Package 'iterativeBMA'

November 12, 2026

Type Package

Title The Iterative Bayesian Model Averaging (BMA) algorithm

Version 1.70.0

Date 2009-7-22

Author Ka Yee Yeung, University of Washington, Seattle, WA, with contributions from Adrian Raftery and Ian Painter

Maintainer Ka Yee Yeung <kayee@u.washington.edu>

Description The iterative Bayesian Model Averaging (BMA) algorithm is a variable selection and classification algorithm with an application of classifying 2-class microarray samples, as described in Yeung, Bumgarner and Raftery (Bioinformatics 2005, 21: 2394-2402).

Depends BMA, leaps, Biobase (>= 2.5.5)

License GPL (>= 2)

URL http://faculty.washington.edu/kayee/research.html

biocViews Microarray, Classification

git_url https://git.bioconductor.org/packages/iterativeBMA

git_branch RELEASE_3_23

git_last_commit 7da42d0

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

iterativeBMA-package 2
bma.predict 3
brier.score 5
BssWssFast 6
imageplot.iterate.bma 7

iterateBMAglm.train 8
iterateBMAglm.train.predict 11
iterateBMAglm.train.predict.test 13
iterateBMAglm.wrapper 15
iterativeBMA-internal 17
testClass 18
testData 18
trainClass 19
trainData 20

Index **21**

`iterativeBMA-package` *The Iterative Bayesian Model Averaging (BMA) algorithm*

Description

The iterative Bayesian Model Averaging (BMA) algorithm is a variable selection and classification algorithm with an application of classifying 2-class microarray samples, as described in Yeung, Bumgarner and Raftery (Bioinformatics 2005, 21: 2394-2402).

Details

Package:	iterativeBMA
Type:	Package
Version:	0.1.0
Date:	2005-12-30
License:	GPL version 2 or higher

The function `iterateBMAglm.train` selects relevant variables by iteratively applying the `bic.glm` function from the `BMA` package. The data is assumed to consist of two classes. The function `iterateBMAglm.train.predict` combines the training and prediction phases, and returns the predicted posterior probabilities that each test sample belongs to class 1. The function `iterateBMAglm.train.predict.test` combines the training, prediction and test phases, and returns a list consisting of the numbers of selected genes and models using the training data, the number of classification errors and the Brier Score on the test set.

Author(s)

Ka Yee Yeung, University of Washington, Seattle, WA, with contributions from Adrian Raftery and Ian Painter

Maintainer: Ka Yee Yeung <kayee@u.washington.edu>

References

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train.predict, iterateBMAglm.train.predict.test, bma.predict, brier.score

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)

## training phase: select relevant genes
ret.bic.glm <- iterateBMAglm.train (train.expr.set=trainData, trainClass, p=100)

## get the selected genes with probne0 > 0
ret.gene.names <- ret.bic.glm$namesx[ret.bic.glm$probne0 > 0]

data (testData)

## get the subset of test data with the genes from the last iteration of bic.glm
curr.test.dat <- t(exprs(testData)[ret.gene.names,])

## to compute the predicted probabilities for the test samples
y.pred.test <- apply (curr.test.dat, 1, bma.predict, postprobArr=ret.bic.glm$postprob, mleArr=ret.bic.glm$mle)

## compute the Brier Score if the class labels of the test samples are known
data (testClass)
brier.score (y.pred.test, testClass)
```

`bma.predict` *Predicted Probabilities from Bayesian Model Averaging*

Description

This function computes the predicted posterior probability that each test sample belongs to class 1. It assumes 2-class data, and requires the true class labels to be known.

Usage

