Package 'MiDA'

April 18, 2019

Type Package

Title Microarray Data Analysis

Version 0.1.2

Maintainer Elena Filatova <filatova@nniiem.ru>

Description Set of functions designed to simplify transcriptome analysis and identification of marker molecules using microarrays data. The package includes a set of functions that allows performing full pipeline of analysis including data normalization, summarisation, binary classification, FDR (False Discovery Rate) multiple comparison and the definition of potential biological markers.

License GPL-3

Encoding UTF-8

LazyLoad true

Imports caret, gbm, genefilter, graphics, limma, preprocessCore, pROC, stats, SQN

Depends R (>= 3.5.0)

RoxygenNote 6.1.1

NeedsCompilation no

Author Elena Filatova [aut, cre],
Nikolai Sakharnov [dtc],
Dmitry Knyazev [dtc],
Oleg Utkin [dtc],
Blokhina Scientific Research Institute of Epidemiology and Microbiology of Nizhny Novgorod, Russia [fnd]

Repository CRAN

Date/Publication 2019-04-18 10:40:03 UTC

R topics documented:

`IMexpression` *Infectious mononucleosis transcriptome*

Description

This is compilied numeric matrix of raw microarray intensities data. Each of 100 rows corresponds to a probe (gene, transcript) and each of 89 column correspondes to a specimen (patient). Specimen is total mRNA samples from human peripheral blood leukocytes' taken from healthy children and children with infectious mononucleosis of different ethiology. Specimen features are in `IMspecimen` data.

Usage

```
data(IMexpression)
```

Format

A matrix with 100 rows and 89 variables

Details

- rownames. Probes IDs. For most transcripts there are several probes.
- colnames. Specimen IDs.

Author(s)

Nikolai A. Sakharnov, Dmitry I. Knyazev, Oleg V. Utkin

IMspecimen *Specimen features*

Description

A dataset containing information about specimens from `IMexpression` data: IDs and diagnoses.

Usage

```
data(IMspecimen)
```

Format

A data frame with 89 rows and 2 variables.

Details

- ID. Specimen ID, factor variable with 89 levels.
- diagnosis. Specimen diagnosis: "ebv" (children with acute EBV mononucleosis), "hhv6" (children with acute HHV6 mononucleosis), "norm" (healthy children). Factor variable with 3 levels.

Author(s)

Nikolai A. Sakharnov, Dmitry I. Knyazev, Oleg V. Utkin

MiBiClassGBODT *Binary classification using gradient boosting over desicion trees*

Description

This function conducts a binary classification of specimens based on microarray gene (transcript) expression data. Gradient boosting over desicion trees algorithm is used. Several generalized boosted regression models are fitted during cross-validation, for each model measurements of classification quality and feature importance are returned.

Usage

```
MiBiClassGBODT(Matrix, specimens, n.crossval = 5, ntrees = 10000,
  shrinkage = 0.1, intdepth = 2, n.terminal = 10, bag.frac = 0.5)
```

Arguments

Matrix	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
specimens	factor vector with two levels specifying specimens in the columns of the Matrix.
n.crossval	integer specifying number of cross-validation folds.
ntrees	integer specifying the total number of decision trees (boosting iterations).
shrinkage	numeric specifying the learning rate. Scales the step size in the gradient descent procedure.
intdepth	integer specifying the maximum depth of each tree.
n.terminal	integer specifying the actual minimum number of observations in the terminal nodes of the trees.
bag.frac	the fraction of the training set observations randomly selected to propose the next tree in the expansion.

Details

Matrix must contain specimens from two classification groups only. To sample expression matrix use MiDataSample.
The order of the variables in specimens and the columns of Matrix must be the same. Levels of specimens are two classification groups. To sample specimens use MiSpecimenSample.
Number of cross-validation folders defines number of models to be fitted. For example, if n.crossval=5 then all specimens are divided into 5 folders, each of them is later used for model testing, so 5 models are fitted. See createFolds for details.
While boosting, basis functions are iteratively adding in a greedy fashion so that each additional basis function further reduces the selected loss function. Gaussian distribution (squared error) is used. ntrees, shrinkage, intdepth are parameters for model tuning. bag.frac introduces randomnesses into the model fit. If bag.frac < 1 then running the same model twice will result in similar but different fits. Number of specimens in train sample must be enough to provide the minimum number of observations in terminal nodes.I.e.
(1-1/n.crossval)*bag.frac > n.terminal.
See gbm for details.

