Package 'geneClassifiers'

April 2, 2025

Type Package

Title Application of gene classifiers

Version 1.31.0

Description This packages aims for easy accessible application of classifiers which have been published in literature using an ExpressionSet as input.

URL https://doi.org/doi:10.18129/B9.bioc.geneClassifiers

BugReports https://github.com/rkuiper/geneClassifiers/issues

License GPL-2

biocViews GeneExpression, BiomedicalInformatics, Classification, Survival, Microarray

LazyData true

Suggests testthat

Depends R (>= 3.6.0)

Imports utils, methods, stats, Biobase, BiocGenerics

Collate aaa.R allGenericFunctions.R Class_TransformationProcess.R Class_FixedExpressionData.R Class_ClassifierParameters.R Class_ClassifierResults.R zzz.R geneClassifiers.R

RoxygenNote 6.1.1

Encoding UTF-8

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

git_branch devel

git_last_commit ef8632b

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-04-02

Author R Kuiper [cre, aut] (ORCID: <https://orcid.org/0000-0002-3703-1762>)

Maintainer R Kuiper <r.kuiper.emc@gmail.com>

Contents

`geneClassifiers-package`

geneClassifiers: Application of gene classifiers

Description

This packages aims for easy accessible application of classifiers which have been published in literature using an ExpressionSet as input.

Details

Combining gene expression profiling data with survival data has led to the development of robust outcome predictors (gene classifiers).This package provides a method for running gene classifiers generating patient specific predictive outcomes. This package is intended to support and enable research. To yield stable results, this package requires a dataset of at least 20 patients.

For detail on how to use this package see the vignette: `vignette("geneClassifiers")`

Author(s)

Maintainer: R Kuiper <r.kuiper.emc@gmail.com> (0000-0002-3703-1762)

References

Kuiper R, Broyl A, de Knegt YA, Van Vliet MH, Van Beers EH, van der Holt B, el Jarari L, Mulligan G, Gregory W, Morgan G, Goldschmidt H. A gene expression signature for high-risk multiple myeloma. Leukemia. 2012 Nov;26(11):2406.

See Also

Useful links:

- https://doi.org/doi:10.18129/B9.bioc.geneClassifiers
- Report bugs at https://github.com/rkuiper/geneClassifiers/issues

---

`ClassifierParameters` *An S4 class to store classifier parameters.*

Description

This class stores classifier related information This is information on probe-sets used and their weightings, means, standard deviations and covariance structure as observed in the classifiers training data, and the description of the procedure on how to preprocess new data prior to application of the classifier.

Slots

`name` A character string indicating the name of the classifier

`description` A short description of the classifier

`citations` A character vector of citations to literature

`normalizationMethod` A character string indicating the normalization method to apply

`eventChain` A list of preprocessing steps

`probeNames` A character vector

`intercept` A numeric value

`weights` A numeric vector

`decisionBoundaries` A numeric vector with values that separate the risk-groups

`doRun` A function which is called for the actual classification

`means` A numeric vector of probe-set means as observed in the trainingsset (if available)

`sds` A numeric vector of probe-set standard deviations as observed in the trainingsset (if available)

`.geneClassifierVersion` An object of class `package_version`

`ClassifierResults` *An S4 class to store classifier results.*

Description

This class stores classifier results as obtained after running the `runClassifier` function.

Slots

`classifierParameters` An object of class `ClassifierParameters` in which the applied classifier parameters are stored.

`score` A numeric vector of resulting classifier scores

`batchCorrection` A character vector indicating wheter batch correction was applied

`weightingType` A character string indicating wheter the weighting type was complete (i.e. no missing data), reweighted (i.e. missing data was handled based on correction using the covariance structure in the classifiers training data), or reduced (i.e. missing data but not reweighting the original probeset weighting)

`.geneClassifierVersion` An object of class `package_version`

`dim,FixedExpressionData-method`
Dimensions of an Object

Description

Retrieve the dimension of an object.

Usage

```
## S4 method for signature 'FixedExpressionData'
dim(x)
```

Arguments

`x`	an R object, for example a matrix, array or data frame.

