Package ‘bbl’

October 12, 2022

Title Boltzmann Bayes Learner

Version 1.0.0

Maintainer Jun Woo <junwoo035@gmail.com>

Depends R (>= 3.6.0)

Imports methods, stats, utils, Rcpp (>= 0.12.16), pROC, RColorBrewer

Description Supervised learning using Boltzmann Bayes model inference, which extends naive Bayes model to include interactions. Enables classification of data into multiple response groups based on a large number of discrete predictors that can take factor values of heterogeneous levels. Either pseudo-likelihood or mean field inference can be used with L2 regularization, cross-validation, and prediction on new data. <doi:10.18637/jss.v101.i05>.

License GPL (>= 2)

LinkingTo Rcpp

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation yes

Suggests glmnet, BiocManager, Biostrings

Author Jun Woo [aut, cre] (<https://orcid.org/0000-0003-3220-2064>)

Repository CRAN

Date/Publication 2022-01-27 23:20:12 UTC

R topics documented:

`bbl` *Boltzmann Bayes Learning Inference*

Description

Main driver for bbl inference

Usage

```
bbl(
  formula,
  data,
  weights,
  xlevels = NULL,
  verbose = 1,
  method = "pseudo",
  novarOk = FALSE,
  testNull = TRUE,
  prior.count = 1,
  ...
)
```

Arguments

`formula`	Formula for modeling
`data`	Data for fitting
`weights`	Vector of weights for each instance in data. Restricted to non-negative integer frequencies, recoding the number of times each row of data must be repeated. If `NULL`, assumed to be all 1. Fractional weights are not supported. Can be a named column in `data`
`xlevels`	List of factor levels for predictors. If `NULL`, will be inferred from data with factor levels ordered alphanumerically.
`verbose`	Output verbosity level. Will be send to down-stream function calls with one level lower
`method`	BB inference algorithm; pseudo-likelihood inference (`'pseudo'`) or mean field (`'mf'`)
`novarOk`	If `TRUE`, will proceed with predictors having only one level
`testNull`	Repeat the inference for the 'pooled' sample; i.e., under the null hypothesis of all rows in data belonging to a single group
`prior.count`	Prior count for computing single predictor and pairwise frequencies
`...`	Other parameters to `mlestimate`.

Details

Formula argument and data are used to tabulate xlevels unless explicitly given as list. Data are expected to be factors or integers. This function is a driver interepreting formula and calls `bbi.fit`. Will stop with error if any predictor has only one level unless `novarOk='TRUE'`. Use `removeConst` to remove the non-varying predictors before calling if this happens.

Value

A list of class `bbl` with the following elements:

`coefficients`	List of inferred coefficients with elements `h`, `J`, `h0`, and `J0`. The bias parameter `h` is a list of length equal to no. of response groups, each of which is a list of the same struture as `xlevels`: length equal to no. of predictors, containing vectors of length equal to each predictor factor levels: $h_i^{(y)}(x)$ represented by `h[[y]][[i]][x]`. The interaction parameter `J` is a list of lists of dimension $m \times m$, where m is the number of predictors. Each element is a matrix of dimension $L_i \times L_j$, where L_i and L_j are numbers of factor levels in predictor `i` and `j`: $J_{ij}^{(y)}(x_1, x_2)$ represented by `J[[y]][[i]][[j]][x1,x2]`. All elements of lists are named. The pooled parameters `h0` and `J0`, if computed, are of one less dimension, omitting response group argument.
`xlevels`	List of vectors containing predictor levels.
`terms`	The `terms` of `formula` input.
`groups`	Vector of response groups.
`groupname`	Name of the response variable.

`qJ`	Matrix of logicals whose elements record whether `formula` includes interaction between the two predictors.
`model`	Model data frame derived from `formula` and `data`.
`lkh`	Log likelihood.
`lz`	Vector log partition function. Used in `predict`.
`weights`	Vector of integral weights (frequencies).
`call`	Function call.
`df`	Degrees of freedom.