Value

list of 2:
QC - matrix containing quality measures for each fitted model and their summary. Accur - accuracy (percentage of correct predictions), AUC - area under ROC curve (see roc), MCC - Mattew's correlation coefficient
formula ((TP*TN)-(FP*FN))/sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)),
F1sc - F1 score
formula 2xPresxRec/(Pres+Rec).
If all the data points from one class are misclassified into other, MCC and F1 score may get NaN values.

Importance - list of data frames containing for each fitted model: var - probe ID and rel.inf - its feature importance for classification (relative influence).
Feature importance (relative influence) graphs are also produced.

Author(s)

Elena N. Filatova

See Also

createFolds, gbm, MiSpecimenSample, MiDataSample, roc

Examples

```
#get gene expression and specimen data
data("IMexpression");data("IMspecimen")
#sample expression matrix and specimen data for binary classification,
#only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running
BoostRes<-MiBiClassGBODT(SampleMatrix, SampleSpecimen, n.crossval = 3,
                      ntrees = 10, shrinkage = 1, intdepth = 2)
BoostRes[[1]] # QC values for n.crossval = 3 models and its summary
length(BoostRes[[2]]) # n.crossval = 3 data frames of probes feature importance for classification
head(BoostRes[[2]][[1]])
```

MiDataSample *Select matrix columns based on values of attendant vector*

Description

This function selects columns of a matrix that correspond to 1 or 2 factor levels of attendant vector.

Usage

```
MiDataSample(Matrix, specimens, group1, group2 = NULL)
```

Arguments

Matrix	matrix
specimens	factor vector with length equal to number of Matrix columns
group1	value of factor level to sample
group2	additional value of factor level to sample

Details

This function is ment for sampling specimens in gene/transcript expression matrix for binary classification in case when specimens belong to more than two groups. The aim is to create gene/transcript expression matrix that contains specimens for only 1 or 2 groups. Groups are specified in corresponding factor vector that contains specimens description. It should be used together with MiSpecimenSample that samples specimens' description in the same way.

Value

`Matrix` with rescricted number of columns that correspond to specimens from 1 or 2 groups.

Author(s)

Elena N. Filatova

See Also

`MiSpecimenSample`

Examples

```
#get gene expression and specimen data
data("IMexpression");data("IMspecimen")
dim(IMexpression) # 100 columns (genes/transcripts) - 89 specimens
colnames(IMexpression)[1:10] # look at first 10 columns of matrix - specimens IDs
IMspecimen[1:10,] # specimens IDs and group factor - diagnoses in attendant vector
# note that specimens in matrix columns are in the same order as specimens in description data
# select specimens with only EBV and NORM diagnoses (and sample the description data as well)
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis, "ebv", "norm")
SampleSamples<-MiSpecimenSample(IMspecimen$diagnosis, "ebv", "norm")
dim(SampleMatrix)# only 68 specimens with EBV and NORM diagnoses left
colnames(SampleMatrix)[1:10]
SampleSamples[1:10] # corresponding diagnoses
```

MiFracData *Subset an expression matrix based on probe's feature importance*

Description

This function reduces the number of rows (probes) in gene/transcript expression matrix, leaving only those that have the biggest feature importance for binary classification.

Usage

```
MiFracData(Matrix, importance.list, NRows)
```

Arguments

`Matrix`	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient). Probe IDs must be indicated as matrix row names.

`importance.list`
: a list of data frames, containing the result of binary classification: probe IDs in first column and probe's feature importance (relative influence) in the second column in the order from most important to the least important for classification. Such list is the `MiBiClassGBODT` output (`Importance`).

`NRows`
: integer defines how many probes are to be left in the expression matrix.

Details

Function provides gene expression matrix subsetting according to probe's feature importance for binary classification, i.e., feature selection. Feature selection provides better classification and identification of significant genes while "not important" genes are taken away from analysis. The procedure of the pairwise combinations of the feature selection and classification methods are described by Pirooznia et al (2008).
The function is able to use multiple feature importance data at a time to subset one expression matrix. If `importance.list` contains more than one data frame (i.e., the result of a binary classification for more than one model created during cross-validation), the function selects most important probes from each data frame and then removes the repeats. Thus, the output matrix may contain number of probes more than `NRows`.

