Package ‘randomLCA’

July 23, 2017

Type Package
Title Random Effects Latent Class Analysis
Version 1.0-12
Date 2017-07-20
Maintainer Ken Beath <ken.beath@mq.edu.au>
Contact Ken Beath <ken.beath@mq.edu.au>
Author Ken Beath [aut, cre]
Description Fits standard and random effects latent class models. The single level random effects model is described in Qu et al <doi:10.2307/2533043> and the two level random effects model in Beath and Heller <doi:10.1177/1471082X0800900302>. Examples are given for their use in diagnostic testing.
Depends R(>= 3.2.0), lattice
Suggests R.rsp
VignetteBuilder R.rsp
Imports boot, fastGHQuad, Matrix
LazyLoad yes
LazyData yes
NeedsCompilation yes
License GPL (>= 2)
Repository CRAN
Date/Publication 2017-07-23 16:24:46 UTC

R topics documented:

AIC 2
AIC3 3
BIC 3
calcCond2Prob 4
calcCondProb 4
calcMargProb 5
Returns AIC for a randomLCA object.

Usage

```r
# S3 method for class 'randomLCA'
AIC(object, ..., k = 2)
```

Arguments

- `object`: randomLCA object
- `...`: additional argument; currently none is used.
- `k`: penalty per parameter

Value

AIC.

Author(s)

Ken Beath
**AIC3**

*AIC with 3 penalty for randomLCA object*

**Description**

Returns AIC with penalty 3 for a randomLCA object.

**Usage**

`AIC3(object)`

**Arguments**

- `object`: randomLCA object

**Value**

AIC3.

**Author(s)**

Ken Beath

---

**BIC**

*BIC for randomLCA object*

**Description**

Returns BIC for a randomLCA object.

**Usage**

```r
## S3 method for class 'randomLCA'
BIC(object, ...)
```

**Arguments**

- `object`: randomLCA object
- `...`: additional argument; currently none is used.

**Value**

BIC

**Author(s)**

Ken Beath
**calcCond2Prob**  
*Calculate Conditional Outcome Probabilities for 2 Level Models*

**Description**

The conditional probabilities are obtained integrating over the period random effect.

**Usage**

```r
calcCond2Prob(object, conditionalp = 0.5)
```

**Arguments**

- `object` RandomLCA object
- `conditionalp` the percentiles for the random effect

**Value**

Returns a data frame containing class, block, outcome, outcomep (outcome probability) and perc (percentiles of the random effect) if conditionalp is specified. For example a conditionalp of 0.5 is the 50th percentile or the median corresponding to a random effect of zero. 0.025 and 0.975 correspond to the 2.5th and 97.5th percential, so the region between them if 95% of the variation in the data.

**Author(s)**

Ken Beath <kenbeath@mq.edu.au>

---

**calcCondProb**  
*Calculate Conditional Outcome Probabilities*

**Description**

Calculates the conditional outcome probabilities for random effects models or for standard latent class returns the outcome probabilities. For random effects, the outcome probabilities may be calculated for various percentiles of the random effect.

**Usage**

```r
calcCondProb(object, conditionalp = 0.5)
```

**Arguments**

- `object` RandomLCA object
- `conditionalp` the percentiles for the random effect
calcMargProb

Value

Returns a data frame containing class, block, outcome, outcomep (outcome probability) and perc (percentiles of the random effect) if conditionalp is specified. For example a conditionalp of 0.5 is the 50th percentile or the median corresponding to a random effect of zero. 0.025 and 0.975 correspond to the 2.5th and 97.5th percential, so the region between them is 95% of the variation in the data.

Author(s)

Ken Beath <ken.beath@mq.edu.au>

calcMargProb  Calculates Marginal Outcome Probabilities

Description

Calculates the marginal outcome probabilities for a random effects latent class model, by integrating the outcome probability over the random effect. This is performed using Gauss-Hermite quadrature with the number of quadrature points specified for the model fitting.

Usage

calcMargProb(object)

Arguments

object  randomLCA object

Value

Returns a data frame containing class, block, outcome, outcomep (outcome probability).

