Package ‘TBSignatureProfiler’

March 28, 2024

Title  Profile RNA-Seq Data Using TB Pathway Signatures

Version  1.14.0

Description  Gene signatures of TB progression, TB disease, and other TB
disease states have been validated and published previously. This
package aggregates known signatures and provides computational tools
to enlist their usage on other datasets. The TBSignatureProfiler makes
it easy to profile RNA-Seq data using these signatures and includes
common signature profiling tools including ASSIGN, GSVA, and ssGSEA.
Original models for some gene signatures are also available. A shiny
app provides some functionality alongside for detailed command line
accessibility.

License  MIT + file LICENSE

URL  https://github.com/compbiomed/TBSignatureProfiler
     https://compbiomed.github.io/TBSignatureProfiler-docs/

BugReports  https://github.com/compbiomed/TBSignatureProfiler/issues

Depends  R (>= 4.2.0)

Imports  ASSIGN (>= 1.23.1), BiocParallel, ComplexHeatmap, DESeq2, DT,
edgeR, gdata, ggplot2, GSVA, magrittr, methods, RColorBrewer,
reshape2, rlang, ROCit, S4Vectors, singscore, stats,
SummarizedExperiment

Suggests  BiocStyle, caret, circlize, class, covr, dplyr, e1071,
glmnet, HGNChelper, impute, knitr, lintr, MASS, plyr, pROC,
randomForest, rmarkdown, shiny, spelling, sva, testthat

VignetteBuilder  knitr

biocViews  GeneExpression, DifferentialExpression

Encoding  UTF-8

Language  en-US

LazyData  TRUE

RoxygenNote  7.2.3

git_url  https://git.bioconductor.org/packages/TBSignatureProfiler
R topics documented:

- `OriginalModel_NoRetraining` ......................................................... 3
- `OriginalModel_Retraining` ......................................................... 4
- `addTBsignature` ................................................................. 5
- `bootstrapAUC` ................................................................. 6
- `Bootstrap_LOOCV_LR_AUC` ..................................................... 7
- `common_sigAnnotData` ........................................................ 8
- `compareAlgs` ................................................................. 9
- `compareBoxplots` ........................................................... 11
- `COVIDsignatures` .............................................................. 12
- `cv_glmnet_OriginalModel` ..................................................... 16
- `deseq2_norm_rle` ............................................................. 17
- `distinctColors` ............................................................... 17
- `evaluateOriginalModel` ...................................................... 18
- `knn_OriginalModel` .......................................................... 19
- `lda_OriginalModel` ............................................................ 20
- `LOOAUC_simple_multiple_noplot_one_df` .............................. 20
- `mkAssay` ................................................................. 21
- `ObtainSampleScore_OriginalModel` ............................................. 22
- `OriginalTrainingData` .......................................................... 23
- `plotQuantitative` ............................................................... 24
- `randomForest_OriginalModel` ............................................... 25
- `ref_combat_impute` ............................................................ 26
- `runTBsigProfiler` ............................................................. 27
- `sigAnnotData` ................................................................. 30
- `signatureBoxplot` .............................................................. 31
- `signatureGeneHeatmap` ......................................................... 32
- `signatureHeatmap` ............................................................. 35
- `SignatureQuantitative` ......................................................... 37
- `signatureROCplot` ............................................................ 39
- `signatureROCplot_CI` ........................................................ 40

Author  Aubrey R. Odom [aut, cre, dtm]
    (<https://orcid.org/0000-0001-7113-7598>),
    David Jenkins [aut, org] (<https://orcid.org/0000-0002-7451-4288>),
    Xutao Wang [aut],
    Yue Zhao [ctb] (<https://orcid.org/0000-0001-5257-5103>),
    Christian Love [ctb],
    W. Evan Johnson [aut]

Maintainer  Aubrey R. Odom <aodom@bu.edu>
subsetGeneSet . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
SulimanOriginalModel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
svm_OriginalModel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
tableAUC . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
TBcommon . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 45
TBsignatures . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
TBsignaturesSplit . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60
TBSPapp . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61
TB_hiv . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61
TB_indian . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62

Index 64

.OriginalModel_NoRetraining

TB gene signatures that do not require retraining.

Description
A function to obtain predicted score for TB gene signatures that do not need to be retrained.

Usage
.OriginalModel_NoRetraining(input, useAssay, geneSignaturesName, BPPARAM)

Arguments
input A SummarizedExperiment object with gene symbols as the assay row names.
useAssay A character string or an integer specifying the assay in the input. Default is the first assay in the assay list. Used for the test SummarizedExperiment object. Default is 1, indicating the first assay in the input.
geneSignaturesName A character string/vector specifying the signature of interest. If any(geneSignaturesName == "") == TRUE, it will run all available gene signatures’ original models.
BPPARAM An instance inherited from bplapply.