```
bma.predict (newdataArr, postprobArr, mleArr)
```

Arguments

newdataArr	a vector consisting of the data from a test sample.
postprobArr	a vector consisting of the posterior probability of each BMA selected model.
mleArr	matrix with one row per model and one column per variable giving the maximum likelihood estimate of each coefficient for each BMA selected model.

Details

Let Y be the response variable (class labels for samples in our case). In Bayesian Model Averaging (BMA), the posterior probability of Y=1 given the training set is the weighted average of the posterior probability of Y=1 given the training set and model M multiplied by the posterior probability of model M given the training set, summing over a set of models M.

Value

A real number between zero and one, representing the predicted posterior probability.

References

Raftery, A.E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

brier.score, iterateBMAglm.train

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)

## training phase: select relevant genes
ret.bic.glm <- iterateBMAglm.train (train.expr.set=trainData, trainClass, p=100)

## get the selected genes with probne0 > 0
ret.gene.names <- ret.bic.glm$namesx[ret.bic.glm$probne0 > 0]

data (testData)

## get the subset of test data with the genes from the last iteration of bic.glm
curr.test.dat <- t(exprs(testData)[ret.gene.names,])

## to compute the predicted probabilities for the test samples
y.pred.test <- apply (curr.test.dat, 1, bma.predict, postprobArr=ret.bic.glm$postprob, mleArr=ret.bic.glm$mle)
```

```
## compute the Brier Score if the class labels of the test samples are known
data (testClass)
brier.score (y.pred.test, testClass)
```

`brier.score` *Brier Score: assessment of prediction accuracy*

Description

The Brier Score is a probabilistic number of errors that takes the predicted probabilities into consideration. A small Brier Score indicates high prediction accuracy. This function assumes 2-class data, and requires the true class labels to be known.

Usage

```
brier.score (predictedArr, truthArr)
```

Arguments

`predictedArr`	a vector consisting of the predicted probabilities that the test sample belongs to class 1.
`truthArr`	a zero-one vector indicating the known class labels of the test samples. We assume this vector has the same length as `predictedArr`.

Details

The Brier Score computes the sum of squares of the differences between the true class and the predicted probability over all test samples. If the predicted probabilities are constrained to equal to 0 or 1, the Brier Score is equal to the total number of classification errors.

Value

A non-negative real number.

References

Brier, G.W. (1950) Verification of forecasts expressed in terms of probability. Monthly Weather Review 78: 1-3.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

`bma.predict`, `iterateBMAglm.train.predict`

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)
data (testData)

ret.vec <- iterateBMAglm.train.predict (train.expr.set=trainData, test.expr.set=testData, trainClass, p=100)

## compute the Brier Score
data (testClass)
brier.score (ret.vec, testClass)
```

`BssWssFast` *Between-groups sum-of-squares to within-groups sum-of-squares ratio*

Description

This is a univariate technique to select relevant genes in classification of microarray data. In classifying samples of microarray data, this ratio is computed for each gene. A large between-groups to within-groups sum-of-squares ratio indicates a potentially relevant gene.

Usage

```
BssWssFast (X, givenClassArr, numClass = 2)
```

Arguments

`X`	data matrix where columns are variables and rows are observations. In the case of gene expression data, the columns (variables) represent genes, while the rows (observations) represent samples or experiments.
`givenClassArr`	class vector for the observations (samples or experiments). Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in X. In the case of 2-class data, we expect the class vector consists of zero's and one's.
`numClass`	number of classes. The default is 2.

Details

This function is called by `iterateBMAglm.2class`.

Value

A list of 2 elements are returned:

`x`	A vector containing the BSS/WSS ratios in descending order.
`ix`	A vector containing the indices corresponding to the sorted ratios.

References

Dudoit, S., Fridlyand, J. and Speed, T.P. (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97: 77-87.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train, trainData, trainClass

Examples

```
data(trainData)
data(trainClass)

ret.bsswss <- BssWssFast (X=t(exprs(trainData)), givenClassArr=trainClass, numClass = 2)
```

`imageplot.iterate.bma` *An image plot visualization tool*

Description

Create a visualization of the models and variables selected by the iterative BMA algorithm.

Usage

