Package 'sigsquared'

October 18, 2017

Type Package

Title Gene signature generation for functionally validated signaling pathways

Version 1.8.0

Date 2015-01-23

Author UnJin Lee

Maintainer UnJin Lee <unjin@uchicago.edu>

Depends R (>= 3.2.0), methods

Imports Biobase, survival

Suggests RUnit, BiocGenerics

Description By leveraging statistical properties (log-rank test for survival) of patient cohorts defined by binary thresholds, poor-prognosis patients are identified by the sigsquared package via optimization over a cost function reducing type I and II error.

biocViews

License GPL version 3

NeedsCompilation no

R topics documented:

analysisPipeline *Training of thresholds*

Description

The analysisPipeline function is used to train a set of thresholds for predicting survival outcome within the context of a given signaling environment. This signaling environment is encoded in a geneSignature object.

Usage

```
analysisPipeline(dataSet, geneSig, iterPerK=2500, k=3, rand=TRUE, newjpdf=FALSE, jpdf=FALSE, nJPD
```

Arguments

dataSet	ExpressionSet object containing both expression data (exprs) and phenotypic survival data (pData)
geneSig	geneSignature object containing directions, thresholds, and gene symbols
iterPerK	integer number of optimization iterations for each k
k	integer k for k-fold cross-validation
rand	boolean determining whether the k subsets are randomly drawn (otherwise k subsets are selected ordinally)
newjpdf	boolean for generating a joint probability function for alternate smoothed cost function (not recommended)
jpdf	solnSpace object containing empirical joint probability function for alternate smoothed cost function (not recommended)
nJPDF	value determining the number of samples with which to estimate the empirical joint probability function for alternate smoothed cost function (not recommended)
disc	vector of discretation thresholds for discretized cost function
MFS	variable name for survival-time data in dataSet object
met	variable name for metastasis event data in dataSet object
optMeth	optimization method used by R function 'optim'

Details

The analysisPipeline function optimizes over a cost function designed to minize both type I and II error. There is a discretized and smoothed cost function available, however implementation of the smoothed cost function relies on sampling of the solution space. This sampling may be precomputed and implemented through the 'jpdf' argument, however overall usage of the smoothed cost function is not recommended.

Value

A geneSignature object containing newly trained thresholds

Author(s)

UnJin Lee

Examples

```
## Load in example data
data("BrCa443")

## Create initial geneSignature object
## Note it is not necessary to define thresholds at this point
gs <- setGeneSignature(g=new("geneSignature"), direct=c(-1,1,1,1,1,1,1), genes=c("RKIP", "HMGA2", "SPP1", "C

## Generate thresholds
gs <- analysisPipeline(dataSet=BrCa443, geneSig=gs, iterPerK=50, k=2, rand=FALSE)
```

`BrCa443` *Breast Cancer 443 Data Set*

Description

BrCa443 is an ExpressionSet object that contains gene expression values for 7 genes, RKIP, HMGA2, SPP1, CXCR4, MMP1, metaBACH1, and metaLET7 for 443 breast cancer patients. It also contains paired survival data for each patient in the form of survival and event data.

Usage

```
data(BrCa443)
```

Format

ExpressionSet with expression data and survival data

Source

GSE5327, GSE2034, and GSE2603

`ensembleAdjustable` *Application of geneSignature object*

Description

The ensembleAdjustable function applies a geneSignature object to a data matrix containing expression values and gene symbols or an ExpressionSet object.

Usage

```
ensembleAdjustable(dataSet, geneSig, index=F)
```

Arguments

`dataSet`	data set object, may be numeric matrix or an ExpressionSet
`geneSig`	geneSignature object containing directions, thresholds, and gene symbols
`index`	index to indicate which samples are to be subsetted, may be FALSE for no subsetting or a vector of column numbers

Value

A logical vector with length equal to the number of samples (or samples subsetted), TRUE indicating a positive, FALSE indicating a negative

Author(s)