Author(s)

Ken Beath
classProbs

Determines class probabilities for fitted model

Description

The class probabilities for the model are returned.

Usage

classProbs(object)

Arguments

object randomLCA object

Details

Simply extracts the corresponding variable from the randomLCA object.

Value

A vector of class probabilities for each class.

Author(s)

Ken Beath

dentistry

Dental X-ray data

Description

Six dentists evaluated dental x-rays for incipient caries in Handelman et al (1986), data consistss of 5 of the dentists analysed by Espeland and Handelman (1989) using a latent class model. Further analysis incorporating a random effects latent class model by Qu et al (1996), and by Albert and Dodd (2004)

Usage

dentistry
dentistry

Format

A data frame with 32 observations on the following 6 variables.

V1 Dentist 1
V2 Dentist 2
V3 Dentist 3
V4 Dentist 4
V5 Dentist 5
freq Number of subjects

Source

Espeland and Handelman (1989)

References


Examples

```r
## Not run:
# fit LCR model from Qu et al (1996)
dentistry.lca <- randomLCA(dentistry[,1:5],freq=dentistry$freq)
# start with constant loading
dentistry.lcarandom <- randomLCA(dentistry[,1:5],freq=dentistry$freq,
random=TRUE,probit=TRUE)
# allow loading to vary by dentist
dentistry.lcarandomunequal <- randomLCA(dentistry[,1:5],freq=dentistry$freq,
random=TRUE,constload=FALSE,probit=TRUE)

## End(Not run)
```
fitted \hspace{1cm} fitted values

Description

Extract fitted values for randomLCA object.

Usage

```r
## S3 method for class 'randomLCA'
fitted(object, ...)
```

Arguments

- `object`: randomLCA object
- `...`: additional argument; currently none is used.

Value

A data frame. The first columns of the data frame correspond to the patterns, followed by the frequency of each pattern, and then the fitted number for each pattern.

Author(s)

Ken Beath <ken.beath@mq.edu.au>

genderrole \hspace{1cm} Gender Role Opinion Items

Description

Opinions collected on gender roles in a study by Felling et al (1987). This was originally published in Heinen (1996) and subsequently in Galindo Garre and Vermunt (2006).

Usage

`genderrole`
**genderrole**

**Format**
A data frame with 16 observations on the following 5 variables.

Q1 Women’s liberation sets women against men.
Q2 It’s better for a wife not to have a job because that always poses problems in the household, especially if there are children.
Q3 The most natural situation occurs when the man is the breadwinner and the woman runs the household and takes care of the children.
Q4 It isn’t really as important for a girl to get a good education as it is for a boy.
Q5 A woman is better suited to raise small children than a man.

Freq Number of subjects

**Source**
Galindo Garre and Vermunt (2006)

**References**


**Examples**
```r
# Not run:
# standard latent class
genderrole.lca1 <- randomLCA(genderrole[,1:5],freq=genderrole$Freq,nclass=1)
genderrole.lca2 <- randomLCA(genderrole[,1:5],freq=genderrole$Freq)
genderrole.lca3 <- randomLCA(genderrole[,1:5],freq=genderrole$Freq,nclass=3)
# repeat with random effect with constant loading
# increase quadrature points and/or use higher penalty to obtain
# convergence
genderrole.lca1random <- randomLCA(genderrole[,1:5],freq=genderrole$Freq, nclass=1,random=TRUE)
genderrole.lca2random <- randomLCA(genderrole[,1:5],freq=genderrole$Freq, random=TRUE,penalty=0.1,quadpoints=61)
genderrole.lca3random <- randomLCA(genderrole[,1:5],freq=genderrole$Freq, nclass=3,random=TRUE,penalty=0.1,quadpoints=61)
# improved BIC for 1 class random
print(c(BIC(genderrole.lca1),BIC(genderrole.lca2),BIC(genderrole.lca3)))
print(c(BIC(genderrole.lca1random),BIC(genderrole.lca2random),
BIC(genderrole.lca3random)))
# can also repeat fits without constant loading to give mixture of IRT models
genderrole.lca1random2 <- randomLCA(genderrole[,1:5],freq=genderrole$Freq, nclass=1,random=TRUE,constload=FALSE)
genderrole.lca2random2 <- randomLCA(genderrole[,1:5],freq=genderrole$Freq, random=TRUE,constload=FALSE,quadpoints=61,penalty=0.1)
```
hivtests <- randomLCA(genderrole[,1:5], freq=genderrole$Freq, nclass=3, random=TRUE, constload=FALSE, quadpoints=61, penalty=0.1)
# no improvement in fit
print(c(BIC(genderrole.lca3random2), BIC(genderrole.lca2random2), BIC(genderrole.lca3random2)))