Author(s)

Jun Woo, <`junwoo035@gmail.com`>

References

doi: 10.18637/jss.v101.i05

Examples

```
titanic <- as.data.frame(Titanic)
b <- bbl(Survived ~ (Class + Sex + Age)^2, data = titanic, weights = Freq)
b
```

`bbl.fit` *bbl Inference with model matrix*

Description

Performs bbl inference using response vector and predictor matrix

Usage

```
bbl.fit(
  x,
  y,
  qJ = NULL,
  weights = NULL,
  xlevels = NULL,
  verbose = 1,
  method = "pseudo",
  prior.count = 1,
  ...
)
```

Arguments

`x`	Data frame of factors with each predictor in columns.
`y`	Vector of response variables.
`qJ`	Matrix of logicals indicating which predictor combinations are interacting.
`weights`	Vector of non-negative integer frequencies, recoding the number of times each row of data must be repeated. If `NULL`, assumed to be all 1. Fractional weights are not supported.
`xlevels`	List of factor levels for predictors. If `NULL`, will be inferred from data with factor levels ordered alphanumerically.
`verbose`	Verbosity level of output. Will be propagated to `mlestimate` with one level down.
`method`	`c('pseudo','mf')`; inference method.
`prior.count`	Prior count for computing single predictor and pairwise frequencies
`...`	Other arguments to `mlestimate`.

Details

This function would normally be called by `bbl` rather than directly. Expects the predictor data `x` and response vector `y` instead of formula input to `bbl`.

Value

List of named components `h`, `J`, `lkh`, and `lz`; see `bbl` for information regarding these components.

Examples

```
titanic <- as.data.frame(Titanic)
freq <- titanic$Freq
x <- titanic[,1:3]
y <- titanic$Survived
b <- bbl.fit(x=x,y=y, weights=freq)
b
```

`crossVal` *Cross-Validation of BB Learning*

Description

Run multiple fittings of `bbl` model with training/validation division of data

Usage

```
crossVal(
  formula,
  data,
  weights,
  novarOk = FALSE,
  lambda = 1e-05,
  lambdah = 0,
  eps = 0.9,
  nfold = 5,
  method = "pseudo",
  use.auc = TRUE,
  verbose = 1,
  progress.bar = FALSE,
  storeOpt = TRUE,
  ...
)
```

Arguments

`formula`	Formula for model. Note that intercept has no effect.
`data`	Data frame of data. Column names must match `formula`.
`weights`	Frequency vector of how many times each row of `data` must be repeated. If `NULL`, defaults to vector of 1s. Fractional values are not supported.
`novarOk`	Proceed even when there are predictors with only one factor level.
`lambda`	Vector of L2 penalizer values for `method = 'pseudo'`. Inferences will be repeated for each value. Restricited to non-negative values.
`lambdah`	L2 penalizer in `method = 'pseudo'` applied to parameter `h`. In contrast to `lambda`, only a single value is allowed.
`eps`	Vector of regularization parameters, $\epsilon \in [0,1]$, for `method = 'mf'`. Inference will be repeated for each value.
`nfold`	Number of folds for training/validation split.
`method`	`c('pseudo','mf')` for pseudo-likelihood maximization or mean field.
`use.auc`	Use AUC as the measure of prediction accuracy. Only works if response groups are binary. If `FALSE`, mean prediction group accuracy will be used as score.
`verbose`	Verbosity level. Downgraded when relayed into `bbl`.
`progress.bar`	Display progress bar in `predict`.
`storeOpt`	Store the optimal fitted object of class `bbl`.
`...`	Other parameters to `mlestimate`.

Details

The `data` slot of `object` is split into training and validation subsets of (`nfold`-1):1 ratio. The model is trained with the former and validated on the latter. Individual division/fold results are combined into validation result for all instances in the data set and prediction score is evaluated using the known response group identity.