```
imageplot.iterate.bma (bicreg.out, color="default", ...)
```

Arguments

`bicreg.out`	An object of type 'bicreg', 'bic.glm' or 'bic.surv'
`color`	The color of the plot. The value "default" uses the current default R color scheme for image. The value "blackandwhite" produces a black and white image.
`...`	Other parameters to be passed to the image and axis functions.

Details

This function is a modification of the `imageplot.bma` function from the `BMA` package. The difference is that variables (genes) with `probne0` equal to 0 are removed before plotting. The arguments of this function is identical to those in `imageplot.bma`.

Value

An heatmap-style image, with the BMA selected variables on the vertical axis, and the BMA selected models on the horizontal axis. The variables (genes) are sorted in descreasing order of the posterior probability that the variable is not equal to 0 (`probne0`) from top to bottom. The models are sorted in descreasing order of the model posterior probability (`postprob`) from left to right.

Note

The `BMA` and `Biobase` packages are required.

References

Clyde, M. (1999) Bayesian Model Averaging and Model Search Strategies (with discussion). In Bayesian Statistics 6. J.M. Bernardo, A.P. Dawid, J.O. Berger, and A.F.M. Smith eds. Oxford University Press, pages 157-185.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

`iterateBMAglm.train`

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)

## training phase: select relevant genes
ret.bic.glm <- iterateBMAglm.train (train.expr.set=trainData, trainClass, p=100)

## produce an image plot to visualize the selected genes and models
imageplot.iterate.bma (ret.bic.glm)
```

`iterateBMAglm.train` *Iterative Bayesian Model Averaging: training step*

Description

Classification and variable selection on microarray data. This is a multivariate technique to select a small number of relevant variables (typically genes) to classify microarray samples. This function performs the training phase. The data is assumed to consist of two classes. Logistic regression is used for classification.

Usage

```
iterateBMAglm.train (train.expr.set, train.class, p=100, nbest=10, maxNvar=30, maxIter=20000, thresPro
```

Arguments

train.expr.set	an ExpressionSet object. We assume the rows in the expression data represent variables (genes), while the columns represent samples or experiments. This training data is used to select relevant genes (variables) for classification.
train.class	class vector for the observations (samples or experiments) in the training data. Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in train.dat. Since we assume 2-class data, we expect the class vector consists of zero's and one's.
p	a number indicating the maximum number of top univariate genes used in the iterative BMA algorithm. This number is assumed to be less than the total number of genes in the training data. A larger p usually requires longer computational time as more iterations of the BMA algorithm are potentially applied. The default is 100.
nbest	a number specifying the number of models of each size returned to bic.glm in the BMA package. The default is 10.
maxNvar	a number indicating the maximum number of variables used in each iteration of bic.glm from the BMA package. The default is 30.
maxIter	a number indicating the maximum of iterations of bic.glm. The default is 20000.
thresProbne0	a number specifying the threshold for the posterior probability that each variable (gene) is non-zero (in percent). Variables (genes) with such posterior probability less than this threshold are dropped in the iterative application of bic.glm. The default is 1 percent.

Details

The training phase consists of first ordering all the variables (genes) by a univariate measure called between-groups to within-groups sums-of-squares (BSS/WSS) ratio, and then iteratively applying the bic.glm algorithm from the BMA package. In the first application of the bic.glm algorithm, the top maxNvar univariate ranked genes are used. After each application of the bic.glm algorithm, the genes with probne0 < thresProbne0 are dropped, and the next univariate ordered genes are added to the BMA window.

Value

An object of class bic.glm returned by the last iteration of bic.glm. The object is a list consisting of the following components:

namesx	the names of the variables in the last iteration of bic.glm.
postprob	the posterior probabilities of the models selected.
deviance	the estimated model deviances.
label	labels identifying the models selected.

bic	values of BIC for the models.
size	the number of independent variables in each of the models.
which	a logical matrix with one row per model and one column per variable indicating whether that variable is in the model.
probne0	the posterior probability that each variable is non-zero (in percent).
postmean	the posterior mean of each coefficient (from model averaging).
postsd	the posterior standard deviation of each coefficient (from model averaging).
condpostmean	the posterior mean of each coefficient conditional on the variable being included in the model.
condpostsd	the posterior standard deviation of each coefficient conditional on the variable being included in the model.
mle	matrix with one row per model and one column per variable giving the maximum likelihood estimate of each coefficient for each model.
se	matrix with one row per model and one column per variable giving the standard error of each coefficient for each model.
reduced	a logical indicating whether any variables were dropped before model averaging.
dropped	a vector containing the names of those variables dropped before model averaging.
call	the matched call that created the bma.lm object.