Value

expression matrix with only selected probes in alphabetical order as rows and all specimens as columns.

Author(s)

Elena N. Filatova

References

Pirooznia M., Yang J.Y., Yang M.Q., Deng Y. (2008) A comparative study of different machine learning methods on microarray gene expression data. BMC Genomics 9 (Suppl1), S13. https://doi.org/10.1186/1471-2164-9-S1-S13

See Also

`MiBiClassGBODT`

Examples

```
# get gene expression and specimen data
data("IMexpression");data("IMspecimen")
#sample expression matrix and specimen data for binary classification,
#only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
dim(SampleMatrix) # 100 probes
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running
ClassRes<-MiBiClassGBODT(SampleMatrix, SampleSpecimen, n.crossval = 3,
                         ntrees = 10, shrinkage = 1, intdepth = 2)
```

```
# List of influence data frames for all 3 models build using cross-validation
# is the 2nd element of BiClassGBODT results
# take 10 most important probes from each model
Sample2Matrix<-MiFracData(SampleMatrix, importance.list = ClassRes[[2]], 10)
dim(Sample2Matrix) # less than 100 probes left
```

`MiInflCount` *Mean microarray probes' feature importance from binary classification*

Description

Counts mean of probes' feature importance for multiple models of binary classification built on microarray gene/transcript expression data

Usage

```
MiInflCount(importance.list)
```

Arguments

`importance.list`
: a list of data frames, containing the result of binary classification: probe IDs in first column and probe's feature importance (relative influence) in the second column in the order from most important to the least important for classification. Such list is the `MiBiClassGBODT` output (`Importance`).

Details

This function takes the result of binary classification performed with cross-validation and counts mean of each probe's feature importance (relative influence) gained in all fitted models.

Value

a list of 2
`data.mean` - data frame of probe names (in alphabetical order), their mean feature importance and standard deviation.
`data.importance` - data frame of probe IDs (in alphabetical order) and their original feature importance values gained in all cross-validation models.

Author(s)

Elena N. Filatova

See Also

`MiBiClassGBODT`

Examples

```
# get gene expression and specimen data
data("IMexpression");data("IMspecimen")
# sample expression matrix and specimen data for binary classification,
# only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running
ClassRes<-MiBiClassGBODT(SampleMatrix, SampleSpecimen, n.crossval = 3,
                         ntrees = 10, shrinkage = 1, intdepth = 2)
# List of influence data frames for all 3 models build using cross-validation
# is the 2nd element of BiClassGBODT results
Importances<-MiInflCount(ClassRes[[2]])
Importances[[1]][1:10,] # mean and sd. 0s are for low feature importance
Importances[[2]][1:10,] # original values for n.crossval = 3 models
```

`MiIntDepthAjust` *Ajust maximum depth parameter for fitting generalized boosted regression models*

Description

Test maximum depth parameter for microarray data binary classification using gradient boosting over desicion trees.

Usage

```
MiIntDepthAjust(Matrix, specimens, test.frac = 5, times = 5,
  ntrees = 1000, shrinkage = 0.1, intdepth = c(1:4),
  n.terminal = 10, bag.frac = 0.5)
```

Arguments

`Matrix`	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
`specimens`	factor vector with two levels specifying specimens in the columns of the `Matrix`
`test.frac`	integer specifying fraction of data to use for model testing
`times`	integer specifying number of trials
`ntrees`	integer specifying the total number of decision trees (boosting iterations).
`shrinkage`	numeric specifying the learning rate. Scales the step size in the gradient descent procedure.
`intdepth`	vector of integers specifying the maximum depth of each tree. The tested parameter.

n.terminal	integer specifying the actual minimum number of observations in the terminal nodes of the trees.
bag.frac	the fraction of the training set observations randomly selected to propose the next tree in the expansion.

Details

test.frac defines fraction of specimens that will be used for model testing. For example, if test.frac=5 then 4/5th of specimens will be used for model fitting (train data) and 1/5th of specimens will be used for model testing (test data). Specimens for test and train data will be selected by random. So with times>1, train and test data will differ each time.
While boosting basis functions are iteratively adding in a greedy fashion so that each additional basis function further reduces the selected loss function. Gaussian distribution (squared error) is used. ntrees, shrinkage, intdeep are parameters for model tuning. bag.frac introduces randomnesses into the model fit. If bag.frac < 1 then running the same model twice will result in similar but different fits. Number of specimens in train sample must be enough to provide the minimum number of observations in terminal nodes.I.e.
(1-1/test.frac)*bag.frac > n.terminal.
See gbm for details.
Use MiNTreesAjust and MiShrinkAjust for ajusting other parameters.
Function is rather time-costing. If specimens are not equally distributed between two classified groups, NA may be produced.

