Package ‘genefu’

November 21, 2016

Type Package

Title Computation of Gene Expression-Based Signatures in Breast Cancer

Version 2.6.0

Date 2015-10-28

Description Description: This package contains functions implementing various tasks usually required by gene expression analysis, especially in breast cancer studies: gene mapping between different microarray platforms, identification of molecular subtypes, implementation of published gene signatures, gene selection, and survival analysis.

Author Deena M.A. Gendoo, Natchar Ratanasirigulchai, Markus S. Schroder, Laia Pare, Joel S. Parker, Aleix Prat, and Benjamin Haibe-Kains

Maintainer Benjamin Haibe-Kains <benjamin.haibe.kains@utoronto.ca>, Markus Schroeder <markus.schroeder@ucdconnect.ie>

biocViews DifferentialExpression, GeneExpression, Visualization, Clustering, Classification

Depends survcomp, mclust, limma,biomaRt, iC10, AIMS, R (>= 2.10)

Suggests GeneMeta, breastCancerVDX, breastCancerMAINZ, breastCancerTRANSBIG, breastCancerUPP, breastCancerUNT, breastCancerNKI, rmeta, Biobase, xtable, knitr, caret, survival

Imports amap

VignetteBuilder knitr

License Artistic-2.0

URL http://www.pmgenomics.ca/bhklab/software/genefu

LazyData yes

NeedsCompilation no

R topics documented:

  genefu-package .................................................. 3
  bimod .............................................................. 4
  boxplotplus2 ...................................................... 6
  claudinLow ........................................................ 7
  claudinLowData .................................................. 8
  collapseIDs ....................................................... 9
topics documented:

- compare.proto.cor .................................................. 9
- compute.pairw.cor.meta .......................................... 10
- compute.pairw.cor.z ............................................... 12
- compute.proto.cormeta ........................................... 12
- cordiff.dep .......................................................... 14
- endoPredict .......................................................... 15
- expos ................................................................. 16
- fuzzy.ttest ........................................................... 17
- gene70 ................................................................. 18
- gene76 ................................................................. 20
- geneid.map ............................................................ 21
- genius ................................................................. 22
- ggi .................................................................... 23
- ihc4 ................................................................. 24
- intrinsic.cluster .................................................... 26
- intrinsic.cluster.predict ......................................... 28
- map.datasets ........................................................ 29
- medianCtr ............................................................ 31
- mod1 ................................................................. 31
- mod2 ................................................................. 32
- modelOvcAngiogenic ............................................... 33
- molecular.subtyping ............................................... 33
- nkis ................................................................. 36
- npi ................................................................. 37
- oncotypedx .......................................................... 38
- ovcAngiogenic ...................................................... 39
- ovcCrijns ............................................................ 40
- ovcTCGA ............................................................ 42
- ovcYoshihara ........................................................ 43
- overlapSets .......................................................... 44
- pam50 ............................................................... 45
- pik3cags ............................................................. 46
- power.cor ............................................................ 47
- ps.cluster ............................................................ 48
- read.m.file .......................................................... 49
- readArray ........................................................... 50
- rename.duplicate .................................................... 50
- rescale .............................................................. 51
- rorS ................................................................. 52
- scmgene.robust ...................................................... 53
- scmod1.robust ....................................................... 54
- scmod2.robust ....................................................... 55
- setcolclass.df ....................................................... 56
- sig.endoPredict ..................................................... 56
- sig.gene70 ........................................................... 57
- sig.gene76 ........................................................... 58
- sig.genius ............................................................ 58
- sig.ggi ............................................................... 59
- sig.oncotypedx ....................................................... 60
- sig.pik3cags ........................................................ 60
- sig.score ........................................................... 61
- sig.tamr13 .......................................................... 62
Description

This package contains functions implementing various tasks usually required by gene expression analysis, especially in breast cancer studies: gene mapping between different microarray platforms, identification of molecular subtypes, implementation of published gene signatures, gene selection, survival analysis, ...

Details

Package: genefu
Type: Package
Version: 2.6.0
Date: 2015-10-28
License: Artistic-2.0

Author(s)

Benjamin Haibe-Kains
- Bioinformatics and Computational Genomics Laboratory, Princess Margaret Cancer Center, University Health Network, Toronto, Ontario, Canada
http://www.pmgenomics.ca/bhklab/
Former labs:
- Computational Biology and Functional Genomics, Dana-Farber Cancer Institute, Boston, MA, USA
  http://compbio.dfci.harvard.edu
- Center for Cancer Computational Biology, Dana-Farber Cancer Institute, Boston, MA, USA
  http://cccdb.dfci.harvard.edu/home
- Machine Learning Group (MLG), Universite Libre de Bruxelles, Bruxelles, Belgium
  http://www.ulb.ac.be/di/mlg/
- Breast Cancer Translational Laboratory (BCTL), Institut Jules Bordet, Bruxelles, Belgium

Maintainer: Benjamin Haibe-Kains
<benjamin.haibe.kains@utoronto.ca>

Markus Schroeder
<markus.schroeder@ucdconnect.ie>

See Also
survcomp

**bimod**

*Function to identify bimodality for gene expression or signature score*

**Description**

This function fits a mixture of two Gaussians to identify bimodality. Useful to identify ER of HER2 status of breast tumors using ESR1 and ERBB2 expressions respectively.

**Usage**

```r
bimod(x, data, annot, do.mapping = FALSE, mapping, model = c("E", "V"),
      do.scale = TRUE, verbose = FALSE, ...)
```

**Arguments**

- `x`: Matrix containing the gene(s) in the gene list in rows and at least three columns: "probe", "EntrezGene.ID" and "coefficient" standing for the name of the probe, the NCBI Entrez Gene id and the coefficient giving the direction and the strength of the association of each gene in the gene list.
- `data`: Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
- `annot`: Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
- `do.mapping`: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping`: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
model Model name used in Mclust.

do.scale TRUE if the gene expressions or signature scores must be rescaled (see rescale), FALSE otherwise.

verbose TRUE to print informative messages, FALSE otherwise.

... Additional parameters to pass to sig.score.

Value

status Status being 0 or 1.

status1.proba Probability p to be of status 1, the probability to be of status 0 being 1-p.

gaussians Matrix of parameters fitted in the mixture of two Gaussians. Matrix of NA values if EM algorithm did not converge.

BIC Values (gene expressions or signature scores) used to identify bimodality.

BI Bimodality Index (BI) as defined by Wang et al., 2009.

x Values (gene expressions or signature scores) used to identify bimodality.

Author(s)

Benjamin Haibe-Kains

References


See Also

Mclust

Examples

## load NKI data
data(nkis)
## load gene modules from Desmedt et al. 2008
data(mod1)
## retrieve esr1 affy probe and Entrez Gene id
esr1 <- mod1$ESR1[, , drop=FALSE]
## computation of signature scores
esr1.bimod <- bimod(x=esr1, data=data.nkis, annot=annot.nkis, do.mapping=TRUE, model="V", verbose=TRUE)
table("ER.IHC"=demo.nkis[, "er"], "ER.GE"=esr1.bimod$status)
**boxplotplus2**

*Box plot of group of values with corresponding jittered points*

**Description**

This function allows for display a boxplot with jittered points.

**Usage**

```r
boxplotplus2(x, .jit = 0.25, .las = 1, .ylim, box.col = "lightgrey",
   pt.col = "blue", pt.cex = 0.5, pt.pch = 16, med.line = FALSE,
   med.col = "goldenrod", ...)
```

**Arguments**

- `x`: x could be a list of group values or a matrix (each group is a row).
- `jit`: Amount of jittering noise.
- `las`: Numeric in 0,1,2,3; the style of axis labels.
- `ylim`: Range for y axis.
- `box.col`: Color for boxes.
- `pt.col`: Color for groups (jittered points).
- `pt.cex`: A numerical value giving the amount by which plotting jittered points should be magnified relative to the default.
- `pt.pch`: Either an integer specifying a symbol or a single character to be used as the default in plotting jittered points. See `points` for possible values and their interpretation.
- `med.line`: TRUE if a line should link the median of each group, FALSE otherwise.
- `med.col`: Color of `med.line`.
- `...`: Additional parameters for `boxplot` function.

**Value**

Number of samples in each group.

**Note**

2.21.2006 - Christos Hatzis, Nuvera Biosciences

**Author(s)**

Christos Hatzis

**See Also**

`boxplot`, `jitter`
Examples

```r
dd <- list("G1"=runif(20), "G2"=rexp(30) * -1.1, "G3"=rnorm(15) * 1.3)
boxplotplus2(x=dd, .las=3, .jit=0.75, .ylim=c(-3,3), pt.cex=0.75,
pt.col=c(rep("darkred", 20), rep("darkgreen", 30), rep("darkblue", 15)),
pt.pch=c(0, 9, 17))
```

Description

Subtyping method for identifying Claudin-Low Breast Cancer Samples. Code generously provided by Aleix Prat.

Usage

```r
claudinLow(x,classes="",y,nGenes="",priors="equal",std=F,distm="euclidean",centroids=F)
```

Arguments

- `x` the data matrix of training samples, or pre-calculated centroids
- `classes` a list labels for use in coloring the points
- `y` the data matrix of test samples
- `nGenes` the number of genes selected when training the model
- `priors` 'equal' assumes equal class priors, 'class' calculates them based on proportion in the data
- `std` when true, the training and testing samples are standardized to mean=0 and var=1
- `distm` the distance metric for determining the nearest centroid, can be one of euclidean, pearson, or spearman
- `centroids` when true, it is assumed that x consists of pre-calculated centroids

References


See Also

`medianCtr`, `claudinLowData`
Examples

data(claudinLowData)

# Training Set
train<-claudinLowData
train$xd<- medianCtr(train$xd)
# Testing Set
test<-claudinLowData
test$xd<- medianCtr(test$xd)

# Generate Predictions
predout<-claudinLow(train$xd, as.matrix(train$classes$Group,ncol=1), test$xd)

# Obtain results
results <- cbind(predout$predictions, predout$distances)
# write.table(results,"T.E.9CELL.LINE_results.txt",sep="\t",col=T, row=F)
**collapseIDs**  
*Utility function to collapse IDs*

**Description**  
Utility function called within the claudinLow classifier

**Usage**

```r
collapseIDs<-x,allids=row.names(x),method="mean"
```

**Arguments**

- `x` Matrix of numbers  
- `allids` Defaults to rownames of matrix  
- `method` Default method is "mean"

**References**

`citation("claudinLow")`

**See Also**

`claudinLow`

---

**compare.proto.cor**  
*Function to statistically compare correlation to prototypes*

**Description**

This function performs a statistical comparison of the correlation coefficients as computed between each probe and prototype.

**Usage**

```r
compare.proto.cor(gene.cor, proto.cor, nn,
                 p.adjust.m = c("none", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr"))
```

**Arguments**

- `gene.cor` Correlation coefficients between the probes and each of the prototypes.  
- `proto.cor` Pairwise correlation coefficients of the prototypes.  
- `nn` Number of samples used to compute the correlation coefficients between the probes and each of the prototypes.  
- `p.adjust.m` Correction method as defined in `p.adjust`.  

Value

Data frame with probes in rows and with three columns: "proto" is the prototype to which the probe is the most correlated, "cor" is the actual correlation, and "signif" is the (corrected) p-value for the superiority of the correlation to this prototype compared to the second highest correlation.

Author(s)

Benjamin Haibe-Kains

See Also

compute.proto.cor.meta, compute.pairw.cor.meta

Examples

```r
## load VDX dataset
data(vdxs)
## load NKI dataset
data(nkis)
## reduce datasets
ginter <- ginter[!is.na(ginter)][1:30]
myx <- unique(c(match(ginter, annot.vdxs[ ,"EntrezGene.ID"]),
              sample(x=1:nrow(annot.vdxs), size=20)))
data2.vdxs <- data.vdxs[, , myx]
annot2.vdxs <- annot.vdxs[, , myx]
myx <- unique(c(match(ginter, annot.nkis[ ,"EntrezGene.ID"]),
              sample(x=1:nrow(annot.nkis), size=20)))
data2.nkis <- data.nkis[, , myx]
annot2.nkis <- annot.nkis[, , myx]
## mapping of datasets
datas <- list("VDX"=data2.vdxs,"NKI"=data2.nkis)
annots <- list("VDX"=annot2.vdxs, "NKI"=annot2.nkis)
datas.mapped <- map.datasets(datas=datas, annots=annots, do.mapping=TRUE)
## define some prototypes
protos <- paste("geneid", ginter[1:3], sep=".")
## compute meta-estimate of correlation coefficients to the three prototype genes
probecor <- compute.proto.cor.meta(datas=datas.mapped$datas, proto=protos,
method="pearson")
## compute meta-estimate of pairwise correlation coefficients between prototypes
protocor <- compute.pairw.cor.meta(datas=datas.mapped$datas, proto=protos,
method="pearson")
## compare correlation coefficients to each prototype
res <- compare.proto.cor(gene.cor=probecor$cor, proto.cor=protocor$cor,
nn=probecor$cor.n, p.adjust.m="fdr")
head(res)
```
**compute.pairw.cor.meta**

**Description**

This function computes meta-estimate of pairwise correlation coefficients for a set of genes from a list of gene expression datasets.