Note

The BMA and Biobase packages are required.

References

Raftery, A.E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train.predict, iterateBMAglm.train.predict.test, bma.predict, brier.score

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)

## training phase: select relevant genes
ret.bic.glm <- iterateBMAglm.train (train.expr.set=trainData, trainClass, p=100)
```

```
## get the selected genes with probne0 > 0
ret.gene.names <- ret.bic.glm$namesx[ret.bic.glm$probne0 > 0]

## show the posterior probabilities of selected models
ret.bic.glm$postprob

data (testData)

## get the subset of test data with the genes from the last iteration of bic.glm
curr.test.dat <- t(exprs(testData)[ret.gene.names,])

## to compute the predicted probabilities for the test samples
y.pred.test <- apply (curr.test.dat, 1, bma.predict, postprobArr=ret.bic.glm$postprob, mleArr=ret.bic.glm$mle)

## compute the Brier Score if the class labels of the test samples are known
data (testClass)
brier.score (y.pred.test, testClass)
```

`iterateBMAglm.train.predict`
Iterative Bayesian Model Averaging: training and prediction

Description

Classification and variable selection on microarray data. This is a multivariate technique to select a small number of relevant variables (typically genes) to classify microarray samples. This function performs the training, and prediction steps. The data is assumed to consist of two classes. Logistic regression is used for classification.

Usage

```
iterateBMAglm.train.predict (train.expr.set, test.expr.set, train.class, p=100, nbest=10, maxNvar=30,
```

Arguments

`train.expr.set`	an `ExpressionSet` object. We assume the rows in the expression data represent variables (genes), while the columns represent samples or experiments. This training data is used to select relevant genes (variables) for classification.
`test.expr.set`	an `ExpressionSet` object. We assume the rows in the expression data represent variables (genes), while the columns represent samples or experiments. The variables selected using the training data is used to classify samples on this test data.
`train.class`	class vector for the observations (samples or experiments) in the training data. Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in train.dat. Since we assume 2-class data, we expect the class vector consists of zero's and one's.

p	a number indicating the maximum number of top univariate genes used in the iterative BMA algorithm. This number is assumed to be less than the total number of genes in the training data. A larger p usually requires longer computational time as more iterations of the BMA algorithm are potentially applied. The default is 100.
nbest	a number specifying the number of models of each size returned to bic.glm in the BMA package. The default is 10.
maxNvar	a number indicating the maximum number of variables used in each iteration of bic.glm from the BMA package. The default is 30.
maxIter	a number indicating the maximum of iterations of bic.glm. The default is 20000.
thresProbne0	a number specifying the threshold for the posterior probability that each variable (gene) is non-zero (in percent). Variables (genes) with such posterior probability less than this threshold are dropped in the iterative application of bic.glm. The default is 1 percent.

Details

This function consists of the training phase and the prediction phase. The training phase consists of first ordering all the variables (genes) by a univariate measure called between-groups to within-groups sums-of-squares (BSS/WSS) ratio, and then iteratively applying the bic.glm algorithm from the BMA package. The prediction phase uses the variables (genes) selected in the training phase to classify the samples in the test set.

Value

A vector consisting of the predicted probability that each test sample belongs to class 1 is returned.

Note

The BMA and Biobase packages are required.

References

Raftery, A.E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train, iterateBMAglm.train.predict.test, brier.score

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)
data (testData)

ret.vec <- iterateBMAglm.train.predict (train.expr.set=trainData, test.expr.set=testData, trainClass, p=100)

## compute the Brier Score
data (testClass)
brier.score (ret.vec, testClass)
```

---

iterateBMAglm.train.predict.test

Iterative Bayesian Model Averaging: training, prediction and testing

---

Description

Classification and variable selection on microarray data. This is a multivariate technique to select a small number of relevant variables (typically genes) to classify microarray samples. This function performs the training, prediction and testing steps. The data is assumed to consist of two classes, and the classes of the test data is assumed to be known. Logistic regression is used for classification.