Value

Retrieves the 'dim attribute of the object. It is 'NULL' or a vector of mode 'integer'.

See Also

Other fixed data information extraction functions: `[,FixedExpressionData,ANY,ANY-method`, `getNormalizationMethod`, `getTargetValue`

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0", targetValue=500)
dim(myData)
dim(myData[1:10,1:3])
```

---

exampleMAS5 *Example MAS5.0 ExpressionSet*

Description

An ExpressionSet. The data contains a sample of gene expression data from patients included in the HOVON65/GMMG-HD4 trial on multiple myeloma. The data was MAS5.0 normalized to a target value of 500.

Usage

```
exampleMAS5
```

Format

An object of class ExpressionSet with 374 rows and 25 columns.

FixedExpressionData *An S4 class to store classifier parameters.*

Description

This class stores gene expression data together with information on the normalization method and additional normalization related parameters. In order to ensure the data is not manipulated in unforeseen ways, manipulation is strictly controled through adding transformations which are predefined in the TransformationProcess-class. Upon reading the data by the exprs function, the transformations areperformed in the order the were added.

Slots

normalizationMethod A character string indicating the normalization method that was applied to the data. Possible values are give by getNormalizationMethods.

expressionEnvironment A locked environment in which the expression matrix is stored.

normalizationParameters A list with normalization specific values.

transformationProcess A locked environment to which the transformation processes are added.

.geneClassifierVersion An object of class package_version

getBatchCorrection *Obtain the batch correction status for a classifier result.*

Description

getBatchCorrection returns TRUE or FALSE indicating whether correction was applied

Usage

```
getBatchCorrection(object)

## S4 method for signature 'ClassifierResults'
getBatchCorrection(object)
```

Arguments

object	An object of class ClassifierResults as returned by runClassifier

Value

TRUE or FALSE

See Also

Other classifier results: getClassifications, getScores, getWeightingType

Examples

```
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('EMC92', myData)
getBatchCorrection( results )
```

getCitations *Obtain citations to the classifier*

Description

getCitations Obtain citations to the classifier

Usage

```
getCitations(object)

## S4 method for signature 'ClassifierParameters'
getCitations(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

A character vector

See Also

Other classifier information functions: getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getCitations(aClassifier)
```

---

getClassifications *Obtain classifier classifications.*

Description

getClassifications returns the resulting classifications.

Usage

```
getClassifications(object)

## S4 method for signature 'ClassifierResults'
getClassifications(object)
```

Arguments

object	An object of class ClassifierResults

Value

A vector of orderd factors with classifications per sample

See Also

Other classifier results: getBatchCorrection, getScores, getWeightingType

Examples

```
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('EMC92', myData)
getScores( results )
getClassifications( results )
```

getClassifier *Obtain a classifier definition.*

Description

getClassifier returns a requested classifier definition.

Usage

```
getClassifier(value)

## S4 method for signature 'ClassifierResults'
getClassifier(value)

## S4 method for signature 'character'
getClassifier(value)
```

Arguments

value	Either a text value indicating a classifier name (see showClassifierList), or an object of class ClassifierResults as returned by the runClassifier function.

Value

The return value is a classifier definition which is encoded in an object of class ClassifierParameters. This can be used as input argument for the runClassifier function.

See Also

ClassifierParameters and runClassifier

Other classifier information functions: getCitations, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
getClassifier("EMC92")
```

getDecisionBoundaries *Obtain the decision boundaries defined for the classifier.*

Description

getDecisionBoundaries returns the a numeric vector of boundary values that separate the risk groups.

Usage

```
getDecisionBoundaries(object)

## S4 method for signature 'ClassifierParameters'
getDecisionBoundaries(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

A numeric vector

See Also

Other classifier information functions: getCitations, getClassifier, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getDecisionBoundaries(aClassifier)
```

getDescription *Obtain classifiers' description.*

Description

getDescription returns the descriptive text associated with the classifier.

