Package 'plfm'

July 23, 2025

Type Package

Title Probabilistic Latent Feature Analysis

Version 2.2.6

Date 2024-02-14

Author Michel Meulders [aut, cre],
Philippe De Bruecker [ctb]

Maintainer Michel Meulders <michel.meulders@kuleuven.be>

Depends sfsmisc, abind

Description Functions for estimating probabilistic latent feature models with a disjunctive, conjunctive or additive mapping rule on (aggregated) binary three-way data.

License GPL (>= 2)

Repository CRAN

LazyLoad yes

LazyData yes

NeedsCompilation yes

Date/Publication 2024-02-15 00:40:02 UTC

Contents

`plfm-package` *Probabilistic Latent Feature Analysis*

Description

Functions for estimating disjunctive, conjunctive or additive probabilistic latent feature models on (aggregated) binary three-way data

Details

Probabilistic latent feature models can be used to model three-way three-mode binary observations (e.g. persons who indicate for each of a number of products and for each of a set of attributes whether a product has a certain attribute). A basic probabilistic feature model (referred to as `plfm`) uses aggregated three-way three-mode binary data as input, namely the two-way two-mode frequency table that is obtained by summing the binary three-way three-mode data across persons. The basic probabilistic feature model (Maris, De Boeck and Van Mechelen, 1996) is based on the assumption that observations are statistically independent and that model parameters are homogeneous across persons. The `plfm` function can be used to locate the posterior mode(s) of basic probabilistic feature models, and to compute information criteria for model selection, and measures of statistical and descriptive model fit. The `stepplfm` function can be used to fit a series of disjunctive, conjunctive or additive basic probabilistic feature models with different number of latent features. In addition, the `bayesplfm` function can be used to compute a sample of the posterior distribution of the basic probabilistic feature model in the neigbourhood of a specific posterior mode.

Latent class extensions of the probabilistic feature model (referred to as `LCplfm`) take binary three-way three-mode observations as input. In contrast to the basic probabilistic feature model, latent class probabilistic feature models allow to model dependencies between (subsets of) observations

(Meulders, De Boeck and Van Mechelen, 2003) and/or to account for heterogeneity in model parameters across persons (Meulders, Tuerlinckx, and Vanpaemel, 2013). The `LCplfm` function can be used to compute posterior mode estimates (of different types of) latent class probabilistic feature models as well as to compute information criteria for model selection, and measures of descriptive model fit. The `stepLCplfm` function can be used to compute a series of latent class probabilistic feature models with different numbers of latent features and latent classes.

To see the preferable citation of the package, type citation("plfm").

Author(s)

Michel Meulders

Maintainer: <michel.meulders@kuleuven.be>

References

Candel, M. J. J. M., and Maris, E. (1997). Perceptual analysis of two-way two-mode frequency data: probability matrix decomposition and two alternatives. *International Journal of Research in Marketing, 14*, 321-339.

Gelman, A., Van Mechelen, I., Verbeke, G., Heitjan, D. F., and Meulders, M. (2005). Multiple imputation for model checking: Completed-data plots with missing and latent data. *Biometrics, 61*, 74-85.

Maris, E., De Boeck, P., and Van Mechelen, I. (1996). Probability matrix decomposition models. *Psychometrika, 61*, 7-29.

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

Meulders, M., De Boeck, P., Kuppens, P., and Van Mechelen, I. (2002). Constrained latent class analysis of three-way three-mode data. *Journal of Classification, 19*, 277-302.

Meulders, M., De Boeck, P., and Van Mechelen, I. (2001). Probability matrix decomposition models and main-effects generalized linear models for the analysis of replicated binary associations. *Computational Statistics and Data Analysis, 38*, 217-233.

Meulders, M., De Boeck, P., and Van Mechelen, I. (2003). A taxonomy of latent structure assumptions for probability matrix decomposition models. *Psychometrika, 68*, 61-77.

Meulders, M., De Boeck, P., Van Mechelen, I., and Gelman, A. (2005). Probabilistic feature analysis of facial perception of emotions. *Applied Statistics, 54*, 781-793.

Meulders, M., De Boeck, P., Van Mechelen, I., Gelman, A., and Maris, E. (2001). Bayesian inference with probability matrix decomposition models. *Journal of Educational and Behavioral Statistics, 26*, 153-179.

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1)*, 1-45.

Meulders, M., Gelman, A., Van Mechelen, I., and De Boeck P. (1998). Generalizing the probability matrix decomposition model: An example of Bayesian model checking and model expansion. In J. Hox, and E. De Leeuw (Eds.), *Assumptions, robustness, and estimation methods in multivariate modeling* (pp. 1-19). TT Publicaties: Amsterdam.

Meulders, M., Tuerlinckx, F., and Vanpaemel, W. (2013). Constrained multilevel latent class models for the analysis of three-way three-mode binary data. *Journal of Classification, 30 (3)*, 306-337.

anger *Situational determinants of anger-related behavior*

Description

The raw data consist of the binary judgments of 101 first-year psychology students who indicated whether or not they would display each of 8 anger-related behaviors when being angry at someone in each of 6 situations. The 8 behaviors consist of 4 pairs of reactions that reflect a particular strategy to deal with situations in which one is angry at someone, namely, (1) fighting (fly off the handle, quarrel), (2) fleeing (leave, avoid), (3) emotional sharing (pour out one's heart, tell one's story), and (4) making up (make up, clear up the matter). The six situations are constructed from two factors with three levels: (1) the extent to which one likes the instigator of anger (like, dislike, unfamiliar), and (2) the status of the instigator of anger (higher, lower, equal). Each situation is presented as one level of a factor, without specifying a level for the other factor.

Usage

```
data(anger)
```

Format

The data consist of a list of 5 objects:

1. freq1: A 6 X 8 matrix of frequencies. The frequency in cell *(j,k)* indicates how many of 101 respondents would display reaction *k* in situation *j*.
2. freqtot: A 6 X 8 matrix of frequencies. The frequency in cell *(j,k)* indicates the total number of respondents who judged the situation-response pair *(j,k)*.
3. rowlabels: A vector of labels for the situations.
4. columnlabels: A vector of labels for the anger-related reactions.
5. data: A 101 X 6 X 8 array of binary (0/1) values. the value in cell *(i,j,k)* equals 1 if person *i* would display behavior *k* in situation *j*, and 0 otherwise.

Source

Meulders, M., De Boeck, P., Kuppens, P., and Van Mechelen, I. (2002). Constrained latent class analysis of three-way three-mode data. *Journal of Classification, 19*, 277-302.

References

Kuppens, P., Van Mechelen, I., and Meulders, M. (2004). Every cloud has a silver lining: Interpersonal and individual differences determinants of anger-related behaviors. *Personality and Social Psychology Bulletin, 30*, 1550-1564.

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

Vermunt, J. K. (2007). A hierarchical mixture model for clustering three-way data sets. *Computational Statistics and Data Analysis, 51*, 5368-5376.

anger2 *Situational determinants of anger-related behavior*

Description

The raw data consist of the binary judgments of 115 first-year psychology students who indicated whether or not they would display each of 14 anger-related behaviors when being angry at someone in each of 9 situations. The 14 behaviors consist of 7 pairs of reactions that reflect a particular strategy to deal with situations in which one is angry at someone:

1. Anger-out: (a) You flew off the handle, (b) You started a fight
2. Avoidance: (a) You avoided a confrontation, (b) You went out of the other's way
3. Social sharing (a) You unburdened your heart to others, (b) You told others what had happened
4. Assertive behavior: (a) You said what was bothering you in a direct and sober way, (b) You calmly explained what was bothering you
5. Indirect behavior (a) You showed something was bothering you without saying anything, (b) You started to sulk
6. Anger-in: (a) You suppressed your anger, (b) You bottled up your anger
7. Reconciliation (a) You reconciled, (b) You talked things out

The six situations are constructed by crossing the levels of two factors with three levels: (1) the extent to which one likes the instigator of anger (like, unfamilar, dislike), and (2) the status of the instigator of anger (lower status, equal status, higher status)

Usage

```
data(anger2)
```

Format

The data consist of a list of 5 objects:

1. data: A 115 X 9 X 14 matrix of binary observations (0/1). The observation in cell *(i,j,k)* equals 1 if person *i* would display behavior *k* in situation *j* and 0 otherwise.
2. freq1: A 9 X 14 matrix of frequencies. The frequency in cell *(j,k)* indicates the number of respondents who indicate that they would display behavior *k* in situation *j*.
3. freqtot: A 9 X 14 matrix of frequencies. The frequency in cell *(j,k)* indicates the total number of respondents who judged the situation-response pair *(j,k)*.
4. rowlabels: A vector of labels for the situations.
5. columnlabels: A vector of labels for the anger-related behaviors.

Source

Kuppens, P., Van Mechelen, I., and Meulders, M. (2004). Every cloud has a silver lining: Interpersonal and individual differences determinants of anger-related behaviors. *Personality and Social Psychology Bulletin, 30*, 1550-1564.

References

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1), 1-45*, 1-29.

`bayesplfm` *Bayesian analysis of probabilistic latent feature models for two-way two-mode frequency data*

Description

Computation of a sample of the posterior distribution for disjunctive or conjunctive probabilistic latent feature models with *F* features.

Usage

```
bayesplfm(data,object,attribute,rating,freq1,freqtot,F,
          Nchains=2,Nburnin=0,maxNiter=4000,
          Nstep=1000,Rhatcrit=1.2,maprule="disj",datatype="freq",
          start.bayes="best",fitted.plfm=NULL)
```

Arguments

`data`	A data frame that consists of three components: the variables `object`, `attribute` and `rating`. Each row of the data frame describes the outcome of a binary rater judgement about the association between a certain object and a certain attribute.
`object`	The name of the `object` component in the data frame `data`. The values of the vector `data$object` should be (non-missing) numeric or character values.
`attribute`	The name of the `attribute` component in the data frame `data`. The values of the vector `data$attribute` should be (non-missing) numeric or character values.
`rating`	The name of the `rating` component in the data frame `data`. The elements of the vector `data$rating` should be the numeric values 0 (no association) or 1 (association), or should be specified as missing (NA).
`freq1`	A *J X K* matrix of observed association frequencies.
`freqtot`	A *J X K* matrix with the total number of binary ratings in each cell *(j,k)*. If the total number of ratings is the same for all cells of the matrix it is sufficient to enter a single numeric value rather than a matrix. For instance, if *N* raters have judged *J X K* associations, one may specify `freqtot=`*N*
`F`	The number of latent features included in the model.
`Nchains`	The number of Markov-chains that are simulated using a data-augmented Gibbs sampling algorithm.
`Nburnin`	The number of burn-in iterations.
`maxNiter`	The maximum number of iterations that will be computed for each chain.