UnJin Lee

Examples

```
require(Biobase)
## Generate test geneSignature object with 0s for thresholds
gs <- setGeneSignature(g=new("geneSignature"), direct=c(1,1,1), genes=c("A", "B", "C"), thresholds=c(0, 0, 0

## Generate randomly distributed matrix and ExpressionSet
mat <- matrix(rnorm(9, 0, 1), nrow=3)
rownames(mat) <- c("A", "B", "C")
posmat <- abs(mat)
expset <- new("ExpressionSet", exprs=mat)

## Apply geneSignature to matrices
ensembleAdjustable(mat, gs)
ensembleAdjustable(posmat, gs)

## Apply geneSignature to ExpressionSet
ensembleAdjustable(expset, gs)

## Apply geneSignature with subsetting
ensembleAdjustable(mat, gs, c(1, 3))
ensembleAdjustable(expset, gs, c(1, 3))
```

---

geneSignature-class *Class* "geneSignature"

Description

The geneSignature object contains the necessary elements defining the signaling environment on which a prognostic gene signature will be created.

Objects from the Class

Objects can be created by calls of the form `new("geneSignature", ...)`. Objects all contain 4 slots - geneSet, geneDirect, thresholds, dirMat (unused).

Slots

geneSet: Object of class "character" ~~

geneDirect: Object of class "numeric" ~~

thresholds: Object of class "numeric" ~~

dirMat: Object of class "matrix" ~~

Methods

analysisPipeline `signature(dataSet = "ExpressionSet", geneSig = "geneSignature")`: ...

ensembleAdjustable `signature(dataSet = "ExpressionSet", geneSig = "geneSignature")`: ...

ensembleAdjustable `signature(dataSet = "matrix", geneSig = "geneSignature")`: ...

getDirect `signature(g = "geneSignature")`: ...

getGenes `signature(g = "geneSignature")`: ...

getNGenes `signature(g = "geneSignature")`: ...

getThresholds `signature(g = "geneSignature")`: ...

setDirect `signature(g = "geneSignature", direct = "numeric")`: ...

setGenes `signature(g = "geneSignature")`: ...

setGeneSignature `signature(g = "geneSignature")`: ...

setThresholds `signature(g = "geneSignature")`: ...

Author(s)

UnJin lee

References

Lee U, Frankenberger C, Yun J, Bevilacqua E, Caldas C, et al. (2013) A Prognostic Gene Signature for Metastasis-Free Survival of Triple Negative Breast Cancer Patients. PLoS ONE 8(12): e82125. doi:10.1371/journal.pone.0082125

Examples

```
showClass("geneSignature")
```

`setGeneSignature` *geneSignature functions*

Description

The geneSignature object contains the necessary elements defining the signaling environment on which a prognostic gene signature will be created. This collection of functions are used to manipulate or retrieve the data slots of a given geneSignature object.

Usage

```
setGeneSignature(g, direct=NA, thresholds=c(0), genes=NA, mat=matrix())
setDirect(g, direct)
setThresholds(g, thresholds)
setGenes(g, genes)
getDirect(g)
getThresholds(g)
getGenes(g)
getNGenes(g)
```

Arguments

g	geneSignature object
direct	vector of -1s or 1s representing down- or up-regulation respectively
thresholds	vector of values containing thresholds for the geneSignature object
genes	character vector of gene names
mat	matrix of interactions between genes (unused)

Value

All setting functions return objects of class geneSignature. getDirect yields a vector of -1s or 1s, getThesholds yields a vector of theshold values, getGenes yields a character vector of gene names, getNGenes yields the number of genes in the geneSignature

Author(s)

UnJin Lee

Examples

```
## Generate and read out values of a geneSignature object
gs <- setGeneSignature(new("geneSignature"), c(1, 1), c(0, 0), c("BACH1", "RKIP"), matrix())
getDirect(gs)
getThresholds(gs)
getGenes(gs)
getNGenes(gs)
```

---

sigsquared *The sigsquared package*

Description

The sigsquared package attempts to detect the presence of alternate signaling states of an input pathway that significantly predict differential survival outcome within a mixed cohort of patients. The main goal of this package is to generate gene signatures given known signaling pathways that are predictive of differential survival outcome. The two main functions used to accomplish this goal are first, the ability to train model parameters for a given linear network model, and second, the ability to apply the model and trained parameters to transcript data.

Index