Usage

```
getDescription(object)

## S4 method for signature 'ClassifierParameters'
getDescription(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

A character string describing the classifier

See Also

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getDescription(aClassifier)
```

---

getEventChain *Obtain classifiers' event chain.*

Description

getEventChain returns the event chain encoded in the in the classifier. The eventchain indicates what preprocessing steps are performed by the runClassifier function prior to classification.

Usage

```
getEventChain(object)

## S4 method for signature 'ClassifierParameters'
getEventChain(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

Returns the event chain encoded in the in the classifier encoded as a named list.

See Also

showClassifierList getClassifier runClassifier

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getEventChain(aClassifier)
```

`getIntercept` *Obtain classifiers' intercept.*

Description

`getIntercept` returns the numeric value of the classifier's intercept.

Usage

```
getIntercept(object)

## S4 method for signature 'ClassifierParameters'
getIntercept(object)
```

Arguments

`object`	An object of class `ClassifierParameters` as returned by `getClassifier`

Value

A numeric value

See Also

Other classifier information functions: `getCitations`, `getClassifier`, `getDecisionBoundaries`, `getDescription`, `getEventChain`, `getMeans`, `getNormalizationMethod`, `getProbeNames`, `getSds`, `getTrainingData`, `getWeights`

Examples

```
aClassifier <- getClassifier("EMC92")
getIntercept(aClassifier)
```

getMeans *Obtain classifiers' reference means.*

Description

getMeans returns the reference means encoded in the in the classifier.

Usage

```
getMeans(object)

## S4 method for signature 'ClassifierParameters'
getMeans(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

Returns a numeric vector of probe set means as observed in the reference data

See Also

showClassifierList getClassifier runClassifier

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getNormalizationMethod, getProbeNames, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getMeans(aClassifier)
```

getName *Obtain object names.*

Description

getName returns the name associated with the requested object.

Usage

```
getName(object)

## S4 method for signature 'TransformationProcess'
getName(object)

## S4 method for signature 'ClassifierParameters'
getName(object)

## S4 method for signature 'ClassifierResults'
getName(object)
```

Arguments

object	The object to get the name of.

Value

The return value is a character string

See Also

ClassifierParameters

ClassifierResults

Examples

```
aClassifier <- getClassifier("EMC92")
getName( aClassifier )
```

---

getNormalizationMethod

Obtain normalization method

Description

The function getNormalizationMethod returns the normalization method associated with the object

getNormalizationMethods returns a character vector of currenlty available normalization methods.

Usage

```
getNormalizationMethod(object)

getNormalizationMethods()

## S4 method for signature 'FixedExpressionData'
getNormalizationMethod(object)

## S4 method for signature 'ClassifierParameters'
getNormalizationMethod(object)
```

Arguments

object	An object of class FixedExpressionData or ClassifierParameters

Details

The given normlization methods can be used in the

Value

A character string indicating the normalization method.

See Also

getNormalizationMethods

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getProbeNames, getSds, getTrainingData, getWeights

Other fixed data information extraction functions: [,FixedExpressionData,ANY,ANY-method, dim,FixedExpressionData-method, getTargetValue

Other workflow functions: runClassifier, setNormalizationMethod, showClassifierList

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0", targetValue=500)
aClassifier <- getClassifier("EMC92")
getNormalizationMethod( myData )
getNormalizationMethod( aClassifier )
data(exampleMAS5)

showClassifierList()
getNormalizationMethods()

myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('UAMS70', myData)

getScores( results )
```

```
getClassifications( results )
```

---

getProbeNames *Obtain probe-set names.*

Description

getProbeNames returns the probe names associated with the requested classifier.

Usage

```
getProbeNames(object)

## S4 method for signature 'ClassifierParameters'
getProbeNames(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

The return value is a character vector of probe-set names.

See Also

ClassifierParameters

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getSds, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getProbeNames( aClassifier )
```

getScores *Obtain classifier score.*

Description

getScores returns the resulting scores from a classifier run

Usage

```
getScores(object)

## S4 method for signature 'ClassifierResults'
getScores(object)
```

Arguments

object	An object of class ClassifierResults

Value

A numeric vector with scores per sample

See Also

Other classifier results: getBatchCorrection, getClassifications, getWeightingType

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0", targetValue=500)
results <- runClassifier('EMC92', myData)
getScores( results )
getClassifications( results )
```

getSds *Obtain classifiers' reference standard deviations.*

Description

getSds returns the reference standard deviations encoded in the classifier.

