Package ‘lcda’

February 20, 2015

Type Package

Title Latent Class Discriminant Analysis

Version 0.3

Date 2011-04-06

Depends R (>= 2.6.0), poLCA

Author Michael Buecker

Maintainer Michael Buecker <buecker@statistik.uni-dortmund.de>

Description Local Discrimination via Latent Class Models

License GPL

Repository CRAN

Date/Publication 2011-04-06 10:12:13

NeedsCompilation no

R topics documented:

cclcda *Common Components Latent Class Discriminant Analysis (CCLCDA)*

Description

Local Discrimination via Latent Class Models with common components.

Usage

```
cclcda(x, ...)


## Default S3 method:
cclcda(x, grouping=NULL, prior=NULL,
                         probs.start=NULL, nrep=1, m=3,
                         maxiter = 1000, tol = 1e-10,
                         subset, na.rm = FALSE, ...)

## S3 method for class 'formula'
cclcda(formula, data, ...)
```

Arguments

`x`	Matrix or data frame containing the explanatory variables. Manifest variables must contain only integer values, and must be coded with consecutive values from 1 to the maximum number of outcomes for each variable. All missing values should be entered as NA.
`grouping`	A factor specifying the class for each observation; if not specified, the first column of 'data' is taken. The class must be coded by integer values with consecutive values from 1 to the maximum number of classes.
`formula`	Formula of the form `'groups ~ x1 + x2 + ...'`.
`data`	Data frame from which variables specified in formula are to be taken.
`prior`	The prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.
`probs.start`	A list of matrices (per variable) of response probabilities θ_{mkdr} to be used as the starting values for the estimation algorithm. Each matrix in the list corresponds to one manifest variable, with one row for each latent class, and one column for each outcome. The default is `NULL`, producing random starting values. Note that if `nrep>1`, then any user-specified `probs.start` values are only used in the first of the nrep attempts.
`nrep`	Number of times to estimate the model, using different random values of `probs.start`. The default is one. Setting `nrep>1` automates the search for the global – rather than just a local – maximum of the log-likelihood function. `cclcda` uses the parameter estimates corresponding to the model with the greatest log-likelihood.
`m`	The number of subclasses. Can be either a vector containing the number of subclasses per class or a number of subclasses for all classes. Default is `m=3`.
`maxiter`	The maximum number of iterations through which the estimation algorithm will cycle.
`tol`	A tolerance value for judging when convergence has been reached. When the one-iteration change in the estimated log-likelihood is less than `tol`, the estimation algorithm stops updating and considers the maximum log-likelihood to have been found.

`subset`	An index vector specifying the cases to be used in the training sample.
`na.rm`	Logical, for how `cclcda` handles cases with missing values on the manifest variables. If `TRUE`, those cases are removed (listwise deleted) before estimating the model. If `FALSE`, cases with missing values are retained. Cases with missing covariates are always removed. The default is `TRUE`.
`...`	Further arguments to be passed to `cclcda.default`.

Details

The `cclcda`-function performs a Common Components Latent Class Discriminant Analysis (CCLCDA). The model to estimate is

$$f(x) = \sum_{m=1}^{M} w_m \prod_{d=1}^{D} \prod_{r=1}^{R_d} \theta_{mdr}^{x_{dr}},$$

where m is the latent subclass index, d is the variable index and r is the observation index. The variable x_{dr} is 1 if the variable d of this observation is r. This common Latent Class Modell will be estimated for all classes by the `poLCA`-function (see `poLCA`) and class conditional mixing proportions w_{mk} are computed afterwards. These weights are computed by

$$\frac{1}{N_k} \sum_{n=1}^{N_k} \hat{P}(m, k | X = x_n),$$

where k is the class index and N_k the number of observations in class k.

The LCA uses the assumption of local independence to estimate a mixture model of latent multi-way tables, the number of which (`m`) is specified by the user. Estimated parameters include the latent-class-conditional response probabilities for each manifest variable θ_{mdr} and the class conditional mixing proportions w_{mk} denoting population share of observations corresponding to each latent multi-way table per class.

Posterior class probabilities can be estimated with the `predict` method.

Value

A list of class `cclcda` containing the following components:

`call`	The (matched) function call.
`lca.theta`	The estimated class conditional response probabilities of the LCA given as a list of matrices like `probs.start`.
`lca.w`	The estimated mixing proportions of the LCA.
`lca.wmk`	The estimated class conditional mixing proportions of the LCA.
`prior`	Prior probabilites.
`m`	Number of latent subclasses.
`r`	Number of different responses per variable.
`k`	Number of classes.
`d`	Number of variables.
`aic`	Value of the AIC for each class conditional Latent Class Model.