Nstep	The convergence of the chains to the true posterior will be checked for each parameter after c*Nstep iterations with c=1,2,... The convergence will only be checked when Nchains>1.
Rhatcrit	The estimation procedure will be stopped if the Rhat convergence diagnostic is smaller than Rhatcrit for each object- and attribute parameter. By default Rhatcrit=1.2.
maprule	Disjunctive (*maprule="disj"*) or conjunctive (*maprule="conj"*) mapping rule of the probabilistic latent feature model.
datatype	The type of data used as input. When datatype="freq" one should specify frequency data freq1 and freqtot, and when datatype="dataframe" one should specify the name of the data frame data, and its components, object, attribute and rating.
start.bayes	This argument can be used to define the type of starting point for the Bayesian analysis. If start.bayes="best" the best solution of a plfm analysis is used as the starting point for the Bayesian analysis, and if start.bayes = "fitted.plfm", the starting point is read from the (plfm) object assigned to the argument fitted.plfm. If start.bayes="random", a random starting point is used for the Bayesian analysis.
fitted.plfm	The name of the plfm object that contains posterior mode estimates for the specified model.

Details

The function bayesplfm can be used to compute a sample of the posterior distribution of disjunctive or conjunctive probabilistic latent feature models with a particular number of features using a data-augmented Gibbs sampling algorithm (Meulders, De Boeck, Van Mechelen, Gelman, and Maris, 2001; Meulders, De Boeck, Van Mechelen, and Gelman, 2005; Meulders, 2013).

By specifying the parameter Nchains the function can be used to compute one single chain, or multiple chains. When only one chain is computed, no convergence measure is reported. When more than one chain is computed, for each parameter, convergence to the true posterior distribution is assessed using the Rhat convergence diagnostic proposed by Gelman and Rubin (1992).

When using bayesplfm for Bayesian analysis the same starting point will be used for each simulated chain. The reason for using the same starting point for each of the chains is that the posterior distribution of probabilistic feature models with $F>2$ is always multimodal (local maxima may exist, and one may switch feature labels), so that the aim of the Bayesian analysis is to compute a sample in the neigbourhood of one specific posterior mode. It is recommended to use the best posterior mode obtained with the plfm function as a starting point for the Bayesian analysis (use start.bayes="best", or specify start.bayes="fitted.plfm" and fitted.plfm=object) with "object" being a plfm object that contains posterior mode estimates for the specified model. As an alternative to using the plfm(), function one may use random starting points for the Bayesian analysis (start.bayes="random") to explore the posterior distribution.

The function bayesplfm() will converge well if the distinct posterior modes are well-separated and if the different chains only visit the same mode during the estimation process. However, if the posterior distribution is multimodal, it may fail to converge if the Gibbs sampler starts visiting different posterior modes within one chain, or if different chains sample from distinct posterior modes.

Value

`call`	Parameters used to call the function.
`sample.objpar`	A *J X F X Niter X Nchains* array with parameter values for the object parameters. The matrix `sample.objpar[,,i,c]` contains the draw of the object parameters in iteration *i* of chain *c*. Note: when `Nchains=1` the chain length *Niter* equals `maxNiter`, and when `Nchains>1` the chain length *Niter* equals the number of iterations required to obtain convergence.
`sample.attpar`	A *K X F X Niter X Nchains* array with parameter values for the attribute parameters. The matrix `sample.attpar[,,i,c]` contains the draw of the attribute parameters in iteration *i* of chain *c*. Note: when `Nchains=1` the chain length *Niter* equals `maxNiter`, and when `Nchains>1` the chain length *Niter* equals the number of iterations required to obtain convergence.
`pmean.objpar`	A *J X F* matrix with the posterior mean of the object parameters computed on all iterations and chains in the sample.
`pmean.attpar`	A *K X F* matrix with the posterior mean of the attribute parameters computed on all iterations and chains in the sample.
`p95.objpar`	A *3 X J X F* array which contains for each object parameter the percentiles 2.5, 50 and 97.5.
`p95.attpar`	A *3 X K X F* array which contains for each attribute parameter the percentiles 2.5, 50 and 97.5.
`Rhat.objpar`	A *J X F* matrix with Rhat convergence values for the object parameters.
`Rhat.attpar`	A *K X F* matrix with Rhat convergence values for the attribute parameters.
`fitmeasures`	A list with two measures of descriptive fit on the *J X K* table: (1) the correlation between observed and expected frequencies, and (2) the proportion of the variance in the observed frequencies accounted for by the model. The association probabilities and corresponding expected frequencies are computed using the posterior mean of the parameters.
`convstat`	The number of object-and attribute parameters that do not meet the convergence criterion.

Author(s)

Michel Meulders

References

Gelman, A., and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science, 7* , 457-472.

Meulders, M., De Boeck, P., Van Mechelen, I., Gelman, A., and Maris, E. (2001). Bayesian inference with probability matrix decomposition models. *Journal of Educational and Behavioral Statistics, 26*, 153-179.

Meulders, M., De Boeck, P., Van Mechelen, I., and Gelman, A. (2005). Probabilistic feature analysis of facial perception of emotions. *Applied Statistics, 54*, 781-793.

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1)*, 1-45.

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

See Also

plfm, summary.bayesplfm,print.summary.bayesplfm

Examples

```
## Not run:
## example 1: Bayesian analysis using data generated under the model

## define number of objects
J<-10
## define number of attributes
K<-10
## define number of features
F<-2

## generate true parameters
set.seed(43565)
objectparameters<-matrix(runif(J*F),nrow=J)
attributeparameters<-matrix(runif(K*F),nrow=K)

## generate data for conjunctive model using N=100 replications
gdat<-gendat(maprule="conj",N=100,
             objpar=objectparameters,attpar=attributeparameters)

## Use stepplfm to compute posterior mode(s) for 1 up to 3 features

conj.lst<-stepplfm(minF=1,maxF=3,maprule="conj",freq1=gdat$freq1,freqtot=100,M=5)


## Compute a sample of the posterior distribution
## for the conjunctive model with two features
## use the posterior mode obtained with stepplfm as starting point
conjbayes2<-bayesplfm(maprule="conj",freq1=gdat$freq1,freqtot=100,F=2,
                      maxNiter=3000,Nburnin=0,Nstep=1000,Nchains=2,
                      start.bayes="fitted.plfm",fitted.plfm=conj.lst[[2]])


## End(Not run)

## Not run:
## example 2: Bayesian analysis of situational determinants of anger-related behavior

## load data
data(anger)

## Compute one chain of 500 iterations (including 250 burn-in iterations)
## for the disjunctive model with two features
## use a random starting point
```

```
bayesangerdisj2a<-bayesplfm(maprule="disj",freq1=anger$freq1,freqtot=anger$freqtot,F=2,
                      maxNiter=500,Nstep=500,Nburnin=250,Nchains=1,start.bayes="random")

##print a summary of the output
summary(bayesangerdisj2a)


## Compute a sample of the posterior distribution
## for the disjunctive model with two features
## compute starting points with plfm
## run 2 chains with a maximum length of 10000 iterations
## compute convergence after each 1000 iterations

bayesangerdisj2b<-bayesplfm(maprule="disj",freq1=anger$freq1,freqtot=anger$freqtot,F=2,
                      maxNiter=10000,Nburnin=0,Nstep=1000,Nchains=2,start.bayes="best")


## print the output of the disjunctive 2-feature model for the anger data
print(bayesangerdisj2b)


## print a summary of the output of the disjunctive 2-feature model
##for the anger data
summary(bayesangerdisj2b)

## End(Not run)
```

---

car *Ratings of associations between car models and car attributes*

Description

The data describe the ratings of 78 respondents about the association between each of 14 car models and each of 27 car attributes.

Usage

```
data(car)
```

Format

The data consist of a list of 4 objects:

1. datalongformat: A data frame that consists of 6 components: Each row of the data frame describes the outcome of a binary rater judgement about the association between a certain car and a certain attribute. The components *IDobject* and *objectlabel* contain an ID and label for the car models, the components *IDattribute* and *attributelabel* contain an ID and label for the attributes, the component *IDrater* contains a rater ID, and the component *rating* contains the binary ratings (1 if the car model has the attribute according to the rater, and 0 otherwise).

2. data3w: A 78 X 14 X 27 array of binary judgements. The observation in cell *(i,j,k)* equals 1 if respondent *i* indicates that car *j* has attribute *k*, and 0 otherwise.
3. freq1: A 14 X 27 matrix of frequencies. The frequency in cell *(j,k)* indicates how many of 78 respondents indicate an association between car model *j* and attribute *k*.
4. freqtot: A 14 X 27 matrix of frequencies. The frequency in cell *(j,k)* indicates the total number of respondents who judged the car-attribute pair *(j,k)*.

Source

Van Gysel, E. (2011). *Perceptuele analyse van automodellen met probabilistische feature modellen.* [translation from Dutch: Perceptual analysis of car models with probabilistic feature models] Master thesis. Hogeschool-Universiteit Brussel.

References

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

---

car2 *Judgements on associations between car models and car attributes*

Description

The data consist of the binary judgements of 147 respondents about the association between each of 12 car models and each of 23 car attributes.

Usage

```
data(car2)
```

Format

The data consist of a list of 5 objects:

1. data3w: A 147 X 12 X 23 array of binary judgements. The observation in cell *(i,j,k)* equals 1 if respondent *i* indicates that car *j* has attribute *k*, and 0 otherwise.
2. freq1: A 12 X 23 matrix of frequencies. The frequency in cell *(j,k)* indicates how many of 147 respondents indicate an association between car model *j* and attribute *k*.
3. freqtot: A 12 X 23 matrix of frequencies. The frequency in cell *(j,k)* indicates the total number of respondents who judged the car-attribute pair *(j,k)*.

Source

M\"ahler, R. (2014). *Analyse van perceptie en preferentie van middelgrote wagens.* [translation from Dutch: Analysis of perception and preference for midsize cars.] Master thesis. KU Leuven.

References

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1)*, 1-45.

`gendat` *Data generation*

Description

Computation of association probabilities and data generation for disjunctive, conjunctive or additive probabilistic latent feature models.

Usage

```
gendat(maprule="disj", N, objpar, attpar)
```

Arguments

`maprule`	Disjunctive (`maprule="disj"`), conjunctive (`maprule="conj"`) or additive (`maprule="add"`) mapping rule of the probabilistic latent feature model.
`N`	Number of replications for which binary associations are generated.
`objpar`	True objectparameters. As object parameters are probabilities they should be between 0 and 1.
`attpar`	True attributeparameters. As attribute parameters are probabilities they should be between 0 and 1.

Details

The function `gendat` computes for all pairs of *J* objects and *K* attributes association probabilities and it generates association frequencies (i.e. the number of replications `N` for which an object is associated to an attribute), according to a disjunctive, conjunctive or additive probabilistic latent feature model. In addition, the function computes a matrix with in each cell the total number of replications `N`. If the requested number of replications `N` equals 0, the function only computes association probabilities and does not generate new data.