Usage

```
getSds(object)

## S4 method for signature 'ClassifierParameters'
getSds(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

Returns a numeric vector of probe set standard deviations as observed in the reference data

See Also

showClassifierList getClassifier runClassifier

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getTrainingData, getWeights

Examples

```
aClassifier <- getClassifier("EMC92")
getSds(aClassifier)
```

---

getTargetValue *Obtain the targetValue*

Description

getTargetValue returns the current applied targetValue in the MAS5.0 gene expression data.

Usage

```
getTargetValue(object)

## S4 method for signature 'FixedExpressionData'
getTargetValue(object)
```

Arguments

object	An object of class FixedExpressionData

Value

A numeric value

See Also

Other fixed data information extraction functions: [,FixedExpressionData,ANY,ANY-method, dim,FixedExpressionData-method, getNormalizationMethod

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0", targetValue=500)
getTargetValue( myData )
```

`getTrainingData` *Obtain classifier training data.*

Description

`getTrainingData` returns the training data that was used for building the classifier.

Usage

```
getTrainingData(object)

## S4 method for signature 'ClassifierParameters'
getTrainingData(object)
```

Arguments

`object`	An object of class `ClassifierParameters` as returned by `getClassifier`

Value

An object of class `ExpressionSet`

See Also

Other classifier information functions: `getCitations`, `getClassifier`, `getDecisionBoundaries`, `getDescription`, `getEventChain`, `getIntercept`, `getMeans`, `getNormalizationMethod`, `getProbeNames`, `getSds`, `getWeights`

Examples

```
aClassifier <- getClassifier("EMC92")
getTrainingData(aClassifier)
```

getWeightingType *Obtain the weighting type used to obtain a classifier result.*

Description

getWeightingType returns weigthing type

getWeightingTypes returns weigthing type

Usage

```
getWeightingType(object)

getWeightingTypes()

## S4 method for signature 'ClassifierResults'
getWeightingType(object)
```

Arguments

object	An object of class ClassifierResults as returned by runClassifier

Value

one of the values in getWeightingTypes()

either "complete" or "reweighted"

See Also

Other classifier results: getBatchCorrection, getClassifications, getScores

Other classifier results: getBatchCorrection, getClassifications, getScores

Examples

```
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('EMC92', myData)
getWeightingType( results )
getWeightingTypes()
```

getWeights *Obtain classifier weights.*

Description

getWeights returns the probe weights associated with the classifier.

Usage

```
getWeights(object)

## S4 method for signature 'ClassifierParameters'
getWeights(object)
```

Arguments

object	An object of class ClassifierParameters as returned by getClassifier

Value

A numeric vector.

See Also

Other classifier information functions: getCitations, getClassifier, getDecisionBoundaries, getDescription, getEventChain, getIntercept, getMeans, getNormalizationMethod, getProbeNames, getSds, getTrainingData

Examples

```
aClassifier <- getClassifier("EMC92")
getWeights(aClassifier)
```

runClassifier *Perform classification.*

Description

runClassifier performs classification by applying a classifier to gene expression data.

Usage

```
runClassifier(classifierParameters, fixedExpressionData, ...)


  ## S4 method for signature 'character,FixedExpressionData'
runClassifier(classifierParameters,
  fixedExpressionData, ...)


  ## S4 method for signature 'ClassifierParameters,FixedExpressionData'
runClassifier(classifierParameters,
  fixedExpressionData, ...)
```

Arguments

classifierParameters	Either a text value indicating a classifier name (see showClassifierList), or an object of class ClassifierParameters as returned by the getClassifier function.
fixedExpressionData	The data to be classified in the form of a FixedExpressionData object as returned by the setNormalizationMethod function.
...	see details