`bic`	Value of the BIC for each class conditional Latent Class Model.
`Gsq`	The likelihood ratio/deviance statistic for each class conditional model.
`Chisq`	The Pearson Chi-square goodness of fit statistic for fitted vs. observed multiway tables for each class conditional model.
`entropy`	Value of the weighted entropy as described below.
`gini`	Value of the weighted Gini coefficient as described below.
`chi.stat`	Value of the Chi-square test statistik of the test of latent class membership and class membership as described below.
`chi.p`	P Value of the Chi-square of the test of latent class membership and class membership as described below.

Note

If the number of latent classes per class is unknown a model selection must be accomplished to determine the value of `m`. For this goal there are some model selection criteria implemented. The AIC, BIC, likelihood ratio statistic and the Chi-square goodness of fit statistic are taken from the poLCA-function (see `poLCA`).

Additionally `cclcda` provides quality criteria which should give insight into the model's classification potential. These criteria are similar to the splitting criteria of classification trees. The impurity measures are

– Weighted entropy: The weighted entropy is given by

$$H := -\sum_{m=1}^{M} P(m) \sum_{k=1}^{K} \left(P(k|m) \cdot \log_K P(k|m)\right).$$

– Weighted Gini coefficient: The weighted Gini coefficient is given by

$$G := \sum_{m=1}^{M} P(m) \left[1 - \sum_{k=1}^{K} \left(P(k|m)\right)^2\right].$$

– Pearson's Chi-square test: A Pearson's Chi-square test is performed to test the independence of latent class membership and class membership.

Author(s)

Michael B\"ucker

See Also

`predict.cclcda`, `lcda`, `predict.lcda`, `cclcda2`, `predict.cclcda2`, `poLCA`

Examples

```
# response probabilites
probs1 <- list()

probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1,
                        0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=4, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1,
                        0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7,
                        0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1,
                        0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=4, byrow=TRUE)

prior <- c(0.5,0.5)
wmk <- matrix(c(0.45,0.45,0.05,0.05,0.05,0.05,0.45,0.45),
              ncol=4, nrow=2, byrow=TRUE)
wkm <- apply(wmk*prior, 2, function(x) x/sum(x))

# generation of training data
data_temp <- poLCA.simdata(N = 1000, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data <- data_temp$dat
lclass <- data_temp$trueclass
grouping <- numeric()
for (i in 1:length(lclass))
{
grouping[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# generation of test data
data_temp <- poLCA.simdata(N = 500, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data.test <- data_temp$dat
lclass <- data_temp$trueclass
grouping.test <- numeric()
for (i in 1:length(lclass))
{
grouping.test[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# cclcda-procedure
object <- cclcda(data, grouping, m=4)
object
```

cclcda2 — *Common Components Latent Class Discriminant Analysis 2 (CCLCDA2)*

Description

Local Discrimination via Latent Class Models with common components

Usage

```
cclcda2(x, ...)

## Default S3 method:
cclcda2(x, grouping=NULL, prior=NULL,
                         probs.start=NULL, wmk.start=NULL,
                         nrep=1, m=3, maxiter = 1000,
                         tol = 1e-10, subset, na.rm = FALSE, ...)

## S3 method for class 'formula'
cclcda2(formula, data, ...)
```

Arguments

`x`	Matrix or data frame containing the explanatory variables. Manifest variables must contain only integer values, and must be coded with consecutive values from 1 to the maximum number of outcomes for each variable. All missing values should be entered as NA.
`grouping`	A factor specifying the class for each observation; if not specified, the first column of 'data' is taken. The class must be coded by integer values with consecutive values from 1 to the maximum number of classes.
`formula`	Formula of the form `'groups ~ x1 + x2 + ...'`.
`data`	Data frame from which variables specified in formula are to be taken.
`prior`	The prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.
`probs.start`	A list of matrices (per variable) of response probabilities θ_{mkdr} to be used as the starting values for the estimation algorithm. Each matrix in the list corresponds to one manifest variable, with one row for each latent class, and one column for each outcome. The default is `NULL`, producing random starting values. Note that if `nrep>1`, then any user-specified `probs.start` values are only used in the first of the nrep attempts.
`wmk.start`	A matrix of starting values for the parameter w_{mk} (see details). The default is `NULL`, producing random starting values. Note that if `nrep>1`, then any user-specified `wmk.start` values are only used in the first of the nrep attempts.