**Usage**

```r
compute.pairw.cor.meta(datas, method = c("pearson", "spearman"))
```

**Arguments**

- `datas`: List of datasets. Each dataset is a matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined. All the datasets must have the same probes.
- `method`: Estimator for correlation coefficient, can be either `pearson` or `spearman`.

**Value**

- `cor`: Matrix of meta-estimate of correlation coefficients with probes in rows and prototypes in columns.
- `cor.n`: Number of samples used to compute meta-estimate of correlation coefficients.

**Author(s)**

Benjamin Haibe-Kains

**See Also**

`map.datasets`, `compute.proto.cor.meta`

**Examples**

```r
## load VDX dataset
data(vdxs)
## load NKI dataset
data(nkis)
## reduce datasets
ginter <- intersect(annot.vdxs[,"EntrezGene.ID"], annot.nkis[,"EntrezGene.ID"])
ginter <- ginter[!is.na(ginter)][1:30]
myx <- unique(c(match(ginter, annot.vdxs[,"EntrezGene.ID"]),
sample(x=1:nrow(annot.vdxs), size=20))))
data2.vdxs <- data.vdxs[,myx]
annot2.vdxs <- annot.vdxs[myx,]
myx <- unique(c(match(ginter, annot.nkis[,"EntrezGene.ID"]),
sample(x=1:nrow(annot.nkis), size=20))))
data2.nkis <- data.nkis[,myx]
annot2.nkis <- annot.nkis[myx,]
## mapping of datasets
datas <- list("VDX"=data2.vdxs,"NKI"=data2.nkis)
annots <- list("VDX"=annot2.vdxs, "NKI"=annot2.nkis)
datas.mapped <- map.datasets(datas=datas, annots=annots, do.mapping=TRUE)
## compute meta-estimate of pairwise correlation coefficients
pairwcor <- compute.pairw.cor.meta(datas=datas.mapped$datas, method="pearson")
str(pairwcor)
```
compute.pairw.cor.z  
*Function to compute the Z transformation of the pairwise correlations for a list of datasets*

**Description**

This function computes the Z transformation of the meta-estimate of pairwise correlation coefficients for a set of genes from a list of gene expression datasets.

**Usage**

```r
compute.pairw.cor.z(datas, method = c("pearson"))
```

**Arguments**

- `datas` List of datasets. Each dataset is a matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined. All the datasets must have the same probes.
- `method` Estimator for correlation coefficient, can be either `pearson` or `spearman`.

**Value**

- `z` Z transformation of the meta-estimate of correlation coefficients.
- `se` Standard error of the Z transformation of the meta-estimate of correlation coefficients.
- `nn` Number of samples used to compute the meta-estimate of correlation coefficients.

**Author(s)**

Benjamin Haibe-Kains

**See Also**

`map.datasets`, `compute.pairw.cor.meta`, `compute.proto.cor.meta`

---

compute.proto.cor.meta  
*Function to compute correlations to prototypes in a meta-analytical framework*

**Description**

This function computes meta-estimate of correlation coefficients between a set of genes and a set of prototypes from a list of gene expression datasets.

**Usage**

```r
compute.proto.cor.meta(datas, proto, method = c("pearson", "spearman"))
```
compute.proto.cor.meta

Arguments

datas  List of datasets. Each dataset is a matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined. All the datasets must have the same probes.

proto  Names of prototypes (e.g. their EntrezGene ID).

method  Estimator for correlation coefficient, can be either pearson or spearman.

Value

cor  Matrix of meta-estimate of correlation coefficients with probes in rows and prototypes in columns.

cor.n  Number of samples used to compute meta-estimate of correlation coefficients.

Author(s)

Benjamin Haibe-Kains

See Also

map.datasets

Examples

## load VDX dataset
data(vdxs)
## load NKI dataset
data(nkis)
## reduce datasets
ginter <- intersect(annot.vdxs[, "EntrezGene.ID"], annot.nkis[, "EntrezGene.ID"])
ginter <- ginter[!is.na(ginter)][1:30]
myx <- unique(c(match(ginter, annot.vdxs[, "EntrezGene.ID"]),
              sample(x=1:nrow(annot.vdxs), size=20)))
data2.vdxs <- data.vdxs[, myx]
annot2.vdxs <- annot.vdxs[, myx]
data2.nkis <- data.nkis[, myx]
annot2.nkis <- annot.nkis[, myx]
## mapping of datasets
datas <- list("VDX"=data2.vdxs, "NKI"=data2.nkis)
annots <- list("VDX"=annot2.vdxs, "NKI"=annot2.nkis)
datas.mapped <- map.datasets(datas=datas, annots=annots, do.mapping=TRUE)
## define some prototypes
protos <- paste("geneid", ginter[1:3], sep="."")
## compute meta-estimate of correlation coefficients to the three prototype genes
probecor <- compute.proto.cor.meta(datas=datas.mapped$datas, proto=protos,
                      method="pearson")
str(probecor)
**Function to estimate whether two dependent correlations differ**

**Description**
This function tests for statistical differences between two dependent correlations using the formula provided on page 56 of Cohen & Cohen (1983). The function returns a t-value, the DF and the p-value.

**Usage**
```r
cordiff.dep(r.x1y, r.x2y, r.x1x2, n,
  alternative = c("two.sided", "less", "greater"))
```

**Arguments**
- `r.x1y` The correlation between x1 and y where y is typically your outcome variable.
- `r.x2y` The correlation between x2 and y where y is typically your outcome variable.
- `r.x1x2` The correlation between x1 and x2 (the correlation between your two predictors).
- `n` The sample size.
- `alternative` A character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

**Details**
This function is inspired from the `cordif.dep`.

**Value**
Vector of three values: t statistics, degree of freedom, and p-value.

**Author(s)**
Benjamin Haibe-Kains

**References**

**See Also**
cor, t.test, compare.proto.cor
Examples

```r
## load VDX dataset
data(vdxs)
## retrieve ESR1, AURKA and MKI67 gene expressions
x1 <- data.vdxs[, "208079_s_at"]
x2 <- data.vdxs[, "205225_at"]
y <- data.vdxs[, "212022_s_at"]
## is MKI67 significantly more correlated to AURKA than ESR1?
cc.ix <- complete.cases(x1, x2, y)
cordiff.dep(r.x1y=abs(cor(x=x1[cc.ix], y=y[cc.ix], use="everything", method="pearson")), r.x2y=abs(cor(x=x2[cc.ix], y=y[cc.ix], use="everything", method="pearson")), r.x1x2=abs(cor(x=x1[cc.ix], y=x2[cc.ix], use="everything", method="pearson")), n=sum(cc.ix), alternative="greater")
```

Description

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the endoPredict signature as published by Filipits et al 2011.

Usage

```r
endoPredict(data, annot, do.mapping = FALSE, mapping, verbose = FALSE)
```

Arguments

- **data**: Matrix of gene expressions with samples in rows and probes in columns, dim-names being properly defined.
- **annot**: Matrix of annotations with at least one column named "EntrezGene.ID", dim-names being properly defined.
- **do.mapping**: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise. Note that for Affymetrix HGU datasets, the mapping is not necessary.
- **mapping**: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- **verbose**: TRUE to print informative messages, FALSE otherwise.

Details

The function works best if data have been noralized with MAS5. Note that for Affymetrix HGU datasets, the mapping is not necessary.

Value

- **score**: Continuous signature scores
- **risk**: Binary risk classification, 1 being high risk and 0 being low risk.
- **mapping**: Mapping used if necessary.
- **probe**: If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.
Author(s)
Benjamin Haibe-Kains

References

Examples
```r
## load GENE70 signature
data(sig.endoPredict)
## load NKI dataset
data(vdxs)
## compute relapse score
rs.vdxs <- endoPredict(data=data.vdxs, annot=annot.vdxs, do.mapping=FALSE)
```

Description
This dataset contains (part of) the gene expression, annotations and clinical data from the expO dataset collected by the International Genomics Consortium (http://www.intgen.org/expo/).

Usage
data(expos)

Format
expos is a dataset containing three matrices:

- data.expos Matrix containing gene expressions as measured by Affymetrix hgu133plus2 technology (single-channel, oligonucleotides)
- annot.expos Matrix containing annotations of fymetrix hgu133plus2 microarray platform
- demo.expos Clinical information of the breast cancer patients whose tumors were hybridized

Details
This dataset has been generated by the International Genomics Consortium using Affymetrix hgu133plus2 technology. The gene expressions have been normalized using fRMA. Only part of the gene expressions (966) are contained in data.expos.

Source
fuzzy.ttest

References


Examples

data(expos)

fuzzy.ttest

Function to compute the fuzzy Student t test based on weighted mean and weighted variance

Description

This function allows for computing the weighted mean and weighted variance of a vector of continuous values.

Usage

fuzzy.ttest(x, w1, w2, alternative=c("two.sided", "less", "greater"), check.w = TRUE, na.rm = FALSE)

Arguments

x an object containing the observed values.
w1 a numerical vector of weights of the same length as x giving the weights to use for elements of x in the first class.
w2 a numerical vector of weights of the same length as x giving the weights to use for elements of x in the second class.
alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
check.w TRUE if weights should be checked such that 0 <= w <= 1 and (w1[i] + w2[i]) < 1 for 1 <= i <= length(x), FALSE otherwise. Beware that weights greater than one may inflate over-optimistically resulting p-values, use with caution.
na.rm TRUE if missing values should be removed, FALSE otherwise.

Details

The weights w1 and w2 should represent the likelihood for each observation stored in x to belong to the first and second class, respectively. Therefore the values contained in w1 and w2 should lay in [0,1] and 0 <= (w1[i] + w2[i]) <= 1 for i in {0,1,...,n} where n is the length of x.

The Welch’s version of the t test is implemented in this function, therefore assuming unequal sample size and unequal variance. The sample size of the first and second class are calculated as the sum(w1) and sum(w2), respectively.
Value

A numeric vector of six values that are the difference between the two weighted means, the value of the t statistic, the sample size of class 1, the sample size of class 2, the degree of freedom and the corresponding p-value.

Author(s)

Benjamin Haibe-Kains

References

http://en.wikipedia.org/wiki/T_test

See Also

weighted.mean

Examples

```r
set.seed(54321)
## random generation of 50 normally distributed values for each of the two classes
xx <- c(rnorm(50), rnorm(50)+1)
## fuzzy membership to class 1
ww1 <- runif(50) + 0.3
ww1[ww1 > 1] <- 1
ww1 <- c(ww1, 1 - ww1)
## fuzzy membership to class 2
ww2 <- 1 - ww1
## Welch's t test weighted by fuzzy membership to class 1 and 2
wt <- fuzzy.ttest(x=xx, w1=ww1, w2=ww2)
print(wt)
## Not run:
## permutation test to compute the null distribution of the weighted t statistic
wt <- wt[2]
rands <- t(sapply(1:1000, function(x,y) { return(sample(1:y)) }, y=length(xx)))
randst <- apply(rands, 1, function(x, xx, ww1, ww2)
{ return(fuzzy.ttest(x=xx, w1=ww1[x], w2=ww2[x])[[2]]) }, xx=xx, ww1=ww1, ww2=ww2)
ifelse(wt < 0, sum(randst <= wt), sum(randst >= wt)) / length(randst)
## End(Not run)
```

gene70

Function to compute the 70 genes prognosis profile (GENE70) as published by van’t Veer et al. 2002

Description

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the 70 genes prognosis profile (GENE70) as published by van’t Veer et al. 2002.
Usage
gene70(data, annot, do.mapping = FALSE, mapping,
   std = c("none", "scale", "robust"), verbose = FALSE)

Arguments
data Matrix of gene expressions with samples in rows and probes in columns, dimension names being properly defined.
annot Matrix of annotations with at least one column named "EntrezGene.ID", dimension names being properly defined.
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
mapping Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
std Standardization of gene expressions: scale for traditional standardization based on mean and standard deviation, robust for standardization based on the 0.025 and 0.975 quantiles, none to keep gene expressions unchanged.
verbose TRUE to print informative messages, FALSE otherwise.

Value
score Continuous signature scores
risk Binary risk classification, 1 being high risk and 0 being low risk.
mapping Mapping used if necessary.
probe If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.

Author(s)
Benjamin Haibe-Kains

References

See Also
nkis

Examples
## load GENE70 signature
data(sig.gene70)
## load NKI dataset
data(nkis)
## compute relapse score
rs.nkis <- gene70(data=data.nkis)
table(rs.nkis$risk)
## note that the discrepancies compared to the original publication
## are closed to the official cutoff, raising doubts on its exact value.
## computation of the signature scores on a different microarray platform
## load VDX dataset
data(vdxs)
## compute relapse score
rs.vdxs <- gene70(data=data.vdxs, annot=annot.vdxs, do.mapping=TRUE)
table(rs.vdxs$risk)

---

**gene76**  
*Function to compute the Relapse Score as published by Wang et al. 2005*

### Description

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the Relapse Score (GENE76) as published by Wang et al. 2005.

### Usage

```r
gene76(data, er)
```

### Arguments

- **data**: Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
- **er**: Vector containing the estrogen receptor (ER) status of breast cancer patients in the dataset.

### Value

- **score**: Continuous signature scores
- **risk**: Binary risk classification, 1 being high risk and 0 being low risk.