Details

A list of possible classifiers is obtained by showClassifierList. The data to be classified is first to be processed by the setNormalizationMethod function. By default the data is assumed to contain many (n>=25) samples with corresponding probe-sets needed for classification. If one of these conditions is not met, a classifier outcome might be seriously affected. By default an error is given. Although strongly discouraged, it is possible to circumvent the security checks. If not all required probe-sets are included in the input set, you can explicitly pass the parameter `allow.reweighted = TRUE` to the `runClassifier` function in order to determine the classifier outcome using less probe-sets (e.g. possible if the missing probe-sets are known to have minimal contribution).See `vignette("MissingCovariates")` for more information. If the input data has a small number of samples, the default batch correction becomes ineffective. If you are aware of the possible negative effects you can force to not use batch correction by passing the parameter `do.batchcorrection=FALSE`.

Value

The classification results as an object of class ClassifierResults.

See Also

Other workflow functions: getNormalizationMethod, setNormalizationMethod, showClassifierList

Examples

```
data(exampleMAS5)
myData<-setNormalizationMethod(exampleMAS5,"MAS5.0",targetValue=500)
runClassifier("EMC92",myData)
```

---

setNormalizationMethod *Prepare data.*

Description

setNormalizationMethod is to be called prior to running a classifier.

Usage

```
setNormalizationMethod(expressionSet, method, ...)
```

Arguments

expressionSet	An object of class ExpressionSet containing the gene expression data.
method	A character string indicating the normalization that was applied to the data. Possible values are given by getNormalizationMethods().
...	see details.

Details

The FixedExpressionData class forms together with the ClassifierParameters class the basis for input to the runClassifier function. The data inside the FixedExpressionData-class has to be stored as it is right after normalization. This function may require some additional arguments:

- isLog2Transformed = TRUE Use this argument if the data already underwent a log2transformation, as is common e.g. in case of MAS5.0 normalization.
- targetValue = value This is a MAS5.0 specific argument. It is the sample intensity mean when the lowest and highest 2% of intensities are discarded. If only part of the original expression set is given to this function, then this argument is required.

Value

An object of class FixedExpressionData

See Also

Other workflow functions: getNormalizationMethod, runClassifier, showClassifierList

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('EMC92', myData)
getScores( results )
getClassifications( results )
```

---

showClassifierList *Show classifier names and descriptions.*

Description

showClassifierList gives a data.frame of all implemented classifiers.

Usage

```
showClassifierList(normalizations)
```

Arguments

normalizations	an optional text argument of one or more normalization methods in order to filter the classifiers to be shown.

Details

The names of the classifiers shown can be used as input for the runClassifier function and the getClassifier function.

Value

A data.frame with columns: "name","normalizationMethod" and "description"

See Also

Other workflow functions: getNormalizationMethod, runClassifier, setNormalizationMethod

Examples

```
showClassifierList()
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0",targetValue=500)
results <- runClassifier('UAMS70', myData)
getScores( results )
getClassifications( results )
```

`[,FixedExpressionData,ANY,ANY-method` *Extract*

Description

Extract Parts of an Object

Usage

```
## S4 method for signature 'FixedExpressionData,ANY,ANY'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'FixedExpressionData,ANY,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'FixedExpressionData,missing,ANY'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'FixedExpressionData,ANY,missing'
x[[i, j, ...]]

## S4 method for signature 'FixedExpressionData,missing,ANY'
x[[i, j, ...]]

## S4 method for signature 'FixedExpressionData,ANY,ANY'
x[[i, j, ...]]
```

Arguments

`x`	An object of class `FixedExpressionData`
`i`	the rows index
`j`	the column index
`...`	unused
`drop`	unused

Value

An object of class `FixedExpressionData`

See Also

Other fixed data information extraction functions: `dim,FixedExpressionData-method`, `getNormalizationMethod`, `getTargetValue`

Examples

```
data(exampleMAS5)
myData <- setNormalizationMethod(exampleMAS5, "MAS5.0", targetValue=500)
dim(myData)
dim(myData[1:10,1:3])
dim(myData[[1:10,1:3]])
```

Index