Details
Anderson_42 and Anderson_OD_51 used difference of sums to calculate prediction scores. Difference of sums is obtained by subtracting the sum of the expression of genes within signatures that are down-regulated from the sum of the expression of genes that are up-regulated within signatures. Kaforou_27, Kaforou_OD_44, and Kaforou_OD_53 used difference of arithmetic means to calculate prediction scores. Sweeney_OD_3 used difference of arithmetic mean to calculate prediction score.

Value
A SummarizedExperiment object with predicted scores for each sample obtained from the signature’s original model.
Description

A function to obtain predicted score for TB gene signatures that need retraining of original models.

Usage

.OriginalModel_Retraining(input, useAssay, geneSignaturesName, adj, BPPARAM)

Arguments

input
A SummarizedExperiment object with gene symbols as the assay row names.

useAssay
A character string or an integer specifying the assay in the input. Default is the first assay in the assay list. Used for the test SummarizedExperiment object. Default is 1, indicating the first assay in the input.

geneSignaturesName
A character string/vector specifying the signature of interest. If any(geneSignaturesName == "") == TRUE, it will run all available gene signatures’ original models.

adj
A small positive real number used in ComBat to solve for genes with 0 counts(rare cases). Default is 1e-3.

BPPARAM
An instance inherited from bplapply.

Details

Maertzdorf_4 and Maertzdorf_15 were trained using a random forest to distinguish patients with active TB from healthy controls.

Verhagen_10 was also trained using a random forest to distinguish samples with active TB from either latent infection or healthy controls. The random forest model was built using randomForest.

Jacobsen_3 were trained using linear discriminant analysis (LDA) to distinguish samples with active TB from latent infection status.

Sambarey_HIV_10 were also trained using LDA to distinguish samples with active TB from either latent infection, healthy control, or other disease (HIV). The LDA model was built using lda.

Berry_OD_86 and Berry_393 were trained using K-nearest neighbors (KNN) model to differentiate samples with active TB from latent infection status. The KNN model was built using knn.

Suliman_RISK_4 and Zak_RISK_16 were trained using support vector machines (SVM) to distinguish TB progressor from non-progressors. The input gene expression features for Suliman_RISK_4 used the paired ratio of GAS6/CD1C, SEPTIN4/BLK, SEPTIN4/CD1C, GAS6/BLK. The SVM model was built using svm.
addTBsignature

Value

A SummarizedExperiment object with predicted scores for each sample obtained from the signature’s original model.

Description

This function allows users to integrate new signatures into the TBSP with a function that updates the TBsignature, TBcommon, sigAnnotData and common_sigAnnotData objects. Users that wish to use this function should do so with the downloaded package as a working directory, and not as a casual package function. This function does not complete all required updates to the package for a signature to be full added; users should check the vignette "Submitting Signatures to the TBSP Package" on the [TBSP website](https://compbiomed.github.io/TBSignatureProfiler-docs/) for a walkthrough of this complete process. Also note that this function only adds one signature at a time, and must me run multiple times to add subsequent signatures.

Usage

```
addTBsignature(
  sigsymbols,  
  authname,     
  signame_common = NULL, 
  sigtype,      
  tissue_type,  
  saveobjs = FALSE,  
  views = TRUE
)
```

Arguments

- **sigsymbols** a character vector of the gene symbols that compose the signature to be added. Required.
- **authname** a character string containing the last name of the primary author of the publication where the signature was first identified. If spaces are present, omit them, and use proper capitalization. Required.
- **signame_common** a character string of the alternate name of the signature given by the publication, if it exists. If NULL, no assigned name is assumed to exist. Default is null.
- **sigtype** a character string that gives the context that the signature was developed under. Most commonly, it will distinguish TB from LTBI ("Disease"), TB from some combination of other diseases and possibly LTBI ("Disease/Other Diseases"), TB from Human Immunodeficiency Virus ("Disease/HIV"), TB from pneumonia ("Disease/Pneumonia"), or identify risk of progression to TB ("risk"), risk of TB treatment failure ("failure"), or classify treatment responses (i.e., failures from cures, "response"). Required.
tissuetype a character string that denotes whether the signature was developed using samples of either whole blood/paxgene ("whole blood") or peripheral blood mononuclear cells("PBMC"). Due to the manipulation of cells inherently required to obtain PBMCs, many scientists prefer to use only whole blood samples for analysis. Accepts "whole blood", "PBMC" or "mixed". Required.

saveobjs logical. If TRUE, the contents of the data file (TBsignatures, TBcommon, sigAnnotData, common_sigAnnotData) will be overwritten and updated to include the new signature. If FALSE, no files will be overwritten, but you can check function output for errors before writing RDS objects by setting views = TRUE. Default is saveobjs = FALSE.

views logical. If TRUE, all objects will be sent to a data view in a new window to check for errors. Default is TRUE.

