Package 'mmtfa'

June 15, 2018

Type Package

Title Model-Based Clustering and Classification with Mixtures of Modified t Factor Analyzers

Version 0.3

Date 2018-06-11

Author Jeffrey L. Andrews, Paul D. McNicholas, and Mathieu Chalifour

Maintainer Jeffrey L. Andrews <jeff.andrews@ubc.ca>

Description Fits a family of mixtures of multivariate t-distributions under a continuous t-distributed latent variable structure for the purpose of clustering or classification. The alternating expectation-conditional maximization algorithm is used for parameter estimation.

License GPL (>= 2)

LazyLoad yes

Imports parallel, mvnfast, matrixStats

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-15 07:49:20 UTC

R topics documented:

`mmtfa-package` *mmtfa: Model-Based Clustering and Classification with Mixtures of Modified Multivariate t Factor Analyzers*

Description

Fits mixtures of multivariate modified t-factor analyzers via the alternating expectation-conditional maximization algorithm under a clustering or classification paradigm in serial or parallel.

Details

Package:	mmtfa
Type:	Package
Version:	0.3
Date:	2018-06-11
License:	GPL (>=2)
LazyLoad:	yes

Author(s)

Jeffrey L. Andrews, Paul D. McNicholas, and Mathieu Chalifour

Maintained by: Jeffrey L. Andrews <jeff.andrews@ubc.ca>

References

Andrews JL and McNicholas PD (2011a), 'Extending mixtures of multivariate t-factor analyzers'. *Statistics and Computing* 21(3), 361–373.

Andrews JL and McNicholas PD (2011b), 'Mixtures of modified t-factor analyzers for model-based clustering, classification, and discriminant analysis'. *Journal of Statistical Planning and Inference* 141(4), 1479–1486.

See Also

`mmtfa` for main function

`mmtfa` *mmtfa: Function for Model-Based Clustering and Classification with Mixtures of Multivariate t Factor Analyzers*

Description

Fits mixtures of multivariate modified t-factor analyzers via the alternating expectation-conditional maximization algorithm to the given data under a clustering (default) or classification paradigm (by giving either training index or percentage of data taken to be known) in serial or parallel.

Usage

```
mmtfa(x, Gs=1:4, Qs=1:2, clas=0, init="kmeans", scale=TRUE, models = "all",
        dfstart=50, dfupdate="approx", gauss=FALSE, eps=0.05, known=NULL,
        parallel.cores=FALSE)
```

Arguments

x	A numeric matrix or data frame.
Gs	An integer or integer vector indicating the number of groups to fit. Default is 1-4.
Qs	An integer or integer vector indicating the number of factors to fit. Default is 1-2.
clas	Integer between 0–100 giving the percentage of data taken to be known. Simulates a classification scenario. Additional options to be provided in future updates...
init	A list of initializing classification of the form that `init[[G]]` contains the initializing vector for all G considered (see example below). Alternatively, the user can use a character string indicating initialization method. Currently the user can choose from `"kmeans"` (default), 'hard" random - `"hard"`, 'soft" random - `"soft"`, and `"uniform"` (classification only).
scale	Logical indicating whether or not the function should scale the data. Default is `TRUE` and is the prescribed method — tEIGEN models are not scale invariant.
models	A character string or character vector giving the models to fit. See details for instructions on choices.
dfstart	The initialized value for the degrees of freedom. The default is 50.
dfupdate	Character string (`"approx"` uses a closed form aqpproximation, `"numeric"` uses the `uniroot` function) or logical `FALSE` to skip degrees of freedom updates.
known	A vector of known classifications that can be numeric or character - optional for clustering, necessary for classification. Must be the same length as the number of rows in the data set.
gauss	Logical indicating if the algorithm should use the gaussian distribution. Currently equivalent to setting `dfstart=200` and `dfupdate=FALSE`. Will be improved in later updates...
eps	Tolerance value for the convergence criterion for the AECM algorithm.
parallel.cores	Logical or integer specifying number of computing cores to utilize for coarse-grain parallelization of the algorithm. If `FALSE` (default), then the algorithm is run in serial. If `TRUE`, then `detectCores()` from the `parallel` package will determine the number of cores and use all of them.

