A Reference Class which contains parameters of a SNMoE model.*

Description

ParamSNMoE contains all the parameters of a SNMoE model.

Fields

`fData` FData object representing the sample.

`K` The number of mixture components.

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`nu` degree of freedom

`alpha` is the parameter vector of the logistic model with $alpha_K$ being the null vector.

`beta` is the vector of regression coefficients of component k, the updates for each of the expert component parameters consist in analytically solving a weighted Gaussian linear regression problem.

`sigma` The variances for the *K* mixture component.

`lambda` skewness parameter

`delta` the skewness parameter lambda (by equivalence delta)

See Also

FData

`ParamStMoE-class` *A Reference Class which contains parameters of a MRHLP model.*

Description

ParamMRHLP contains all the parameters of a MRHLP model.

Fields

`fData` FData object representing the sample.

`K` The number of mixture components.

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`nu` degree of freedom

`alpha` is the parameter vector of the logistic model with $alpha_K$ being the null vector.

`beta` is the vector of regression coefficients of component k, the updates for each of the expert component parameters consist in analytically solving a weighted Gaussian linear regression problem.

`sigma` The variances for the *K* mixture component.

`lambda` skewness parameter

`delta` the skewness parameter lambda (by equivalence delta)

`nuk` degrees of freedom

Methods

`initParam(try_EM, segmental = FALSE)` Method to initialize parameters `alpha`, `beta` and `sigma`.

`MStep(statStMoE, verbose_IRLS)` Method used in the EM algorithm to learn the parameters of the StMoE model based on statistics provided by `statStMoE`.

See Also

FData

`ParamTMoE-class` *A Reference Class which contains parameters of a TMoE model.*

Description

ParamTMoE contains all the parameters of a TMoE model.

Fields

`fData` FData object representing the sample.

`K` The number of mixture components.

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`nu` degree of freedom

`alpha` is the parameter vector of the logistic model with $alpha_K$ being the null vector.

`beta` is the vector of regression coefficients of component k, the updates for each of the expert component parameters consist in analytically solving a weighted Gaussian linear regression problem.

`sigma` The variances for the *K* mixture component.

`delta` the skewness parameter lambda (by equivalence delta)

See Also

FData

`sampleUnivNMoE` *Draw a sample from a normal mixture of linear experts model.*

Description

Draw a sample from a normal mixture of linear experts model.

Usage

```
sampleUnivNMoE(alphak, betak, sigmak, x)
```

Arguments

`alphak`	The parameters of the gating network. `alphak` is a matrix of size *(q + 1, K - 1)*, with *K - 1*, the number of regressors (experts) and *q* the order of the logistic regression
`betak`	Matrix of size *(p + 1, K)* representing the regression coefficients of the experts network.
`sigmak`	Vector of length *K* giving the standard deviations of the experts network.
`x`	A vector og length *n* representing the inputs (predictors).

Value

A list with the output variable `y` and statistics.

- y Vector of length *n* giving the output variable.
- zi A vector of size *n* giving the hidden label of the expert component generating the i-th observation. Its elements are $zi[i] = k$, if the i-th observation has been generated by the k-th expert.
- z A matrix of size *(n, K)* giving the values of the binary latent component indicators Z_{ik} such that $Z_{ik} = 1$ iff $Z_i = k$.
- stats A list whose elements are:
  - Ey_k Matrix of size *(n, K)* giving the conditional expectation of Yi the output variable given the value of the hidden label of the expert component generating the ith observation *zi = k*, and the value of predictor *X = xi*.
  - Ey Vector of length *n* giving the conditional expectation of Yi given the value of predictor *X = xi*.
  - Vary_k Vector of length *k* representing the conditional variance of Yi given *zi = k*, and *X = xi*.
  - Vary Vector of length *n* giving the conditional expectation of Yi given *X = xi*.

`sampleUnivSNMoE` *Draw a sample from a skew-normal mixture of linear experts model.*

Description

Draw a sample from a skew-normal mixture of linear experts model.

Usage

```
sampleUnivSNMoE(alphak, betak, sigmak, lambdak, x)
```

Arguments

`alphak`	The parameters of the gating network. `alphak` is a matrix of size *(q + 1, K - 1)*, with *K - 1*, the number of regressors (experts) and *q* the order of the logistic regression
`betak`	Matrix of size *(p + 1, K)* representing the regression coefficients of the experts network.
`sigmak`	Vector of length *K* giving the standard deviations of the experts network.
`lambdak`	Vector of length *K* giving the skewness parameter of each experts.
`x`	A vector og length *n* representing the inputs (predictors).