Usage

```
iterateBMAglm.train.predict.test (train.expr.set, test.expr.set, train.class, test.class, p=100, nbes
```

Arguments

`train.expr.set`	an `ExpressionSet` object. We assume the rows in the expression data represent variables (genes), while the columns represent samples or experiments. This training data is used to select relevant genes (variables) for classification.
`test.expr.set`	an `ExpressionSet` object. We assume the rows in the expression data represent variables (genes), while the columns represent samples or experiments. The variables selected using the training data is used to classify samples on this test data.
`train.class`	class vector for the observations (samples or experiments) in the training data. Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in train.dat. Since we assume 2-class data, we expect the class vector consists of zero's and one's.
`test.class`	class vector for the observations (samples or experiments) in the test data. Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in test.dat. Since we assume 2-class data, we expect the class vector consists of zero's and one's.

p	a number indicating the maximum number of top univariate genes used in the iterative BMA algorithm. This number is assumed to be less than the total number of genes in the training data. A larger p usually requires longer computational time as more iterations of the BMA algorithm are potentially applied. The default is 100.
nbest	a number specifying the number of models of each size returned to bic.glm in the BMA package. The default is 10.
maxNvar	a number indicating the maximum number of variables used in each iteration of bic.glm from the BMA package. The default is 30.
maxIter	a number indicating the maximum of iterations of bic.glm. The default is 20000.
thresProbne0	a number specifying the threshold for the posterior probability that each variable (gene) is non-zero (in percent). Variables (genes) with such posterior probability less than this threshold are dropped in the iterative application of bic.glm. The default is 1 percent.

Details

This function consists of the training phase, prediction phase, and the testing phase. The training phase consists of first ordering all the variables (genes) by a univariate measure called between-groups to within-groups sums-of-squares (BSS/WSS) ratio, and then iteratively applying the bic.glm algorithm from the BMA package. The prediction phase uses the variables (genes) selected in the training phase to classify the samples in the test set. The testing phase assumes that the class labels of the samples in the test set are known, and computes the number of classification errors and the Brier Score.

Value

A list consisting of 4 elements are returned:

num.genes	The number of relevant genes selected using the training data.
num.model	The number of models selected using the training data.
num.err	The number of classification errors produced when the the predicted class labels of the test samples are compared to the known class labels.
brierScore	The Brier Score computed using the predicted and known class labels of the test samples. The Brier Score represents a probabilistic number of errors. A small Brier Score implies high prediction accuracy.

Note

The BMA and Biobase packages are required.

References

Raftery, A.E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train, iterateBMAglm.train.predict

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)
data (testData)
data (testClass)

iterateBMAglm.train.predict.test (train.expr.set=trainData, test.expr.set=testData, trainClass, testClass, p=100
```

---

iterateBMAglm.wrapper *Iterative Bayesian Model Averaging*

Description

This function repeatedly calls bic.glm from the BMA package until all variables are exhausted. The data is assumed to consist of two classes. Logistic regression is used for classification.

Usage