To compute association probabilities the function *gendat* uses a *J X F* matrix of object parameters and a *K X F* matrix of attribute parameters as input. The *F* object parameters of object *j* represent, for each of *F* features, the probability that object *j* has feature *f*. Similarly, the *F* attribute parameters of attribute *k* reflect, for each of *F* features, the probability that attribute *k* is linked to feature *f*.

According to the *disjunctive* probabilistic latent feature model, object *j* is associated to attribute *k* if the object and the attribute have at least one feature in common. More specifically, the association probability in cell *(j,k)* for the disjunctive model can be computed as:

$$p(j,k) = 1 - \prod_f (1 - objpar[j,f] * attpar[k,f]).$$

According to the *conjunctive* probabilistic latent feature model, object *j* and attribute *k* are associated if object *j* has all the features that are linked to attribute *k*. For the conjunctive model the association probability in cell *(j,k)* is computed as:

$$p(j,k) = \prod_f (1 - (1 - objpar[j,f]) * attpar[k,f]).$$

The *additive* mapping rule states that an object and attribute are more likely to associated if they have more common features. More specifically, the association probability for the additive model is computed as:

$$p(j,k) = \frac{1}{F} * \sum_f (objpar[j,f]) * attpar[k,f]).$$

Value

`call`	Parameters used to call the function.
`prob1`	*J X K* matrix of association probabilities.
`freq1`	*J X K* matrix of association frequencies.
`freqtot`	*J X K* matrix with number of replications.

Author(s)

Michel Meulders

References

Maris, E., De Boeck, P., and Van Mechelen, I. (1996). Probability matrix decomposition models. *Psychometrika, 61*, 7-29.

Meulders, M., De Boeck, P., Van Mechelen, I., Gelman, A., and Maris, E. (2001). Bayesian inference with probability matrix decomposition models. *Journal of Educational and Behavioral Statistics, 26*, 153-179.

Meulders, M., De Boeck, P., Van Mechelen, I., & Gelman, A. (2005). Probabilistic feature analysis of facial perception of emotions. *Applied Statistics, 54*, 781-793.

See Also

`plfm`

Examples

```
## define constants
J<-20
K<-15
F<-2

## generate true parameters
set.seed(43565)
```

```
objectparameters<-matrix(runif(J*F),nrow=J)
attributeparameters<-matrix(runif(K*F),nrow=K)

## compute association probabilities for a conjunctive model
probconj<-gendat(maprule="conj",N=0,
             objpar=objectparameters,attpar=attributeparameters)

## generate data for a disjunctive model using N=200 replications
gdat<-gendat(maprule="disj",N=200,
             objpar=objectparameters,attpar=attributeparameters)

## generate data for a additive model using N=200 replications
gdat<-gendat(maprule="add",N=200,
             objpar=objectparameters,attpar=attributeparameters)
```

---

gendatLCplfm *Data generation*

Description

Data generation for disjunctive, conjunctive and additive latent class probabilistic latent feature models.

Usage

```
gendatLCplfm(N,objpar,attpar,sizepar,maprule="disj",model=1)
```

Arguments

N	Number of replications (e.g. persons) for which binary object-attribute associations are generated.
objpar	True objectparameters. If model=1, model=3, model=4, or model=6 objpar is a $J\ X\ F\ X\ T$ array, if model=2 or model=5 objpar is a $J\ X\ F$ matrix. As object parameters are probabilities they should be between 0 and 1.
attpar	True attributeparameters. If model=2, model=3, model=5, or model=6 attpar is a $K\ X\ F\ X\ T$ array, if model=1 or model=4 attpar is a $K\ X\ F$ matrix. As attribute parameters are probabilities they should be between 0 and 1.
sizepar	A T-vector of true class size parameters.
maprule	Disjunctive (maprule="disj"), conjunctive (maprule="conj") or additive (maprule="add") mapping rule of the latent class probabilistic latent feature model.
model	The type of dependency and heterogeneity assumption included in the model. model=1, model=2, model=3 represent models with a constant object-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters. model=4, model=5, model=6 represent models with a constant attribute-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters.

Details

The function `gendatLCplfm` generates binary object-attribute associations for *N* replications according to a disjunctive, conjunctive or additive latent class probabilistic latent feature model of a specific model type. In addition, the function computes the *J X K* matrix of marginal object-attribute association probabilities and a *J X K X T* array of class-specific object-attribute association probabilities. To compute association probabilities the function *gendatLCplfm* uses a vector of class size parameters (`sizepar`) a matrix or array of object parameters (`objpar`) and a matrix or array of true attribute parameters (`attpar`) as input.

According to the *disjunctive* probabilistic latent feature model, object *j* is associated to attribute *k* if the object and the attribute have at least one feature in common. More specifically, for `model=1` the class-specific object-attribute association probability in cell *(j,k)* for the disjunctive model can be computed as:

$$p(j,k|t) = 1 - \prod_f (1 - objpar[j,f,t] * attpar[k,f]).$$

According to the *conjunctive* probabilistic latent feature model, object *j* and attribute *k* are associated if object *j* has all the features that are linked to attribute *k*.

In particular, for `model=1`,the class-specific object-attribute association probability in cell *(j,k)* for the conjunctive model can be computed as:

$$p(j,k|t) = \prod_f (1 - (1 - objpar[j,f,t]) * attpar[k,f]).$$

According to the *additive* probabilistic latent feature model, an object and an attribute are more likely to be associated if they have more features in common.

In particular, for `model=1`,the class-specific object-attribute association probability in cell *(j,k)* for the additive model can be computed as:

$$p(j,k|t) = \frac{1}{F} * \sum_f (objpar[j,f,t]) * attpar[k,f]).$$

The marginal object-attribute association probability can be computed as follows:

$$p(j,k) = \sum_t sizepar[t] * p(j,k|t).$$

Value

`call`	Parameters used to call the function.
`data`	*I X J X K* matrix of association probabilities.
`class`	*I*-vector that contains latent class membership of each replication.
`condprob.JKT`	*J X K X T* array of class-specific conditional object-attribute association probabilities.
`margprob.JK`	*J X K* matrix of marginal object-attribute association probabilities.

Author(s)

Michel Meulders

References

Meulders, M., Tuerlinckx, F., and Vanpaemel, W. (2013). Constrained multilevel latent class models for the analysis of three-way three-mode binary data. *Journal of Classification, 30 (3)*, 306-337.

See Also

LCplfm

Examples

```
## Not run:
# define constants
I<-500
J<-10
K<-8
F<-2
T<-2

# model 1

# generate true parameters
objpar<-array(runif(J*F*T),c(J,F,T))
attpar<-matrix(runif(K*F),c(K,F))
sizepar<-rep(1/T,T)
# generate data
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=1)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=1,maprule="conj")


# model 2

# generate true parameters
objpar<-matrix(runif(J*F),nrow=J)
attpar<-array(runif(K*F*T),c(K,F,T))
sizepar<-rep(1/T,T)
# generate data
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=2)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=2,maprule="conj")

# model 3

# generate true parameters
objpar<-array(runif(J*F*T),c(J,F,T))
attpar<-array(runif(K*F*T),c(K,F,T))
sizepar<-rep(1/T,T)
# generate data
```

```
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=3)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=3,maprule="conj")

# model 4

# generate true parameters
objpar<-array(runif(J*F*T),c(J,F,T))
attpar<-matrix(runif(K*F),c(K,F))
sizepar<-rep(1/T,T)
# generate data
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=4)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=4,maprule="conj")

# model 5

# generate true parameters
objpar<-matrix(runif(J*F),nrow=J)
attpar<-array(runif(K*F*T),c(K,F,T))
sizepar<-rep(1/T,T)
# generate data
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=5)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=5,maprule="conj")


# model 6
# generate true parameters
objpar<-array(runif(J*F*T),c(J,F,T))
attpar<-array(runif(K*F*T),c(K,F,T))
sizepar<-rep(1/T,T)
# generate data
d<-gendatLCplfm(N=I,objpar=objpar,attpar=attpar,sizepar=sizepar,maprule="conj",model=6)
# estimate parameters of true model
res<-LCplfm(data=d$data,F=2,T=2,model=6,maprule="conj")

## End(Not run)
```

`hostility` *self-reported hostile behavior in frustrating situations*

Description

The data consist of the judgments of 316 first-year psychology students who indicated on a three point scale the extent to which they would display each of 4 hostile behaviors in each of 14 frustrating situations (0= you do not display this response in this situation, 1= you display this response to a limited extent in this situation, 2= you display this response to a strong extent in this situation).

Usage

```
data(hostility)
```

Format

The data consist of a list of 6 objects:

1. data: A 316 X 14 X 4 array of dichotomized judgements (0 versus 1 or 2). The observation in cell *(i,j,k)* equals 1 if person *i* would display behavior *k* in situation *j* to a limited or strong extent and 0 if person *i* would not display behavior *k* in situation *j*.
2. freq1: A 14 X 4 matrix of frequencies. The frequency in cell *(j,k)* indicates the number of respondents who indicate that they would display behavior *k* in situation *j*.
3. freqtot: A 14 X 4 matrix of frequencies. The frequency in cell *(j,k)* indicates the total number of respondents who judged the situation-response pair *(j,k)*.
4. situation: A vector with descriptions of the situations.
5. rowlabels: A vector of labels for the situations.
6. columnlabels: A vector of labels for the anger-related behaviors.

Source

Vansteelandt, K. (1999). A formal model for the competency-demand hypothesis. *European Journal of Personality, 13*, 429-442.

References

Vansteelandt, K. and Van Mechelen, I. (1998). Individual differences in situation-behavior profiles: A triple typology model. *Journal of Personality and Social Psychology, 75*, 751-765.

LCplfm — *Latent class probabilistic feature analysis of three-way three-mode binary data*

Description

Computation of parameter estimates, standard errors, criteria for model selection, and measures of descriptive fit for disjunctive, conjunctive and additive latent class probabilistic feature models.