Value

A list with the output variable y and statistics.

- y Vector of length *n* giving the output variable.
- zi A vector of size *n* giving the hidden label of the expert component generating the i-th observation. Its elements are $zi[i] = k$, if the i-th observation has been generated by the k-th expert.
- z A matrix of size *(n, K)* giving the values of the binary latent component indicators Z_{ik} such that $Z_{ik} = 1$ iff $Z_i = k$.
- stats A list whose elements are:
  - Ey_k Matrix of size *(n, K)* giving the conditional expectation of Yi the output variable given the value of the hidden label of the expert component generating the ith observation *zi = k*, and the value of predictor *X = xi*.
  - Ey Vector of length *n* giving the conditional expectation of Yi given the value of predictor *X = xi*.
  - Vary_k Vector of length *k* representing the conditional variance of Yi given *zi = k*, and *X = xi*.
  - Vary Vector of length *n* giving the conditional expectation of Yi given *X = xi*.

`sampleUnivSTMoE` *Draw a sample from a univariate skew-t mixture.*

Description

Draw a sample from a univariate skew-t mixture.

Usage

```
sampleUnivSTMoE(alphak, betak, sigmak, lambdak, nuk, x)
```

Arguments

`alphak`	The parameters of the gating network. `alphak` is a matrix of size *(q + 1, K - 1)*, with *K - 1*, the number of regressors (experts) and *q* the order of the logistic regression
`betak`	Matrix of size *(p + 1, K)* representing the regression coefficients of the experts network.
`sigmak`	Vector of length *K* giving the standard deviations of the experts network.
`lambdak`	Vector of length *K* giving the skewness parameter of each experts.
`nuk`	Vector of length *K* giving the degrees of freedom of the experts network t densities.
`x`	A vector og length *n* representing the inputs (predictors).

Value

A list with the output variable y and statistics.

- y Vector of length *n* giving the output variable.
- zi A vector of size *n* giving the hidden label of the expert component generating the i-th observation. Its elements are $zi[i] = k$, if the i-th observation has been generated by the k-th expert.
- z A matrix of size *(n, K)* giving the values of the binary latent component indicators Z_{ik} such that $Z_{ik} = 1$ iff $Z_i = k$.
- stats A list whose elements are:
  - Ey_k Matrix of size *(n, K)* giving the conditional expectation of Yi the output variable given the value of the hidden label of the expert component generating the ith observation *zi = k*, and the value of predictor *X = xi*.
  - Ey Vector of length *n* giving the conditional expectation of Yi given the value of predictor *X = xi*.
  - Vary_k Vector of length *k* representing the conditional variance of Yi given *zi = k*, and *X = xi*.
  - Vary Vector of length *n* giving the conditional expectation of Yi given *X = xi*.

`sampleUnivTMoE` *Draw a sample from a univariate t mixture of experts (TMoE).*

Description

Draw a sample from a univariate t mixture of experts (TMoE).

Usage

```
sampleUnivTMoE(alphak, betak, sigmak, nuk, x)
```

Arguments

`alphak`	The parameters of the gating network. `alphak` is a matrix of size *(q + 1, K - 1)*, with *K - 1*, the number of regressors (experts) and *q* the order of the logistic regression
`betak`	Matrix of size *(p + 1, K)* representing the regression coefficients of the experts network.
`sigmak`	Vector of length *K* giving the standard deviations of the experts network.
`nuk`	Vector of length *K* giving the degrees of freedom of the experts network t densities.
`x`	A vector of length *n* representing the inputs (predictors).

Value

A list with the output variable y and statistics.

- y Vector of length n giving the output variable.
- zi A vector of size n giving the hidden label of the expert component generating the i-th observation. Its elements are $zi[i] = k$, if the i-th observation has been generated by the k-th expert.
- z A matrix of size *(n, K)* giving the values of the binary latent component indicators Z_{ik} such that $Z_{ik} = 1$ iff $Z_i = k$.
- stats A list whose elements are:
  - Ey_k Matrix of size *(n, K)* giving the conditional expectation of Yi the output variable given the value of the hidden label of the expert component generating the ith observation *zi = k*, and the value of predictor *X = xi*.
  - Ey Vector of length n giving the conditional expectation of Yi given the value of predictor *X = xi*.
  - Vary_k Vector of length k representing the conditional variance of Yi given *zi = k*, and *X = xi*.
  - Vary Vector of length n giving the conditional expectation of Yi given *X = xi*.