## End(Not run)

---

**hivtests**  
*HIV testing data*

**Description**

Serum samples are tested for HIV by 4 different biossays in Alvord et al (1988) and sensitivity and specificity determined using latent class analysis. Qu et al (1996) repeat the analysis using a model incorporating a random effect.

**Usage**

hivtests

**Format**

A data frame with 16 observations on the following 5 variables.

- **v1** Test 1
- **v2** Test 2
- **v3** Test 3
- **v4** Test 4
- **freq** Number of subjects

**Source**

Qu, Tan and Kutner (1989)

**References**


Examples

```r
## Not run:
# fit standard latent class
hivtests.lca2 <- randomLCA(hivtests[,1:4],freq=hivtests$freq)
# with random effect and constant loading
hivtests.lca2random <- randomLCA(hivtests[,1:4],freq=hivtests$freq,random=TRUE,penalty=1.0)
# with random effect and variable loading
# for this model there are 13 parameters fitted to 16 observations, so model is fairly unstable
hivtests.lca2random2 <- randomLCA(hivtests[,1:4],freq=hivtests$freq,random=TRUE,
   constload=FALSE,penalty=1.0)
# BIC shows best model is random effects with constant loading
print(c(BIC(hivtests.lca2),BIC(hivtests.lca2random),BIC(hivtests.lca2random2)))
## End(Not run)
```

logLik

### log Likelihood for randomLCA object

Description

Returns log Likelihood for a randomLCA object.

Usage

```r
## S3 method for class 'randomLCA'
logLik(object, ...)
```

Arguments

- `object`: randomLCA object
- `...`: additional argument; currently none is used.

Value

The loglikelihood.

Author(s)

Ken Beath
Description

Four tests were performed on hospital patients to determine if a myocardial infarction had occurred.

Usage

myocardial

Format

A data frame with 32 observations on the following 6 variables.

qwave result from ECG test
history clinical history
LDH flipped, enzyme related to tissue breakdown
CPK high, creatine kinase or creatine phosphokinase, related to muscle damage
freq Number of subjects

Source

Rindskopf and Rindskopf (1986)

References


Examples

# fit 2 class model from Rindskopf and Rindskopf (1986)
myocardial.lca2 <- randomLCA(myocardial[,1:4], freq=myocardial$freq)
outcomeProbs

Extract outcome probabilities for randomLCA object

Description

Extract outcome probabilities and confidence intervals for a randomLCA object.

Usage

```r
## S3 method for class 'randomLCA'
outcomeProbs(object, level = 0.95, boot = FALSE, type = "norm",
             R = ifelse(type == "norm", 199, 999), scale = c("prob", "raw"), ...)
```

Arguments

- `object`: randomLCA object
- `level`: confidence interval
- `boot`: use parametric bootstrap to obtain confidence interval
- `type`: type of bootstrap confidence intervals to use, with "perc" or "norm" valid, see `boot.ci` for description. It seems reasonable to use the normal approximation.
- `R`: replications for parametric bootstrap
- `scale`: either "prob" where probabilities are returned, the default, or "raw" where the probabilities are returned on the logit or probit scale, depending on which scale was selected in the `texttrandomLCA` function
- `...`: additional argument; currently none is used.

Details

Confidence intervals are calculated based on asymptotic normality of the estimates transformed by either the inverse of the probit or logistic, or using parametric bootstrap. The asymptotic confidence intervals are currently only available for models without random effects. For the confidence intervals obtained from the parametric bootstrap, the bootstrap is performed on the data that has been transformed to the logit or probit scale, as appropriate. The samples are close to normal allowing for the use of confidence intervals based on the normal approximation. About 199 replications gives similar accuracy to percentile with 999.