Value

Object of class `cv.bbl` extending `bbl`, a list with extra components: `regstar`, Value of regularization parameter, `lambda` and `eps` for `method='pseudo'` and `method='mf'`,respectively, at which the accuracy score is maximized; `maxscore`, Value of maximum accuracy; `cvframe`, Data frame of regularization parameters and scores scanned. If `use.auc=TRUE`, also contains 95

Examples

```
set.seed(513)
m <- 5
n <- 100
predictors <- list()
for(i in 1:m) predictors[[i]] <- c('a','c','g','t')
names(predictors) <- paste0('v',1:m)
par <- list(randompar(predictors), randompar(predictors, h0=0.1, J0=0.1))
dat <- randomsamp(predictors, response=c('ctrl','case'), par=par, nsample=n)
cv <- crossVal(y ~ .^2, data=dat, method='mf', eps=seq(0.1,0.9,0.1))
cv
```

`fitted.bbl` *Fitted Response Group Probabilities*

Description

Response group probabilities from BBL fit

Usage

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

Arguments

`object`	Object of class `bbl`.
`...`	Other arguments

Details

This method returns predicted response group probabilities of trainig data

Value

Matrix of response group probabities with data points in rows and response groups in columns

Examples

```
titanic <- as.data.frame(Titanic)
fit <- bbl(Survived ~ Class + Sex + Age, data=titanic, weights=titanic$Freq)
```

`formula.bbl` *Formula in BBL Fitting*

Description

Returns the formula used in BBL fit

Usage

```
## S3 method for class 'bbl'
formula(x, ...)
```

Arguments

`x`	Object of class `bbl`
`...`	Other arguments

Value

Formula object

Examples

```
titanic <- as.data.frame(Titanic)
fit <- bbl(Survived ~ Class + Sex + Age, data=titanic, weights=titanic$Freq)
formula(fit)
```

`freq2raw` *Convert Frequency Table into Raw Data*

Description

Data with unique rows and a frequency column is converted into data with duplicate rows.

Usage

```
freq2raw(data, freq)
```

Arguments

`data`	Data frame with factors in columns
`freq`	Vector of frequency of each row in `data`; can be a named column in `data`; if missing, the column `Freq` is looked for in `data`

Details

The ouput data frame can be used as input to `bbl`.

Value

Data frame with one row per instances

Examples

```
Titanic
x <- as.data.frame(Titanic)
head(x)
titanic <- freq2raw(data=x[,1:3], freq=x$Freq)
head(titanic)
```

`logLik.bbl` *Log likelihood for bbl object*

Description

Compute log likelihood from a fitted `bbl` object

Usage

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

Arguments

`object`	Object of class `bbl`
`...`	Other arguments to methods

Details

This method uses inferred parameters from calls to `bbl` and data to compute the log likelihood.

Value

An object of class `logLik`, the Log likelihood value and the attribute "df" (degrees of freedom), the number of parameters.

`mcSample` *Sample Predictor Distributions*

Description

Uses fitted BBL model to explore predictor distributions

Usage

```
mcSample(object, nsteps = 1000, verbose = 1, progress.bar = TRUE)
```

Arguments

`object`	Object of class `bbl`
`nsteps`	Total number of MC steps
`verbose`	Verbosity level of output
`progress.bar`	Display progress bar

Details

After `bbl` fit, the resulting model is used by this function to sample predictor distributions in each response group and find the most likely preditor set using MCMC.

Examples

```
titanic <- as.data.frame(Titanic)
b <- bbl(Survived~., data=titanic[,1:4], weights=titanic$Freq)
pxy <- mcSample(b)
pxy
```

`mlestimate` *Maximum likelihood estimate*

Description

Perform inference of bias and interaction parameters for a single response group

Usage

```
mlestimate(
  xi,
  weights = NULL,
  qJ = NULL,
  method = "pseudo",
  L = NULL,
  lambda = 1e-05,
  lambdah = 0,
  symmetrize = TRUE,
  eps = 0.9,
  nprint = 100,
  itmax = 1e+05,
  tolerance = 1e-04,
  verbose = 1,
  prior.count = 1,
  naive = FALSE,
  lz.half = FALSE
)
```