Usage

```
LCplfm(data,F=2,T=2,M=5,maprule="disj",emcrit1=1e-3,emcrit2=1e-8,
       model=1,start.objectparameters=NULL,start.attributeparameters=NULL,
       start.sizeparameters=NULL,delta=0.0001,printrun=FALSE,
       update.objectparameters=NULL,update.attributeparameters=NULL,
       Nbootstrap=2000)
```

Arguments

data	A *I X J X K* data array of binary observations. Observation *(i,j,k) (i=1,..,I; j=1,..,J; k=1,..,K)* indicates whether object *j* is associated to attribute *k* according to rater *i*.
F	The number of latent features included in the model.
T	The number of latent classes included in the model.
M	The number of times a particular model is estimated using random starting points.
maprule	Disjunctive (maprule="disj"), conjunctive (maprule="conj") or additive (maprule="add") mapping rule of the probabilistic latent feature model.
emcrit1	Convergence criterion to be used for the estimation of candidate models.
emcrit2	Convergence criterion to be used for the estimation of the best model.
model	The type of dependency and heterogeneity assumption included in the model. model=1, model=2, model=3 represent models with a constant object-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters. model=4, model=5, model=6 represent models with a constant attribute-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters.
start.objectparameters	An array of object parameters to be used as starting value for each run. The size of the array equals *J x F x T x M* when model = 1,4,3,6 and *J x F x M* when model = 2,5. If start.objectparameters=NULL randomly generated object parameters are used as starting values.
start.attributeparameters	An array of attribute parameters to be used as starting value for each run. The size of the array equals *K x F x T x M* when model = 2,5,3,6 and *K x F x M* when model = 1,3. If start.attributeparameters=NULL randomly generated attribute parameters are used as starting values.
start.sizeparameters	A *T x M* matrix of latent class size parameters to be used as starting value for each run. If start.sizeparameters=NULL randomly generated class size parameters are used as starting values.
delta	The precision used to compute standard errors of the parameters with the method of finite differences.
printrun	printrun=TRUE prints the analysis type (disjunctive, conjunctive, additive), the number of features (*F*), the number of latent classes (*T*) and the number of the run to the output screen, whereas printrun=FALSE suppresses the printing.
update.objectparameters	A binary valued array that indicates for each object parameter whether it has to be estimated from the data or constrained to the starting value. A value of 1 means that the corresponding object parameter is estimated and a value of 0 means that the corresponding object parameter is constrained to the starting

value provided by the user. The size of the array equals *J x F x T* when `model = 1,4,3,6` and *J x F* when `model = 2,5`. If `update.objectparameters = NULL` all object parameters are estimated from the data.

`update.attributeparameters`
: A binary valued array that indicates for each attribute parameter whether it has to be estimated from the data or constrained to the starting value. A value of 1 means that the corresponding attribute parameter is estimated and a value of 0 means that the corresponding attribute parameter is constrained to the starting value provided by the user. The size of the array equals *K x F x T* when `model = 2,5,3,6` and *K x F* when `model = 1,3`. If `update.attributeparameters = NULL` all attribute parameters are estimated from the data.

`Nbootstrap`
: Number of bootstrap iterations to be used for simulating the reference distribution of odds-ratio dependency measures.

Details

Estimation The estimation algorithm includes two steps. In a first exploratory step an EM algorithm is used to conduct `M` runs using random starting points. Each exploratory run is terminated if the convergence criterium (i.e., the sum of absolute differences between parameter values in subsequent iterations) is smaller than `emcrit1`. In a second step, the best solution among the `M` runs (i.e., with the highest posterior density) is used as the starting point of the EM algorithm for conducting a final analysis. The final analysis is terminated if the convergence criterion `emcrit2` is smaller than the convergence criterion.

Model selection criteria, goodness-of-fit and statistical dependency measures

To choose among models with different numbers of features, or with different mapping rules, one may use information criteria such as the Akaike Information Criterion (AIC, Akaike, 1973, 1974), or the Schwarz Bayesian Information Criterion (BIC, Schwarz, 1978). AIC and BIC are computed as *-2*loglikelihood+k*Npar*. For AIC *k* equals 2 and for BIC *k* equals *log(N)*, with *N* the observed number of replications (*I*) for which object-attribute associations are collected. *Npar* represents the number of model parameters. Models with the lowest value for AIC or BIC should be selected.

The descriptive goodness-of-fit of the model is assessed with the correlation between observed and expected frequencies in the *J X K* table, and the proportion of the variance in the observed frequencies accounted for by the model (VAF) (i.e. the squared correlation between observed and expected frequencies).

To assess to which extent the model can capture observed statistical dependencies between object-attribute pairs with a common object or attribute, a parametric bootstrap procedure is used to evaluate whether observed dependencies are within the simulated 95 or 99 percent confidence interval. Let *D(i,j,k)* be equal to 1 if rater i indicates that object *j* is associated to attribute *k*. The statistical dependency between pairs *(j,k)* and *(j*,k*)* is measured with the odds ratio (OR) statistic:

$$OR(j,k,j^*,k^*) = \log\left[\frac{N_{11} * N_{00}}{N_{10} * N_{01}}\right]$$

with

$$N_{11} = \sum_i D(i,j,k)D(i,j^*,k^*) + 0.5$$

$$N_{00} = \sum_i (1 - D(i,j,k))(1 - D(i,j^*,k^*)) + 0.5$$

$$N_{10} = \sum_i D(i,j,k)(1 - D(i,j^*,k^*)) + 0.5$$

$$N_{01} = \sum_i (1 - D(i,j,k))D(i,j^*,k^*) + 0.5$$

The model selection criteria AIC and BIC, the descriptive goodness-of-fit measures (correlation observed and expected frequencies, and VAF) and a summary of the OR dependency measures (i.e., proportion of observed OR dependencies of a certain type that are in the simulated 95 or 99 percent confidence interval) are stored in the object `fitmeasures` of the output list

Value

`call`	Parameters used to call the function.
`logpost.runs`	A list with the logarithm of the posterior density for each of the *M* computed models.
`best`	An index which indicates the model with the highest posterior density among each of the *M* computed models.
`objpar`	Estimated object parameters for the best model.
`attpar`	Estimated attribute parameters for the best model.
`sizepar`	A vector of `T` class size parameters for the best model.
`SE.objpar`	Estimated standard errors for the object parameters of the best model.
`SE.attpar`	Estimated standard errors for the attribute parameters of the best model.
`SE.sizepar`	Estimated standard errors for the class size parameters of the best model.
`gradient.objpar`	Gradient of the object parameters for the best model.
`gradient.attpar`	Gradient of the attribute parameters for the best model.
`gradient.sizepar`	Gradient of the class size parameters for the best model.
`fitmeasures`	A list of model selection criteria, goodness-of-fit measures and OR dependency measures for the model with the highest posterior density.
`postprob`	A *I X T* matrix of posterior probabilities for the best model.
`margprob.JK`	A *J X K* matrix of marginal object-attribute association probabilities.
`condprob.JKT`	A *J X K X T* array of conditional object-attribute association probabilities (i.e., probability of object-attribute association given latent class membership).
`report.OR.attpair`	A matrix that contains for all attribute pairs per object the observed OR dependency (OR.obs), the expected OR dependency (OR.mean) and the upper and lower bounds of the corresponding simulated 95 and 99 percent confidence interval (OR.p025, OR.p975, OR.p005, OR.p995).

`report.OR.objpair`
: A matrix that contains for all object pairs per attribute the observed OR dependency (OR.obs), the expected OR dependency (OR.mean) and the upper and lower bounds of the corresponding simulated 95 and 99 percent confidence interval (OR.p025, OR.p975, OR.p005, OR.p995).

Author(s)

Michel Meulders and Philippe De Bruecker

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki (Eds.), *Second international symposium on information theory* (p. 271-283). Budapest: Academiai Kiado.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*, 716-723.

Candel, M. J. J. M., and Maris, E. (1997). Perceptual analysis of two-way two-mode frequency data: probability matrix decomposition and two alternatives. *International Journal of Research in Marketing, 14*, 321-339.

Louis, T. A. (1982). Finding observed information using the em algorithm. *Journal of the Royal Statistical Society, Series B, 44*, 98-130.

Maris, E. (1999). Estimating multiple classification latent class models. *Psychometrika, 64*, 187-212.

Maris, E., De Boeck, P., and Van Mechelen, I. (1996). Probability matrix decomposition models. *Psychometrika, 61*, 7-29.

Meulders, M., De Boeck, P., and Van Mechelen, I. (2001). Probability matrix decomposition models and main-effects generalized linear models for the analysis of replicated binary associations. *Computational Statistics and Data Analysis, 38*, 217-233.

Meulders, M., De Boeck, P., Van Mechelen, I., Gelman, A., and Maris, E. (2001). Bayesian inference with probability matrix decomposition models. *Journal of Educational and Behavioral Statistics, 26*, 153-179.

Meulders, M., De Boeck, P., Van Mechelen, I., and Gelman, A. (2005). Probabilistic feature analysis of facial perception of emotions. *Applied Statistics, 54*, 781-793.

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1)*, 1-45.

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

Meulders, M., Tuerlinckx, F., and Vanpaemel, W. (2013). Constrained multilevel latent class models for the analysis of three-way three-mode binary data. *Journal of Classification, 30 (3)*, 306-337.

Tanner, M. A. (1996). *Tools for statistical inference: Methods for the exploration of posterior distributions and likelihood functions* (Third ed.). New York: Springer-Verlag.

Tatsuoka, K. (1984). *Analysis of errors in fraction addition and subtraction problems*. Final Report for NIE-G-81-0002, University of Illinois, Urbana-Champaign.

Schwarz, G. (1978). Estimating the dimensions of a model. *Annals of Statistics, 6*, 461-464.