### Author(s)

Benjamin Haibe-Kains

### References


### See Also

- `ggi`
Examples

```r
## load GENE76 signature
data(sig.gene76)
## load VDX dataset
data(vdxs)
## compute relapse score
rs.vdxs <- gene76(data=data.vdxs, er=demo.vdxs[ ,"er"])
table(rs.vdxs$risk)
```

---

geneid.map

*Function to find the common genes between two datasets or a dataset and a gene list*

Description

This function allows for fast mapping between two datasets or a dataset and a gene list. The mapping process is performed using Entrez Gene id as reference. In case of ambiguities (several probes representing the same gene), the most variant probe is selected.

Usage

geneid.map(geneid1, data1, geneid2, data2, verbose = FALSE)

Arguments

geneid1  
first vector of Entrez Gene ids. The name of the vector cells must be the name of the probes in the dataset data1.

data1  
First dataset with samples in rows and probes in columns. The dimnames must be properly defined.

geneid2  
Second vector of Entrez Gene ids. The name of the vector cells must be the name of the probes in the dataset data1 if it is not missing, proper names must be assigned otherwise.

data2  
First dataset with samples in rows and probes in columns. The dimnames must be properly defined. It may be missing.

verbose  
TRUE to print informative messages, FALSE otherwise.

Value

geneid1  
Mapped gene list from geneid1.

data1  
Mapped dataset from data1.

geneid2  
Mapped gene list from geneid2.

data2  
Mapped dataset from data2.

Note

It is mandatory that the names of geneid1 and geneid2 must be the probe names of the microarray platform.
Author(s)
Benjamin Haibe-Kains

Examples
```r
## load NKI data
data(nkis)
nkis.gid <- annot.nkis[, "EntrezGene.ID"]
names(nkis.gid) <- dimnames(annot.nkis)[[1]]
## load GGI signature
data(sig.ggi)
ggi.gid <- sig.ggi[, "EntrezGene.ID"]
names(ggi.gid) <- as.character(sig.ggi[, "probe"])
## mapping through Entrez Gene ids of NKI and GGI signature
res <- geneid.map(geneid1=nkis.gid, data1=data.nkis,
                  geneid2=ggi.gid, verbose=FALSE)
str(res)
```

Description
This function computes the Gene Expression progNostic Index Using Subtypes (GENIUS) as published by Haibe-Kains et al. 2010. Subtype-specific risk scores are computed for each subtype signature separately and an overall risk score is computed by combining these scores with the posterior probability to belong to each of the breast cancer molecular subtypes.

Usage
```r
genius(data, annot, do.mapping = FALSE, mapping, do.scale = TRUE)
```

Arguments
- **data**: Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
- **annot**: Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
- **do.mapping**: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- **mapping**: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- **do.scale**: TRUE if the ESR1, ERBB2 and AURKA (module) scores must be rescaled (see `rescale`), FALSE otherwise.

Value
- **GENIUSM1**: Risk score from the ER-/HER2- subtype signature in GENIUS model.
- **GENIUSM2**: Risk score from the HER2+ subtype signature in GENIUS model.
- **GENIUSM3**: Risk score from the ER+/HER2- subtype signature in GENIUS model.
- **score**: Overall risk prediction as computed by the GENIUS model.
**gg**

**Author(s)**

Benjamin Haibe-Kains

**References**


**See Also**

`subtype.cluster.predict.sig.score`

**Examples**

```r
## load NKI dataset
data(nkis)
## compute GENIUS risk scores based on GENIUS model fitted on VDX dataset
genius.nkis <- genius(data=data.nkis, annot=annot.nkis, do.mapping=TRUE)
str(genius.nkis)
## the performance of GENIUS overall risk score predictions are not optimal
## since only part of the NKI dataset was used
```

---

`gg()`  
*Function to compute the raw and scaled Gene expression Grade Index (GGI)*

**Description**

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the Gene expression Grade Index (GGI).

**Usage**

```r
ggi(data, annot, do.mapping = FALSE, mapping, hg, verbose = FALSE)
```

**Arguments**

- `data` Matrix of gene expressions with samples in rows and probes in columns, dim-names being properly defined.
- `annot` Matrix of annotations with at least one column named "EntrezGene.ID", dim-names being properly defined.
- `do.mapping` TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping` Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- `hg` Vector containing the histological grade (HG) status of breast cancer patients in the dataset.
- `verbose` TRUE to print informative messages, FALSE otherwise.
Value

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>score</td>
<td>Continuous signature scores</td>
</tr>
<tr>
<td>risk</td>
<td>Binary risk classification, 1 being high risk and 0 being low risk.</td>
</tr>
<tr>
<td>mapping</td>
<td>Mapping used if necessary.</td>
</tr>
<tr>
<td>probe</td>
<td>If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.</td>
</tr>
</tbody>
</table>

Author(s)

Benjamin Haibe-Kains

References


See Also

gene76

Examples

```r
## load GGI signature
data(sig.ggi)
## load NKI dataset
data(nkis)
## compute relapse score
ggi.nkis <- ggi(data=data.nkis, annot=annot.nkis, do.mapping=TRUE,
              ,hg=demo.nkis[, "grade"]
table(ggi.nkis$risk)
```

---

**ihc4**

*Function to compute the IHC4 prognostic score as published by Paik et al. in 2004.*

Description

This function computes the prognostic score based on four measured IHC markers (ER, PGR, HER2, Ki-67), following the algorithm as published by Cuzick et al. 2011. The user has the option to either obtain just the shrinkage-adjusted IHC4 score (IHC4) or the overall score that also combines the clinical score (IHC4+C).

Usage

```
ihc4(ER, PGR, HER2, Ki67, age, size, grade, node, ana, scoreWithClinical=FALSE, na.rm = FALSE)
```
### Arguments

- **ER**
  - ER score between 0-10, calculated as (H-score/30)

- **PGR**
  - Progesterone Receptor score between 0-10

- **HER2**
  - Her2/neu status (0 or 1).

- **Ki67**
  - Ki67 score based on percentage of positively staining malignant cells

- **age**
  - Patient age

- **size**
  - Tumor size in cm.

- **grade**
  - Histological grade, i.e. low (1), intermediate (2) and high (3) grade.

- **node**
  - Nodal status.

- **ana**
  - Treatment with anastrozole

- **scoreWithClinical**
  - TRUE to get IHC4+C score, FALSE to get just the IHC4 score.

- **na.rm**
  - TRUE if missing values should be removed, FALSE otherwise.

### Value

Shrinkage-adjusted IHC4 score or the Overall Prognostic Score based on IHC4+C (IHC4+Clinical Score)

### Author(s)

Deena M.A. Gendoo

### References


### Examples

```r
## load NKI dataset
data(nkis)
## compute shrinkage-adjusted IHC4 score
count<-nrow(demo.nkis)
ihc4(ER=sample(x=1:10, size=count,replace=TRUE), PGR=sample(x=1:10, size=count,replace=TRUE),
HER2=sample(x=0:1, size=count,replace=TRUE), Ki67=sample(x=1:100, size=count,replace=TRUE),
scoreWithClinical=FALSE, na.rm=TRUE)

## compute IHC4+C score
ihc4(ER=sample(x=1:10, size=count,replace=TRUE), PGR=sample(x=1:10, size=count,replace=TRUE),
HER2=sample(x=0:1, size=count,replace=TRUE), Ki67=sample(x=1:100, size=count,replace=TRUE),
age=demo.nkis[, "age"], size=demo.nkis[, "size"], grade=demo.nkis[, "grade"], node=demo.nkis[, "node"],
ana=sample(x=0:1, size=count,replace=TRUE), scoreWithClinical=TRUE, na.rm=TRUE)
```
intrinsic.cluster  Function to fit a Single Sample Predictor (SSP) as in Perou, Sorlie, Hu, and Parker publications

Description

This function fits the Single Sample Predictor (SSP) as published in Sorlie et al 2003, Hu et al 2006 and Parker et al 2009. This model is actually a nearest centroid classifier where the centroids representing the breast cancer molecular subtypes are identified through hierarchical clustering using an "intrinsic gene list".

Usage

intrinsic.cluster(data, annot, do.mapping = FALSE, mapping, std = c("none", "scale", "robust"), rescale.q = 0.05, intrinsicg, number.cluster = 3, mins = 5, method.cor = c("spearman", "pearson"), method.centroids = c("mean", "median", "tukey"), filen, verbose = FALSE)

Arguments

data                  Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
annot                 Matrix of annotations with at least one column named "EntrezGene.ID", dimnames being properly defined.
do.mapping           TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
mapping              Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
std                   Standardization of gene expressions: scale for traditional standardization based on mean and standard deviation, robust for standardization based on the 0.025 and 0.975 quantiles, none to keep gene expressions unchanged.
rescale.q             Proportion of expected outliers for (robust) rescaling the gene expressions.
intrinsicg           Intrinsic gene lists. May be specified by the user as a matrix with at least 2 columns named probe and EntrezGene.ID for the probe names and the corresponding Entrez Gene ids. The intrinsic gene lists published by Sorlie et al. 2003, Hu et al. 2006 and Parker et al. 2009 are stored in ssp2003, ssp2006 and pam50 respectively.
number.cluster       The number of main clusters to be identified by hierarchical clustering.
mins                  The minimum number of samples to be in a main cluster.
method.cor            Correlation coefficient used to identified the nearest centroid. May be spearman or pearson.
method.centroids      Method to compute a centroid from gene expressions of a cluster of samples: mean, median or tukey (Tukey’s Biweight Robust Mean).
filen                 Name of the csv file where the subtype clustering model must be stored.
verbose               TRUE to print informative messages, FALSE otherwise.
**Value**

- **model** Single Sample Predictor
- **subtype** Subtypes identified by the SSP. For published intrinsic gene lists, subtypes can be either "Basal", "Her2", "LumA", "LumB" or "Normal".
- **subtype.proba** Probabilities to belong to each subtype estimated from the correlations to each centroid.
- **cor** Correlation coefficient to each centroid.

**Author(s)**

Benjamin Haibe-Kains

**References**


Hu, Zhiyuan and Fan, Cheng and Oh, Daniel and Marron, JS and He, Xiaping and Qaqish, Bahjat and Livasy, Chad and Carey, Lisa and Reynolds, Evangeline and Dressler, Lynn and Nobel, Andrew and Parker, Joel and Ewend, Matthew and Sawyer, Lynda and Wu, Junyuan and Liu, Yudong and Nanda, Rita and Tretiakova, Maria and Orrico, Alejandra and Dreher, Donna and Palazzo, Juan and Perreard, Laurent and Nelson, Edward and Mone, Mary and Hansen, Heidi and Mullins, Michael and Quackenbush, John and Ellis, Matthew and Olopade, Olufunmilayo and Bernard, Philip and Perou, Charles (2006) "The molecular portraits of breast tumors are conserved across microarray platforms", *BMC Genomics*, 7(96)


**See Also**

- `subtype.cluster`
- `intrinsic.cluster.predict`
- `ssp2003`
- `ssp2006`
- `pam50`

**Examples**

```r
## load SSP signature published in Sorlie et al. 2003
data(ssp2003)
## load NKI data
data(nkis)
## load VDX data
data(vdxs)
ssp2003.nkis <- intrinsic.cluster(data=data.nkis, annot=annot.nkis,
do.mapping=TRUE, std="robust",
intrinsicg=ssp2003$centroids.map[,c("probe", "EntrezGene.ID")],
number.cluster=5, mins=5, method.cor="spearman",
method.centroids="mean", verbose=TRUE)
str(ssp2003.nkis, max.level=1)
```
intrinsic.cluster.predict

*Function to identify breast cancer molecular subtypes using the Single Sample Predictor (SSP)*

**Description**

This function identifies the breast cancer molecular subtypes using a Single Sample Predictor (SSP) fitted by `intrinsic.cluster`.

**Usage**

```r
intrinsic.cluster.predict(sbt.model, data, annot, do.mapping = FALSE, mapping, do.prediction.strength = FALSE, verbose = FALSE)
```

**Arguments**

- `sbt.model`: Subtype Clustering Model as returned by `intrinsic.cluster`.
- `data`: Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
- `annot`: Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
- `do.mapping`: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping`: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- `do.prediction.strength`: TRUE if the prediction strength must be computed (Tibshirani and Walther 2005), FALSE otherwise.
- `verbose`: TRUE to print informative messages, FALSE otherwise.

**Value**

- `subtype`: Subtypes identified by the SSP. For published intrinsic gene lists, subtypes can be either "Basal", "Her2", "LumA", "LumB" or "Normal".
- `subtype.proba`: Probabilities to belong to each subtype estimated from the correlations to each centroid.
- `cor`: Correlation coefficient to each centroid.
- `prediction.strength`: Prediction strength for subtypes.
- `subtype.train`: Classification (similar to subtypes) computed during fitting of the model for prediction strength.
- `centroids.map`: Mapped probes from the intrinsic gene list used to compute the centroids.
- `profiles`: Intrinsic gene expression profiles for each sample.

**Author(s)**

Benjamin Haibe-Kains
map.datasets

Function to map a list of datasets through EntrezGene IDs in order to get the union of the genes

Description
This function maps a list of datasets through EntrezGene IDs in order to get the union of the genes.