Arguments

`xi`	Data matrix; expected to be numeric with elements ranging from zero to positive integral upper bound `L-1`.
`weights`	Frequency vector of number of times each row of `xi` is to be repeated. If `NULL`, defaults to 1. Expected to be non-negative integers.
`qJ`	Matrix of logicals indicating which predictor pairs are interacting. If `NULL`, all are allowed.
`method`	`c('pseudo','mf')` for pseudo-likelihood maximization or mean field inference.
`L`	Vector of number of factor levels in each predictor. If `NULL`, will be inferred from `xi`.
`lambda`	Vector of L2 regularization parameters for `method = 'pseudo'`. Applies to interaction parameters `J`.
`lambdah`	L2 parameters for `h` in `'pseudo'`. If `NULL`, it is set equal to `lambda`. `lambdah = 0` will free `h` from penalization.
`symmetrize`	Enforce the symmetry of interaction parameters by taking mean values of the matrix and its trace: $J_{ij}^{(y)}(x_1, x_2) = J_{ji}^{(y)}(x_2, x_1)$.
`eps`	Vector of regularization parameters for `mf`. Must be within the range of $\epsilon \in [0, 1]$.
`nprint`	Frequency of printing iteration progress under `'pseudo'`.
`itmax`	Maximum number of iterations for `'pseudo'`.
`tolerance`	Upper bound for fractional changes in pseduo-likelihood values before termiating iteration in `'pseudo'`.

`verbose`	Verbosity level.
`prior.count`	Prior count for `method = 'mf'` to reduce numerical instability.
`naive`	Naive Bayes inference. Equivalent to `method = 'mf'` together with `eps = 0`.
`lz.half`	Divide interaction term in approximation to $\ln Z_{iy}$ in `'pseudo'`.

Details

Given numeric data matrix, either pseudo-likelihood of mean-field theory is used to find the maximum likelihood estimate of bias `h` and interaction `J` parameters. Normally called by `bbl` rather than directly.

Value

List of inferred parameters `h` and `J`. See `bbl` for parameter structures.

Examples

```
set.seed(535)
predictors <- list()
for(i in 1:5) predictors[[i]] <- c('a','c','g','t')
par <- randompar(predictors)
par
xi <- sample_xi(nsample=5000, predictors=predictors, h=par$h, J=par$J,
                code_out=TRUE)
head(xi)
ps <- mlestimate(xi=xi, method='pseudo', lambda=0)
ps$h
ps$J[[1]]
mf <- mlestimate(xi=xi, method='mf', eps=0.9)
plot(x=unlist(par$h), y=unlist(ps$h), xlab='True', ylab='Inferred')
segments(x0=-2, x1=2, y0=-2, y1=2, lty=2)
points(x=unlist(par$J), y=unlist(ps$J), col='red')
points(x=unlist(par$h), y=unlist(mf$h), col='blue')
points(x=unlist(par$J), y=unlist(mf$J), col='green')
```

`model.frame.bbl` *Model Frame for BBL*

Description

Returns the model frame used in BBL fit

Usage

```
## S3 method for class 'bbl'
model.frame(formula, ...)
```

Arguments

`formula`	Object of class `bbl`
`...`	Other arguments

Value

Data frame used for fitting

Examples

```
titanic <- as.data.frame(Titanic)
fit <- bbl(Survived ~ Class + Sex + Age, data=titanic[,1:4], weights=titanic$Freq)
head(model.frame(fit))
```

`nobs.bbl` *Number of Observations in BBL Fit*

Description

Returns the number of observations from a BBL fit

Usage

```
## S3 method for class 'bbl'
nobs(object, ...)
```

Arguments

`object`	Object of class `bbl`
`...`	Other arguments

Value

An integer of number of observations

Examples

```
titanic <- as.data.frame(Titanic)
fit <- bbl(Survived ~ Class + Sex + Age, data=titanic[,1:4], weights=titanic$Freq)
nobs(fit)
```

`plot.bbl` *Plot bbl object*

Description

Visualize bias and interaction parameters

Usage

```
## S3 method for class 'bbl'
plot(x, layout = NULL, hcol = NULL, Jcol = NULL, npal = 100, ...)
```

Arguments

`x`	Object of class `bbl`
`layout`	Matrix of layouts for arrangment of linear and interaction parameters. If `NULL`, the top half will be used for linear parameter barplot and bottom half will be divided into interaction heatmaps for each response group.
`hcol`	Color for linear barplots. Grayscale if `NULL`.
`Jcol`	Color for interaction heatmaps. Default (`NULL`) is `RdBu` from `RColorBrewer`.
`npal`	Number of color scales.
`...`	Other graphical parameters for `plot`.