See Also

print.LCplfm

Examples

```
## Not run:

# example 1: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# estimate a disjunctive LCplfm model with F=2 and T=2
# assume constant situation-feature classification
# and class-specific situation parameters (i.e. model 1)
# use 10 exploratory runs with random starting points
anger.LCplfm.disj<-LCplfm(data=anger$data,F=2, T=2, M=10)

# print the output of the model
print (anger.LCplfm.disj)


# estimate an additive LCplfm model with F=2 and T=2
# assume constant situation-feature classification
# and class-specific situation parameters (i.e. model 1)
# use 10 exploratory runs with random starting points
anger.LCplfm.add<-LCplfm(data=anger$data,F=2, T=2, M=10, maprule="add")

# print the output of the model
print (anger.LCplfm.add)


# estimate a disjunctive LCplfm model with F=4 and T=2
# assume constant situation-feature classifications
# and class-specific situation parameters (i.e. model 1)
# use 20 exploratory runs with random starting points (M=20)
# constrain parameters of subsequent behavior pairs to "load"
# on only one feature

# specify which attribute parameters have to be estimated from the data
update.attribute<-matrix(rep(0,8*4),ncol=4)
update.attribute[1:2,1]<-c(1,1)
update.attribute[3:4,2]<-c(1,1)
update.attribute[5:6,3]<-c(1,1)
update.attribute[7:8,4]<-c(1,1)

# specify starting values for attribute parameters in each of M=20 runs
# for parameters with update.attribute==0 starting values are constrained to 1e-6
# for parameters with update.attribute==1 starting values are sampled from a unif(0,1)
start.attribute<-array(runif(8*4*20),c(8,4,20))
start.attribute[update.attribute%o%rep(1,20)==0]<-1e-6
```

```
# estimate the constrained model
anger.LCplfm.constr<-LCplfm(data=anger$data,F=4, T=2, M=20,
                     update.attributeparameters=update.attribute,
                     start.attributeparameters=start.attribute)

# estimate a disjunctive LCplfm model with F=4 and T=2
# assume constant situation-feature classifications
# class-specific situation and bahavior parameters (i.e. model 3)
# use 20 exploratory runs with random starting points (M=20)
# constrain parameters of subsequent behavior pairs to "load"
# on only one feature

# specify which attribute parameters have to be estimated from the data

update.attribute<-matrix(rep(0,8*4),ncol=4)
update.attribute[1:2,1]<-c(1,1)
update.attribute[3:4,2]<-c(1,1)
update.attribute[5:6,3]<-c(1,1)
update.attribute[7:8,4]<-c(1,1)
update.attribute<-update.attribute%o%rep(1,2)

# specify starting values for attribute parameters in each of M=20 runs
# for parameters with update.attribute==0 starting values are constrained to 1e-6
# for parameters with update.attribute==1 starting values are sampled from a unif(0,1)
start.attribute<-array(runif(8*4*2*20),c(8,4,2,20))
start.attribute[update.attribute%o%rep(1,20)==0]<-1e-6

# estimate the constrained model
anger.LCplfm.m3.constr<-LCplfm(data=anger$data,F=4, T=2, M=20, model=3,
                     update.attributeparameters=update.attribute,
                     start.attributeparameters=start.attribute)


## End(Not run)

## Not run:
# example 2: analysis of car perception data

# load car data
data(car)

# estimate a disjunctive LCplfm with F=3 and T=2
# assume constant attribute-feature classification
# and class-specific car parameters (i.e. model 4)
# use 10 exploratory runs with random starting points
car.LCplfm.disj<-LCplfm(data=car$data3w,F=3, T=2, M=10,model=4)

# print the output of the model
print(car.LCplfm.disj)

# estimate an additive LCplfm with F=3 and T=2
# assume constant attribute-feature classification
# and class-specific car parameters (i.e. model 4)
```

```
# use 10 exploratory runs with random starting points
car.LCplfm.add<-LCplfm(data=car$data3w,F=3, T=2, M=10, model=4, maprule="add")

# print the output of the model
print(car.LCplfm.add)


## End(Not run)

## Not run:

# example 3: estimation of multiple classification latent class
# model (Maris, 1999) for cognitive diagnosis


# load subtraction data
library(CDM)
data(fraction.subtraction.data)
data(fraction.subtraction.qmatrix)


# create three-way data as input for LCplfm
I<-536
J<-1
K<-20
data3w<-array(c(as.matrix(fraction.subtraction.data)),c(I,J,K))

# add item labels

itemlabel<-c("5/3 - 3/4",
"3/4 - 3/8",
"5/6 - 1/9",
"3 1/2 - 2 3/2",
"4 3/5 - 3 4/10",
"6/7 - 4/7",
"3 - 2 1/5",
"2/3 - 2/3",
"3 7/8 - 2",
"4 4/12 - 2 7/12",
"4 1/3 - 2 4/3",
"1 1/8 - 1/8",
"3 3/8 - 2 5/6",
"3 4/5 - 3 2/5",
"2 - 1/3",
"4 5/7 - 1 4/7",
"7 3/5 - 4/5",
"4 1/10 - 2 8/10",
"4 - 1 4/3",
"4 1/3 - 1 5/3")

dimnames(data3w)[[3]]<-itemlabel

# estimate multiple classification latent class model (Maris, 1999)
```

```
set.seed(537982)
subtract.m1.lst<-stepLCplfm(data3w,minF=3,maxF=5,minT=1,maxT=3,model=1,M=20,maprule="conj")


# print BIC values
sumar<-summary(subtract.m1.lst)
as.matrix(sort(sumar[,5]))

# print output best model
subtract.m1.lst[[5,2]]

# correlation between extracted skills and qmatrix
round(cor(fraction.subtraction.qmatrix,subtract.m1.lst[[5,2]]$attpar),2)

## End(Not run)
```

`plfm` *Probabilistic latent feature analysis of two-way two-mode frequency data*

Description

Computation of parameter estimates, standard errors, criteria for model selection, and goodness-of-fit criteria for disjunctive, conjunctive or additive probabilistic latent feature models with F features.

Usage

```
plfm(data,object,attribute,rating,freq1,freqtot,F,
     datatype="freq",maprule="disj",M=5,emcrit1=1e-2,
     emcrit2=1e-10,printrun=TRUE)
```

Arguments

`data`	A data frame that consists of three components: the variables `object`, `attribute` and `rating`. Each row of the data frame describes the outcome of a binary rater judgement about the association between a certain object and a certain attribute.
`object`	The name of the `object` component in the data frame `data`. The values of the vector `data$object` should be (non-missing) numeric or character values.
`attribute`	The name of the `attribute` component in the data frame `data`. The values of the vector `data$attribute` should be (non-missing) numeric or character values.
`rating`	The name of the `rating` component in the data frame `data`. The elements of the vector `data$rating` should be the numeric values 0 (no association) or 1 (association), or should be specified as missing (NA).
`freq1`	A J X K matrix of observed association frequencies.

freqtot	A *J X K* matrix with the total number of binary ratings in each cell *(j,k)*. If the total number of ratings is the same for all cells of the matrix it is sufficient to enter a single numeric value rather than a matrix. For instance, if *N* raters have judged *J X K* associations, one may specify freqtot=*N*
F	The number of latent features included in the model.
datatype	The type of data used as input. When datatype="freq" one should specify frequency data freq1 and freqtot, and when datatype="dataframe" one should specify the name of the data frame data, and its components, object, attribute and rating.
maprule	Disjunctive (maprule="disj"), conjunctive (maprule="conj") or additive (maprule="add") mapping rule of the probabilistic latent feature model.
M	The number of times a particular model is estimated using random starting points.
emcrit1	Convergence criterion which indicates when the estimation algorithm should switch from Expectation-Maximization (EM) steps to EM+Newton-Rhapson steps.
emcrit2	Convergence criterion which indicates final convergence to a local maximum.
printrun	printrun=TRUE prints the analysis type (disjunctive, conjunctive or additive), the number of features (*F*) and the number of the run to the output screen, whereas printrun=FALSE suppresses the printing.

Details

Estimation

The function plfm uses an accelerated EM-algorithm to locate the posterior mode(s) of the probabilistic latent feature model. The algorithm starts with a series of Expectation-Maximization (EM) steps until the difference between subsequent values of the logarithm of the posterior density becomes smaller than the convergence criterion emcrit1, and then switches to an accelerated algorithm which consists of EM + Newton-Rhapson steps. The accelerated algorithm stops when the difference between subsequent values of the logarithm of the posterior density becomes smaller than the convergence criterion emcrit2. Computational details about the implementation of the EM-steps for PLFMs are described in Maris, De Boeck, and Van Mechelen (1996). The general scheme of the accelerated algorithm is described in Louis (1982) and Tanner (1996). Computational details about implementing the accelerated algorithm for PLFMs are described in Meulders (2013).

When using the function plfm to estimate a particular PLFM (i.e. with a certain number of latent features and specific mapping rule), one may locate the distinct posterior mode(s) by running the algorithm M times using random starting points. The estimated object-and attribute parameters of each run are stored in the objects objpar.runs and attpar.runs of the output list. Next, a number of additional statistics (estimated object- and attribute parameters, asymptotic standard errors of object- and attribute parameters, model selecion criteria and goodness-of-fit measures) are computed for the best model (i.e. the model among M runs with the highest posterior density).

Model selection criteria and goodness-of-fit measures

To choose among models with different numbers of features, or with different mapping rules, one may use information criteria such as the Akaike Information Criterion (AIC, Akaike, 1973, 1974), or the Schwarz Bayesian Information Criterion (BIC, Schwarz, 1978). AIC and BIC are computed

as *-2*loglikelihood+k*Npar*. For AIC *k* equals 2 and for BIC *k* equals *log(N)*, with *N* the observed number of replications for which object-attribute associations are collected. *Npar* represents the number of model parameters; for probabilistic latent feature models this equals *(J+K)F*. Models with the lowest value for AIC or BIC should be selected.

To assess the statistical fit of the probabilistic feature model one may use a Pearson chi-square measure on the *J X K* frequency table to evaluate whether predicted frequencies deviate significantly from observed frequencies (see Meulders et al., 2001). In addition, one may assess the descriptive fit of the model using the correlation between observed and expected frequencies in the *J X K* table, and the proportion of the variance in the observed frequencies accounted for by the model (VAF) (i.e. the squared correlation between observed and expected frequencies).

The model selection criteria AIC and BIC, the results of the Pearson goodness-of fit test, and the descriptive fit measures (correlation observed and expected frequencies, and VAF) are stored in the object `fitmeasures` of the output list

Value

`call`	Parameters used to call the function.
`objpar`	A *J X F* matrix of object parameters.
`attpar`	A *K X F* matrix of attribute parameters.
`fitmeasures`	A list of model selection criteria and goodness-of-fit criteria for the model with the highest posterior density.
`logpost.runs`	A list with the logarithm of the posterior density for each of the *M* computed models.
`objpar.runs`	A *M X J X F* array which contains the object parameters for each of the *M* computed models.
`attpar.runs`	A *M X K X F* array which contains the attribute parameters for each of the *M* computed models.
`bestsolution`	An index which indicates the model with the highest posterior density among each of the *M* computed models.
`gradient.objpar`	A *J X F* gradient matrix for the object parameters in the best solution.
`gradient.attpar`	A *K X F* gradient matrix for the attribute parameters in the best solution.
`SE.objpar`	A *J X F* matrix of asymptotic standard errors for the object parameters in the best solution.
`SE.attpar`	A *K X F* matrix of asymptotic standard errors for the attribute parameters in the best solution.
`prob1`	A *J X K* matrix of expected association probabilities for the best solution.

Author(s)

Michel Meulders

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki (Eds.), *Second international symposium on information theory* (p. 271-283). Budapest: Academiai Kiado.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*, 716-723.

Candel, M. J. J. M., and Maris, E. (1997). Perceptual analysis of two-way two-mode frequency data: probability matrix decomposition and two alternatives. *International Journal of Research in Marketing, 14*, 321-339.

Louis, T. A. (1982). Finding observed information using the em algorithm. *Journal of the Royal Statistical Society, Series B, 44*, 98-130.

Maris, E., De Boeck, P., and Van Mechelen, I. (1996). Probability matrix decomposition models. *Psychometrika, 61*, 7-29.

Meulders, M., De Boeck, P., and Van Mechelen, I. (2001). Probability matrix decomposition models and main-effects generalized linear models for the analysis of replicated binary associations. *Computational Statistics and Data Analysis, 38*, 217-233.

Meulders, M., De Boeck, P., Van Mechelen, I., Gelman, A., and Maris, E. (2001). Bayesian inference with probability matrix decomposition models. *Journal of Educational and Behavioral Statistics, 26*, 153-179.

Meulders, M., De Boeck, P., Van Mechelen, I., and Gelman, A. (2005). Probabilistic feature analysis of facial perception of emotions. *Applied Statistics, 54*, 781-793.