Usage
map.datasets(datas, annots, do.mapping = FALSE, mapping.coln = "EntrezGene.ID", mapping, verbose = FALSE)
Arguments

datas  List of matrices of gene expressions with samples in rows and probes in columns, dimnames being properly defined.

annots  List of matrices of annotations with at least one column named "EntrezGene.ID", dimnames being properly defined.

do.mapping  TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.

mapping.coln  Name of the column containing the biological annotation to be used to map the different datasets, default is "EntrezGene.ID".

mapping  Matrix with columns "EntrezGene.ID" and "probe.x" used to force the mapping such that the probes of platform x are not selected based on their variance.

verbose  TRUE to print informative messages, FALSE otherwise.

Details

In case of several probes representing the same EntrezGene ID, the most variant is selected if mapping is not specified. When a EntrezGene ID does not exist in a specific dataset, NA values are introduced.

Value

datas  List of datasets (gene expression matrices)

annots  List of annotations (annotation matrices)

Author(s)

Benjamin Haibe-Kains

Examples

```r
## load VDX dataset
data(vdxs)
## load NKI dataset
data(nkis)
## reduce datasets
ginter <- intersect(annot.vdxs[, "EntrezGene.ID"], annot.nkis[, "EntrezGene.ID"])
ginter <- ginter[!is.na(ginter)][1:30]
myx <- unique(c(match(ginter, annot.vdxs[, "EntrezGene.ID"]),
               sample(x=1:nrow(annot.vdxs), size=20)))
data2.vdxs <- data.vdxs[, myx]
annot2.vdxs <- annot.vdxs[myx, ]
myx <- unique(c(match(ginter, annot.nkis[, "EntrezGene.ID"]),
               sample(x=1:nrow(annot.nkis), size=20)))
data2.nkis <- data.nkis[, myx]
annot2.nkis <- annot.nkis[myx, ]
## mapping of datasets
datas <- list("VDX"=data2.vdxs,"NKI"=data2.nkis)
annots <- list("VDX"=annot2.vdxs, "NKI"=annot2.nkis)
datas.mapped <- map.datasets(datas=datas, annots=annots, do.mapping=TRUE)
str(datas.mapped, max.level=2)
```
medianCtr  Center around the median

Description
Utility function called within the claudinLow classifier

Usage
medianCtr(x)

Arguments
x  Matrix of numbers

Value
Returns a matrix of median-centered numbers

References
citation("claudinLow")

See Also
claudinLow

mod1  Gene modules published in Desmedt et al. 2008

Description
List of seven gene modules published in Desmedt et al. 2008, i.e. ESR1 (estrogen receptor pathway), ERBB2 (her2/neu receptor pathway), AURKA (proliferation), STAT1 (immune response), PLAU (tumor invasion), VEGF (angogenesis) and CASP3 (apoptosis).

Usage
data(mod1)

Format
mod1 is a list of seven signatures, i.e. matrices with 3 columns containing the annotations and information related to the signatures themselves.

Source
http://clincancerres.aacrjournals.org/content/14/16/5158.abstract?ck=nck
References

Examples
data(mod1)

mod2
Gene modules published in Wirapati et al. 2008

Description
List of seven gene modules published in Wirapati et al. 2008, i.e. ESR1 (estrogen receptor pathway), ERBB2 (her2/neu receptor pathway) and AURKA (proliferation).

Usage
data(mod2)

Format
mod2 is a list of three gene signatures, i.e. matrices with 3 columns containing the annotations and information related to the signatures themselves.

Source
http://breast-cancer-research.com/content/10/4/R65

References

Examples
data(mod2)
**modelOvcAngiogenic**

Model used to classify ovarian tumors into Angiogenic and NonAngiogenic subtypes.

---

**Description**

Object containing the set of parameters for the mixture of Gaussians used as a model to classify ovarian tumors into Angiogenic and NonAngiogenic subtypes.

**Usage**

`data(modelOvcAngiogenic)`

**Format**

`modelOvcAngiogenic`

**References**


**Examples**

`data(modelOvcAngiogenic)`

`head(modelOvcAngiogenic)`

---

**molecular.subtyping**

Function to identify breast cancer molecular subtypes using the Subtype Clustering Model

---

**Description**

This function identifies the breast cancer molecular subtypes using a Subtype Clustering Model fitted by `subtype.cluster`.

**Usage**

`molecular.subtyping(sbt.model = c("scmgene", "scmod1", "scmod2", "pam50", "ssp2006", "ssp2003", "intClust", "AIMS","claudinLow"), data, annot, do.mapping = FALSE)`
molecular.subtyping

Arguments

sbt.model Subtyping classification model, can be either "scmgene", "scmod1", "scmod2", "pam50", "ssp2006", "ssp2003", "intClust", "AIMS", or "claudinLow".
data Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
annot Matrix of annotations with at least one column named "EntrezGene.ID" (for ssp, scm, AIMS, and claudinLow models) or "Gene.Symbol" (for the intClust model), dimnames being properly defined.
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.

Value

subtype Subtypes identified by the subtyping classification model.
subtype.proba Probabilities to belong to each subtype estimated by the subtyping classification model.
subtype.crisp Crisp classes identified by the subtyping classification model.

Author(s)

Benjamin Haibe-Kains

References

Hu, Zhiyuan and Fan, Cheng and Oh, Daniel and Marron, JS and He, Xiaping and Qaqish, Bahjat and Livasy, Chad and Carey, Lisa and Reynolds, Evangeline and Dressler, Lynn and Nobel, Andrew and Parker, Joel and Ewend, Matthew and Sawyer, Lynda and Wu, Junyuan and Liu, Yudong and Nanda, Rita and Tretiakova, Maria and Orrico, Alejandra and Dreher, Donna and Palazzo, Juan and Perreard, Laurent and Nelson, Edward and Mone, Mary and Hansen, Heidi and Mullins, Michael and Quackenbush, John and Ellis, Matthew and Olopade, Olufunmilayo and Bernard, Philip and Perou, Charles (2006) "The molecular portraits of breast tumors are conserved across microarray platforms", BMC Genomics, 7(96)


See Also

`subtype.cluster.predict`, `intrinsic.cluster.predict`

Examples

```r
##### without mapping (affy hgu133a or plus2 only)
## load VDX data
data(vdxs)

## Subtype Clustering Model fitted on EXPO and applied on VDX
sbt.vdx.SCMGENE <- molecular.subtyping(sbt.model="scmgene",
data=vdxs, annot=annot.vdxs, do.mapping=FALSE)
table(sbt.vdx.SCMGENE$subtype)

## Using the AIMS molecular subtyping algorithm
sbt.vdxs.AIMS <- molecular.subtyping(sbt.model="AIMS", data=vdxs,
annot=annot.vdxs, do.mapping=FALSE)
table(sbt.vdxs.AIMS$subtype)

## Using the IntClust molecular subtyping algorithm
colnames(annot.vdxs)[3]<-"Gene.Symbol"
sbt.vdxs.intClust <- molecular.subtyping(sbt.model="intClust", data=vdxs,
annot=annot.vdxs, do.mapping=FALSE)
table(sbt.vdxs.intClust$subtype)

##### with mapping
## load NKI data
data(nkis)

## Subtype Clustering Model fitted on EXPO and applied on NKI
sbt.nkis <- molecular.subtyping(sbt.model="scmgene", data=nkis,
annot=nkis, do.mapping=TRUE)
table(sbt.nkis$subtype)

##### with mapping
## load vdxs data
data(vdxs)

## Claudin-Low classification of 150 VDXS samples
```

nxis

Gene expression, annotations and clinical data from van de Vijver et al. 2002

Description

This dataset contains (part of) the gene expression, annotations and clinical data as published in van de Vijver et al. 2002.

Usage

data(nxis)

Format

nxis is a dataset containing three matrices:

- data.nxis: Matrix containing gene expressions as measured by Agilent technology (dual-channel, oligonucleotides)
- annot.nxis: Matrix containing annotations of Agilent microarray platform
- demo.nxis: Clinical information of the breast cancer patients whose tumors were hybridized

Details

This dataset represent only partially the one published by van de Vijver et al. in 2008. Indeed, only part of the patients (150) and gene expressions (922) are contained in data.nxis.

Source

http://www.nature.com/nature/journal/v415/n6871/full/415530a.html

References


Examples

data(nxis)
npi

Function to compute the Nottingham Prognostic Index

Description

This function computes the Nottingham Prognostic Index (NPI) as published in Galeat et al, 1992. NPI is a clinical index shown to be highly prognostic in breast cancer.

Usage

npi(size, grade, node, na.rm = FALSE)

Arguments

size  tumor size in cm.
grade  Histological grade, i.e. low (1), intermediate (2) and high (3) grade.
node  Nodal status. If only binary nodal status (0/1) is available, map 0 to 1 and 1 to 3.
na.rm  TRUE if missing values should be removed, FALSE otherwise.

Details

The risk prediction is either Good if score < 3.4, Intermediate if 3.4 <= score <= 5.4, or Poor if score > 5.4.

Value

score  Continuous signature scores
risk  Binary risk classification, 1 being high risk and 0 being low risk.

Author(s)

Benjamin Haibe-Kains

References


See Also

st.gallen

Examples

```r
## load NKI dataset
data(nkis)
## compute NPI score and risk classification
npi(size=demo.nkis[,"size"], grade=demo.nkis[,"grade"],
    node=ifelse(demo.nkis[,"node"] == 0, 1, 3), na.rm=TRUE)
```
oncotypedx

Function to compute the OncotypeDX signature as published by Paik et al. in 2004.

Description

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the OncotypeDX signature as published by Paik et al. 2004.

Usage

oncotypedx(data, annot, do.mapping = FALSE, mapping, verbose = FALSE)

Arguments

data Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
annot Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise. Note that for Affymetrix HGU datasets, the mapping is not necessary.
mapping Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
verbose TRUE to print informative messages, FALSE otherwise.

Details

Note that for Affymetrix HGU datasets, the mapping is not necessary.

Value

score Continuous signature scores
risk Binary risk classification, 1 being high risk and 0 being low risk.
mapping Mapping used if necessary.
probe If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.

Author(s)

Benjamin Haibe-Kains

References

Examples

```r
## load GENE70 signature
data(sig.oncotypedx)
## load NKI dataset
data(nkis)
## compute relapse score
rs.nkis <- oncotypedx(data=data.nkis, annot=annot.nkis, do.mapping=TRUE)
table(rs.nkis$risk)
```

---

**ovcAngiogenic**

Function to compute the subtype scores and risk classifications for the angiogenic molecular subtype in ovarian cancer.

### Description

This function computes subtype scores and risk classifications from gene expression values following the algorithm developed by Bentink, Haibe-Kains et al. to identify the angiogenic molecular subtype in ovarian cancer.

### Usage

```r
ovcAngiogenic(data, annot, hgs, hgs, 
gmap = c("entrezgene", "ensembl_gene_id", "hgnc_symbol", "unigene"),
do.mapping = FALSE, verbose = FALSE)
```

### Arguments

- `data` Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
- `annot` Matrix of annotations with one column named as gmap, dimnames being properly defined.
- `hgs` vector of booleans with TRUE represents the ovarian cancer patients who have a high grade, late stage, serous tumor, FALSE otherwise. This is particularly important for properly rescaling the data. If hgs is missing, all the patients will be used to rescale the subtype score.
- `gmap` character string containing the bioMaRt attribute to use for mapping if do.mapping=TRUE.
- `do.mapping` TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `verbose` TRUE to print informative messages, FALSE otherwise.

### Value

- `score` Continuous signature scores
- `risk` Binary risk classification, 1 being high risk and 0 being low risk.
- `mapping` Mapping used if necessary.
- `probe` If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.
- `subtype` data frame reporting the subtype score, maximum likelihood classification and corresponding subtype probabilities
Author(s)
Benjamin Haibe-Kains

References

See Also
sigOvcAngiogenic

Examples
## load the ovcAngiogenic signature
data(sigOvcAngiogenic)
## load NKI dataset
data(nkis)
colnames(annot.nkis)[is.element(colnames(annot.nkis), "EntrezGene.ID")]<-"entrezgene"
## compute relapse score
ovcAngiogenic.nkis <- ovcAngiogenic(data=data.nkis, annot=annot.nkis, gmap="entrezgene", do.mapping=TRUE)
table(ovcAngiogenic.nkis$risk)

ovcCrijns

Function to compute the subtype scores and risk classifications for the prognostic signature published by Crijns et al.

Description
This function computes subtype scores and risk classifications from gene expression values using the weights published by Crijns et al.

Usage
ovcCrijns(data, annot, hgs, gmap = c("entrezgene", "ensembl_gene_id", "hgnc_symbol", "unigene"), do.mapping = FALSE, verbose = FALSE)

Arguments
data
Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.

annot
Matrix of annotations with one column named as gmap, dimnames being properly defined.

hgs
vector of booleans with TRUE represents the ovarian cancer patients who have a high grade, late stage, serous tumor, FALSE otherwise. This is particularly important for properly rescaling the data. If hgs is missing, all the patients will be used to rescale the subtype score.
ovcCrijns

gmap character string containing the biomaRt attribute to use for mapping if do.mapping=TRUE
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
verbose TRUE to print informative messages, FALSE otherwise.

Details

Note that the original algorithm has not been implemented as it necessitates refitting of the model weights in each new dataset. However the current implementation should give similar results.

Value

score Continuous signature scores
risk Binary risk classification, 1 being high risk and 0 being low risk.
mapping Mapping used if necessary.
probe If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.