Details

This method displays a barplot of bias parameters and heatmaps (one per response group) of interaction parameters. All parameters are offset by the pooled values (single group inference) unless missing.

`plot.cv.bbl` *Plot Cross-validation Outcome*

Description

Plot cross-validation score as a function of regularization parameter

Usage

```
## S3 method for class 'cv.bbl'
plot(
  x,
  type = "b",
  log = "x",
  pch = 21,
  bg = "white",
  xlab = NULL,
  ylab = NULL,
  las = 1,
  ...
)
```

Arguments

`x`	Object of class `cv.bbl` from a call to `crossVal`
`type`	Symbol type in `plot`, present here to set default.
`log`	Log scale argument to `plot`.
`pch`	Symbol type code in `par`.
`bg`	Symbol background color in `par`.
`xlab`	X axis label
`ylab`	Y axis label
`las`	Orientation of axis labels in `par`.
`...`	Other arguments to `plot`.

Details

This function will plot accuracy score as a function of regularization parameter from a call to `crossVal`.

`predict.bbl` *Predict Response Group Using* `bbl` *Model*

Description

Make prediction of response group identity based on trained model

Usage

```
## S3 method for class 'bbl'
predict(object, newdata, type = "link", verbose = 1, progress.bar = FALSE, ...)
```

Arguments

`object`	Object of class `bbl` containing trained model
`newdata`	Data frame of new data for which prediction is to be made. Columns must contain all of those in `model@data`. If column names are present, the columns will be matched based on them. Extra columns will be ignored. If column names are not provided, the columns should exactly match `model@data` predictor parts. If `NULL`, replaced by `model@data` (self-prediction).
`type`	Return value type. If `'link'`, the logit scale probabilities. If `'prob'` the probability itself.
`verbose`	Verbosity level
`progress.bar`	Display progress of response group probability. Useful for large samples.
`...`	Other arguments to methods

Details

This method uses a new data set for predictors and trained `bbl` model parameters to compute posterior probabilities of response group identity.

Value

Data frame of predicted posterior probabilities with samples in rows and response groups in columns. The last column is the predicted response group with maximum probability.

Examples

```
set.seed(154)

m <- 5
L <- 3
n <- 1000

predictors <- list()
for(i in 1:m) predictors[[i]] <- seq(0,L-1)
names(predictors) <- paste0('v',1:m)
par <- list(randompar(predictors=predictors, dJ=0.5),
            randompar(predictors=predictors, h0=0.1, J0=0.1, dJ=0.5))
dat <- randomsamp(predictors=predictors, response=c('ctrl','case'), par=par,
                  nsample=n)
dat <- dat[sample(n),]
dtrain <- dat[seq(n/2),]
dtest <- dat[seq(n/2+1,n),]

model <- bbl(y ~ .^2, data=dtrain)
pred <- predict(model, newdata=dtest)
score <- mean(dtest$y==pred$yhat)
score

auc <- pROC::roc(response=dtest$y, predictor=pred$case, direction='<')$auc
auc
```

`predict.cv.bbl` *Predict using Cross-validation Object*

Description

Use the optimal fitted model from cross-validation run to make prediction

Usage

```
## S3 method for class 'cv.bbl'
predict(object, ...)
```

Arguments

`object`	Object of class `cv.bbl`.
`...`	Other parameters to `predict.bbl`.

Details

This method will use the fitted model with maximum accuracy score returned by a call to `crossVal` to make prediction on new data

Value

Data frame of prediction; see `predict.bbl`.