`simulatedstructureddata` *Toy sample*

Description

Toy sample

Usage

```
simulatedstructureddata
```

Format

An object of class `data.frame` with 500 rows and 2 columns.

`StatNMoE-class` *A Reference Class which contains statistics of a NMoE model.*

Description

StatNoE contains all the parameters of a NMoE model.

Fields

piik Matrix of size (n, K) representing the probabilities $P(zi = k; W) = P(z_{ik} = 1; W)$ of the latent variable $zi,\ i = 1, \dots, m$.

z_ik Hard segmentation logical matrix of dimension (n, K) obtained by the Maximum a posteriori (MAP) rule: $z_{ik} = 1$ if $z_{ik} = \arg\max_k P(z_i = k|Y, W, \beta);$ 0 otherwise, $k = 1, \dots, K$.

klas Column matrix of the labels issued from z_ik. Its elements are $klas(i) = k$, $k = 1, \dots, K$.

Wik Matrix of dimension *(nm,K)*.

Ey_k Matrix of dimension *(n,K)*.

Ey Column matrix of dimension *n*.

Var_yk Column matrix of dimension *K*.

Vary Column matrix of dimension *n*.

log_lik Numeric. Log-likelihood of the StMoE model.

com_loglik Numeric. Complete log-likelihood of the StMoE model.

stored_loglik Numeric vector. Stored values of the log-likelihood at each EM iteration.

BIC Numeric. Value of the BIC (Bayesian Information Criterion) criterion. The formula is $BIC = log\_lik - nu \times \log(n)/2$ with *nu* the degree of freedom of the StMoE model.

ICL Numeric. Value of the ICL (Integrated Completed Likelihood) criterion. The formula is $ICL = com\_loglik - nu \times \log(n)/2$ with *nu* the degree of freedom of the StMoE model.

AIC Numeric. Value of the AIC (Akaike Information Criterion) criterion. The formula is $AIC = log\_lik - nu$.

log_piik_fik Matrix of size (n, K) giving the values of the logarithm of the joint probability $P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

log_sum_piik_fik Column matrix of size *n* giving the values of $\sum_{k=1}^{K} \log P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

tik Matrix of size (n, K) giving the posterior probability that Y_i originates from the k-th regression model $P(zi = k|Y, W, \beta)$.

Methods

MAP() calcule une partition d'un echantillon par la regle du Maximum A Posteriori ?? partir des probabilites a posteriori Entrees : post_probas , Matrice de dimensions [n x K] des probabibiltes a posteriori (matrice de la partition floue) n : taille de l'echantillon K : nombres de classes klas(i) = arg max (post_probas(i,k)) , for all i=1,...,n 1<=k<=K = arg max p(zi=k|xi;theta) 1<=k<=K = arg max p(zi=k;theta)p(xi|zi=k;theta)/suml=1^Kp(zi=l;theta) p(xi|zi=l;theta) 1<=k<=K Sorties : classes : vecteur collones contenant les classe (1:K) Z : Matrice de dimension [nxK] de la partition dure : ses elements sont zik, avec zik=1 si xi appartient ?? la classe k (au sens du MAP) et zero sinon.

See Also

ParamNMoE, FData

`StatSNMoE-class` *A Reference Class which contains statistics of a SNMoE model.*

Description

StatMRHLP contains all the parameters of a SNMoE model.

Fields

`piik` Matrix of size (n, K) representing the probabilities $P(zi = k; W) = P(z_{ik} = 1; W)$ of the latent variable $zi,\ i = 1, \dots, m$.

`z_ik` Hard segmentation logical matrix of dimension (n, K) obtained by the Maximum a posteriori (MAP) rule: $z_{ik} = 1 \text{ if } z_{ik} = \arg\max_k P(z_i = k|Y, W, \beta);\ 0 \text{ otherwise}, k = 1, \dots, K$.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \dots, K$.