Value

Data frame consisting of outcome probabilities and confidence intervals. One for each class.

Author(s)

Ken Beath
Examples

```r
# standard latent class with 2 classes
dentistry.lca2 <- randomLCA(dentistry[,1:5], freq=dentistry$freq, nclass=2)
print(outcomeProbs(dentistry.lca2))
# print on the default logit scale
print(outcomeProbs(dentistry.lca2, scale="raw"))
# convert back to probabilities
print(1.0/(1.0+exp(-outcomeProbs(dentistry.lca2, scale="raw")[[1]])))
print(1.0/(1.0+exp(-outcomeProbs(dentistry.lca2, scale="raw")[[2]])))
```

### Description

The Positive Action program is a series of interventions designed to reduce negative behaviours in elementary-school students. In a study in Hawaii (Beets et al, 2006) information was recorded from students in the treatment group about whether the various parts of the program were implemented. While it is useful to describe the proportion of students experiencing implementation of each part of the program, latent class analysis will reveal if there are specific patterns to the implementation of the program (Alcock, 2008).

### Usage

pap

### Format

A data frame with 606 observations summarising the answers for 1566 students on the following 11 variables. For each variable the data has been dichotomized so that a 0 represents no implementation and a 1 represents some implementation.

- Q1 you receive stickers from your teacher for doing positive actions?
- Q2 you receive a word of the week card from your teacher?
- Q3 you put notes in an icu box?
- Q4 your teacher read notes about you from the icu box?
- Q5 your teacher read your notes from the icu box?
- Q6 your class receive a token for meeting your classroom goals?
- Q7 you participate in a positive action assembly?
- Q8 your class receive a balloon in an assembly for achieving their classroom goals?
- Q9 your class participate in whole school positive action celebrations?
- Q10 most weeks were you taught a positive action lesson?

Freq Number of subjects
Source

Alcock (2008)

References


Examples

```r
## Not run:
# standard latent class
pap.lca1 <- randomLCA(pap[,1:10],freq=pap$Freq,nclass=1)
pap.lca2 <- randomLCA(pap[,1:10],freq=pap$Freq,nclass=2)
pap.lca3 <- randomLCA(pap[,1:10],freq=pap$Freq,nclass=3)
pap.lca4 <- randomLCA(pap[,1:10],freq=pap$Freq,nclass=4)
pap.lca5 <- randomLCA(pap[,1:10],freq=pap$Freq,nclass=5)

# repeat with random effect with constant loading
# once BIC increases fitting further models is unnecessary
pap.lca1random <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=1,random=TRUE)
pap.lca2random <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=2,random=TRUE)
pap.lca3random <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=3,random=TRUE)

# can also repeat fits without constant loading to give mixture of IRT models
pap.lca1random2 <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=1,random=TRUE,constload=FALSE)
pap.lca2random2 <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=2,random=TRUE,constload=FALSE)
pap.lca3random2 <- randomLCA(pap[,1:10],
  freq=pap$Freq,nclass=3,random=TRUE,constload=FALSE)

# produce table of BIC values
# shows 4 class best of standard latent class
# but 2 class latent class with constant loading has better BIC
pap.bic <- data.frame(bic=c(BIC(pap.lca1),BIC(pap.lca2),BIC(pap.lca3),BIC(pap.lca4),BIC(pap.lca5)), bic2=c(BIC(pap.lca1random),BIC(pap.lca2random),BIC(pap.lca3random),NA,NA), bic3=c(BIC(pap.lca1random2),BIC(pap.lca2random2),BIC(pap.lca3random2),NA,NA))
print(pap.bic)
# plot 4 class standard
plot(pap.lca4,type="b")
# plot 2 class standard
plot(pap.lca2random,type="b")
```
Description

Plots the outcome probabilities for a randomLCA object, for random effects objects this can be either marginal or conditional or both. For a 2 level random effects model conditional2 will condition on the subject random effect and integrate over the period random effects. Note that plot is based on the xyplot function.