`nrep`	Number of times to estimate the model, using different random values of `probs.start`. The default is one. Setting `nrep>1` automates the search for the global – rather than just a local – maximum of the log-likelihood function. `cclcda2` uses the parameter estimates corresponding to the model with the greatest log-likelihood.
`m`	The number of subclasses. Can be either a vector containing the number of subclasses per class or a number of subclasses for all classes. Default is `m=3`.
`maxiter`	The maximum number of iterations through which the estimation algorithm will cycle.
`tol`	A tolerance value for judging when convergence has been reached. When the one-iteration change in the estimated log-likelihood is less than `tol`, the estimation algorithm stops updating and considers the maximum log-likelihood to have been found.
`subset`	An index vector specifying the cases to be used in the training sample.
`na.rm`	Logical, for how `cclcda2` handles cases with missing values on the manifest variables. If `TRUE`, those cases are removed (listwise deleted) before estimating the model. If `FALSE`, cases with missing values are retained. Cases with missing covariates are always removed. The default is `TRUE`.
`...`	Further arguments to be passed to `cclcda2.default`.

Details

The `cclcda2`-function performs a Common Components Latent Class Discriminant Analysis 2 (CCLCDA2). The class conditional model to estimate is

$$f_k(x) = \sum_{m=1}^{M_k} w_{mk} \prod_{d=1}^{D} \prod_{r=1}^{R_d} \theta_{mdr}^{x_{dr}},$$

where m is the latent subclass index, d is the variable index and r is the observation index. The variable x_{dr} is 1 if the variable d of this observation is r. This Latent Class Modell will be estimated. The class conditional mixing proportions w_{mk} and the parameters θ_{mdr} are computed in every step of the EM-Algorithm.

The LCA uses the assumption of local independence to estimate a mixture model of latent multi-way tables, the number of which (`m`) is specified by the user. Estimated parameters include the latent-class-conditional response probabilities for each manifest variable θ_{mdr} and the class conditional mixing proportions w_{mk} denoting population share of observations corresponding to each latent multi-way table per class.

Posterior class probabilities can be estimated with the `predict` method.

Value

A list of class `cclcda2` containing the following components:

`call`	The (matched) function call.
`lca.theta`	The estimated class conditional response probabilities of the LCA given as a list of matrices like `probs.start`.
`lca.w`	The estimated mixing proportions of the LCA.

`lca.wmk`	The estimated class conditional mixing proportions of the LCA.
`prior`	Prior probabilites.
`m`	Number of latent subclasses.
`r`	Number of different responses per variable.
`k`	Number of classes.
`d`	Number of variables.
`aic`	Value of the AIC for each class conditional Latent Class Model.
`bic`	Value of the BIC for each class conditional Latent Class Model.
`Gsq`	The likelihood ratio/deviance statistic for each class conditional model.
`Chisq`	The Pearson Chi-square goodness of fit statistic for fitted vs. observed multiway tables for each class conditional model.
`entropy`	Value of the weighted entropy as described below.
`gini`	Value of the weighted Gini coefficient as described below.
`chi.stat`	Value of the Chi-square test statistik of the test of latent class membership and class membership as described below.
`chi.p`	P Value of the Chi-square of the test of latent class membership and class membership as described below.

Note

If the number of latent classes per class is unknown a model selection must be accomplished to determine the value of `m`. For this goal there are some model selection criteria implemented. The AIC, BIC, likelihood ratio statistic and the Chi-square goodness of fit statistic are taken from the poLCA-function (see `poLCA`).

Additionally `cclcda2` provides quality criteria which should give insight into the model's classification potential. These criteria are similar to the splitting criteria of classification trees. The impurity measures are

– Weighted entropy: The weighted entropy is given by

$$H := -\sum_{m=1}^{M} P(m) \sum_{k=1}^{K} \left(P(k|m) \cdot \log_K P(k|m)\right).$$

– Weighted Gini coefficient: The weighted Gini coefficient is given by

$$G := \sum_{m=1}^{M} P(m) \left[1 - \sum_{k=1}^{K} \left(P(k|m)\right)^2\right].$$

– Pearson's Chi-square test: A Pearson's Chi-square test is performed to test the independence of latent class membership and class membership.