Details

Model specification (via the `models` argument) follows nomenclature developed from the factor analyzer decomposition of the covariance matrix. The nomenclature refers to the decomposition and constraints on the covariance matrix:

$$\Sigma_g = \Lambda_g \Lambda_g' + \omega_g \delta_g$$

The first letter can be a `"C"` (constrained across groups) or `"U"` (unconstrained) and that refers to setting $\Lambda_g = \Lambda$ or not, respectively. The second letter has the same choices, resulting in $\omega_g = \omega$ or not. The third letter is permitted as a `"C"`, `"U"`, or `"I"` (constrained to be the identity matrix),

applying those constraints to δ_g. The fourth, and final, letter refers to the degrees of freedom, and again is permitted "C" or "U".

As many models as desired can be selected and ran via the vector supplied to models. The complete list of possible names is: "UUUU", "UUUC", "UCCU","UCCC", "UUIU", "UUIC", "UCIU", "UCIC", "CUUU", "CUUC", "CCCU", "CCCC", "CUIU", "CUIC", "CCIU", "CCIC", "CUCU", "CUCC", "UUCU", "UUCC", "UCUU", "UCUC", "CCUU", "CCUC".

More commonly, subsets can be called by the following character strings: "all" runs all 24 MMtFA models (default), "dfunconstrained" runs the 12 unconstrained degrees of freedom models, "dfconstrained" runs the 12 constrained degrees of freedom models,

Also note that for G=1, several models are equivalent (for example, UUUU and CCCC). Thus, for G=1 only one model from each set of equivalent models will be run.

Value

x	Data used for clustering/classification.
classification	Vector of group classifications as determined by the BIC.
bic	BIC of the best fitted model.
modelname	Name of the best model according to the BIC.
allbic	Matrix of BIC values according to model and G. A value of -Inf is returned when the model did not converge.
bestmodel	Character string giving best model (BIC) details.
G	Value corresponding to the number of components chosen by the BIC.
tab	Classification table for BIC-selected model (only available when known is given). When classification is used the 'known' observations are left out of the table.
fuzzy	The fuzzy clustering matrix for the model selected by the BIC.
logl	The log-likelihood corresponding to the model with the best BIC.
iter	The number of iterations until convergence for the model selected by the BIC.
parameters	List containing the fitted parameters: mean - matrix of means where the rows correspond to the component and the columns are the variables; sigma - array of covariance matrices (multivariate). Upcoming updates will increase the number of saved parameters...
iclresults	List containing all the previous outputs, except x and index, pertaining to the model chosen by the best ICL (all under the same name except allicl and icl are the equivalent of allbic and bic, respectively).

Author(s)

Jeffrey L. Andrews, Paul D. McNicholas, and Mathieu Chalifour

References

Andrews JL and McNicholas PD (2011a), 'Extending mixtures of multivariate t-factor analyzers'. *Statistics and Computing* 21(3), 361–373.

Andrews JL and McNicholas PD (2011b), 'Mixtures of modified t-factor analyzers for model-based clustering, classification, and discriminant analysis'. *Journal of Statistical Planning and Inference* 141(4), 1479–1486.

See Also

See package manual MMtFA

Examples

```
###Note that only one model is run for each example
###in order to reduce computation time

#Clustering iris data with hard random start
tirisr <- mmtfa(iris[,-5], models="UUUU", Gs=1:3, Qs=1, init="hard")

#Clustering iris data with hierarchical starting values
initial_list <- list()
clustree <- hclust(dist(iris[,-5]))
for(i in 1:3){
initial_list[[i]] <- cutree(clustree,i)
}
tirish <- mmtfa(iris[,-5], models="CUCU", Gs=1:3, Qs=1, init=initial_list)

#Classification with the iris data set via percentage of data taken to have known membership
tirisc <- mmtfa(iris[,-5], Qs=1, models="CUIU", init="uniform",clas=50, known=iris[,5])
tirisc$tab
```

Index