Meulders, M. and De Bruecker, P. (2018). Latent class probabilistic latent feature analysis of three-way three-mode binary data. *Journal of Statistical Software, 87(1)*, 1-45.

Meulders, M. (2013). An R Package for Probabilistic Latent Feature Analysis of Two-Way Two-Mode Frequencies. *Journal of Statistical Software, 54(14)*, 1-29. URL http://www.jstatsoft.org/v54/i14/.

Tanner, M. A. (1996). *Tools for statistical inference: Methods for the exploration of posterior distributions and likelihood functions* (Third ed.). New York: Springer-Verlag.

Schwarz, G. (1978). Estimating the dimensions of a model. *Annals of Statistics, 6*, 461-464.

See Also

`gendat`, `print.plfm`, `summary.plfm`, `print.summary.plfm`

Examples

```
## Not run:
## example 1: Analysis of data generated under the model

# define constants
J<-20
K<-15
F<-2

# generate true parameters
set.seed(43565)
objectparameters<-matrix(runif(J*F),nrow=J)
```

```
attributeparameters<-matrix(runif(K*F),nrow=K)

# generate data for disjunctive model using N=200 replications
gdat.disj<-gendat(maprule="disj",N=200,
              objpar=objectparameters,attpar=attributeparameters)

# Estimate a disjunctive probabilistic latent feature model with 2 features
# suppress printing the type of analysis to the output screen
disj2<-plfm(maprule="disj",freq1=gdat.disj$freq1,freqtot=200,F=2,M=1,printrun=FALSE)

# generate data for an additive model using N=200 replications
gdat.add<-gendat(maprule="add",N=200,
              objpar=objectparameters,attpar=attributeparameters)

# Estimate an additive probabilistic latent feature model with 2 features
# suppress printing the type of analysis to the output screen
add2<-plfm(maprule="add",freq1=gdat.add$freq1,freqtot=200,F=2,M=1,printrun=FALSE)

## End(Not run)


## Not run:
# example 2:Perceptual analysis of associations between car models and car attributes

# load car data
data(car)

# compute 1 run of a disjunctive model with 4 features
# use components of a data frame as input
cardisj4<-plfm(datatype="dataframe",data=car$datalongformat,object=objectlabel,
               attribute=attributelabel,rating=rating,maprule="disj",F=4,M=1)

# print the output of a disjunctive 4-feature model
# for data on the perception of car models
print (cardisj4)


# print a summary of the output of a disjunctive 4-feature model
# for data on the perception of car models
sumcardisj4<-summary(cardisj4)
sumcardisj4

## End(Not run)

## Not run:
# example 3: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 1 run of a disjunctive model with 2 features
# use frequency data as input
angerdisj2<-plfm(maprule="disj",freq1=anger$freq1,freqtot=anger$freqtot,F=2,M=1)
```

```
# print the output of a disjunctive 2-feature model
# for data on the situational determinants of anger-related behaviors
print (angerdisj2)


# print a summary of the output of a disjunctive 2-feature model
# for data on the situational determinants of anger-related behaviors
sumangerdisj2<-summary(angerdisj2)
sumangerdisj2

## End(Not run)
```

`plot.LCplfm` *plot parameters in* `LCplfm` *object*

Description

Plot method to visualize the parameters of latent class probabilistic feature models with different numbers of features/classes.

Usage

```
## S3 method for class 'LCplfm'
plot(x, feature=1, class=0, element="object", cexsymb=1, cexlabel=1,
                minpositionlabel = -1, positionlabel = -0.8, xlegend = "topright",
                       ylegend=NULL, x.intersplegend=1, y.intersplegend=1, ...)
```

Arguments

`x`	Latent class probabilistic feature model object returned by `LCplfm`.
`feature`	Latent feature for which parameters are visualized.
`class`	Latent class for which parameters are visualized. When the model contains class-specific object- or attribute parameters, `class=0` means that parameters of all classes are included in the plot.
`element`	Object parameters are plotted if `element`="object" and attribute parameters are plotted if `element`="attribute".
`cexsymb`	Size of symbol used for plotting points.
`cexlabel`	Size of object- or attribute labels in plot.
`minpositionlabel`	Value smaller than 0 that defines space for plotting object- or attribute labels.
`positionlabel`	Value between `minpositionlabel` and 0 to align object- or attribute labels.

xlegend, ylegend
: The x and y co-ordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by xy.coords: See "Details" of legend.

x.intersplegend
: Character interspacing factor for horizontal (x) spacing in legend.

y.intersplegend
: Character interspacing factor for vertical (y) line distances in legend.

...
: Further arguments are ignored.

Examples

```
## Not run:
# example 1: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 5 runs of disjunctive latent class probabilistic feature model
# with 4 features and 2 latent classes
# assume constant situation classification per person
# and class-specific situation parameters (i.e. model=1)

anger.m1<-LCplfm(data=anger$data,F=4,T=2,maprule="disj",
                 M=5,emcrit1=1e-3,emcrit2=1e-8,model=1)

# visualize object and attribute parameters
# of both classes per feature in one figure

par(mfrow=c(2,2),pty="s")
plot(anger.m1,element="attribute",feature=1, main="Feature 1",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m1,element="attribute",feature=2, main="Feature 2",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m1,element="attribute",feature=3, main="Feature 3",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m1,element="attribute",feature=4, main="Feature 4",
     minpositionlabel=-2, positionlabel=-1)

par(mfrow=c(2,2),pty="s")
plot(anger.m1,element="object",feature=1,main="Feature 1",
     minpositionlabel=-1.5, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m1,element="object",feature=2,main="Feature 2",
     minpositionlabel=-1.5, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m1,element="object",feature=3,main="Feature 3",
     minpositionlabel=-1.5, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m1,element="object",feature=4,main="Feature 4",
     minpositionlabel=-1.5, positionlabel=-1, y.intersplegend=0.7)

# compute 5 runs of disjunctive latent class probabilistic feature model
# with 2 features and 2 latent classes
# assume constant situation classification per person
```

```
# and class-specific situation and behavior parameters (i.e. model=3)

anger.m3<-LCplfm(data=anger$data,F=2,T=2,maprule="disj",
                 M=5,emcrit1=1e-3,emcrit2=1e-8,model=3)

# visualize object and attribute parameters of feature 1,2
# for class 1
par(mfrow=c(2,2))
plot(anger.m3,element="attribute",feature=1, class=1,main="Feature 1, class 1",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m3,element="attribute",feature=2, class=1,main="Feature 2, class 1",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m3,element="object",feature=1, class=1,main="Feature 1, class 1",
     minpositionlabel=-2, positionlabel=-1)
plot(anger.m3,element="object",feature=2, class=1,main="Feature 2, class 1",
     minpositionlabel=-2, positionlabel=-1)


# visualize object and attribute parameters of feature 1,2
# for class 2
par(mfrow=c(2,2))
plot(anger.m3,element="attribute",feature=1, class=2,main="Feature 1, class 2",
     minpositionlabel=-1.7, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m3,element="attribute",feature=2, class=2,main="Feature 2, class 2",
     minpositionlabel=-1.7, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m3,element="object",feature=1, class=2,main="Feature 1, class 2",
     minpositionlabel=-1.7, positionlabel=-1, y.intersplegend=0.7)
plot(anger.m3,element="object",feature=2, class=2,main="Feature 2, class 2",
     minpositionlabel=-1.7, positionlabel=-1, y.intersplegend=0.7)


## End(Not run)
```

`plot.plfm` *plot parameters in* `plfm` *object*

Description

Plot method to visualize the parameters of probabilistic feature models.

Usage

```
## S3 method for class 'plfm'
plot(x,feature=1,element="object",cexsymb=1,cexlabel=1,...)
```

Arguments

`x`	Probabilistic feature model object returned by `plfm`.
`feature`	Latent feature for which parameters are visualized.

element	Object parameters are plotted if element="object" and attribute parameters are plotted if element="attribute".
cexsymb	Size of symbol used for plotting points.
cexlabel	Size of object- or attribute labels in plot.
...	Further arguments are ignored.

Examples

```
# examples


## Not run:
# example 1:Perceptual analysis of associations between car models and car attributes

# load car data
data(car)

# compute 1 run of a disjunctive model with 4 features
# use components of a data frame as input
cardisj4<-plfm(datatype="dataframe",data=car$datalongformat,object=objectlabel,
               attribute=attributelabel,rating=rating,maprule="disj",F=4,M=1)

# plot car and attribute parameters per feature
par(mfrow=c(1,2))
plot(cardisj4,feature=1,element="object",main="Car parameters Feature 1")
plot(cardisj4,feature=1,element="attribute",main="Attribute parameters Feature 1")

par(mfrow=c(1,2))
plot(cardisj4,feature=2,element="object",main="Car parameters Feature 2")
plot(cardisj4,feature=2,element="attribute",main="Attribute parameters Feature 2")

par(mfrow=c(1,2))
plot(cardisj4,feature=3,element="object",main="Car parameters Feature 3")
plot(cardisj4,feature=3,element="attribute",main="Attribute parameters Feature 3")

par(mfrow=c(1,2))
plot(cardisj4,feature=4,element="object",main="Car parameters Feature 4")
plot(cardisj4,feature=4,element="attribute",main="Attribute parameters Feature 4")

## End(Not run)



par(mfrow=c(1,2))

# example 2: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 1 run of a disjunctive model with 4 features
# use frequency data as input
```

```
angerdisj2<-plfm(maprule="disj",freq1=anger$freq1,freqtot=anger$freqtot,F=2,M=1)

# plot situation and behavior parameters
par(mfrow=c(2,2))
for (f in 1:2){
plot(angerdisj2,feature=f,element="object",main=paste("Situation parameters Feature",f,sep=" "))}
for (f in 1:2){
plot(angerdisj2,feature=f,element="attribute",main=paste("Behavior parameters Feature",f,sep=" "))}
```

`plot.stepLCplfm` *Plot fit of* `stepLCplfm` *objects*

Description

Plot method to visualize the fit of latent class probabilistic feature models with different numbers of features/classes.

Usage

```
## S3 method for class 'stepLCplfm'
plot(x,which="BIC",...)
```

Arguments

`x`	List of latent class probabilistic latent feature analysis objects returned by `stepLCplfm`.
`which`	Fit criterion for which models with different numbers of features are compared. The argument `which` can take the following values: "AIC", "BIC", "Deviance", "Correlation", "VAF"
`...`	Further arguments are ignored.