Usage

```r
## S3 method for class 'randomLCA'
plot(x, ..., graphtype=ifelse(x$random, "marginal","conditional"),
     conditionalP=0.5, classhorizontal=TRUE)
```

Arguments

- `x`: randomLCA object
- `graphtype`: Type of graph
- `conditionalP`: For a conditional graph the percentile corresponding to the random effect at which the outcome probability is to be calculated
- `classhorizontal`: classes to be plotted across the page
- `...`: additional parameters to xyplot

Author(s)

Ken Beath <ken.beath@mq.edu.au>

See Also

calcCondProb, calcMargProb

Examples

```r
## Not run:
# standard latent class with 2 classes
uterinecarcinoma.lca2 <- randomLCA(uterinecarcinoma[,1:7], freq=uterinecarcinoma$freq)
plot(uterinecarcinoma.lca2)
uterinecarcinoma.lcarandom2 <- randomLCA(uterinecarcinoma[,1:7],
    freq=uterinecarcinoma$freq, random=TRUE, probit=TRUE, quadpoints=61)
# default for random effects models is marginal
plot(uterinecarcinoma.lcarandom2)
```
postClassProbs

# default for random effects models conditional is p=0.5 i.e. median
plot(uterinecarcinoma.1carandom2,graphtype="conditional")
# look at variability by plotting conditional probabilities at 0.05,0.5 and 0.95
plot(uterinecarcinoma.1carandom2,graphtype="conditional",conditionalp=c(0.05,0.5,0.95))

## End(Not run)

---

postClassProbs

Determines posterior class probabilities for fitted model

Description

The posterior class probabilities for each observed pattern and class is determined. These are
returned as a data frame together with the patterns for each observation. If class=0 is requested then
all classes are returned, otherwise only the selected class.

Usage

postClassProbs(object, class=0)

Arguments

- object: randomLCA object
- class: class to be returned. Zero returns all classes.

Details

Extracts the corresponding data from the randomLCA object.

Value

A data frame. The first columns of the data frame correspond to the patterns, followed by the
frequency of each pattern, and then the posterior class probabilities for either the selected class
or for all classes. The returned result is for the summarised data. If raw data is used, that is no
frequencies, and it is required to calculated the posterior class probability for each observation then
it is simply required to merge the class probabilities with the raw data, possibly removing any
variable "freq" in the raw data.

Author(s)

Ken Beath
print.randomLCA  

print for randomLCA object

Description

Prints a randomLCA object. Prints summary.

Usage

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

Arguments

- `x`  randomLCA object
- `...` additional argument; currently none is used.

Author(s)

Ken Beath

randomLCA  

Fits a Latent Class Model including a Random Effect

Description

Fit latent class models, which may include a random effect.

Usage

```r
randomLCA(patterns, freq=NULL, nclass=2, calcSE=TRUE, notrials=20,
random=FALSE, byclass=FALSE, quadpoints=21, constload=TRUE, blocksize=dim(patterns)[2],
level2=FALSE, probit=FALSE, level2size=blocksize,
qnterations=5, penalty=0.01, EMtol=1.0e-9, verbose=FALSE,
seed = as.integer(runif(1, 0, .Machine$integer.max)))
```

Arguments

- `patterns`  Data frame or matrix of 0 and 1 defining the outcome patterns. May also include missing values, with randomLCA using maximum likelihood to fit the models using all available data.
- `freq`  Frequency for each outcome pattern, if missing this is calculated from the patterns, and the patterns are summarised to remove duplicate values.
- `nclass`  Number of classes to be fitted
- `calcSE`  Calculate standard errors for parameters. Useful for bootstrapping.
notrials For a standard latent class model, the number of random starting values used
random Include random effect?
byclass Vary random effect loading(s) by class?
quadpoints Number of quadrature points for adaptive quadrature
constload Outcome loadings are constant for random effects model?
blocksize Where a random effects (single level) model is broken into blocks, that is the
loadings are repeated, this defines the size of the blocks
probit Probit model for random effect?
level2 Fit 2 level random effects model (further details to follow)?
level2size Size of level 2 blocks if fitting 2 level models
qniterations Number of Quasi-Newton iterations within each EM/adaptive cycle. Decrease
if there is a failure to converge
penalty penalty applied to likelihood for outcome probabilities. Shrinks outcome prob-
abilities in slightly and can prevent extreme values. Setting penalty to 0 will
produce an unpenalized fit.
EMtol convergence tolerance for EM algorithm for fixed effect latent class
verbose Prints fit progress if true
seed Initial random seed for generating starting values. This can be set to guarantee
that the fit is the same each time, including the order of the classes.