`print.bbl` *Print Boltzmann Bayes Learning Fits*

Description

This method displays model structure and first elements of coefficients

Usage

```
## S3 method for class 'bbl'
print(x, showcoeff = TRUE, maxcoeff = 3L, ...)
```

Arguments

`x`	An object of class `bbl`, usually dervied from a call to `bbl`.
`showcoeff`	Display first few fit coefficients
`maxcoeff`	Maximum number of coefficients to display
`...`	Further arguments passed to or from other methods

Details

Displays the call to `bbl`, response variable and its levels, predictors and their levels, and the first few fit coefficients.

`print.cv.bbl` *Display Cross-validation Result*

Description

Print cross-validation optimal result and data frame

Usage

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

Arguments

`x`	Object of class `cv.bbl`
`...`	Other arguments to methods

Details

This method prints `crossVal` object with the optimal regularization condition and maximum accuracy score on top and the entire score profile as a data frame below.

`print.summary.bbl` *Print Summary of Boltzmann Bayes Learning*

Description

This method prints the summary of `bbl` object

Usage

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

Arguments

`x`	Object of class `summary.bbl`
`...`	Other arguments to methods

Details

The naive Bayes summary of `summary.bbl` object is displayed.

`randompar` *Generate Random Parameters*

Description

Random values of bias and interaction parameters are generated using either uniform or normal distributions.

Usage

```
randompar(predictors, distr = "unif", h0 = 0, dh = 1, J0 = 0, dJ = 1)
```

Arguments

`predictors`	List of predictor factor levels. See `bbl`.
`distr`	`c('unif','norm')` for uniform or normal distributions.
`h0`	Mean of bias parameters
`dh`	`sd` of bias if `distr = 'unif'`. If `distr = 'norm'`, $h = [h_0 - dh, h_0 + dh]$.
`J0`	Mean of interaction parameters.
`dJ`	`sd` of interactions if `distr = 'unif'`. If `distr = 'norm'`, $J = [J_0 - dJ, J_0 + dJ]$.

Details

Input argument `predictors` is used to set up proper list structures of parameters.

Value

List of parameters, `h` and `J`.

Examples

```
set.seed(311)
predictors <- list()
for(i in 1:5) predictors[[i]] <- c('a','c')
par <- randompar(predictors=predictors)
par
```

`randomsamp` *Generate Random Boltzmann Bayes Model Data*

Description

Predictor-response paired data are generated

Usage

```
randomsamp(predictors, response, prob = NULL, par, nsample = 100)
```

Arguments

`predictors`	List of vectors of predictor levels
`response`	Vector of response variables
`prob`	Vector of probabilities for sampling each response group
`par`	List of `bbl` parameters for each response group; e.g., generated from calls to `randompar`.
`nsample`	Sample size

Details

The argument `response` is used to set up all possible levels of response groups and likewise for `predictors`. The parameter argument `par` must have the appropriate structure consistent with `response` and `predictors`. This function is a wrapper calling `sample_xi` multiple times.

Value

Data frame of response and predictor variables.

`readFasta` *Read FASTA File*

Description

Read nucleotide sequence files in FASTA format

Usage

```
readFasta(file, rownames = FALSE)
```

Arguments

`file`	File name of FASTA input.
`rownames`	Use the sequence annotation line in file (starts with `'>'`) as the row names. Will fail if there are duplicate items.

Details

Sequence data in FASTA files are converted into data frame suitable as input to `bbl`. If sequence lengths are different, instances longer than those already read will be truncated. Empty sequences are skipped.

Value

Data frame of each sequence in rows.

Examples

```
file <- tempfile('data')
write('>seq1', file)
write('atgcc', file, append=TRUE)
write('>seq2', file, append=TRUE)
write('gccaa', file, append=TRUE)
system(paste0('cat ',file))
x <- readFasta(file)
x
```

`removeConst` *Remove Non-varying Predictors*

Description

Constant predictor is identified and removed

Usage

```
removeConst(x)
```

Arguments

`x`	Data frame containing discrete factor variables in each column

Details

Variables with only one factor level is removed from data. Intended for use before calling `bbl`.

