Package 'a4Classif'

April 3, 2025

Type Package

Title Automated Affymetrix Array Analysis Classification Package

Version 1.55.0

Date 2021-07-13

Description Functionalities for classification of Affymetrix microarray data, integrating within the Automated Affymetrix Array Analysis set of packages.

Depends a4Core, a4Preproc

Imports methods, Biobase, ROCR, pamr, glmnet, varSelRF, utils, graphics, stats

Suggests ALL, hgu95av2.db, knitr, rmarkdown

License GPL-3

biocViews Microarray, GeneExpression, Classification

VignetteBuilder knitr

RoxygenNote 7.1.1

git_url https://git.bioconductor.org/packages/a4Classif

git_branch devel

git_last_commit 37f77ac

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-04-03

Author Willem Talloen [aut], Tobias Verbeke [aut], Laure Cougnaud [cre]

Maintainer Laure Cougnaud <laure.cougnaud@openanalytics.eu>

Contents

`lassoClass` *Classify using the Lasso*

Description

Classify using the Lasso

Usage

```
lassoClass(object, groups)
```

Arguments

`object`	object containing the expression measurements; currently the only method supported is one for ExpressionSet objects
`groups`	character string indicating the column containing the class membership

Value

object of class `glmnet`

Author(s)

Willem Talloen

References

Goehlmann, H. and W. Talloen (2009). Gene Expression Studies Using Affymetrix Microarrays, Chapman \& Hall/CRC, pp. 183, 205 and 212.

See Also

`glmnet`

Examples

```
if (require(ALL)){
  data(ALL, package = "ALL")
  ALL <- addGeneInfo(ALL)
  ALL$BTtype <- as.factor(substr(ALL$BT,0,1))

  resultLasso <- lassoClass(object = ALL, groups = "BTtype")
  plot(resultLasso, label = TRUE,
    main = "Lasso coefficients in relation to degree of
  penalization.")
  topTable(resultLasso, n = 15)
}
```

`pamClass` *Classify using Prediction Analysis for MicroArrays*

Description

Classify using the Prediction Analysis for MicroArrays (PAM) algorithm as implemented in the pamr package

Usage

```
pamClass(object, groups, probe2gene = TRUE)
```

Arguments

`object`	object containing the expression measurements; currently the only method supported is one for ExpressionSet objects
`groups`	character string indicating the column containing the class membership
`probe2gene`	logical; if `TRUE` Affymetrix probeset IDs are translated into gene symbols; if `FALSE` no such translation is conducted

Value

object of class `pamClass`

Author(s)

Willem Talloen

References

Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu (1999). Diagnosis of multiple cancer types by shrunken centroids of gene expression. PNAS 99: 6567-6572. Available at www.pnas.org

Goehlmann, H. and W. Talloen (2009). Gene Expression Studies Using Affymetrix Microarrays, Chapman \& Hall/CRC, p. 221.

See Also

pamr.train

Examples

```
if(require(ALL)){
 data(ALL, package = "ALL")
 ALL <- addGeneInfo(ALL)
 ALL$BTtype <- as.factor(substr(ALL$BT,0,1))
 resultPam <- pamClass(object = ALL, groups = "BTtype")
 plot(resultPam)
 topTable(resultPam, n = 5)
 confusionMatrix(resultPam)
}
```

---

rfClass *Classify using Random Forests*

Description

Classify using the Random Forest algorithm of Breiman (2001)

Usage

```
rfClass(object, groups, probe2gene = TRUE)
```

Arguments

object	object containing the expression measurements; currently the only method supported is one for ExpressionSet objects
groups	character string indicating the column containing the class membership
probe2gene	logical; if TRUE Affymetrix probeset IDs are translated into gene symbols in the output object; if FALSE no such translation is conducted

Value

Object of class 'rfClass'

Note

topTable and plot methods are available for 'rfClass' objects.

Author(s)

Tobias Verbeke and Willem Talloen

References

Breiman, L. (2001), *Random Forests*, Machine Learning 45(1), 5-32.

See Also

randomForest

Examples

```
if(require(ALL)){
 data(ALL, package = "ALL")
 ALL <- addGeneInfo(ALL)
 ALL$BTtype <- as.factor(substr(ALL$BT,0,1))
 # select only a subset of the data for computation time reason
 ALLSubset <- ALL[sample.int(n = nrow(ALL), size = 100, replace = TRUE), ]
 resultRf <- rfClass(object = ALLSubset, groups = "BTtype")
 plot(resultRf)
 topTable(resultRf, n = 15)
}
```

ROCcurve *Receiver operating curve*

Description

A ROC curve plots the fraction of true positives (TPR = true positive rate) versus the fraction of false positives (FPR = false positive rate) for a binary classifier when the discrimination threshold is varied. Equivalently, one can also plot sensitivity versus (1 - specificity).

Usage

```
ROCcurve(
  object,
  groups,
  probesetId = NULL,
  geneSymbol = NULL,
  main = NULL,
  probe2gene = TRUE,
  ...
)
```

Arguments

object	ExpressionSet object for the experiment
groups	String containing the name of the grouping variable. This should be a the name of a column in the pData of the expressionSet object.
probesetId	The probeset ID. These should be stored in the featureNames of the expressionSet object.

geneSymbol	The gene symbol. These should be stored in the column `Gene Symbol` in the featureData of the expressionSet object.
main	Main title on top of the graph
probe2gene	Boolean indicating whether the probeset should be translated to a gene symbol (used for the default title of the plot)
...	Possibility to add extra plot options. See par

Value

a plot is drawn in the current device. prediction object is returned invisibly.

Author(s)

Willem Talloen

References

Some explanation about ROC can be found on http://en.wikipedia.org/wiki/ROC_curve and http://www.anaesthetist.com/mnm/stats/roc/Findex.htm. The latter has at the bottom a nice interactive tool to scroll the cut-off and to see how it affects the FP/TP table and the ROC curve.

Examples

```
# simulated data set
esSim <- simulateData()
ROCcurve(probesetId = 'Gene.1', object = esSim, groups = 'type', addLegend = FALSE)
```

---

topTable,pamClass-method

Top table for pamClass *object*

Description

Top table for pamClass object

Usage

```
## S4 method for signature 'pamClass'
topTable(fit, n)
```

Arguments

fit	object for which to obtain a top table, generally a fit object for a given model class
n	number of features (variables) to list in the top table, ranked by importance

Value

topTablePam object

---

topTable,rfClass-method

Top table for rfClass *object*

Description

Top table for rfClass object

Usage

```
## S4 method for signature 'rfClass'
topTable(fit, n)
```

Arguments

fit	object for which to obtain a top table, generally a fit object for a given model class
n	number of features (variables) to list in the top table, ranked by importance

Value

topTableRfClass object

Index