Details
The structure of the patterns is assumed to be a number of blocks of different outcomes each of
level2size, allowing outcomes to be repeated. Each outcome is assumed to have it’s own loading.
An example is the width of the patterns is n and the level2size is n, resulting in n outcomes and
therefore n loadings. Alternatively if the level2size is 1, then there are n repeats of the same outcome
(but with different probabilities) with the same loading. In practice they may not be the same type
of outcome, but usually will be.

The algorithm used is EM for the standard latent class and adaptive (in the sense of moving the
location of the quadrature points) Gauss-Hermite quadrature for the random effects models. The
number of quadrature points defaults to 21.

Value
randomLCA object This contains
fit Fit object from optim
nclass Number of classes
classp Class probabilities
outcomep Outcome probability
lambdacoef Loadings
se Standard errors corresponding to results returned by optim
np Number of parameters
nobs    Number of observations in total
logLik   log likelihood for fitted model
penlogLik Penalised log likelihood for fitted model
observed Observed numbers corresponding to each pattern
fitted   Fitted number corresponding to each pattern
deviance Deviance
classprob Posterior class probability for each pattern
bics     BIC obtained for each trial when fitting initial latent class models
call     call to randomLCA
random   random parameter to randomLCA
constload constload parameter to randomLCA
level2   level2 parameter to randomLCA
level2size level2size parameter to randomLCA
byclass  byclass parameter to randomLCA
probit   probit parameter to randomLCA
quadpoints quadpoints parameter to randomLCA
blocksize blocksize parameter to randomLCA
freq     frequency of each pattern
qniterations qniterations parameter to randomLCA
penalty  penalty parameter to randomLCA

Note
In the returned object there a fields for patterns and frequencies. If frequencies are not supplied then
the patterns and frequencies are constructed. If frequencies are supplied then zero rows are removed.
When frequencies are supplied it is assumed that the data has been simplified. The returned fitted,
posterior class probabilities etc, all correspond to the simplified patterns, not to the original data.

Author(s)
Ken Beath

Examples

## Not run:
# standard latent class with 2 classes
dentistry.lca2 <- randomLCA(dentistry[,1:5],freq=dentistry$freq,nclass=2)
# random effects model with constant random effect loading
dentistry.lca2random <- randomLCA(dentistry[,1:5],freq=dentistry$freq,
  nclass=2,random=TRUE,constload=TRUE,probit=TRUE)
# allow loading to vary by dentist
# this is the 2LCR model from Qu et al (1996)
dentistry.lca2random1 <- randomLCA(dentistry[,1:5],freq=dentistry$freq,
  nclass=2,random=TRUE,constload=FALSE,probit=TRUE)

## End(Not run)
**ranef**

*Extract random effects from a randomLCA object*

**Description**

Extracts the Empirical Bayes estimates of the random effects.

**Usage**

```r
## S3 method for class 'randomLCA'
ranef(object, ...)
```

**Arguments**

- `object`: randomLCA object with a random effect
- `...`: additional argument; currently none is used.

**Value**

A matrix with the first column containing the random effects and the second column the standard error of the random effects.

**Author(s)**

Ken Beath

---

**refit**

*Refit an randomLCA object*

**Description**

Refits an randomLCA object using new data. For an example, see the simulate method.