Author(s)

Benjamin Haibe-Kains

References


See Also

sigOvcCrijns

Examples

## load the ovsCrijns signature
data(sigOvcCrijns)
## load NKI dataset
data(nkis)
colnames(annot.nkis)[is.element(colnames(annot.nkis), "EntrezGene.ID")]<- "entrezgene"
## compute relapse score
ovcCrijns.nkis <- ovcCrijns(data=data.nkis, annot=annot.nkis, gmap="entrezgene", do.mapping=TRUE)
table(ovcCrijns.nkis$risk)
ovcTCGA Function to compute the prediction scores and risk classifications for the ovarian cancer TCGA signature

Description
This function computes signature scores and risk classifications from gene expression values following the algorithm developed by the TCGA consortium for ovarian cancer.

Usage
ovcTCGA(data, annot,
gmap = c("entrezgene", "ensembl_gene_id", "hgnc_symbol", "unigene"),
do.mapping = FALSE, verbose = FALSE)

Arguments
data Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
annot Matrix of annotations with one column named as gmap, dimnames being properly defined.
gmap character string containing the biomaRt attribute to use for mapping if do.mapping=TRUE
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
verbose TRUE to print informative messages, FALSE otherwise.

Value
score Continuous signature scores
risk Binary risk classification, 1 being high risk and 0 being low risk.
mapping Mapping used if necessary.
probe If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.

Author(s)
Benjamin Haibe-Kains

References

See Also
sigOvcTCGA
Examples

```r
## load the ovcTCGA signature
data(sigOvcTCGA)
## load NKI dataset
data(nkis)
colnames(annot.nkis)[is.element(colnames(annot.nkis), "EntrezGene.ID")]
<- "entrezgene"
## compute relapse score
ovcTCGA.nkis <- ovcTCGA(data=data.nkis, annot=annot.nkis, gmap="entrezgene", do.mapping=TRUE)
table(ovcTCGA.nkis$risk)
```

### ovcYoshihara

Function to compute the subtype scores and risk classifications for the prognostic signature published by Yoshihara et al.

#### Description

This function computes subtype scores and risk classifications from gene expression values following the algorithm developed by Yoshihara et al, for prognosis in ovarian cancer.

#### Usage

```r
ovcYoshihara(data, annot, hgs,
gmap = c("entrezgene", "ensembl_gene_id", "hgnc_symbol", "unigene", "refseq_mrna"),
do.mapping = FALSE, verbose = FALSE)
```

#### Arguments

- `data` Matrix of gene expressions with samples in rows and probes in columns, dimnames being properly defined.
- `annot` Matrix of annotations with one column named as gmap, dimnames being properly defined.
- `hgs` vector of booleans with TRUE represents the ovarian cancer patients who have a high grade, late stage, serous tumor, FALSE otherwise. This is particularly important for properly rescaling the data. If hgs is missing, all the patients will be used to rescale the subtype score.
- `gmap` character string containing the biomaRt attribute to use for mapping if do.mapping=TRUE
- `do.mapping` TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `verbose` TRUE to print informative messages, FALSE otherwise.

#### Value

- `score` Continuous signature scores
- `risk` Binary risk classification, 1 being high risk and 0 being low risk.
- `mapping` Mapping used if necessary.
- `probe` If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.
Author(s)

Benjamin Haibe-Kains

References


See Also

sigOvcYoshihara

Examples

```r
## load the ovcYoshihara signature
data(sigOvcYoshihara)
## load NKI dataset
data(nkis)
colnames(annot.nkis)[is.element(colnames(annot.nkis), "EntrezGene.ID")]<- "entrezgene"
## compute relapse score
ovcYoshihara.nkis <- ovcYoshihara(data=data.nkis,
annot=annot.nkis, gmap="entrezgene", do.mapping=TRUE)
table(ovcYoshihara.nkis$risk)
```

overlapSets

Overlap two datasets

Description

Utility function called within the claudinLow classifier

Usage

`overlapSets(x, y)`

Arguments

- `x`: Matrix1
- `y`: Matrix2

Value

Overlapped dataset

References

`citation("claudinLow")`

See Also

`claudinLow`
**PAM50 classifier for identification of breast cancer molecular subtypes (Parker et al 2009)**

**Description**

List of parameters defining the PAM50 classifier for identification of breast cancer molecular subtypes (Parker et al 2009).

**Usage**

```
data(pam50)
data(pam50.scale)
data(pam50.robust)
```

**Format**

List of parameters for PAM50:

- `centroids` Gene expression centroids for each subtype.
- `centroids.map` Mapping for centroids.
- `method.cor` Method of correlation used to compute distance to the centroids.
- `method.centroids` Method used to compute the centroids.
- `std` Method of standardization for gene expressions ("none", "scale" or "robust").
- `mins` Minimum number of samples within each cluster allowed during the fitting of the model.

**Details**

Three versions of the model are provided, each of ones differs by the gene expressions standardization method since it has an important impact on the subtype classification:

- `pam50` Use of the official centroids without scaling of the gene expressions.
- `pam50.scale` Use of the official centroids with traditional scaling of the gene expressions (see `scale`).
- `pam50.robust` Use of the official centroids with robust scaling of the gene expressions (see `rescale`).

The model `pam50.robust` has been shown to reach the best concordance with the traditional clinical parameters (ER IHC, HER2 IHC/FISH and histological grade). However the use of this model is recommended only when the dataset is representative of a global population of breast cancer patients (no sampling bias, the 5 subtypes should be present).

**Source**

[http://jco.ascopubs.org/cgi/content/short/JCO.2008.18.1370v1](http://jco.ascopubs.org/cgi/content/short/JCO.2008.18.1370v1)

**References**

**pik3cags**

*Function to compute the PIK3CA gene signature (PIK3CA-GS)*

**Description**

This function computes signature scores from gene expression values following the algorithm used for the PIK3CA gene signature (PIK3CA-GS).

**Usage**

```r
pik3cags(data, annot, do.mapping = FALSE, mapping, verbose = FALSE)
```

**Arguments**

- `data` Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
- `annot` Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
- `do.mapping` TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping` Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- `verbose` TRUE to print informative messages, FALSE otherwise.

**Value**

Vector of signature scores for PIK3CA-GS

**Author(s)**

Benjamin Haibe-Kains

**References**


**See Also**

gene76
Examples

## load GGI signature
data(sig.pik3cags)
## load NKI dataset
data(nkis)
## compute relapse score
pik3cags.nkis <- pik3cags(data=data.nkis, annot=annot.nkis, do.mapping=TRUE)
head(pik3cags.nkis)

power.cor

Function for sample size calculation for correlation coefficients

Description

This function enables to compute the sample size requirements for estimating pearson, kendall and spearman correlations

Usage

power.cor(rho, w, alpha = 0.05, method = c("pearson", "kendall", "spearman"))

Arguments

rho Correlation coefficients rho (Pearson, Kendall or Spearman)
w a numerical vector of weights of the same length as x giving the weights to use for elements of x in the first class.
alpha alpha level
method a character string specifying the method to compute the correlation coefficient, must be one of "pearson" (default), "kendall" or "spearman". You can specify just the initial letter.

Value

sample size requirement

Author(s)

Benjamin Haibe-Kains

References


Examples

power.cor(rho=0.5, w=0.1, alpha=0.05, method="spearman")
ps.cluster Function to compute the prediction strength of a clustering model

Description

This function computes the prediction strength of a clustering model as published in R. Tibshirani and G. Walther 2005.

Usage

ps.cluster(cl.tr, cl.ts, na.rm = FALSE)

Arguments

cl.tr Clusters membership as defined by the original clustering model, i.e. the one that was not fitted on the dataset of interest.
cl.ts Clusters membership as defined by the clustering model fitted on the dataset of interest.
na.rm TRUE if missing values should be removed, FALSE otherwise.

Value

ps the overall prediction strength (minimum of the prediction strengths at cluster level).
ps.cluster Prediction strength for each cluster
ps.individual Prediction strength for each sample.

Author(s)

Benjamin Haibe-Kains

References


Examples

## load SSP signature published in Sorlie et al. 2003
data(ssp2003)
## load NKI data
data(nkis)
## SP2003 fitted on NKI
ssp2003.nkis <- intrinsic.cluster(data=data.nkis, annot=annot.nkis, do.mapping=TRUE, std="robust",
intrinsicg=ssp2003$centroids.map[,c("probe", "EntrezGene.ID")],
number.cluster=5, mins=5, method.cor="spearman",
method.centroids="mean", verbose=TRUE)
## SP2003 published in Sorlie et al 2003 and applied in VDX
ssp2003.nkis <- intrinsic.cluster.predict(sbt.model=ssp2003,
data=data.nkis, annot=annot.nkis, do.mapping=TRUE, verbose=TRUE)
## prediction strength of sp2003 clustering model
**read.m.file**

Function to read a ’csv’ file containing gene lists (aka gene signatures)

**Description**

This function allows for reading a ’csv’ file containing gene signatures. Each gene signature is composed of at least four columns: "gene.list" is the name of the signature on the first line and empty fields below, "probes" are the probe names, "EntrezGene.ID" are the EntrezGene IDs and "coefficient" are the coefficients of each probe.

**Usage**

read.m.file(file, ...)

**Arguments**

- **file**  
  Filename of the ’csv’ file.

- **...**  
  Additional parameters for `read.csv` function.

**Value**

List of gene signatures.

**Author(s)**

Benjamin Haibe-Kains

**See Also**

`mod1`, `mod2`, `extdata/desmedt2008_genemodules.csv`, `extdata/haibekains2009_sig_genius.csv`

**Examples**

```r
## read the seven gene modules as published in Desmedt et al 2008
genemods <- read.m.file(system.file("extdata/desmedt2008_genemodules.csv", package = "genefu"))
str(genemods, max.level=1)
## read the three subtype signatures from GENIUS
geniusm <- read.m.file(system.file("extdata/haibekains2009_sig_genius.csv", package = "genefu"))
str(geniusm, max.level=1)
```
readArray  

**Overlap two datasets**

**Description**

Formatting function to read arrays and format for use in the claudinLow classifier

**Usage**

```r
readarray<-function(dataFile, designFile=NA, hr=1, impute=T, method="mean")
```

**Arguments**

- `dataFile` file with matrix to be read
- `designFile` Design of file
- `hr` Header rows as Present (2) or Absent (1)
- `impute` whether data will be imputed or not.
- `method` Default method is "mean"

**References**

```r
citation("claudinLow")
```

**See Also**

- `claudinLow`

rename.duplicate  

**Function to rename duplicated strings.**

**Description**

This function renames duplicated strings by adding their number of occurrences at the end.

**Usage**

```r
rename.duplicate(x, sep = "_", verbose = FALSE)
```

**Arguments**

- `x` vector of strings.
- `sep` a character to be the separator between the number added at the end and the string itself.
- `verbose` TRUE to print informative messages, FALSE otherwise.

**Value**

- `new.x` new strings (without duplicates).
- `duplicated.x` strings which were originally duplicated.
**rescale**

**Author(s)**

Benjamin Haibe-Kains

**Examples**

```r
nn <- sample(letters[1:10], 30, replace=TRUE)
table(nn)
rename.duplicate(x=nn, verbose=TRUE)
```

**Description**

This function rescales values x based on quantiles specified by the user such that x' = (x - q1) / (q2 - q1) where q is the specified quantile, q1 = q / 2, q2 = 1 - q/2 and x' are the new rescaled values.

**Usage**

```r
rescale(x, na.rm = FALSE, q = 0)
```

**Arguments**

- **x**
- **na.rm** TRUE if missing values should be removed, FALSE otherwise.
- **q** Quantile (must lie in [0,1]).

**Details**

In order to rescale gene expressions, q = 0.05 yielded comparable scales in numerous breast cancer microarray datasets (data not shown). The rational behind this is that, in general, ‘extreme cases’ (e.g. low and high proliferation, high and low expression of ESR1, ...) are often present in microarray datasets, making the estimation of ‘extreme’ quantiles quite stable. This is specially true for genes exhibiting some multi-modality like ESR1 or ERBB2.

**Value**

Vector of rescaled values with two attributes q1 and q1 containing the values of the lower and the upper quantiles respectively.

**Author(s)**

Benjamin Haibe-Kains

**See Also**

`scale`
Examples

# load VDX dataset
data(vdxs)
# load NKI dataset
data(nkis)
# example of rescaling for ESR1 expression
par(mfrow=c(2,2))
hist(data.vdxs[, "205225_at"], xlab="205225_at", breaks=20,
main="ESR1 in VDX")
hist(data.nkis[, "NM_000125"], xlab="NM_000125", breaks=20,
main="ESR1 in NKI")
hist((rescale(x=data.vdxs[, "205225_at"], q=0.05) - 0.5) * 2,
  xlab="205225_at", breaks=20, main="ESR1 in VDX
rescaled")
hist((rescale(x=data.nkis[, "NM_000125"], q=0.05) - 0.5) * 2,
  xlab="NM_000125", breaks=20, main="ESR1 in NKI
rescaled")

describe

rorS Function to compute the rorS signature as published by Parker et al 2009

Description

This function computes signature scores and risk classifications from gene expression values following the algorithm used for the rorS signature as published by Parker et al 2009.