Value

Data frame omitting non-varying variables from `x`.

Examples

```
set.seed(351)
nt <- c('a','c','g','t')
x <- data.frame(v1=sample(nt,size=50,replace=TRUE),
                v2=rep('a',50),v3=sample(nt,size=50,replace=TRUE))
y <- sample(c('case','ctrl'),size=50,replace=TRUE)
dat <- cbind(data.frame(y=y), x)
summary(dat)
dat <- removeConst(dat)
summary(dat)
```

`residuals.bbl` *Residuals of BBL fit*

Description

Binary-valued vector of fitted vs. true response group

Usage

```
## S3 method for class 'bbl'
residuals(object, ...)
```

Arguments

`object`	Object of class `bbl`
`...`	Other arguments

Details

Discrete response group identity for each data point is compared with the fitted group and 0 (discordant) or 1 (concordant) is returned

Value

Vector binary values for each data point

Examples

```
titanic <- as.data.frame(Titanic)
dat <- freq2raw(titanic[,1:4], freq=titanic$Freq)
fit <- bbl(Survived ~ .^2, data=dat)
x <- residuals(fit)
table(x)
```

`sample_xi` *Generate Random Samples from Boltzmann Distribution*

Description

Random samples are drawn from Boltzmann distribution

Usage

```
sample_xi(nsample = 1, predictors = NULL, h, J, code_out = FALSE)
```

Arguments

`nsample`	Sample size
`predictors`	List of predictor factor levels.
`h`	Bias parameter; see `bbl`.
`J`	Interaction parameters; see `bbl`.
`code_out`	Ouput in integer codes; $a_i = 0, \cdots, L_i - 1$. If `FALSE`, output in factors in `predictors`.

Details

All possible factor states are enumerated exhaustively using input argument `predictors`. If the number of predictors m or the number of factor levels L_i for each predictor i are even moderately large ($m \geq 10$ or $L_i \geq 5$), this function will likely hang because the number of all possible states grows exponentially.

Value

Data frame of samples in rows and predictors in columns.

Examples

```
set.seed(512)
m <- 5
n <- 1000
predictors <- list()
for(i in 1:m) predictors[[i]] <- c('a','c','g','t')
par <- randompar(predictors)
xi <- sample_xi(nsample=n, predictors=predictors, h=par$h, J=par$J)
head(xi)
```

`summary.bbl` *Naive Bayes Summary*

Description

Estimate significant of predictor-group association using naive Bayes model

Usage

```
## S3 method for class 'bbl'
summary(object, prior.count = 0, ...)
```

Arguments

`object`	Object of class `bbl`
`prior.count`	Prior count to be used for computing naive Bayes coefficients and test results. If `0`, will produce `NA`s for factor levels without data points.
`...`	Other arguments to methods.

Details

This `summary.bbl` method gives a rough overview of associations within a `bbl` fit object via naive Bayes coefficients and test p-values. Note that naive Bayes results displayed ignore interactions even when interactions are present in the model being displayed. This feature is because simple analytic results exist for naive Bayes coefficients and test p-values. The likelihood ratio test is with respect to the null hypothesis that coefficients are identical for all response groups.

Value

Object of class `summary.bbl` extending `bbl` class; a list with extra components

`h`	List of bias coefficients of response groups under naive Bayes approximation
`h0`	Bias coefficients of pooled group under naive Bayes
`chisqNaive`	Vector of chi-square statistics for likelihood ratio test for each predictor
`dfNaive`	Vector of degrees of freedom for likelihood ratio test for each predictor
`pvNaive`	Vector p-values for each predictor

`weights.bbl` *Weights in BBL Fit*

Description

This method returns weights used in BBL fit.

Usage

```
## S3 method for class 'bbl'
weights(object, ...)
```

Arguments

`object`	Object of class `bbl`.
`...`	Other arguments

Details

Note that weithts are integral frequency values specifying repeat number of each instance in `bbl`. If no weights were used (default of 1s), `NULL` is returned.

Value

Vector of weights for each instance

Index