Author(s)

Michael B\"ucker

See Also

predict.cclcda2, lcda, predict.lcda, cclcda, predict.cclcda, poLCA

Examples

```
# response probabilites
probs1 <- list()

probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1,
                        0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=4, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1,
                        0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7,
                        0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1,
                        0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=4, byrow=TRUE)

prior <- c(0.5,0.5)
wmk <- matrix(c(0.45,0.45,0.05,0.05,0.05,0.05,0.45,0.45),
              ncol=4, nrow=2, byrow=TRUE)
wkm <- apply(wmk*prior, 2, function(x) x/sum(x))

# generation of training data
data_temp <- poLCA.simdata(N = 1000, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data <- data_temp$dat
lclass <- data_temp$trueclass
grouping <- numeric()
for (i in 1:length(lclass))
{
grouping[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# generation of test data
data_temp <- poLCA.simdata(N = 500, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data.test <- data_temp$dat
lclass <- data_temp$trueclass
grouping.test <- numeric()
for (i in 1:length(lclass))
{
grouping.test[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# cclcda-procedure
object <- cclcda2(data, grouping, m=4)
```

`object`

`lcda` *Latent Class Discriminant Analysis (LCDA)*

Description

Local Discrimination via Latent Class Models

Usage

```
lcda(x, ...)


## Default S3 method:
lcda(x, grouping=NULL, prior=NULL,
                       probs.start=NULL, nrep=1, m=3,
                       maxiter = 1000, tol = 1e-10,
                       subset, na.rm = FALSE, ...)

## S3 method for class 'formula'
lcda(formula, data, ...)
```

Arguments

`x`	Matrix or data frame containing the explanatory variables. Manifest variables must contain only integer values, and must be coded with consecutive values from 1 to the maximum number of outcomes for each variable. All missing values should be entered as NA.
`grouping`	A factor specifying the class for each observation; if not specified, the first column of `data` is taken. The class must be coded by integer values with consecutive values from 1 to the maximum number of classes.
`formula`	Formula of the form `'groups ~ x1 + x2 + ...'`.
`data`	Data frame from which variables specified in formula are to be taken.
`prior`	The prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.
`probs.start`	A list (per class) of lists of matrices (per variable) of response probabilities θ_{mkdr} to be used as the starting values for the estimation algorithm. Each matrix in the list corresponds to one manifest variable, with one row for each latent class, and one column for each outcome. The default is `NULL`, producing random starting values. Note that if `nrep>1`, then any user-specified `probs.start` values are only used in the first of the nrep attempts.

`nrep`	Number of times to estimate the model, using different random values of `probs.start`. The default is one. Setting `nrep>1` automates the search for the global – rather than just a local – maximum of the log-likelihood function. `lcda` uses the parameter estimates corresponding to the model with the greatest log-likelihood.
`m`	The number of subclasses per class. Can be either a vector containing the number of subclasses per class or a number of subclasses for all classes. Default is `m=3`.
`maxiter`	The maximum number of iterations through which the estimation algorithm will cycle.
`tol`	A tolerance value for judging when convergence has been reached. When the one-iteration change in the estimated log-likelihood is less than `tol`, the estimation algorithm stops updating and considers the maximum log-likelihood to have been found.
`subset`	An index vector specifying the cases to be used in the training sample.
`na.rm`	Logical, for how `lcda` handles cases with missing values on the manifest variables. If `TRUE`, those cases are removed (listwise deleted) before estimating the model. If `FALSE`, cases with missing values are retained. Cases with missing covariates are always removed. The default is `TRUE`.
`...`	Further arguments to be passed to `lcda`.

Details

The `lcda`-function performs a Latent Class Discriminant Analysis (LCDA). A Latent Class Modell will be estimated for each class by the `poLCA`-function (see `poLCA`). The class conditional model is given by

$$f_k(x) = \sum_{m=1}^{M_k} w_{mk} \prod_{d=1}^{D} \prod_{r=1}^{R_d} \theta_{mkdr}^{x_{kdr}},$$

where k is the class index, m is the latent subclass index, d is the variable index and r is the observation index. The variable x_{kdr} is 1 if the variable d of this observation is r and in class k. The parameter w_{mk} is the class conditional mixture weight and θ_{mkdr} is the probability for outcome r of variable d in subclass m of class k.

These Latent Class Models use the assumption of local independence to estimate a mixture model of latent multi-way tables. The mixture models are estimated by the EM-algorithm. The number of mixture components (`m`) is specified by the user. Estimated parameters include the latent-class conditional response probabilities for each manifest variable θ_{mkdr} and the class conditional mixing proportions w_{mk} denoting the population share of observations corresponding to each latent multi-way table.