Examples

```
## Not run:
# example 1: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 5 runs of disjunctive latent class probabilistic feature models
# with 1 up to 3 features and with 1 up to 2 latent classes
# assume constant situation classification per person
# and class-specific situation parameters (i.e. model=1)

anger.lst<-stepLCplfm(minF=1,maxF=3,minT=1,maxT=2,data=anger$data,
                      maprule="disj",M=5,emcrit1=1e-3,emcrit2=1e-8,model=1)


# visualize BIC of fitted models
```

```
par(pty="s")
plot(anger.lst)

# print overview fit measures for all estimated models

anger.lst

# print model with 3 features and 1 latent class

anger.lst[[3,1]]

## End(Not run)

## Not run:
# example 2:Perceptual analysis of associations between car models and car attributes

# load car data
data(car)


# compute 5 runs of disjunctive models with 4 features and 1 up to 3 latent classes
# assume constant attribute classification per respondent
# and class-specific car parameters (i.e. model 4)

car.lst<-stepLCplfm(minF=4,maxF=4,minT=1,maxT=3,data=car$data3w,
                    maprule="disj",M=5,emcrit1=1e-3,emcrit2=1e-8,model=4,printrun=TRUE)


# visualize BIC of fitted models
plot(car.lst)

# print overview of fitmeasures for all fitted models
car.lst

## End(Not run)
```

`plot.stepplfm` *Plot fit of* `stepplfm` *objects*

Description

Plot method to visualize the fit of probabilistic latent feature analysis objects with different numbers of features.

Usage

```
## S3 method for class 'stepplfm'
plot(x,which="BIC",...)
```

Arguments

x	List of probabilistic latent feature analysis objects returned by stepplfm.
which	Fit criterion for which models with different numbers of features are compared. The argument which can take the following values: "AIC", "BIC", "Deviance", "Chisquare", "Correlation", "VAF"
...	Further arguments are ignored.

Examples

```
## Not run:
## example 1:Perceptual analysis of associations between car models and car attributes

##load car data
data(car)


## compute 5 runs of disjunctive and conjunctive models with 1 up to 4 features
car.lst<-stepplfm(minF=1,maxF=4,maprule="disj/conj",freq1=car$freq1,
                  freqtot=car$freqtot,M=5)

## visualize the fit of models with different mapping rules
## and a different number of features

par(pty="s")
par(mfrow=c(2,2))
plot(car.lst,which="BIC")
plot(car.lst,which="AIC")
plot(car.lst,which="VAF")


## End(Not run)

## Not run:
## example 2: analysis on determinants of anger-related behavior

## load anger data
data(anger)

## compute 1 run of disjunctive and conjunctive models with 1 up to 3 features
anger.lst<-stepplfm(minF=1,maxF=3,maprule="disj/conj",freq1=anger$freq1,
                    freqtot=anger$freqtot,M=1)

## visualize the fit of models with different mapping rules
## and a different number of features

par(pty="s")
par(mfrow=c(2,2))
plot(anger.lst,which="BIC")
plot(anger.lst,which="AIC")
plot(anger.lst,which="VAF")
```

```
## End(Not run)
```

---

print.bayesplfm *Printing* bayesplfm *objects*

Description

Printing method for objects generated by the bayesplfm function.

Usage

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

Arguments

x	Object returned by bayesplfm.
...	Further arguments are ignored.

Details

The printing method for Bayesian probabilistic latent feature analysis objects displays (1) the parameters used to call the bayesplfm function, (2) the number of parameters that do not meet the convergence criterion (3) information on the descriptive fit of the model (i.e. correlation between observed and expected frequencies, and proportion of the variance in the observed frequencies accounted for by the model), and (4) the posterior mean of the object- and attribute parameters.

See Also

bayesplfm,summary.bayesplfm,print.summary.bayesplfm

Examples

```
## Not run:
##load car data
data(car)


## Compute a sample of the posterior distribution
## for the disjunctive model with two features
## compute the starting point using plfm
carbayes2<-bayesplfm(maprule="disj",freq1=car$freq1,freqtot=car$freqtot,F=2,
                     maxNiter=500,Nburnin=0,Nstep=100,Nchains=2,
                     start.bayes="best")


## print the object generated by bayesplfm
carbayes2
```

```
## End(Not run)
```

`print.LCplfm` *Printing LCplfm objects*

Description

Printing method for latent class probabilistic feature analysis objects.

Usage

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

Arguments

`x`	Latent class probabilistic feature analysis object returned by `LCplfm`.
`...`	Further arguments are ignored.

Details

The printing method for latent class probabilistic feature analysis objects displays (1) the parameters used to call the `LCplfm` function, (2) the estimated object-, attribute- and class size parameters, (3) the estimated standard errors of object-, attribute- and class size parameters, (4) fit measures

Examples

```
## Not run:
# example
print(LCplfm(data=anger$data,F=2, T=2, M=1))

## End(Not run)
```

`print.plfm` *Printing plfm objects*

Description

Printing method for probabilistic latent feature analysis objects.

Usage

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

Arguments

x	Probabilistic latent feature analysis object returned by plfm.
...	Further arguments are ignored.

Details

The printing method for probabilistic latent feature analysis objects displays (1) the parameters used to call the plfm function, (2) information on the descriptive fit of the model (i.e. correlation between observed and expected frequencies, and proportion of the variance in the observed frequencies accounted for by the model), and (3) the estimated object- and attribute parameters.

Examples

```
## example print.plfm(plfm(...))
```

---

print.stepLCplfm *Printing* stepLCplfm *objects*

Description

Printing method for a series of latent class probabilistic latent feature analysis objects.

Usage

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

Arguments

x	Latent class probabilistic latent feature analysis object returned by stepLCplfm.
...	Further arguments are ignored.

Details

The printing method for stepLCplfm displays summary tables about the fit of models with different numbers of features and different numbers of latent classes. Two tables are printed which summarize the fit of models with different numbers of features/classes in terms of (1) information criteria (AIC, BIC,...), and (2) the descriptive fit of the model to the *JXK* frequency table (correlation between observed and expected frequencies, and variance accounted for by the model).

Examples

```
# example print.stepLCplfm(stepLCplfm(...))
```

print.stepplfm *Printing* stepplfm *objects*

Description

Printing method for a series of probabilistic latent feature analysis objects.

Usage

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

Arguments

x	probabilistic latent feature analysis object returned by stepplfm.
...	Further arguments are ignored.

Details

The printing method for stepplfm displays summary tables about the fit of models with different numbers of features. For each mapping rule, three tables are printed which summarize the fit of models with different numbers of features in terms of (1) information criteria (AIC, BIC,...), (2) the statistical fit of the model to the *JXK* frequency table (Chi-square value, df and corresponding p-value), and (3) the descriptive fit of the model to the *JXK* frequency table (correlation between observed and expected frequencies, and variance accounted for by the model).

Examples

```
## example print.stepplfm(stepplfm(...))
```

print.summary.bayesplfm
Printing summaries of Bayesian probabilistic latent feature analysis

Description

Printing method for summaries of objects generated by summary.bayesplfm

Usage

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

Arguments

x	Summary of Bayesian probabilistic latent feature analysis returned by summary.bayesplfm
...	Further arguments are ignored

See Also

summary.bayesplfm, bayesplfm

---

print.summary.plfm *Printing summaries of probabilistic latent feature analysis objects*

Description

Printing method for summaries of probabilistic latent feature analysis objects.

Usage

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

Arguments

,

x	Summary of a probabilistic latent feature analysis object returned by summary.plfm
...	Further arguments are ignored

See Also

plfm, summary.plfm

---

print.summary.stepLCplfm *Printing summaries of* stepLCplfm *objects*

Description

Printing method for summaries of stepLCplfm objects.

Usage

```
## S3 method for class 'summary.stepLCplfm'
print(x,digits=2,...)
```

Arguments

x	Summary of a list of latent class probabilistic latent feature analysis objects returned by summary.stepLCplfm.
digits	By default 2 significant digits are used for printing.
...	Further arguments are ignored.

See Also

stepLCplfm, summary.stepLCplfm

---

print.summary.stepplfm

Printing summaries of stepplfm *objects*

Description

Printing method for summaries of stepplfm objects.

Usage

```
## S3 method for class 'summary.stepplfm'
print(x,digits=2,...)
```

Arguments

x	Summary of a list of probabilistic latent feature analysis object returned by summary.stepplfm.
digits	By default 2 significant digits are used for printing.
...	Further arguments are ignored.

See Also

stepplfm, summary.stepplfm

stepLCplfm *Latent class probabilistic latent feature analysis of three-way three-mode binary data*

Description

The function stepLCplfm subsequently applies the LCplfm function to fit disjunctive, conjunctive or additive models with *minF* up to *maxF* latent features and *minT* to *maxT* latent classes. The results of the estimated models are stored in a list with *F X T* components.

Usage

```
stepLCplfm(minF=1,maxF=3,minT=1,maxT=3,
                  data,maprule="disj",M=5,emcrit1=1e-3,emcrit2=1e-8,
                  model=1,delta=0.0001,printrun=FALSE,Nbootstrap=2000)
```

Arguments

minF	Minimum number of latent features included in the model.
maxF	Maximum number of latent features included in the model.
minT	Minimum number of latent classes included in the model.
maxT	Maximum number of latent classes included in the model.
data	A *I X J X K* data array of binary observations. Observation *(i,j,k) (i=1,..,I; j=1,..,J; k=1,..,K)* indicates whether object *j* is associated to attribute *k* according to rater *i*.
maprule	Fit disjunctive models (maprule="disj"), conjunctive models (maprule="conj") or additive models (maprule="add") .
M	The number of exploratory runs of the EM algorithm using random starting points for each model.
emcrit1	Convergence criterion to be used for the estimation of candidate models in the exploration step.
emcrit2	Convergence criterion to be used for the estimation of the best model in the final analysis.
model	The type of dependency and heterogeneity assumption included in the model. model=1, model=2, model=3 represent models with a constant object-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters. model=4, model=5, model=6 represent models with a constant attribute-feature classification per person and with, respectively, class-specific object parameters, class-specific attribute parameters, and class-specific object- and attribute parameters.
delta	The precision used to compute standard errors of the model parameters with the method of finite differences.

printrun	printrun=TRUE prints the analysis type (disjunctive or conjunctive), the number of features (F), the number of latent classes (T) and the number of the run to the output screen, whereas printrun=FALSE suppresses the printing.
Nbootstrap	Number of bootstrap iterations to be used for simulating the reference distribution of odds-ratio dependency measures.

Details

The results of subsequent LCplfm analyses are stored in a matrix of lists with (maxF-minF+1,maxT-minT+1) components.