Value
Either data objects TBsignatures, TBcommon, sigAnnotData, and common_sigAnnotData will be updated with the new signature and overwritten if saveobjs = FALSE, or no output will be produced except errors and messages for checking that the function runs correctly given the inputs.

Examples
# Mock example signature
TBSignatureProfiler:::addTBsignature(sigsymbols = c("GBP5", "BATF2", "GZMA"),
  authname = "Odom",
  signame_common = NULL,
  sigtype = "Disease/HIV",
  tissuetype = "PBMC",
  saveobjs = FALSE,
  views = FALSE)

bootstrapAUC

**Bootstrap the AUC and conduct T-Tests for a collection of signatures.**

Description
Run bootstrapping of the AUC and derive the p-value for a 2-sample t-test for all signatures tested on a given dataset.

Usage
bootstrapAUC(
  SE_scored,
  annotationColName,
  signatureColNames,
  num.boot = 100,
  pb.show = TRUE
)
Arguments

- **SE_scored**: a `SummarizedExperiment` object with genes as the row features and signature scores in the `colData`. There should also be a column of annotation data. Required.
- **annotationColName**: a character string giving the column name in `colData` that contains the annotation data. Required.
- **signatureColNames**: a vector of column names in `colData` that contain the signature score data. Required.
- **num.boot**: integer. The number of times to bootstrap the data. The default is 100.
- **pb.show**: logical for whether to show a progress bar while running code. The default is `TRUE`.

Value

A list of length 5 returning a vector of p-values for a 2-sample t-test, bootstrapped AUC values, an AUC value for using all scored values for all signatures specified in `signatureColNames`, and values for the lower and upper bounds of a bootstrapped AUC confidence interval using `pROC::roc()`.

Examples

```r
# Run signature profiling
choose_sigs <- list("madeupsig" = c("FCRL3", "OAS2", "IFITM3"))
prof_indian <- runTBsigProfiler(TB_indian, useAssay = "logcounts",
                                 algorithm = "ssGSEA",
                                 combineSigAndAlgorithm = TRUE,
                                 signatures = choose_sigs,
                                 parallel.sz = 1)
# Bootstrapping
booted <- bootstrapAUC(SE_scored = prof_indian, annotationColName = "label",
                       signatureColNames = names(choose_sigs), num.boot = 2)
booted
```

Bootstrap_LOOCV_LR_AUC

*Bootstrap on Leave-one-out CV with Logistic Regression.*

Description

Bootstrap on Leave-one-out CV with Logistic Regression.

Usage

```r
Bootstrap_LOOCV_LR_AUC(df, targetVec, nboot)
```
Arguments

df  
a data.frame of gene expression count data. Required.
targetVec  
a binary vector of the response variable. Should be the same number of rows as df. Required.
nboot  
an integer specifying the number of bootstrap iterations.

Value

A list of length 2 with elements

    auc  
    A vector the length of nboot with the AUC from each bootstrap iteration.
    byClass  
    A dataframe with number of rows equal to nboot. Each row contains the sensitivity, specificity, positive predictive value, negative predictive value, precision, recall, F1, prevalence, detection rate, detection prevalence and balanced accuracy for that bootstrap iteration.

---

common_sigAnnotData  
Annotation information for published TB signatures.

Description

A data.frame of annotation information for published tuberculosis signatures. This table differs from that of sigAnnotData as it refers to signatures via the name given in scientific publications, and via a consistent naming system otherwise. Currently, this table includes two variables, disease and tissue type.

Usage

    common_sigAnnotData

Format

data.frame

Details

The disease variable indicates whether the signature was developed to distinguish TB from LTBI ("Disease"), TB from some combination of other diseases and possibly LTBI ("OD"), TB from Human Immunodeficiency Virus ("HIV"), TB from pneumonia ("PNA"), or identify risk of progression to TB ("RISK"), risk of TB treatment failure ("FAIL"), or classify treatment responses (i.e., failures from cures, "RES").

The tissue type variable denotes whether the signature was developed using samples of either whole blood/paxgene or peripheral blood mononuclear cells (PBMCs). Due to the manipulation of cells inherently required to obtain PBMCs, many scientists prefer to use only whole blood samples for analysis.
Source
See ?TBcommon for reference information.

Examples
data("common_sigAnnotData")

compareAlgs
Compare scoring algorithms on a single signature via heatmap or boxplot.