```
iterateBMAglm.wrapper (sortedA, y, nbest=10, maxNvar=30, maxIter=20000, thresProbne0=1)
```

Arguments

sortedA	data matrix where columns are variables and rows are observations. The variables (columns) are assumed to be sorted using a univariate measure. In the case of gene expression data, the columns (variables) represent genes, while the rows (observations) represent samples or experiments.
y	class vector for the observations (samples or experiments) in the training data. Class numbers are assumed to start from 0, and the length of this class vector should be equal to the number of rows in sortedA. Since we assume 2-class data, we expect the class vector consists of zero's and one's.
nbest	a number specifying the number of models of each size returned to bic.glm in the BMA package. The default is 10.
maxNvar	a number indicating the maximum number of variables used in each iteration of bic.glm from the BMA package. The default is 30.
maxIter	a number indicating the maximum of iterations of bic.glm. The default is 20000.
thresProbne0	a number specifying the threshold for the posterior probability that each variable (gene) is non-zero (in percent). Variables (genes) with such posterior probability less than this threshold are dropped in the iterative application of bic.glm. The default is 1 percent.

Details

In this function, the variables are assumed to be sorted, and bic.glm is called repeatedly. In the first application of the bic.glm algorithm, the top maxNvar univariate ranked genes are used. After each application of the bic.glm algorithm, the genes with probne0 < thresProbne0 are dropped, and the next univariate ordered genes are added to the BMA window. The function iterateBMAglm.train calls BssWssFast before calling this function. Using this function, users can experiment with alternative univariate measures.

Value

If all variables are exhausted, an object of class bic.glm returned by the last iteration of bic.glm. Otherwise, -1 is returned. The object of class bic.glm is a list consisting of the following components:

namesx	the names of the variables in the last iteration of bic.glm.
postprob	the posterior probabilities of the models selected.
deviance	the estimated model deviances.
label	labels identifying the models selected.
bic	values of BIC for the models.
size	the number of independent variables in each of the models.
which	a logical matrix with one row per model and one column per variable indicating whether that variable is in the model.
probne0	the posterior probability that each variable is non-zero (in percent).
postmean	the posterior mean of each coefficient (from model averaging).
postsd	the posterior standard deviation of each coefficient (from model averaging).
condpostmean	the posterior mean of each coefficient conditional on the variable being included in the model.
condpostsd	the posterior standard deviation of each coefficient conditional on the variable being included in the model.
mle	matrix with one row per model and one column per variable giving the maximum likelihood estimate of each coefficient for each model.
se	matrix with one row per model and one column per variable giving the standard error of each coefficient for each model.
reduced	a logical indicating whether any variables were dropped before model averaging.
dropped	a vector containing the names of those variables dropped before model averaging.
call	the matched call that created the bma.lm object.

Note

The BMA and Biobase packages are required.

References

Raftery, A.E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

Yeung, K.Y., Bumgarner, R.E. and Raftery, A.E. (2005) Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21: 2394-2402.

See Also

iterateBMAglm.train, iterateBMAglm.train.predict, iterateBMAglm.train.predict.test, BssWssFast

Examples

```
library (Biobase)
library (BMA)
library (iterativeBMA)
data(trainData)
data(trainClass)

## Use the BSS/WSS ratio to rank all genes in the training data
sorted.vec <- BssWssFast (t(exprs(trainData)), trainClass, numClass = 2)
## get the top ranked 50 genes
sorted.train.dat <- t(exprs(trainData[sorted.vec$ix[1:50], ]))

## run iterative bic.glm
ret.bic.glm <- iterateBMAglm.wrapper (sorted.train.dat, y=trainClass)

## The above commands are equivalent to the following
ret.bic.glm <- iterateBMAglm.train (train.expr.set=trainData, trainClass, p=50)
```

---

iterativeBMA-internal *Internal functions for iterativeBMA*

Description

Internal functions for iterativeBMA, not meant to be called directly.

Usage

```
iterateBMAglm (x, y, curr.mat, stopVar=0, nextVar, nbest=10, thresProbne0 = 1, maxNvar = 30, maxIter=200
iterateBMAinit (x, maxNvar = 30)
imageplot.bma.mod (bicreg.out, color = "default", ...)
convertSingleName (curr.name, orig.expr.set)
convertModelName (curr.model, orig.expr.set)
maxExpValueKY
minExpValueKY
```

zero.threshold
bma.punct.string

---

testClass *Sample Test Data for the Iterative BMA Algorithm*

Description

This is an adapted leukemia (ALL, AML) dataset from Golub et al. (1999). This is a zero-one vector for 34 test samples. Class 0 represents an ALL sample, while class 1 represents an AML sample.

Usage