Value

list of 2
train.accuracy - a data frame of train data classification accuracy for each intdepth value in each trial and their median.
test.accuracy - a data frame of test data classification accuracy for each intdepth value in each trial and their median.
Also a plot for intdepth versus Accuracy is produced.

Author(s)

Elena N. Filatova

See Also

gbm, MiNTreesAjust, MiShrinkAjust

Examples

```
#get gene expression and specimen data
data("IMexpression");data("IMspecimen")
#sample expression matrix and specimen data for binary classification,
#only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running. Test intdepth
set.seed(1)
```

```
ClassRes<-MiIntDepthAjust(SampleMatrix, SampleSpecimen, test.frac = 5, times=3,
                          ntrees = 10, shrinkage = 1, intdepth =  c(1,2))
ClassRes[[1]] # train accuracy
ClassRes[[2]] # test accuracy
```

MiNorm *Microarray data normalization*

Description

Normalizes microarray expression intensities using different methods with or without background correction.

Usage

```
MiNorm(Matrix, posNC, method = "none", leaveNC = TRUE, BGcor = FALSE)
```

Arguments

Matrix	numeric matrix of intensities data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
posNC	numeric vector specifying numbers of rows containing negative controls (non-coding areas). Used for method="SQN" only. Rows with negative controls will be removed from an intensity matrix after the normalization if leaveNC=FALSE.
method	character string specifying normalization method. Possible values are: "none" (no normalization) "center" (subtracting the row mean), "scale" (dividing by row standard deviation), "standardize" (subtracting the row mean and dividing by row standard deviation - z-score transformation), "range" (ranges from 0 to 1), "QN" (normalization based upon quantiles), "SQN" (subset quantile normalization using negative control features), "Loess" (cyclicly applying loess normalization).
leaveNC	logical value indicating whether rows with negative control should be deleted from intensity matrix after normalization.
BGcor	logical value indicating whether background correction should be done before normalization. Could be used for background correction only (without data normalization) if method="none".

Details

This function is intended to normalize microarray intensities data between arrays. Background correction is optional.
Background correction method is "normexp", which is based on a convolution model (Ritchie, 2007). See backgroundCorrect for details.
Quantile normalization method implies that we can give each array the same distribution See normalize.quantiles for details.

Subset quantile normalization is performed based on a subset of negative (or non-coding) controls according to (Wu and Aryee, 2010). Number of normal distributions in the mixture approximation is 5, weight given to the parametric normal mixture model is 0.9. See SQN for details.
Cyclic loess normalization implements method of Ballman et al (2004), whereby each array is normalized to the average of all the arrays. See normalizeCyclicLoess for details.

Value

A matrix of the same dimensions as Matrix containing normalized values with or without background correction. If leaveNC=FALSE the function returns a matrix with normalized values without rows containing negative controls.

Author(s)

Elena N. Filatova

References

Ballman K.V., Grill D.E., Oberg A.L. and Therneau T.M. (2004). Faster cyclic loess: normalizing RNA arrays via linear models. Bioinformatics 20, 2778-2786. https://doi.org/10.1093/bioinformatics/bth327

Bolstad B.M., Irizarry R.A., Astrand M. and Speed T.P. (2003) A Comparison of Normalization Methods for High Density Oligonucleotide Array Data Based on Bias and Variance. Bioinformatics 19(2), 185-193. https://doi.org/10.1093/bioinformatics/19.2.185

Ritchie M.E., Silver J., Oshlack A., Silver J., Holmes M., Diyagama D., Holloway A. and Smyth G.K. (2007). A comparison of background correction methods for two-colour microarrays. Bioinformatics 23, 2700-2707. https://doi.org/10.1093/bioinformatics/btm412

Wu Z and Aryee M. (2010). Subset Quantile Normalization using Negative Control Features. Journal of Computational Biology 17(10), 1385-1395. https://doi.org/10.1089/cmb.2010.0049

See Also

backgroundCorrect, normalizeCyclicLoess, normalize.quantiles, SQN

Examples

```
data("IMexpression")
# Loess normalization
LoMatrix<-MiNorm(IMexpression, method="Loess")
par(mfrow=c(1,2))
boxplot(log2(IMexpression),main="Before normalization")
boxplot(log2(LoMatrix),main="Loess normalization")
par(mfrow=c(1,1))
```

MiNTreesAjust *Ajust number of trees parameter for fitting generalized boosted regression models*

Description

Test total number of trees parameter for microarray data binary classification using gradient boosting over desicion trees.