Usage

rorS(data, annot, do.mapping = FALSE, mapping, verbose = FALSE)

Arguments

data Matrix of gene expressions with samples in rows and probes in columns, dimensions being properly defined.
annot Matrix of annotations with at least one column named "EntrezGene.ID", dimensions being properly defined.
do.mapping TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise. Note that for Affymetrix HGU datasets, the mapping is not necessary.
mapping Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
verbose TRUE to print informative messages, FALSE otherwise.

Value

score Continuous signature scores
risk Binary risk classification, 1 being high risk and 0 being low risk.
mapping Mapping used if necessary.
probe If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.
Author(s)
Benjamin Haibe-Kains

References

Examples
```r
## load NKI dataset
data(vdxs)
## compute relapse score
rs.vdxs <- rorS(data=data.vdxs, annot=annot.vdxs, do.mapping=TRUE)
```

scmgene.robust

**Subtype Clustering Model using only ESR1, ERBB2 and AURKA genes for identification of breast cancer molecular subtypes**

Description

Usage
data(scmgene.robust)

Format
List of parameters for SCMGENE:

- `parameters`: List of parameters for the mixture of three Gaussians (ER-/HER2-, HER2+ and ER+/HER2-) that define the Subtype Clustering Model. The structure is the same than for an Mclust object.
- `cutoff.AURKA`: Cutoff for AURKA module score in order to identify ER+/HER2- High Proliferation (aka Luminal B) tumors and ER+/HER2- Low Proliferation (aka Luminal A) tumors.
- `mod`: ESR1, ERBB2 and AURKA modules.

Source
[http://clincancerres.aacrjournals.org/content/14/16/5158.abstract?ck=nck](http://clincancerres.aacrjournals.org/content/14/16/5158.abstract?ck=nck)

References
scmod1.robust

Examples

```r
data(scmgene.robust)
str(scmgene.robust, max.level=1)
```

### Description

List of parameters defining the Subtype Clustering Model as published in Desmedt et al 2008.

#### Usage

```r
data(scmod1.robust)
```

#### Format

List of parameters for SCMOD1:

- **parameters** List of parameters for the mixture of three Gaussians (ER-/HER2-, HER2+ and ER+/HER2-) that define the Subtype Clustering Model. The structure is the same than for an `Mclust` object.
- **cutoff.AURKA** Cutoff for AURKA module score in order to identify ER+/HER2- High Proliferation (aka Luminal B) tumors and ER+/HER2- Low Proliferation (aka Luminal A) tumors.
- **mod** ESR1, ERBB2 and AURKA modules.

#### Source

[http://clincancerres.aacrjournals.org/content/14/16/5158.abstract?ck=nck](http://clincancerres.aacrjournals.org/content/14/16/5158.abstract?ck=nck)

#### References


#### Examples

```r
data(scmod1.robust)
str(scmod1.robust, max.level=1)
```
scmod2.robust

Subtype Clustering Model using ESR1, ERBB2 and AURKA modules for identification of breast cancer molecular subtypes (Wirapati et al 2008)

Description

List of parameters defining the Subtype Clustering Model as published in Wirapati et al 2008.

Usage

data(scmod2.robust)

Format

List of parameters for SCMOD2:

- `parameters` List of parameters for the mixture of three Gaussians (ER-/HER2-, HER2+ and ER+/HER2-) that define the Subtype Clustering Model. The structure is the same than for an Mclust object.
- `cutoff.AURKA` Cutoff for AURKA module score in order to identify ER+/HER2- High Proliferation (aka Luminal B) tumors and ER+/HER2- Low Proliferation (aka Luminal A) tumors.
- `mod` ESR1, ERBB2 and AURKA modules.

Source

http://breast-cancer-research.com/content/10/4/R65

References


Examples

data(scmod2.robust)
str(scmod2.robust, max.level=1)
setcolclass.df  
*Function to set the class of columns in a data.frame*

**Description**

This function enables to set the class of each column in a data.frame.

**Usage**

```r
setcolclass.df(df, colclass, factor.levels)
```

**Arguments**

- `df`: data.frame for which columns’ class need to be updated.
- `colclass`: class for each column of the data.frame.
- `factor.levels`: list of levels for each factor.

**Value**

A data.frame with columns’ class and levels properly set.

**Author(s)**

Benjamin Haibe-Kains

**Examples**

```r
tt <- data.frame(matrix(NA, nrow=3, ncol=3, dimnames=list(1:3, paste("column", 1:3, sep="."))), stringsAsFactors=FALSE)
tt <- setcolclass.df(df=tt, colclass=c("numeric", "factor", "character"), factor.levels=list(NULL, c("F1", "F2", "F3"), NULL))
```

---

sig.endoPredict  
*Signature used to compute the endoPredict signature as published by Filipits et al 2011*

**Description**

List of 11 genes included in the endoPredict signature. The EntrezGene.ID allows for mapping and the mapping to affy probes is already provided.

**Usage**

```r
data(sig.endoPredict)
```

**Format**

`sig.endoPredict` is a matrix with 5 columns containing the annotations and information related to the signature itself (including a mapping to Affymetrix HGU platform).
References


Examples

data(sig.endoPredict)
head(sig.endoPredict)

---

**sig.gene70**

*Signature used to compute the 70 genes prognosis profile (GENE70) as published by van’t Veer et al. 2002*

Description

List of 70 agilent probe ids representing 56 unique genes included in the GENE70 signature. The EntrezGene.ID allows for mapping and the "average.good.prognosis.profile" values allows for signature computation.

Usage

data(sig.gene70)

Format

`sig.gene70` is a matrix with 9 columns containing the annotations and information related to the signature itself.

Source

[http://www.nature.com/nature/journal/v415/n6871/full/415530a.html](http://www.nature.com/nature/journal/v415/n6871/full/415530a.html)

References


Examples

data(sig.gene70)
head(sig.gene70)
sig.gene76

**Signature used to compute the Relapse Score (GENE76) as published in Wang et al. 2005**

**Description**

List of 76 affymetrix hgu133a probesets representing 60 unique genes included in the GENE76 signature. The EntrezGene.ID allows for mapping and the coefficient allows for signature computation.

**Usage**

data(sig.gene76)

**Format**

`sig.gene76` is a matrix with 10 columns containing the annotations and information related to the signature itself.

**Source**

http://www.thelancet.com/journals/lancet/article/PIIS0140-6736(05)17947-1/abstract

**References**


**Examples**

data(sig.gene76)
head(sig.gene76)

sig.genius

**Gene Expression progNostic Index Using Subtypes (GENIUS) as published by Haibe-Kains et al. 2010.**

**Description**

List of three gene signatures which compose the Gene Expression progNostic Index Using Subtypes (GENIUS) as published by Haibe-Kains et al. 2009. GENIUSM1, GENIUSM2 and GENIUSM3 are the ER-/HER2-, HER2+ and ER+/HER2- subtype signatures respectively.

**Usage**

data(sig.genius)
Format

sig.genius is a list a three subtype signatures.

References


Examples

data(sig.genius)
head(sig.genius)

<table>
<thead>
<tr>
<th>sig.ggi</th>
<th>Gene expression Grade Index (GGI) as published in Sotiriou et al. 2006</th>
</tr>
</thead>
</table>

Description

List of 128 affymetrix hgu133a probesets representing 97 unique genes included in the GGI signature. The "EntrezGene.ID" column allows for mapping and "grade" defines the up-regulation of the expressions either in histological grade 1 or 3.

Usage

data(sig.ggi)

Format

sig.ggi is a matrix with 9 columns containing the annotations and information related to the signature itself.

Source

http://jnci.oxfordjournals.org/cgi/content/full/98/4/262/DC1

References


Examples

data(sig.ggi)
head(sig.ggi)
**sig.oncotypedx**  
*Signature used to compute the OncotypeDX signature as published by Paik et al 2004*

**Description**
List of 21 genes included in the OncotypeDX signature. The EntrezGene.ID allows for mapping and the mapping to affy probes is already provided.

**Usage**
```r
data(sig.oncotypedx)
```

**Format**

`sig.oncotypedx` is a matrix with 5 columns containing the annotations and information related to the signature itself (including a mapping to Affymetrix HGU platform).

**References**


**Examples**

```r
data(sig.oncotypedx)
head(sig.oncotypedx)
```

---

**sig.pik3cags**  
*Gene expression Grade Index (GGI) as published in Sotiriou et al. 2006*

**Description**
List of 278 affymetrix hgu133a probesets representing 236 unique genes included in the PIK3CA-GS signature. The "EntrezGene.ID" column allows for mapping and "coefficient" refers to the direction of association with PIK3CA mutation.

**Usage**
```r
data(sig.pik3cags)
```

**Format**

`sig.pik3cags` is a matrix with 3 columns containing the annotations and information related to the signature itself.
**Description**

This function computes a signature score from a gene list (aka gene signature), i.e. a signed average as published in Sotiriou et al. 2006 and Haibe-Kains et al. 2009.

**Usage**

```
sig.score(x, data, annot, do.mapping = FALSE, mapping, size = 0, cutoff = NA, signed = TRUE, verbose = FALSE)
```

**Arguments**

- `x`: Matrix containing the gene(s) in the gene list in rows and at least three columns: "probe", "EntrezGene.ID" and "coefficient" standing for the name of the probe, the NCBI Entrez Gene id and the coefficient giving the direction and the strength of the association of each gene in the gene list.
- `data`: Matrix of gene expressions with samples in rows and probes in columns, dim-names being properly defined.
- `annot`: Matrix of annotations with at least one column named "EntrezGene.ID", dim-names being properly defined.
- `do.mapping`: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping`: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- `size`: Integer specifying the number of probes to be considered in signature computation. The probes will be sorted by absolute value of coefficients.
- `cutoff`: Only the probes with coefficient greater than `cutoff` will be considered in signature computation.
- `signed`: TRUE if only the sign of the coefficient must be considered in signature computation, FALSE otherwise.
- `verbose`: TRUE to print informative messages, FALSE otherwise.
Value

- **score**: Signature score.
- **mapping**: Mapping used if necessary.
- **probe**: If mapping is performed, this matrix contains the correspondence between the gene list (aka signature) and gene expression data.

Author(s)

Benjamin Haibe-Kains

References


Examples

```r
## load NKI data
data(nkis)
## load GGI signature
data(sig.ggi)
## make of ggi signature a gene list
ggi.gl <- cbind(sig.ggi[, c("probe", "EntrezGene.ID")], "coefficient"=ifelse(sig.ggi[, "grade"] == 1, -1, 1))
## computation of signature scores
ggi.score <- sig.score(x=ggi.gl, data=data.nkis, annot=annot.nkis, do.mapping=TRUE, signed=TRUE, verbose=TRUE)
str(ggi.score)
```

```
```

Description

List of 13 clusters of genes (and annotations) and their corresponding coefficient as an additional attribute.

Usage

data(sig.tamr13)

Format

*sig.tamr13* is a list a 13 clusters of genes with their corresponding coefficient.
References


Examples

data(sig.tamr13)
head(sig.tamr13)

---

sigOvcAngiogenic

Description

a

Usage

data(sigOvcAngiogenic)

Format

*sigOvcAngiogenic* a.

References


Examples

data(sigOvcAngiogenic)
head(sigOvcAngiogenic)

---

sigOvcCrijns

Description

a

Usage

data(sigOvcCrijns)
sigOvcSpentzos

Format

```
sigOvcCrijns a.
```

References


Examples

```
data(sigOvcCrijns)
head(sigOvcCrijns)
```

---

sigOvcSpentzos a

Description

a

Usage

```
data(sigOvcSpentzos)
```

Format

```
sigOvcSpentzos a.
```

References


Examples

```
data(sigOvcSpentzos)
head(sigOvcSpentzos)
```
Description

Usage

data(sigOvcTCGA)

Format

References

Examples

data(sigOvcTCGA)
head(sigOvcTCGA)

data(sigOvcYoshihara)
head(sigOvcYoshihara)
spearmanCI  

Function to compute the confidence interval for the Spearman correlation coefficient

Description

This function enables to compute the confidence interval for the Spearman correlation coefficient using the Fischer Z transformation

Usage

spearmanCI(x, n, alpha = 0.05)

Arguments

x  Spearman correlation coefficient rho
n  the sample size used to compute the Spearman rho
alpha  alpha level for confidence interval

Value

a vector containing the lower, upper values for the confidence interval and p-value for Spearman rho

Author(s)

Benjamin Haibe-Kains

Examples

spearmanCI(x=0.2, n=100, alpha=0.05)

ssp2003  


Description


Usage

data(ssp2003)
data(ssp2003.scale)
data(ssp2003.robust)
Format

List of parameters for SSP2003:

- **centroids**: Gene expression centroids for each subtype.
- **centroids.map**: Mapping for centroids.
- **method.cor**: Method of correlation used to compute distance to the centroids.
- **method.centroids**: Method used to compute the centroids.
- **std**: Method of standardization for gene expressions.
- **mins**: Minimum number of samples within each cluster allowed during the fitting of the model.

Details

Three versions of the model are provided, each of ones differs by the gene expressions standardization method since it has an important impact on the subtype classification:

- **ssp2003**: Use of the official centroids without scaling of the gene expressions.
- **ssp2003.scale**: Use of the official centroids with traditional scaling of the gene expressions (see `scale`).
- **ssp2003.robust**: Use of the official centroids with robust scaling of the gene expressions (see `rescale`).