Author(s)

Michel Meulders

Examples

```
## Not run:
# example 1: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 5 runs of disjunctive latent class probabilistic feature models
# with 1 up to 3 features and with 1 up to 2 latent classes
# assume constant situation classification per person
# and class-specific situation parameters (i.e. model=1)

anger.lst<-stepLCplfm(minF=1,maxF=3,minT=1,maxT=2,data=anger$data,
                      maprule="disj",M=5,emcrit1=1e-3,emcrit2=1e-8,model=1)


# visualize BIC of fitted models

par(pty="s")
plot(anger.lst)

# print overview fit measures for all estimated models

anger.lst

# print model with 3 features and 1 latent class

anger.lst[[3,1]]

## End(Not run)

## Not run:
# example 2:Perceptual analysis of associations between car models and car attributes

# load car data
data(car)
```

```
# compute 5 runs of disjunctive models with 4 features and 1 up to 3 latent classes
# assume constant attribute classification per respondent
# and class-specific car parameters (i.e. model 4)

car.lst<-stepLCplfm(minF=4,maxF=4,minT=1,maxT=3,data=car$data3w,
                    maprule="disj",M=5,emcrit1=1e-3,emcrit2=1e-8,model=4,printrun=TRUE)


# visualize BIC of fitted models
plot(car.lst)

# print overview of fitmeasures for all fitted models
car.lst

## End(Not run)
```

`stepplfm` *Probabilistic latent feature analysis of two-way two-mode frequency data*

Description

The function `stepplfm` subsequently applies the `plfm` function to fit disjunctive, conjunctive or additive models with *minF* up to *maxF* latent features. The results of the different models are stored in a list.

Usage

```
stepplfm(minF,maxF,data,object,attribute,
  rating,freq1,freqtot,datatype="freq",
maprule="disj",M=5,emcrit1=1e-2,emcrit2=1e-10,
        printrun=TRUE)
```

Arguments

`minF`	Minimum number of features to be fitted
`maxF`	Maximum number of features to be fitted
`data`	A data frame that consists of three components: the variables `object`, `attribute` and `rating`. Each row of the data frame describes the outcome of a binary rater judgement about the association between a certain object and a certain attribute.
`object`	The name of the `object` component in the data frame `data`. The values of the vector `data$object` should be (non-missing) numeric or character values.
`attribute`	The name of the `attribute` component in the data frame `data`. The values of the vector `data$attribute` should be (non-missing) numeric or character values.

rating	The name of the rating component in the data frame data. The elements of the vector data$rating should be the numeric values 0 (no association) or 1 (association), or should be specified as missing (NA).
freq1	*J X K* matrix of observed association frequencies.
freqtot	A *J X K* matrix with the total number of binary ratings in each cell *(j,k)*. If the total number of ratings is the same for all cells of the matrix it is sufficient to enter a single numeric value rather than a matrix. For instance, if *N* raters have judged *J X K* associations, one may specify freqtot=*N*.
datatype	The type of data used as input. When datatype="freq" one should specify frequency data freq1 and freqtot, and when datatype="dataframe" one should specify the name of the data frame data, and its components, object, attribute and rating.
maprule	Fit disjunctive models (maprule="disj"), conjunctive models (maprule="conj"), or additive models (maprule="add"). Multiple mapping rules can be computed in one analysis. For instance, to compute both disjunctive and conjunctive models one my use maprule="disj/conj". Other combinations are maprule="disj/add", maprule="conj/add" and maprule="disj/conj/add".
M	The number of times a particular model is estimated using random starting points.
emcrit1	Convergence criterion which indicates when the estimation algorithm should switch from Expectation-Maximization (EM) steps to EM+Newton-Rhapson steps.
emcrit2	Convergence criterion which indicates final convergence to a local maximum.
printrun	printrun=TRUE prints the analysis type (disjunctive, conjunctive, additive), the number of features (F) and the number of the run to the output screen, whereas printrun=FALSE suppresses the printing.

Details

When only one type of mapping rule is requested (disjunctive, conjunctive, additive) (i.e., maprule="disj", maprule="conj" or maprule="add") with minF to maxF features, the results of subsequent plfm analyses are stored in a list with maxF-minF+1 components. When analyses with multiple mapping rules are requested (e.g. maprule="disj/conj"), the results for each mapping rule are stored in a list with maxF-minF+1 components (e.g., two lists named "disj" and "conj", respectively) . The final object generated by stepplfm combines the lists "disj" and "conj" in a list with two components.

Author(s)

Michel Meulders

Examples

```
## Not run:
# example 1:Perceptual analysis of associations between car models and car attributes

# load car data
```

```
data(car)


# compute 5 runs of disjunctive and conjunctive models with 1 up to 4 features
car.lst<-stepplfm(minF=1,maxF=4,maprule="disj/conj",freq1=car$freq1,
                  freqtot=car$freqtot,M=5,emcrit1=1e-6)

# print output of the conjunctive model with 4 features
car.lst$conj[[4]]

# print output of the stepplfm analysis on the car data
car.lst

# summarize and print output of stepplfm analysis on car data
sumcar<-summary(car.lst)
sumcar

# visualize fit of models with different mapping rules and a different number of features
par(pty="s")
par(mfrow=c(2,2))
plot(car.lst,which="BIC")
plot(car.lst,which="AIC")
plot(car.lst,which="VAF")

## End(Not run)


## Not run:
# example 2: analysis on determinants of anger-related behavior

# load anger data
data(anger)

# compute 1 run of disjunctive models with 1 up to 3 features
anger.lst<-stepplfm(minF=1,maxF=3,maprule="disj",freq1=anger$freq1,freqtot=anger$freqtot,M=1)

# print output of disjunctive model with 2 features

anger.lst[[2]]

# print output of stepplfm analysis on anger data
anger.lst

# summarize and print output of stepplfm analysis on anger data
sumanger<-summary(anger.lst)
sumanger

# visualize fit of models with different mapping rules and a different number of features

par(pty="s")
par(mfrow=c(2,2))
plot(anger.lst,which="BIC")
plot(anger.lst,which="AIC")
```

```
plot(anger.lst,which="VAF")

## End(Not run)
```

---

summary.bayesplfm *Summarizing Bayesian probabilistic latent feature analysis*

Description

The function summary.bayesplfm summarizes the output of the object generated by the bayesplfm function.

Usage

```
## S3 method for class 'bayesplfm'
summary(object, ...)
```

Arguments

object	Bayesian probabilistic latent feature analysis object returned by bayesplfm
...	Further arguments are ignored

Details

The summary of the Bayesian probabilistic latent feature analysis objects displays:

1. The parameters used to call the bayesplfm function.
2. Information on the descriptive fit of the model (i.e. correlation between observed and expected frequencies, and proportion of the variance in the observed frequencies accounted for by the model).
3. The posterior mean of the object- and attribute parameters.
4. 95 percent posterior intervals for the object- and attribute parameters.
5. Rhat convergence values for object- and attribute parameters (if Nchains>1).

Value

call	Parameters used to call the function.
descriptivefit	A list with two measures of descriptive fit on the J X K table: (1) the correlation between observed and expected frequencies, and (2) the proportion of the variance in the observed frequencies accounted for by the model.
objpar	A J X F matrix with the posterior mean of the object parameters computed on all iterations and chains in the sample.
attpar	A K X F matrix with the posterior mean of the attribute parameters computed on all iterations and chains in the sample.
p95objpar	95 percent posterior intervals of object parameters.
p95attpar	95 percent posterior intervals of attribute parameters.
Rhatobjpar	Rhat convergence values for object parameters.
Rhatattpar	Rhat convergence values for attribute parameters.

See Also

bayesplfm

Examples

```
## Not run:

##load car data
data(car)

## compute 5 runs of disjunctive model with 2 features
carem2<-plfm(maprule="disj",freq1=car$freq1,freqtot=car$freqtot,F=2,M=5)

## Compute a sample of the posterior distribution
## for the disjunctive model with two features
## use the posterior mode obtained with the previous plfm analysis
carbayes2<-bayesplfm(maprule="disj",freq1=car$freq1,freqtot=car$freqtot,F=2,
                     maxNiter=500,Nburnin=0,Nstep=100,Nchains=2,
                     start.bayes="fitted.plfm",fitted.plfm=carem2)


## compute a summary of the object generated by bayesplfm
summarycarbayes2<-summary(carbayes2)

## End(Not run)
```

---

summary.plfm *Summarizing probabilistic latent feature analysis*

Description

The function summary.plfm summarizes the main output of plfm including estimates and standard errors for object- and attribute parameters, model selection criteria, and goodness-of-fit measures.

Usage

```
## S3 method for class 'plfm'
summary(object, ...)
```

Arguments

object	Probabilistic latent feature analysis object returned by plfm
...	Further arguments are ignored

Details

The summary of probabilistic latent feature analysis objects displays:

1. The parameters used to call the `plfm` function.
2. The value of the loglikelihood, the deviance, the logarithm of the posterior density, the information criteria AIC and BIC.
3. The result of a Pearson chi-square goodness-of-fit test on the J X K table.
4. Information on the descriptive fit of the model (i.e. correlation between observed and expected frequencies. and proportion of the variance in the observed frequencies accounted for by the model).
5. The estimated object- and attribute parameters.
6. Asymptotic standard errors of the object- and attribute parameters.

Value

`call`	Parameters used to call the function.
`informationcriteria`	List of information criteria that can be used for model selection.
`chisquaretest`	Pearson Chi-square test to evaluate the statistical goodness-of-fit of the model on the J X K object by attribute table of association frequencies.
`descriptivefit`	A list of measures to evaluate the descriptive goodness-of-fit of the model on the J X K object by attribute table of association frequencies.
`objpar`	A J X F matrix of estimated object parameters.
`SE.objpar`	A J X F matrix of estimated standard errors of object parameters.
`attpar`	A K X F matrix of estimated attribute parameters.
`SE.attpar`	A K X F matrix of estimated standard errors of attribute parameters

Author(s)

Michel Meulders

See Also

`plfm`, `print.plfm`, `print.summary.plfm`

Examples

```
## Perceptual analysis of associations between car models and car attributes

##load car data
data(car)

##compute the disjunctive model with 4 features
carf4<-plfm(maprule="disj",freq1=car$freq1,freqtot=car$freqtot,F=4,M=1)

## display a summary of the results
summary(carf4)
```

summary.stepLCplfm *Summary method for* stepLCplfm *objects*

Description

The function summary.stepLCplfm summarizes the fit measures of a series of LCplfm objects.

Usage

```
## S3 method for class 'stepLCplfm'
summary(object, ...)
```

Arguments

object	Latent class probabilistic latent feature analysis object returned by stepLCplfm
...	Further arguments are ignored

Details

The summary of the stepLCplfm generates a table of fit measures (information criteria, descriptive goodness-of-fit measures, statistical dependency measures) for models with different numbers of latent class/features that are involved in the analysis.

Author(s)

Michel Meulders

See Also

stepLCplfm, print.stepLCplfm

summary.stepplfm *Summary method for* stepplfm *objects*

Description

The function summary.stepplfm summarizes the fit measures of a series of plfm objects.

Usage

```
## S3 method for class 'stepplfm'
summary(object, ...)
```

Arguments

object	Probabilistic latent feature analysis object returned by stepplfm
...	Further arguments are ignored

Details

The summary of the stepplfm generates a table of fit measures (information criteria, Chi-square fit on the *J X K* table, and descriptive fit measures) for models with different numbers of features and/or mapping rules that are involved in the analysis.

Author(s)

Michel Meulders

See Also

stepplfm, print.stepplfm, print.summary.stepplfm

Index