Usage

```
MiNTreesAjust(Matrix, specimens, test.frac = 5, times = 5,
  ntrees = c(100, 500, 1000, 5000, 10000), shrinkage = 0.1,
  intdepth = 2, n.terminal = 10, bag.frac = 0.5)
```

Arguments

Matrix	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
specimens	factor vector with two levels specifying specimens in the columns of the Matrix
test.frac	integer specifying fraction of data to use for model testing
times	integer specifying number of trials
ntrees	vector of integer specifying the total number of decision trees (boosting iterations).The tested parameter.
shrinkage	numeric specifying the learning rate. Scales the step size in the gradient descent procedure.
intdepth	integer specifying the maximum depth of each tree.
n.terminal	integer specifying the actual minimum number of observations in the terminal nodes of the trees.
bag.frac	the fraction of the training set observations randomly selected to propose the next tree in the expansion.

Details

test.frac defines fraction of specimens that will be used for model testing. For example, if test.frac=5 then 4/5th of specimens will be used for model fitting (train data) and 1/5th of specimens will be used for model testing (test data). Specimens for test and train data will be selected by random. So with times>1, train and test data will differ each time.
While boosting basis functions are iteratively adding in a greedy fashion so that each additional basis function further reduces the selected loss function. Gaussian distribution (squared error) is used. ntrees, shrinkage, intdeep are parameters for model tuning. bag.frac introduces randomnesses into the model fit. If bag.frac < 1 then running the same model twice will result in similar but different fits. Number of specimens in train sample must be enough to provide the minimum number of observations in terminal nodes.I.e.
(1-1/test.frac)*bag.frac > n.terminal.

See gbm for details.
Use MiIntDepthAjust and MiShrinkAjust for ajusting other parameters.
Function is rather time-costing. If specimens are not equally distributed between two classified groups, NA may be produced.

Value

list of 2
train.accuracy - a data frame of train data classification accuracy for each ntrees value in each trial and their median.
test.accuracy - a data frame of test data classification accuracy for each ntrees value in each trial and their median.
Also a plot for ntrees versus Accuracy is produced.

Author(s)

Elena N. Filatova

See Also

gbm, MiIntDepthAjust, MiShrinkAjust

Examples

```
#get gene expression and specimen data
data("IMexpression");data("IMspecimen")
#sample expression matrix and specimen data for binary classification,
#only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running. Test ntrees
set.seed(1)
ClassRes<-MiNTreesAjust(SampleMatrix, SampleSpecimen, test.frac = 5, times = 3,
                        ntrees = c(10, 20), shrinkage = 1, intdepth = 2)
ClassRes[[1]] # train accuracy
ClassRes[[2]] # test accuracy
```

MiSelectSignif *Select biological markers with high fold change and classification importance*

Description

Choose probes which change is biologically significant based on binary classification feature importance, gene expression fold change and statistical significance.

Usage

```
MiSelectSignif(probes, mean1, mean2, FC.method, infl, stat.val,
  tresh.FC = 0.75, tresh.infl = 0.75, tresh.stat = 0.05)
```

Arguments

probes	character vector of probe (gene, transcript) names.
mean1	numeric vector of mean values for probes expression in the first group of comparison.
mean2	numeric vector of mean values for probes expression in the second group of comparison.
FC.method	character specifying the method of fold change counting. Possible values are: "absolute" (mean1-mean2), "percent" ((mean1*100/mean2)-100), "ratio" (mean1/mean2), "Log2.ratio" (log2(mean1/mean2)).
infl	numeric vector of mean values for probes feature importance (relative influence) from binary classification.
stat.val	numeric vector of statistical significance (p-value, q-value) for testing differences of mean1 and mean2.
tresh.FC	numeric from o to 1 specifying the tresh hold for fold change FC parameters (quantile).The significant fold change is bigger than tresh.FC.
tresh.infl	numeric from o to 1 specifying the tresh hold for feature importance infl parameters (quantile).The significant feature importance is bigger than tresh.infl.
tresh.stat	numeric from o to 1 specifying the tresh hold for statistical significance stat.val.The significant fold change is lesser than tresh.stat.