Posterior class probabilities and class memberships can be estimated with the `predict` method.

Value

A list of class `lcda` containing the following components:

`call`	The (matched) function call.
`lca.theta`	The estimated class conditional response probabilities of the LCA given as a list of matrices like `probs.start`.

`lca.w`	The estimated mixing proportions of the LCA.
`prior`	Prior probabilites.
`m`	Number of latent subclasses per class.
`r`	Number of possible responses per variable.
`k`	Number of classes.
`d`	Number of variables.
`aic`	Value of the AIC for each class conditional Latent Class Model.
`bic`	Value of the BIC for each class conditional Latent Class Model.
`Gsq`	The likelihood ratio/deviance statistic for each class conditional model.
`Chisq`	The Pearson Chi-square goodness of fit statistic for fitted vs. observed multiway tables for each class conditional model.

Note

If the number of latent classes per class is unknown a model selection must be accomplished to determine the value of `m`. For this goal there are some model selection criteria implemented. The AIC, BIC, likelihood ratio statistic and the Chi-square goodness of fit statistic are taken from the poLCA-function (see `poLCA`). For each class these criteria can be regarded separately and for each class the number of latent classes can be determined.

Author(s)

Michael B\"ucker

See Also

`predict.lcda`, `cclcda`, `predict.cclcda`, `cclcda2`, `predict.cclcda2`, `poLCA`

Examples

```
# response probabilites for class 1
probs1 <- list()
probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=2, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=2, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=2, byrow=TRUE)

# response probabilites for class 2
probs2 <- list()
probs2[[1]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=2, byrow=TRUE)
probs2[[2]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs2[[3]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs2[[4]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=2, byrow=TRUE)

# generation of data
simdata1 <- poLCA.simdata(N = 500, probs = probs1, nclass = 2,
              ndv = 4, nresp = 4, missval = FALSE)

simdata2 <- poLCA.simdata(N = 500, probs = probs2, nclass = 2,
              ndv = 4, nresp = 4, missval = FALSE)

data1 <- simdata1$dat
data2 <- simdata2$dat

data <- cbind(rbind(data1, data2), rep(c(1,2), each=500))
names(data)[5] <- "grouping"
data <- data[sample(1:1000),]
grouping <- data[[5]]
data <- data[,1:4]

# lcda-procedure
object <- lcda(data, grouping=grouping, m=2)
object
```

predict.cclcda *Predict method for Common Components Latent Class Discriminant Analysis (CCLCDA)*

Description

Classifies new observations using parameters determined by the cclcda-function.

Usage

```
## S3 method for class 'cclcda'
predict(object, newdata, ...)
```

Arguments

object	Object of class cclcda.
newdata	Data frame of cases to be classified.
...	Further arguments are ignored.

Details

Posterior probabilities for new observations using parameters determined by the cclcda-function are computed. The classification of the new data is done by the Bayes decision function.

Value

A list with components:

class	Vector (of class factor) of classifications.
posterior	Posterior probabilities for the classes. For details of computation see cclcda.

Author(s)

Michael B\"ucker

See Also

cclcda, lcda, predict.lcda, cclcda2, predict.cclcda2, poLCA

Examples

```
# response probabilites
probs1 <- list()

probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1,
                        0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=4, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1,
                        0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7,
                        0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1,
                        0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=4, byrow=TRUE)

prior <- c(0.5,0.5)
wmk <- matrix(c(0.45,0.45,0.05,0.05,0.05,0.05,0.45,0.45),
              ncol=4, nrow=2, byrow=TRUE)
wkm <- apply(wmk*prior, 2, function(x) x/sum(x))

# generation of training data
data_temp <- poLCA.simdata(N = 1000, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data <- data_temp$dat
lclass <- data_temp$trueclass
grouping <- numeric()
for (i in 1:length(lclass))
{
grouping[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# generation of test data
data_temp <- poLCA.simdata(N = 500, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data.test <- data_temp$dat
lclass <- data_temp$trueclass
grouping.test <- numeric()
for (i in 1:length(lclass))
{
grouping.test[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# cclcda-procedure
object <- cclcda(data, grouping, m=4)
pred <- predict(object, data.test)$class
1-(sum(pred==grouping.test)/500)
```

---

predict.cclcda2 — *Predict method for Common Components Latent Class Discriminant Analysis 2 (CCLCDA2)*

Description

Classifies new observations using parameters determined by the cclcda2-function.