The model `ssp2003.robust` has been shown to reach the best concordance with the traditional clinical parameters (ER IHC, HER2 IHC/FISH and histological grade). However the use of this model is recommended only when the dataset is representative of a global population of breast cancer patients (no sampling bias, the 5 subtypes should be present).

Source

[http://www.pnas.org/content/100/14/8418](http://www.pnas.org/content/100/14/8418)

References


Examples

data(ssp2003)
str(ssp2003)
data(ssp2003.robust)
str(ssp2003.robust)
ssp2006  

**SSP2006 classifier for identification of breast cancer molecular subtypes (Hu et al 2006)**

### Description

List of parameters defining the SSP2006 classifier for identification of breast cancer molecular subtypes (Hu et al 2006).

### Usage

```r
data(ssp2006)
data(ssp2006.scale)
data(ssp2006.robust)
```

### Format

List of parameters for SSP2006:

- `centroids`  Gene expression centroids for each subtype.
- `centroids.map`  Mapping for centroids.
- `method.cor`   Method of correlation used to compute distance to the centroids.
- `method.centroids`   Method used to compute the centroids.
- `std`    Method of standardization for gene expressions.
- `mins`   Minimum number of samples within each cluster allowed during the fitting of the model.

### Details

Three versions of the model are provided, each of ones differs by the gene expressions standardization method since it has an important impact on the subtype classification:

- **ssp2006**  Use of the official centroids without scaling of the gene expressions.
- **ssp2006.scale**  Use of the official centroids with traditional scaling of the gene expressions (see `scale`).
- **ssp2006.robust**  Use of the official centroids with robust scaling of the gene expressions (see `rescale`).

The model `ssp2006.robust` has been shown to reach the best concordance with the traditional clinical parameters (ER IHC, HER2 IHC/FISH and histological grade). However the use of this model is recommended only when the dataset is representative of a global population of breast cancer patients (no sampling bias, the 5 subtypes should be present).

### Source

[http://www.biomedcentral.com/1471-2164/7/96](http://www.biomedcentral.com/1471-2164/7/96)
st.gallen

References
Hu, Zhiyuan and Fan, Cheng and Oh, Daniel and Marron, JS and He, Xiaping and Qaqish, Bahjat and Livasy, Chad and Carey, Lisa and Reynolds, Evangeline and Dressler, Lynn and Nobel, Andrew and Parker, Joel and Ewend, Matthew and Sawyer, Lynda and Wu, Junyuan and Liu, Yudong and Nanda, Rita and Tretiakova, Maria and Orrico, Alejandra and Dreher, Donna and Palazzo, Juan and Perreard, Laurent and Nelson, Edward and Mone, Mary and Hansen, Heidi and Mullins, Michael and Quackenbush, John and Ellis, Matthew and Olopade, Olufunmilayo and Bernard, Philip and Perou, Charles (2006) "The molecular portraits of breast tumors are conserved across microarray platforms", BMC Genomics, 7(96)

Examples
data(ssp2006)
str(ssp2006)
data(ssp2006.robust)
str(ssp2006.robust)

st.gallen

Function to compute the St Gallen consensus criterion for prognosis

Description
This function computes the updated St Gallen consensus criterions as published by Goldhirsh et al 2003.

Usage
st.gallen(size, grade, node, her2.neu, age, vascular.inv, na.rm = FALSE)

Arguments
size
tumor size in cm.
grade
Histological grade, i.e. low (1), intermediate (2) and high (3) grade.
node
Nodal status (0 or 1 for no lymph node invasion and at least 1 invaded lymph node respectively).
her2.neu
Her2/neu status (0 or 1).
age
Age at diagnosis (in years).
vascular.inv
Peritumoral vascular invasion (0 or 1).
na.rm
TRUE if missing values should be removed, FALSE otherwise.

Value
Vector of risk predictions: "Good", "Intermediate", and "Poor".

Author(s)
Benjamin Haibe-Kains
References


See Also

npi

Examples

```r
## load NKI dataset
data(NKI)
## compute St Gallen predictions
st.gallen(size=demo.nkis[ ,"size"], grade=demo.nkis[ ,"grade"], 
node=demo.nkis[ ,"node"], her2.neu=sample(x=0:1, size=nrow(demo.nkis), replace=TRUE), age=demo.nkis[ ,"age"], vascular.inv=sample(x=0:1, 
size=nrow(demo.nkis), replace=TRUE), na.rm=TRUE)
```

### load NKI dataset

data(NKI)

### compute St Gallen predictions

```r
st.gallen(size=demo.nkis[, "size"], grade=demo.nkis[, "grade"], 
node=demo.nkis[, "node"], her2.neu=sample(x=0:1, size=nrow(demo.nkis), replace=TRUE), age=demo.nkis[, "age"], vascular.inv=sample(x=0:1, 
size=nrow(demo.nkis), replace=TRUE), na.rm=TRUE)
```

---

**stab.fs**

*Function to quantify stability of feature selection.*

**Description**

This function computes several indexes to quantify feature selection stability. This is usually estimated through perturbation of the original dataset by generating multiple sets of selected features.

**Usage**

```r
stab.fs(fsets, N, method = c("kuncheva", "davis"), ...)
```

**Arguments**

- `fsets`: list of sets of selected features, each set of selected features may have different size
- `N`: total number of features on which feature selection is performed
- `method`: stability index (see details section)
- `...`: additional parameters passed to stability index (penalty that is a numeric for Davis' stability index, see details section)

**Details**

Stability indices may use different parameters. In this version only the Davis index requires an additional parameter that is penalty, a numeric value used as penalty term.

Kuncheva index (kuncheva) lays in [-1, 1]. An index of -1 means no intersection between sets of selected features, +1 means that all the same features are always selected and 0 is the expected stability of a random feature selection.

Davis index (davis) lays in [0,1]. With a penalty term equal to 0, an index of 0 means no intersection between sets of selected features and +1 means that all the same features are always selected. A penalty of 1 is usually used so that a feature selection performed with no or all features has a Davis stability index equals to 0. None estimate of the expected Davis stability index of a random feature selection was published.
stab.fs.ranking

Value

A numeric that is the stability index

Author(s)

Benjamin Haibe-Kains

References


See Also

stab.fs.ranking

Examples

```r
set.seed(54321)
## 100 random selection of 50 features from a set of 10,000 features
fsets <- lapply(as.list(1:100), function(x, size=50, N=10000) {
  return(sample(1:N, size, replace=FALSE))
})
names(fsets) <- paste("fsel", 1:length(fsets), sep=".")

## Kuncheva index
stab.fs(fsets=fsets, N=10000, method="kuncheva")
## close to 0 as expected for a random feature selection

## Davis index
stab.fs(fsets=fsets, N=10000, method="davis", penalty=1)
```

stab.fs.ranking Function to quantify stability of feature ranking.

Description

This function computes several indexes to quantify feature ranking stability for several number of selected features. This is usually estimated through perturbation of the original dataset by generating multiple sets of selected features.

Usage

```r
stab.fs.ranking(fsets, sizes, N, method = c("kuncheva", "davis"), ...)
```
Arguments

fsets  list or matrix of sets of selected features (in rows), each ranking must have the same size

sizes  Number of top-ranked features for which the stability index must be computed

N     total number of features on which feature selection is performed

method  stability index (see details section)

...  additional parameters passed to stability index (penalty that is a numeric for Davis’ stability index, see details section)

Details

Stability indices may use different parameters. In this version only the Davis index requires an additional parameter that is penalty, a numeric value used as penalty term.

Kuncheva index (kuncheva) lays in [-1, 1]. An index of -1 means no intersection between sets of selected features, +1 means that all the same features are always selected and 0 is the expected stability of a random feature selection.

Davis index (davis) lays in [0,1]. With a penalty term equal to 0, an index of 0 means no intersection between sets of selected features and +1 means that all the same features are always selected. A penalty of 1 is usually used so that a feature selection performed with no or all features has a Davis stability index equals to 0. None estimate of the expected Davis stability index of a random feature selection was published.

Value

A vector of numeric that are stability indices for each size of the sets of selected features given the rankings

Author(s)

Benjamin Haibe-Kains

References


See Also

stab.fs

Examples

## 100 random selection of 50 features from a set of 10,000 features
fsets <- lapply(as.list(1:100), function(x, size=50, N=10000) {
  return(sample(1:N, size, replace=FALSE))
})
names(fsets) <- paste("fsel", 1:length(fsets), sep=".")

## Kuncheva index
stab.fs.ranking(fsets=fsets, sizes=c(1, 10, 20, 30, 40, 50), penalty=0, method=kuncheva)
strescR

N=10000, method="kuncheva")
## close to 0 as expected for a random feature selection
## Davis index
stab.fs.ranking(fsets=fsets, sizes=c(1, 10, 20, 30, 40, 50),
N=10000, method="davis", penalty=1)

---

**strescR**

Utility function to escape LaTeX special characters present in a string

### Description

This function returns a vector of strings in which LaTeX special characters are escaped, this was useful in conjunction with xtable.

### Usage

```r
strescR(strings)
```

### Arguments

- `strings`: A vector of strings to deal with.

### Value

Returns a vector of strings with escaped characters within each string.

### Author(s)

J.R. Lobry

### References

`citation("seqinr")`

### See Also

`stresc`

### Examples

```r
strescR("MISC_RNA")
strescR(c("BB_0001","BB_0002"))
```
Function to fit the Subtype Clustering Model

**Description**

This function fits the Subtype Clustering Model as published in Desmedt et al. 2008 and Wiarapat et al. 2008. This model is actually a mixture of three Gaussians with equal shape, volume and variance (see EEI model in Mclust). This model is adapted to breast cancer and uses ESR1, ERBB2 and AURKA dimensions to identify the molecular subtypes, i.e. ER-/HER2-, HER2+ and ER+/HER2- (Low and High Prolif).

**Usage**

```r
subtype.cluster(module.ESR1, module.ERBB2, module.AURKA, data, annot,
    do.mapping = FALSE, mapping, do.scale = TRUE, rescale.q = 0.05,
    model.name = "EEI", do.BIC = FALSE, plot = FALSE, filen, verbose = FALSE)
```

**Arguments**

- `module.ESR1`: Matrix containing the ESR1-related gene(s) in rows and at least three columns: "probe", "EntrezGene.ID" and "coefficient" standing for the name of the probe, the NCBI Entrez Gene id and the coefficient giving the direction and the strength of the association of each gene in the gene list.
- `module.ERBB2`: Idem for ERBB2.
- `module.AURKA`: Idem for AURKA.
- `data`: Matrix of gene expressions with samples in rows and probes in columns, dim-names being properly defined.
- `annot`: Matrix of annotations with at least one column named "EntrezGene.ID", dim-names being properly defined.
- `do.mapping`: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- `mapping`: **DEPRECATED** Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- `do.scale`: TRUE if the ESR1, ERBB2 and AURKA (module) scores must be rescaled (see `rescale`), FALSE otherwise.
- `rescale.q`: Proportion of expected outliers for rescaling the gene expressions.
- `do.BIC`: TRUE if the Bayesian Information Criterion must be computed for number of clusters ranging from 1 to 10, FALSE otherwise.
- `model.name`: Name of the model used to fit the mixture of Gaussians with the Mclust from the mclust package; default is "EEI" for fitting a mixture of Gaussians with diagonal variance, equal volume, equal shape and identical orientation.
- `plot`: TRUE if the patients and their corresponding subtypes must be plotted, FALSE otherwise.
- `filen`: Name of the csv file where the subtype clustering model must be stored.
- `verbose`: TRUE to print informative messages, FALSE otherwise.
Value

model Subtype Clustering Model (mixture of three Gaussians), like `scmgene.robust`, `scmod1.robust` and `scmod2.robust` when this function is used on expO dataset (International Genomics Consortium) with the gene modules published in the two references cited below.

BIC Bayesian Information Criterion for the Subtype Clustering Model with number of clusters ranging from 1 to 10.

subtype Subtypes identified by the Subtype Clustering Model. Subtypes can be either "ER-/HER2-", "HER2+" or "ER+/HER2-".

subtype.proba Probabilities to belong to each subtype estimated by the Subtype Clustering Model.

subtype2 Subtypes identified by the Subtype Clustering Model using AURKA to discriminate low and high proliferative tumors. Subtypes can be either "ER-/HER2-", "HER2+", "ER+/HER2- High Prolif" or "ER+/HER2- Low Prolif".

subtype.proba2 Probabilities to belong to each subtype (including discrimination between lowly and highly proliferative ER+/HER2- tumors, see subtype2) estimated by the Subtype Clustering Model.

module.scores Matrix containing ESR1, ERBB2 and AURKA module scores.

Author(s)

Benjamin Haibe-Kains

References


See Also

subtype.cluster.predict, intrinsic.cluster, intrinsic.cluster.predict, scmod1.robust, scmod2.robust

Examples

```r
## example without gene mapping
## load expO data
data(expos)
## load gene modules
data(mod1)
## fit a Subtype Clustering Model
scmod1.expos <- subtype.cluster(module.ESR1=mod1$ESR1, module.ERBB2=mod1$ERBB2,
module.AURKA=mod1$AURKA, data=data.expos, annot=annot.expos, do.mapping=FALSE,
do.scale=TRUE, plot=TRUE, verbose=TRUE)
str(scmod1.expos, max.level=1)
table(scmod1.expos$subtype2)
```
## example with gene mapping

```r
## load NKI data
data(nkis)
## load gene modules
data(mod1)
## fit a Subtype Clustering Model
scmod1.nkis <- subtype.cluster(module.ESR1=mod1$ESR1, module.ERBB2=mod1$ERBB2,
                               module.AURKA=mod1$AURKA, data=data.nkis, annot=annot.nkis, do.mapping=TRUE,
                               do.scale=TRUE, plot=TRUE, verbose=TRUE)
str(scmod1.nkis, max.level=1)
table(scmod1.nkis$subtype2)
```
**Value**

- **subtype**: Subtypes identified by the Subtype Clustering Model. Subtypes can be either "ER-/HER2-", "HER2+", or "ER+/HER2-".