```
data(testClass)
```

Format

The vector is called testClass.

Source

The golubEsets bioconductor data package, or http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

References

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286: 531-7.

---

testData *Sample Test Data for the Iterative BMA Algorithm*

Description

This is an adapted leukemia (ALL, AML) ExpressionSet from Golub et al. (1999). This ExpressionSet consists of the expression levels from 38 ALL or AML samples (rows), and 100 genes (columns). This dataset is used as an example test data in our examples.

Usage

```
data(testData)
```

Format

The ExpressionSet is called testData. Each entry in the matrix represents the expression level of one gene from an ALL or AML sample.

Details

For illustration purposes, a subset of 100 genes from the package golubEsets is included in this package.

Source

The golubEsets bioconductor data package, or http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

References

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286: 531-7.

---

trainClass *Sample Training Data for the Iterative BMA Algorithm*

Description

This is an adapted leukemia (ALL, AML) dataset from Golub et al. (1999). This is a zero-one vector for 38 training samples. Class 0 represents an ALL sample, while class 1 represents an AML sample.

Usage

```
data(trainClass)
```

Format

The vector is called trainClass.

Source

The golubEsets bioconductor data package, or http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

References

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286: 531-7.

`trainData` *Sample Training Data for the Iterative BMA Algorithm*

Description

This is an adapted leukemia (ALL, AML) `ExpressionSet` from Golub et al. (1999). This `ExpressionSet` consists of the expression levels from 38 ALL or AML samples (rows), and 100 genes (columns). This dataset is used as an example training data in our examples.

Usage

```
data(trainData)
```

Format

The `ExpressionSet` is called `trainData`. Each entry in the exprs matrix represents the expression level of one gene from an ALL or AML sample.

Details

For illustration purposes, a subset of 100 genes from the package `golubEsets` is included in this package.

Source

The `golubEsets` bioconductor data package, or http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

References

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286: 531-7.

Index

∗ **classif**
bma.predict, 3
brier.score, 5
imageplot.iterate.bma, 7
iterateBMAglm.train, 8
iterateBMAglm.train.predict, 11
iterateBMAglm.train.predict.test, 13
iterateBMAglm.wrapper, 15
iterativeBMA-package, 2
∗ **datasets**
testClass, 18
testData, 18
trainClass, 19
trainData, 20
∗ **internal**
iterativeBMA-internal, 17
∗ **multivariate**
iterateBMAglm.train, 8
iterateBMAglm.train.predict, 11
iterateBMAglm.train.predict.test, 13
iterateBMAglm.wrapper, 15
iterativeBMA-package, 2
∗ **univar**
BssWssFast, 6

bma.predict, *3*, 3, *5*, *10*
bma.punct.string *(*iterativeBMA-internal*)*, 17
brier.score, *3, 4*, 5, *10*, *12*
BssWssFast, 6, *17*

convertModelName *(*iterativeBMA-internal*)*, 17
convertSingleName *(*iterativeBMA-internal*)*, 17

imageplot.bma.mod *(*iterativeBMA-internal*)*, 17
imageplot.iterate.bma, 7
iterateBMAglm *(*iterativeBMA-internal*)*, 17
iterateBMAglm.train, *4*, *7, 8*, 8, *12*, *15*, *17*
iterateBMAglm.train.predict, *3*, *5*, *10*, 11, *15*, *17*
iterateBMAglm.train.predict.test, *3*, *10*, *12*, 13, *17*
iterateBMAglm.wrapper, 15
iterateBMAinit *(*iterativeBMA-internal*)*, 17
iterativeBMA *(*iterativeBMA-package*)*, 2
iterativeBMA-internal, 17
iterativeBMA-package, 2

maxExpValueKY *(*iterativeBMA-internal*)*, 17
minExpValueKY *(*iterativeBMA-internal*)*, 17

testClass, 18
testData, 18
trainClass, *7*, 19
trainData, *7*, 20

zero.threshold *(*iterativeBMA-internal*)*, 17