Usage

```
## S3 method for class 'cclcda2'
predict(object, newdata, ...)
```

Arguments

object	Object of class cclcda2.
newdata	Data frame of cases to be classified.
...	Further arguments are ignored.

Details

Posterior probabilities for new observations using parameters determined by the cclcda2-function are computed. The classification of the new data is done by the Bayes decision function.

Value

A list with components:

class	Vector (of class factor) of classifications.
posterior	Posterior probabilities for the classes. For details of computation see cclcda2.

Author(s)

Michael B\"ucker

See Also

cclcda2, lcda, predict.lcda, cclcda, predict.cclcda, poLCA

Examples

```
# response probabilites
probs1 <- list()

probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1,
                        0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=4, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1,
                        0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7,
                        0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=4, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1,
                        0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=4, byrow=TRUE)

prior <- c(0.5,0.5)
wmk <- matrix(c(0.45,0.45,0.05,0.05,0.05,0.05,0.45,0.45),
              ncol=4, nrow=2, byrow=TRUE)
wkm <- apply(wmk*prior, 2, function(x) x/sum(x))

# generation of training data
data_temp <- poLCA.simdata(N = 1000, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data <- data_temp$dat
lclass <- data_temp$trueclass
grouping <- numeric()
for (i in 1:length(lclass))
{
grouping[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# generation of test data
data_temp <- poLCA.simdata(N = 500, probs = probs1,
                           nclass = 2, ndv = 4, nresp = 4,
                           P=rep(0.25,4))
data.test <- data_temp$dat
lclass <- data_temp$trueclass
grouping.test <- numeric()
for (i in 1:length(lclass))
{
grouping.test[i] <- sample(c(1,2),1, prob=wkm[,lclass[i]])
}

# cclcda2-procedure
object <- cclcda2(data, grouping, m=4)
```

```
pred <- predict(object, data.test)$class
1-(sum(pred==grouping.test)/500)
```

`predict.lcda` *Predict method for Latent Class Discriminant Analysis (LCDA)*

Description

Classifies new observations using the parameters determined by the `lcda`-function.

Usage

```
## S3 method for class 'lcda'
predict(object, newdata, ...)
```

Arguments

`object`	Object of class `lcda2`.
`newdata`	Data frame of cases to be classified.
`...`	Further arguments are ignored.

Details

Posterior probabilities for new observations using parameters determined by the `lcda`-function are computed. The classification of the new data is done by the Bayes decision function.

Value

A list with components:

`class`	Vector (of class `factor`) of classifications.
`posterior`	Posterior probabilities for the classes. For details of computation see `lcda`.

Author(s)

Michael B\"ucker

See Also

`lcda`, `cclcda`, `predict.cclcda`, `cclcda2`, `predict.cclcda2`, `poLCA`

Examples

```
# response probabilites for class 1
probs1 <- list()
probs1[[1]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs1[[2]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=2, byrow=TRUE)
probs1[[3]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=2, byrow=TRUE)
probs1[[4]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=2, byrow=TRUE)

# response probabilites for class 2
probs2 <- list()
probs2[[1]] <- matrix(c(0.1,0.1,0.7,0.1,0.1,0.1,0.1,0.7),
                      nrow=2, byrow=TRUE)
probs2[[2]] <- matrix(c(0.1,0.1,0.1,0.7,0.7,0.1,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs2[[3]] <- matrix(c(0.7,0.1,0.1,0.1,0.1,0.7,0.1,0.1),
                      nrow=2, byrow=TRUE)
probs2[[4]] <- matrix(c(0.1,0.7,0.1,0.1,0.1,0.1,0.7,0.1),
                      nrow=2, byrow=TRUE)

# generation of data
simdata1 <- poLCA.simdata(N = 500, probs = probs1, nclass = 2,
              ndv = 4, nresp = 4, missval = FALSE)

simdata2 <- poLCA.simdata(N = 500, probs = probs2, nclass = 2,
              ndv = 4, nresp = 4, missval = FALSE)

data1 <- simdata1$dat
data2 <- simdata2$dat

data <- cbind(rbind(data1, data2), rep(c(1,2), each=500))
names(data)[5] <- "grouping"
data <- data[sample(1:1000),]
grouping <- data[[5]]
data <- data[,1:4]

# lcda-procedure
object <- lcda(data, grouping=grouping, m=2)
pred.class <- predict(object, newdata=data)$class
sum(pred.class==grouping)/length(pred.class)
```

Index