**Usage**

```r
## S3 method for class 'randomLCA'
refit(object, newpatterns, newfreq, useinit=FALSE, ...)
```

**Arguments**

- `object`: randomLCA object
- `newpatterns`: the new patterns that are to be fitted using the existing model
- `newfreq`: the frequencies corresponding to the patterns if required
- `useinit`: use initial values from randomLCA object
- `...`: additional argument; currently none is used.
**Details**

The useunit parameter determines whether the parameter estimates from the supplied model are used as initial values or whether the complete model fitting process is repeated. If the initial values are used then fitting will be faster, and the fitted classes will be similar to those in the original model. If the data was not generated from the original model there is an increased risk that the fit will not find the global maxima. For this reason when performing a bootstrap Likelihood ratio test it is better to use useinit=FALSE. However when using useinit=FALSE there may be label switching, where the estimated classes are similar, but in a different order. Unless the estimated parameters are assigned to the correct classes this will invalidate the results of a parametric bootstrap for parameter confidence intervals.

**Value**

The fitted model to the new data.

**Author(s)**

Ken Beath

---

**simulate**

**Simulate**

**Description**

Simulate data from a fitted randomLCA model

**Usage**

```r
## S3 method for class 'randomLCA'
simulate(object, nsim, seed, ...)  
```

**Arguments**

- `object`: randomLCA object
- `nsim`: number of data sets to be simulated
- `seed`: random seed
- `...`: additional optional arguments.

**Details**

Generates random data from the supplied object.

**Value**

A simulated data frame or a list of simulated data frames.
Author(s)

Ken Beath

Examples

```r
## Not run:
myocardial.lca1 <- randomLCA(myocardial[,1:4],freq=myocardial$freq,nclass=1)
myocardial.lca2 <- randomLCA(myocardial[,1:4],freq=myocardial$freq)
# calculate observed lrt
obslrt <- 2*(logLik(myocardial.lca1)-logLik(myocardial.lca2))

print(obslrt)

nsims <- 999
# generate the simulations
thesims <- simulate(myocardial.lca1, nsims)
# for each simulation determine lrt
simlrt <- rep(NA,nsims)
for (isim in 1:nsims) {
  submodel <- refit(myocardial.lca1,newpatterns=thesims[[isim]])
  fullmodel <- refit(myocardial.lca2,newpatterns=thesims[[isim]])
  simlrt[isim] <- 2*(logLik(fullmodel)-logLik(submodel))
  print(c(isim,simlrt[isim]))
}
# calculate p value as proportion of simulated lrt greater than observed,
# corrected by adding one to numerator and denominator
print((sum(simlrt>obslrt)+1)/(nsims+1))
## End(Not run)
```

### summary.randomLCA

Summary for randomLCA object

**Description**

Summarises the fit of a randomLCA object.

**Usage**

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

**Arguments**

- `object` randomLCA object
- `...` additional argument; currently none is used.
Value

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loglik</td>
<td>Log Likelihood</td>
</tr>
<tr>
<td>AIC</td>
<td>AIC</td>
</tr>
<tr>
<td>BIC</td>
<td>BIC</td>
</tr>
<tr>
<td>AIC3</td>
<td>AIC with penalty of 3</td>
</tr>
<tr>
<td>nclass</td>
<td>no of classes</td>
</tr>
<tr>
<td>probit</td>
<td>link is probit</td>
</tr>
<tr>
<td>classp</td>
<td>class probabilities</td>
</tr>
<tr>
<td>outcomep</td>
<td>outcome probabilities (conditional)</td>
</tr>
<tr>
<td>margoutcomep</td>
<td>outcome probabilities (marginal), if model contains random effects</td>
</tr>
<tr>
<td>random</td>
<td>model includes random effects</td>
</tr>
<tr>
<td>level2</td>
<td>model has 2 level hierarchy</td>
</tr>
<tr>
<td>constload</td>
<td>loadings are constant by outcome</td>
</tr>
<tr>
<td>byclass</td>
<td>lambda and tau vary by class</td>
</tr>
<tr>
<td>lambdacoef</td>
<td>lambda coefficients</td>
</tr>
<tr>
<td>taucoef</td>
<td>tau coefficients</td>
</tr>
</tbody>
</table>

Author(s)

Ken Beath

Description

This is the data for Beath and Heller (2009).
Allergy and respiratory symptoms for infants 0 to 2 years in six month periods. Outcome is presence or absence of symptom in the six months. Original data was collected at Visits 1-7 over the 2 year period which were summarised to six month periods.

Note that these models can be slow to fit, with the "symptoms.lca2random2" model taking about 1-2 hours.