`Ey_k` Matrix of dimension *(n,K)*.

`Ey` Column matrix of dimension *n*.

`Var_yk` Column matrix of dimension *K*.

`Vary` Column matrix of dimension *n*.

`log_lik` Numeric. Log-likelihood of the SNMoE model.

`com_loglik` Numeric. Complete log-likelihood of the SNMoE model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each EM iteration.

`BIC` Numeric. Value of the BIC (Bayesian Information Criterion) criterion. The formula is $BIC = log\_lik - nu \times \log(n)/2$ with *nu* the degree of freedom of the SNMoE model.

`ICL` Numeric. Value of the ICL (Integrated Completed Likelihood) criterion. The formula is $ICL = com\_loglik - nu \times \log(n)/2$ with *nu* the degree of freedom of the SNMoE model.

`AIC` Numeric. Value of the AIC (Akaike Information Criterion) criterion. The formula is $AIC = log\_lik - nu$.

`log_piik_fik` Matrix of size (n, K) giving the values of the logarithm of the joint probability $P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

`log_sum_piik_fik` Column matrix of size *n* giving the values of $\sum_{k=1}^{K} \log P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

`tik` Matrix of size (n, K) giving the posterior probability that Y_i originates from the k-th regression model $P(zi = k|Y, W, \beta)$.

`E1ik` To define.

`E2ik` To define.

Methods

`EStep(paramSNMoE)` Method used in the EM algorithm to update statistics based on parameters provided by `paramSNMoE` (prior and posterior probabilities).

`MAP()` calcule une partition d'un echantillon par la regle du Maximum A Posteriori ?? partir des probabilites a posteriori Entrees : post_probas , Matrice de dimensions [n x K] des probabibiltes a posteriori (matrice de la partition floue) n : taille de l'echantillon K : nombres de classes klas(i) = arg max (post_probas(i,k)) , for all i=1,...,n 1<=k<=K = arg max p(zi=k|xi;theta) 1<=k<=K = arg max p(zi=k;theta)p(xi|zi=k;theta)/suml=1^Kp(zi=l;theta) p(xi|zi=l;theta) 1<=k<=K Sorties : classes : vecteur collones contenant les classe (1:K) Z : Matrice de dimension [nxK] de la partition dure : ses elements sont zik, avec zik=1 si xi appartient ?? la classe k (au sens du MAP) et zero sinon.

See Also

ParamSNMoE, FData

`StatStMoE-class` *A Reference Class which contains statistics of a StMoE model.*

Description

StatMRHLP contains all the parameters of a StMoE model.

Fields

`piik` Matrix of size (n, K) representing the probabilities $P(zi = k; W) = P(z_{ik} = 1; W)$ of the latent variable $zi,\ i = 1, \dots, m$.

`z_ik` Hard segmentation logical matrix of dimension (n, K) obtained by the Maximum a posteriori (MAP) rule: $z_{ik} = 1$ if $z_{ik} = \arg\max_k\ P(z_i = k|Y, W, \beta);\ 0$ otherwise, $k = 1, \dots, K$.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \dots, K$.

`Ey_k` Matrix of dimension *(n,K)*.

`Ey` Column matrix of dimension *n*.

`Var_yk` Column matrix of dimension *K*.

`Var_y` Column matrix of dimension *n*.

`log_lik` Numeric. Log-likelihood of the StMoE model.

`com_loglik` Numeric. Complete log-likelihood of the StMoE model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each EM iteration.

`BIC` Numeric. Value of the BIC (Bayesian Information Criterion) criterion. The formula is $BIC = log\_lik - nu \times \log(n)/2$ with *nu* the degree of freedom of the StMoE model.

`ICL` Numeric. Value of the ICL (Integrated Completed Likelihood) criterion. The formula is $ICL = com\_loglik - nu \times \log(n)/2$ with *nu* the degree of freedom of the StMoE model.

`AIC` Numeric. Value of the AIC (Akaike Information Criterion) criterion. The formula is $AIC = log\_lik - nu$.

`log_piik_fik` Matrix of size (n, K) giving the values of the logarithm of the joint probability $P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

`log_sum_piik_fik` Column matrix of size *n* giving the values of $\sum_{k=1}^{K} \log P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

`tik` Matrix of size (n, K) giving the posterior probability that Y_i originates from the k-th regression model $P(zi = k|Y, W, \beta)$.