Details

The order must be the same for all parameters.
This function marks as "markers" probes that statistically significant change their expression in two groups of comparison with high (over tresh hold) fold change and feature importance from binary classification.

Value

data frame of probe names, their fold change values, statistical significance values, feature inportance values and marker values.

Author(s)

Elena N. Filatova

Examples

```
probes<-paste("probe", 1:50, sep="") #probes
mean1<-rnorm(50, mean=0, sd=1) #means
mean2<-rnorm(50, mean=5, sd=1)
infl<-c(1:50) # influence
```

```
stat.val<-rep(c(0.05, 0.04), c(20, 30))
Result<-MiSelectSignif(probes, mean1, mean2, FC.method="absolute", infl, stat.val,
                       tresh.FC=0.75, tresh.infl=0.75, tresh.stat=0.05)
Result[1:5,]
```

`MiShrinkAjust` *Ajust learning rate parameter for fitting generalized boosted regression modelsfor fitting generalized boosted regression models*

Description

Test learning rate (shrinkage) parameter for microarrat data binary classification using gradient boosting over desicion trees.

Usage

```
MiShrinkAjust(Matrix, specimens, test.frac = 5, times = 5,
  ntrees = 1000, shrinkage = c(0.001, 0.01, 0.1), intdepth = 2,
  n.terminal = 10, bag.frac = 0.5)
```

Arguments

`Matrix`	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
`specimens`	factor vector with two levels specifying specimens in the columns of the `Matrix`
`test.frac`	integer specifying fraction of data to use for model testing
`times`	integer specifying number of trials
`ntrees`	integer specifying the total number of decision trees (boosting iterations).
`shrinkage`	numeric vector specifying the learning rate. Scales the step size in the gradient descent procedure. The tested parameter.
`intdepth`	vinteger specifying the maximum depth of each tree.
`n.terminal`	integer specifying the actual minimum number of observations in the terminal nodes of the trees.
`bag.frac`	the fraction of the training set observations randomly selected to propose the next tree in the expansion.

Details

`test.frac` defines fraction of specimens that will be used for model testing. For example, if `test.frac`=5 then 4/5th of specimens will be used for model fitting (train data) and 1/5th of specimens will be used for model testing (test data). Specimens for test and train data will be selected by random. So with `times`>1, train and test data will differ each time.
While boosting basis functions are iteratively adding in a greedy fashion so that each additional basis function further reduces the selected loss function. Gaussian distribution (squared error) is used.

ntrees, shrinkage, intdeep are parameters for model tuning. bag.frac introduces randomnesses into the model fit. If bag.frac < 1 then running the same model twice will result in similar but different fits. Number of specimens in train sample must be enough to provide the minimum number of observations in terminal nodes.I.e.
(1-1/test.frac)*bag.frac > n.terminal.
See gbm for details.
Use MiIntDepthAjust and MiNTreesAjust for ajusting other parameters.
Function is rather time-costing. If specimens are not equally distributed between two classified groups, NA may be produced.

Value

list of 2
train.accuracy - a data frame of train data classification accuracy for each shrinkage value in each trial and their median.
test.accuracy - a data frame of test data classification accuracy for each shrinkage value in each trial and their median.
Also a plot for shrinkage versus Accuracy is produced.

Author(s)

Elena N. Filatova

See Also

gbm, MiIntDepthAjust, MiNTreesAjust

Examples

```
data("IMexpression");data("IMspecimen")
#sample expression matrix and specimen data for binary classification,
#only "NORM" and "EBV" specimens are left
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis,"norm", "ebv")
SampleSpecimen<-MiSpecimenSample(IMspecimen$diagnosis, "norm", "ebv")
#Fitting, low tuning for faster running. Test shrinkage
set.seed(1)
ClassRes<-MiShrinkAjust(SampleMatrix, SampleSpecimen, test.frac = 5, times = 3,
                        ntrees = 10, shrinkage = c(0.1, 1), intdepth = 2)
ClassRes[[1]] # train accuracy
ClassRes[[2]] # test accuracy
```

MiSpecimenSample *Select values from factor vector*

Description

This function takes factor vector with multiple levels and selects values for 1 or 2 levels only.