Thanks to the investigators of the CAPS study for making the data available.

Usage

symptoms
Format

A data frame with 444 observations on the following 17 variables.

- NightcoughNQS: Night cough in visits 1-3
- WheezeNQS: Wheeze in visits 1-3
- ItchyrashNQS: Itchy rash in visits 1-3
- FlexDermaNQS: Flexural Dermatitis in visits 1-3
- NightcoughNTU: Night cough in visits 4-5
- WheezeNTU: Wheeze in visits 4-5
- ItchyrashNTU: Itchy rash in visits 4-5
- FlexDermaNTU: Flexural Dermatitis in visits 4-5
- NightcoughNV: Night cough in visit 6
- WheezeNV: Wheeze in visit 6
- ItchyrashNV: Itchy rash in visit 6
- FlexDermaNV: Flexural Dermatitis in visit 6
- NightcoughNW: Night cough in visit 7
- WheezeNW: Wheeze in visit 7
- ItchyrashNW: Itchy rash in visit 7
- FlexDermaNW: Flexural Dermatitis in visit 7
- Freq: Number of subjects

Source

Mihrshai et al (2001)

References


Examples

```r
## Not run:
symptoms.lca2 <- randomLCA(symptoms[,1:16],freq=symptoms$Freq,nclass=2)
symptoms.lcaparam <- randomLCA(symptoms[,1:16],freq=symptoms$Freq,
random=TRUE,nclass=2,blocksize=4,constload=FALSE)
symptoms.lcaparam2 <- randomLCA(symptoms[,1:16],freq=symptoms$Freq,
random=TRUE,level2=TRUE,nclass=2,level2size=4,constload=FALSE)

## End(Not run)
```
uterinecarcinoma  Uterine Carcinoma Data

Description

Classification of 118 histology samples by 118 pathologists. Original classification in Holmquist et al (1967) was to one of five categories, this has been reduced to two. Analysed by a number of authors, with a random effects model in Qu et al (1996).

Usage

uterinecarcinoma

Format

A data frame with 20 observations on the following 8 variables.

v1 Pathologist 1
v2 Pathologist 2
v3 Pathologist 3
v4 Pathologist 4
v5 Pathologist 5
v6 Pathologist 6
v7 Pathologist 7
freq Number of observed pattern

Source

Qu et al (1996)

References


Examples

---

## Not run:

```r
uterinecarcinoma.lcarandom2 <- randomLCA(uterinecarcinoma[,1:7],
  freq=uterinecarcinoma$freq,random=TRUE,probit=TRUE,quadpoints=61)
# LCR! model of Que et al. This is fairly unstable and
# is also slow and doesn't improve the model fit
uterinecarcinoma.lcarandom2by <- randomLCA(uterinecarcinoma[,1:7],freq=uterinecarcinoma$freq,
  byclass=TRUE,random=TRUE,probit=TRUE,quadpoints=71)
## End(Not run)
```
# Index

*Topic **datasets**
- dentistry, 6
- genderrole, 8
- hivtests, 10
- myocardial, 12
- pap, 14
- symptoms, 24
- uterinecarcinoma, 26

*Topic **methods**
- AIC, 2
- AIC3, 3
- BIC, 3
- calcCond2Prob, 4
- calcCondProb, 4
- calcMargProb, 5
- classProbs, 6
- fitted, 8
- logLik, 11
- outcomeProbs, 13
- plot, 16
- postClassProbs, 17
- print.randomLCA, 18
- ranef, 21
- refit, 21
- simulate, 22
- summary.randomLCA, 23

*Topic **multivariate**
- randomLCA, 18

AIC, 2
AIC3, 3
BIC, 3
calcCond2Prob, 4
calcCondProb, 4, 16
calcMargProb, 5, 16
classProbs, 6
dentistry, 6
fitted, 8
genderrole, 8
hivtests, 10
logLik, 11
myocardial, 12
outcomeProbs, 13
pap, 14
plot, 16
postClassProbs, 17
print.randomLCA, 18
ranef, 21
refit, 21
simulate, 22
summary.randomLCA, 23
symptoms, 24
uterinecarcinoma, 26