`wik` To define.

`dik` To define.

`stme_pdf` skew-t mixture of experts density

`E1ik` To define.

`E2ik` To define.

`E3ik` To define.

Methods

`EStep(paramStMoE)` Method used in the EM algorithm to update statistics based on parameters provided by `paramStMoE` (prior and posterior probabilities).

`MAP()` calcule une partition d'un echantillon par la regle du Maximum A Posteriori ?? partir des probabilites a posteriori Entrees : post_probas , Matrice de dimensions [n x K] des probabibiltes a posteriori (matrice de la partition floue) n : taille de l'echantillon K : nombres de classes klas(i) = arg max (post_probas(i,k)) , for all i=1,...,n 1<=k<=K = arg max p(zi=k|xi;theta) 1<=k<=K = arg max p(zi=k;theta)p(xi|zi=k;theta)/suml=1^Kp(zi=l;theta) p(xi|zi=l;theta) 1<=k<=K Sorties : classes : vecteur collones contenant les classe (1:K) Z : Matrice de dimension [nxK] de la partition dure : ses elements sont zik, avec zik=1 si xi appartient et la classe k (au sens du MAP) et zero sinon.

See Also

ParamStMoE, FData

`StatTMoE-class` *A Reference Class which contains statistics of a TMoE model.*

Description

StatTMoE contains all the parameters of a TMoE model.

Fields

`piik` Matrix of size (n, K) representing the probabilities $P(zi = k; W) = P(z_{ik} = 1; W)$ of the latent variable $zi,\ i = 1, \dots, m$.

`z_ik` Hard segmentation logical matrix of dimension (n, K) obtained by the Maximum a posteriori (MAP) rule: $z_{ik} = 1$ if $z_{ik} = \arg\max_k\ P(z_i = k|Y, W, \beta);\ 0$ otherwise, $k = 1, \dots, K$.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \dots, K$.

`Wik` Matrix of dimension *(nm,K)*.

`Ey_k` Matrix of dimension *(n,K)*.

`Ey` Column matrix of dimension *n*.

`Var_yk` Column matrix of dimension *K*.

`Vary` Column matrix of dimension *n*.

`log_lik` Numeric. Log-likelihood of the TMoE model.

`com_loglik` Numeric. Complete log-likelihood of the TMoE model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each EM iteration.

`BIC` Numeric. Value of the BIC (Bayesian Information Criterion) criterion. The formula is $BIC = log\_lik - nu \times \log(n)/2$ with *nu* the degree of freedom of the TMoE model.

`ICL` Numeric. Value of the ICL (Integrated Completed Likelihood) criterion. The formula is $ICL = com\_loglik - nu \times \log(n)/2$ with *nu* the degree of freedom of the TMoE model.

`AIC` Numeric. Value of the AIC (Akaike Information Criterion) criterion. The formula is $AIC = log\_lik - nu$.

`log_piik_fik` Matrix of size (n, K) giving the values of the logarithm of the joint probability $P(Y_i,\ zi = k)$, $i = 1, \dots, n$.

`log_sum_piik_fik` Column matrix of size n giving the values of $\sum_{k=1}^{K} \log P(Y_i,\ zi = k)$, $i = 1, \ldots, n$.

`tik` Matrix of size (n, K) giving the posterior probability that Y_i originates from the k-th regression model $P(zi = k|Y, W, \beta)$.

Methods

`MAP()` calcule une partition d'un echantillon par la regle du Maximum A Posteriori ?? partir des probabilites a posteriori Entrees : post_probas , Matrice de dimensions [n x K] des probabibiltes a posteriori (matrice de la partition floue) n : taille de l'echantillon K : nombres de classes klas(i) = arg max (post_probas(i,k)) , for all i=1,...,n 1<=k<=K = arg max p(zi=k|xi;theta) 1<=k<=K = arg max p(zi=k;theta)p(xi|zi=k;theta)/suml=1^Kp(zi=l;theta) p(xi|zi=l;theta) 1<=k<=K Sorties : classes : vecteur collones contenant les classe (1:K) Z : Matrice de dimension [nxK] de la partition dure : ses elements sont zik, avec zik=1 si xi appartient ?? la classe k (au sens du MAP) et zero sinon.

See Also

ParamTMoE, FData

Index