Usage

```
MiSpecimenSample(x, group1, group2 = NULL)
```

Arguments

x	factor vector
group1	value of factor level to sample
group2	additional value of factor level to sample

Details

This function is ment for sampling specimens for binary classification when they belong to more than two groups. The aim is to create factor vector with only two levels specifying specimens in the columns of corresponding microarray expression matrix. It should be used together with MiDataSample that samples columns of microarray expression matrix in the same way.

Value

factor vector with values for 1 or 2 levels only

Author(s)

Elena N. Filatova

See Also

MiDataSample

Examples

```
#get gene expression and specimen data
data("IMexpression");data("IMspecimen")
dim(IMexpression) # 100 columns (genes/transcripts) - 89 specimens
colnames(IMexpression)[1:10] # look at first 10 columns of matrix - specimens IDs
IMspecimen[1:10,] # specimens IDs and group factor - diagnoses in attendant vector
# note that specimens in matrix columns are in the same order as specimens in description data
# select specimens with only EBV and NORM diagnoses (and sample the description data as well)
SampleMatrix<-MiDataSample(IMexpression, IMspecimen$diagnosis, "ebv", "norm")
SampleSamples<-MiSpecimenSample(IMspecimen$diagnosis, "ebv", "norm")
dim(SampleMatrix)# only 68 specimens with EBV and NORM diagnoses left
colnames(SampleMatrix)[1:10]
SampleSamples[1:10] # corresponding diagnoses
```

MiStatCount *FDR for microarray gene expression data*

Description

Performs descriptive statistics and FDR (False Discovery Rate) test for microarray expression matrix

Usage

```
MiStatCount(Matrix, specimens)
```

Arguments

Matrix	numeric matrix of expression data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient).
specimens	factor vector with two levels specifying specimens in the columns of the Matrix.

Details

This function takes matrix of expression data and performs T-test with FDR correction for two groups for each probe.
T-test is a two-sided, two-class with equal variances against the null hypothesis 'mean1=mean2' for each row. See rowttests for details.

Value

a data frame containing for each probe: mean and sd values for both groups, difference of means, p-value for T-test and q-value for FDR (False Discovery Rate) correction.

Author(s)

Elena N. Filatova

References

Welch B.L.(1951) On the comparison of several mean values: an alternative approach. Biometrika 38, 330-336. https://doi.org/10.1093/biomet/38.3-4.330

See Also

rowttests

Examples

```
data("IMexpression"); data("IMspecimen") # load data and specimen information
#sampling data and specimen information
ExpData<-MiDataSample(IMexpression, IMspecimen$diagnosis,"ebv", "norm")
Specimens<-MiSpecimenSample(IMspecimen$diagnosis, "ebv", "norm")
#Counting statistics
StatRes<-MiStatCount(ExpData, Specimens)
head(StatRes)
```

`MiSummarize` *Microarray data summarization*

Description

Counts median of intensities for multiple probes that target one gene/transcript.

Usage

```
MiSummarize(Matrix, sep, method = "median")
```

Arguments

`Matrix`	numeric matrix of intensities data where each row corresponds to a probe (gene, transcript), and each column correspondes to a specimen (patient). Row names of `Matrix` should contain probe IDs that consist of three terms: gene name - transcript name - probe name.
`sep`	a character string to separate the terms in probe IDs.
`method`	character string specifying summarization method. Possible values are "median" and "mean".

Details

This function is used for summarizing expression intensities data when multiple probes target one gene/transcript. Row names of `Matrix` should contain probe IDs that consist of 3 terms: "Gene name - sep - transcript name - sep - probe name" (for example, "AGTR2.ALL" - for gene, only one probe; "AGTR2.NM_000686.z1" - 1st probe to AGTR2 NM_000686 mRNA transcript).

Value

gene/transcript expression matrix with median/mean of expression intensities for each gene/transcript.

Author(s)

Elena N. Filatova

Examples

```
data("IMexpression") # load data
# See 5 zonds to AGTR2.NM_000686
IMexpression [1:10, 1:5]
SumMatrix<-MiSummarize(IMexpression, sep=".")
# now there is median expression for AGTR2.NM_000686
SumMatrix[ 1:10, 1:5]
```

Index