- **subtype.proba**: Probabilities to belong to each subtype estimated by the Subtype Clustering Model.

- **prediction.strength**: Prediction strength for subtypes.

- **BIC**: Bayesian Information Criterion for the Subtype Clustering Model with number of clusters ranging from 1 to 10.

- **subtype2**: Subtypes identified by the Subtype Clustering Model using AURKA to discriminate low and high proliferative tumors. Subtypes can be either "ER-/HER2-", "HER2+", "ER+/HER2- High Prolif" or "ER+/HER2- Low Prolif".

- **subtype.proba2**: Probabilities to belong to each subtype (including discrimination between lowly and highly proliferative ER+/HER2- tumors, see subtype2) estimated by the Subtype Clustering Model.

- **prediction.strength2**: Prediction strength for subtypes2.

- **module.scores**: Matrix containing ESR1, ERBB2 and AURKA module scores.

- **mapping**: Mapping if necessary (list of matrices with 3 columns: probe, EntrezGene.ID and new.probe).

**Author(s)**

Benjamin Haibe-Kains

**References**


**See Also**

subtype.cluster, scmod1.robust, scmod2.robust

**Examples**

```r
## without mapping (affy hgu133a or plus2 only)
## load VDX data
data(vdxs)
## Subtype Clustering Model fitted on EXPO and applied on VDX
sbt.vdxs <- subtype.cluster.predict(sbt.model=scmgene.robust, data=data.vdxs,
anno=annot.vdxs, do.mapping=FALSE, do.prediction.strength=FALSE,
do.BIC=FALSE, plot=TRUE, verbose=TRUE)
```
## with mapping
## load NKI data
data(nkis)

## Subtype Clustering Model fitted on EXPO and applied on NKI
sbt.nkis <- subtype.cluster.predict(sbt.model=scmgene.robust, data=data.nkis, 
annot=annot.nkis, do.mapping=TRUE, do.prediction.strength=FALSE, 
do.BIC=FALSE, plot=TRUE, verbose=TRUE)
table(sbt.nkis$subtype)
table(sbt.nkis$subtype2)

tamr13  

Function to compute the risk scores of the tamoxifen resistance signature (TAMR13)

### Description
This function computes signature scores from gene expression values following the algorithm used for the Tamoxifen Resistance signature (TAMR13).

### Usage
tamr13(data, annot, do.mapping = FALSE, mapping, verbose = FALSE)

### Arguments
- **data**: Matrix of gene expressions with samples in rows and probes in columns, dim-names being properly defined.
- **annot**: Matrix of annotations with at least one column named "EntrezGene.ID", dim-names being properly defined.
- **do.mapping**: TRUE if the mapping through Entrez Gene ids must be performed (in case of ambiguities, the most variant probe is kept for each gene), FALSE otherwise.
- **mapping**: Matrix with columns "EntrezGene.ID" and "probe" used to force the mapping such that the probes are not selected based on their variance.
- **verbose**: TRUE to print informative messages, FALSE otherwise.

### Value
- **score**: Continuous signature scores
- **risk**: Binary risk classification, 1 being high risk and 0 being low risk (not implemented, the function will return NA values).

### Author(s)
Benjamin Haibe-Kains
References

See Also
gene76

Examples
```r
## load TAMR13 signature
data(sig.tamr13)
## load VDX dataset
data(vdxs)
## compute relapse score
tamr13.vdxs <- tamr13(data=data.vdxs, annot=annot.vdxs, do.mapping=FALSE)
summary(tamr13.vdxs$score)
```

---

**tbrm**  
*Function to compute Tukey’s Biweight Robust Mean*

**Description**
Computation of Tukey’s Biweight Robust Mean, a robust average that is unaffected by outliers.

**Usage**
tbrm(x, C = 9)

**Arguments**
- `x`: a numeric vector
- `C`: a constant. C is preassigned a value of 9 according to the Cook reference below but other values are possible.

**Details**
This is a one step computation that follows the Affy whitepaper below see page 22. This function is called by chron to calculate a robust mean. C determines the point at which outliers are given a weight of 0 and therefore do not contribute to the calculation of the mean. C=9 sets values roughly +/-6 standard deviations to 0. C=6 is also used in tree-ring chronology development. Cook and Kairiukstis (1990) have further details.

Retrieved from tbrm.

**Value**
A numeric mean.
Author(s)
Andy Bunn

References

See Also
chron

Examples
tbrm(rnorm(100))

<table>
<thead>
<tr>
<th>vdxs</th>
<th>Gene expression, annotations and clinical data from Wang et al. 2005 and Minn et al 2007</th>
</tr>
</thead>
</table>

Description
This dataset contains (part of) the gene expression, annotations and clinical data as published in Wang et al. 2005 and Minn et al 2007.

Usage
data(vdxs)

Format
vdxs is a dataset containing three matrices:
data.vdxs Matrix containing gene expressions as measured by Affymetrix hgu133a technology (single-channel, oligonucleotides)
annot.vdxs Matrix containing annotations of Affymetrix hgu133a microarray platform
demo.vdxs Clinical information of the breast cancer patients whose tumors were hybridized

Details
This dataset represent only partially the one published by Wang et al. 2005 and Minn et al 2007. Indeed only part of the patients (150) and gene expressions (966) are contained in data.vdxs.

Source
weighted.meanvar

References


Examples

data(vdxs)

---

### weighted.meanvar

Function to compute the weighted mean and weighted variance of `x`

---

### Description

This function allows for computing the weighted mean and weighted variance of a vector of continuous values.

#### Usage

`weighted.meanvar(x, w, na.rm = FALSE)`

#### Arguments

- `x`: an object containing the values whose weighted mean is to be computed.
- `w`: a numerical vector of weights of the same length as `x` giving the weights to use for elements of `x`.
- `na.rm`: TRUE if missing values should be removed, FALSE otherwise.

#### Details

If `w` is missing then all elements of `x` are given the same weight, otherwise the weights coerced to numeric by `as.numeric`. On the contrary of `weighted.mean` the weights are NOT normalized to sum to one. If the sum of the weights is zero or infinite, NAs will be returned.

#### Value

A numeric vector of two values that are the weighted mean and weighted variance respectively.

#### Author(s)

Benjamin Haibe-Kains

#### References

write.m.file

See Also

weighted.mean

Examples

set.seed(54321)
weighted.meanvar(x=rnorm(100) + 10, w=runif(100))

write.m.file Function to write a 'csv' file containing gene lists (aka gene signatures)

Description

This function allows for writing a 'csv' file containing gene signatures. Each gene signature is composed of at least four columns: "gene.list" is the name of the signature on the first line and empty fields below, "probes" are the probe names, "EntrezGene.ID" are the EntrezGene IDs and "coefficient" are the coefficients of each probe.

Usage

write.m.file(obj, file, ...)

Arguments

obj List of gene signatures.
file Filename of the 'csv' file.
... Additional parameters for read.csv function.

Value

None.

Author(s)

Benjamin Haibe-Kains

Examples

## load gene modules published by Demsedt et al 2009
data(mod1)
## write these gene modules in a 'csv' file
## Not run: write.m.file(obj=mod1, file="desmedt2009_genemodules.csv")
Index

*Topic **breast cancer**
  genefu-package, 3

*Topic **character**
  rename.duplicate, 50

*Topic **clustering**
  genefu-package, 3
  intrinsic.cluster, 26
  intrinsic.cluster.predict, 28
  molecular.subtyping, 33
  ps.cluster, 48
  subtype.cluster, 74
  subtype.cluster.predict, 76

*Topic **correlation**
  compare.proto.cor, 9
  compute.pairw.cor.meta, 10
  compute.pairw.cor.z, 12
  compute.proto.cor.meta, 12

*Topic **datasets**
  pam50, 45
  ssp2003, 66
  ssp2006, 68

*Topic **data**
  claudinLowData, 8
  expos, 16
  mod1, 31
  mod2, 32
  modelOvcAngiogenic, 33
  nki, 36
  scmgene.robust, 53
  scmod1.robust, 54
  scmod2.robust, 55
  sig.endoPredict, 56
  sig.gene70, 57
  sig.gene76, 58
  sig.genius, 58
  sig.ggi, 59
  sig.oncotypedx, 60
  sig.pik3cags, 60
  sig.tamr13, 62
  sigOvcAngiogenic, 63
  sigOvcCrijsns, 63
  sigOvcSpentzos, 64
  sigOvcTCGA, 65
  sigOvcYoshihara, 65
  vdxs, 80

*Topic **feature selection**
  stab.fs, 70
  stab.fs.ranking, 71

*Topic **htest**
  cordiff.dep, 14
  fuzzy.ttest, 17
  power.cor, 47
  spearmanCI, 66

*Topic **mapping**
  geneid.map, 21
  map.datasets, 29

*Topic **misc**
  tbrm, 79

*Topic **models**
  bimod, 4
  genefu-package, 3
  sig.score, 61

*Topic **prognosis**
  endoPredict, 15
  gene70, 18
  gene76, 20
  genefu-package, 3
  genius, 22
  ggi, 23
  ihc4, 24
  npi, 37
  oncotypedx, 38
  ovcAngiogenic, 39
  ovcCrijsns, 40
  ovcTCGA, 42
  ovcYoshihara, 43
  pik3cags, 46
  rorS, 52
  st.gallen, 69
  st.gallen, 69
  tamr13, 78

*Topic **stability**
  stab.fs, 70
  stab.fs.ranking, 71

*Topic **univar**
  weighted.meanvar, 81

annot.expos (expos), 16
annot.nkis (nkis), 36
annot.vdxs (vdxs), 80
bimod, 4
boxplot, 6
boxplotplus2, 6
chron, 79, 80
claudinLow, 7, 8, 9, 31, 44, 50
claudinLowData, 7, 8
collapseIDs, 9
compare.proto.cor, 9, 14
compute.pairw.cor.meta, 10, 10, 12
compute.pairw.cor.z, 12
compute.proto.cor.meta, 10–12, 12
cor, 14
cordif.dep, 14
cordiff.dep, 14
data.expos, 16
data.expos (expos), 16
data.nkis (nkis), 36
data.vdxs, 80
data.vdxs (vdxs), 80
demo.expos, 16
demo.nkis (nkis), 36
demo.vdxs (vdxs), 80
demo.mult, 16
doncolclass, 22
endoPredict, 15
expos, 16, 16
fuzzy.ttest, 17
gene70, 18
gene76, 20, 24, 46, 79
geneid.map, 21
genius, 22
ghi, 20, 23
ihc4, 24
intrinsic.cluster, 26, 28, 29, 75
intrinsic.cluster.predict, 27, 28, 35, 75
jitter, 6
map.datasets, 11–13, 29
Mclust, 5, 53–55, 74
medianCtr, 7, 31
mod1, 31, 31, 49
mod2, 32, 32, 49
modelOvcAngiogenic, 33, 33
molecular.subtyping, 33
nkis, 19, 36, 36
npi, 37, 70
oncotypedx, 38
ovcAngiogenic, 39
ovcCrijns, 40
ovcTCGA, 42
ovcYoshihara, 43
overlapSets, 44
p.adjust, 9
pam50, 27, 29, 45
pik3cags, 46
points, 6
power.cor, 47
ps.cluster, 48
read.csv, 49, 82
read.m.file, 49
readArray, 50
rename.duplicate, 50
rescale, 5, 22, 45, 51, 67, 68, 74
rors, 52
scale, 45, 51, 67, 68
scmgene.robust, 53, 75
scmod1.robust, 54, 75, 77
scmod2.robust, 55, 75, 77
setcolclass.df, 56
sig.endoPredict, 56, 56
sig.gene70, 57, 57
sig.gene76, 58, 58
sig.genius, 58, 59
sig.ggi, 59, 59
sig.oncotypedx, 60, 60
sig.pik3cags, 60, 60
sig.score, 5, 23, 61
sig.tamr13, 62, 62
sigOvcAngiogenic, 40, 63, 63
sigOvcCrijns, 41, 63, 64
sigOvcSpentzos, 64, 64
sigOvcTCGA, 42, 65, 65
sigOvcYoshihara, 44, 65, 65
spearmanCI, 66
ssp2003, 27, 29, 66
ssp2006, 27, 29, 68
st.gallen, 37, 69
stab.fs, 70, 72
stab.fs.ranking, 71, 71
stresc, 73
strescR, 73
INDEX

subtype.cluster, 27, 33, 74, 76, 77
subtype.cluster.predict, 23, 35, 75, 76

t.test, 14
tamr13, 78
tbrm, 79, 79

vdxs, 80, 80

weighted.mean, 18, 81, 82
weighted.meanvar, 81
write.m.file, 82