Description
It may be useful to compare the results of scoring across several different scoring algorithms via a method of visualization, such as a heatmap. The compareSigs function allows the input of a SummarizedExperiment data object and conducts profiling on each signature desired, and outputting a heatmap or boxplot for each signature.

Usage
compareAlgs(input,
  signatures = NULL,
  annotationColName,
  useAssay = "counts",
  algorithm = c("GSVA", "ssGSEA", "ASSIGN", "PLAGE", "Zscore", "singscore"),
  showColumnNames = TRUE,
  showRowNames = TRUE,
  scale = FALSE,
  choose_color = c("blue", "gray95", "red"),
  colList = list(),
  show.pb = FALSE,
  parallel.sz = 0,
  output = "heatmap",
  num.boot = 100,
  column_order = NULL
)

Arguments
input an input data object of the class "SummarizedExperiment". Required.
signatures a list of signatures to run with their associated genes. This list should be in the same format as TBsignatures, included in the TBSignatureProfiler package. If signatures = NULL, the default set of signatures TBsignatures list is used. For details, run ?TBsignatures. If <2 genes in a signature are present in the sample,
that signature will not be evaluated and will not be present in the resulting SE object. The default is NULL.

annotationColName
a character string giving the column name in colData that contains the annotation data. Required.

useAssay
a character string specifying the assay to use for signature profiling when input is a SummarizedExperiment. Required only for input data of the class SummarizedExperiment. If null, the assay used will be "counts". The default is NULL.

algorithm
a vector of algorithms to run, or character string if only one is desired. The default is c("GSVA", "ssGSEA", "ASSIGN", "PLAGE", "Zscore", "singscore").

showColumnNames
logical. Setting showColumnNames = TRUE will show the column names (i.e. sample names) on the heatmap. The default is TRUE.

showRowNames
logical. Setting showColumnNames = TRUE will show the row names (i.e. signature names) on the heatmap. The default is TRUE.

scale
logical. Setting scale = TRUE scales the signature data. The default is FALSE.

colorSets
a vector of names listing the color sets in the order that they should be used in creating the heatmap. By default, this function will use the color sets in the order listed in Usage for annotation information. You may replace the default with the same collection of sets in order that you want to use them, or provide custom color sets with the colList parameter.

choose_color
a vector of color names to be interpolated for the heatmap gradient, or a colorRamp function produced by circlize::colorRamp2. The default is c("blue", "gray95", "red").

colList
a named list of named vectors specifying custom color information to pass to ComplexHeatmap::Heatmap(). The list should have as many elements as there are annotation columns, and each element name should correspond exactly with the name of each annotation column. The colors in the vector elements should be named according to the levels of the factor in that column's annotation data if the annotation is discrete, or it should be produced with circlize::colorRamp2 if the annotation is continuous. By default, ColorBrewer color sets will be used. See the the parameter colorSets for additional details.

show.pb
logical, whether warnings and other output from the profiling should be suppressed (including progress bar output). Default is FALSE.

parallel.sz
an integer identifying the number of processors to use when running the calculations in parallel for the GSVA and ssGSEA algorithms. If parallel.sz = 0, all cores are used. The default is 0.

output
a character string specifying whether the outputted plot should be a "heatmap" or "boxplot". The default is "heatmap".

num.boot
an integer indicating the number of times to bootstrap the data.

column_order
a vector of character strings indicating the order in which to manually arrange the heatmap columns. Default is NULL, such that column order is automatically determined via clustering.
Value

A heatmap or boxplot for each signature specified comparing the enumerated algorithms.

Examples

```r
compareAlgs(TB_indian, 
  signatures = TBsignatures[c("Gliddon_OD_3")], 
  annotationColName = "label", 
  algorithm = c("ssGSEA", "PLAGE"), 
  scale = TRUE, parallel.sz = 1, output = "heatmap")
```

**compareBoxplots**

Create a comparison plot of boxplots for bootstrapped AUC values.

Description

Present the results of AUC bootstrapping for a collection of scored signatures via boxplots.

Usage

```r
compareBoxplots(
  SE_scored, 
  annotationColName, 
  signatureColNames, 
  num.boot = 100, 
  name = "Boxplot Comparison of Signature AUCs", 
  pb.show = TRUE, 
  abline.col = "red", 
  fill.col = "gray79", 
  outline.col = "black", 
  rotateLabels = FALSE, 
  violinPlot = FALSE 
)
```

Arguments

- **SE_scored** a SummarizedExperiment object with genes as the row features and signature scores in the colData. There should also be a column of annotation data. Required.
- **annotationColName** a character string giving the column name in colData that contains the annotation data. Required.
- **signatureColNames** a vector of column names in the colData that contain the signature score data. Required.
- **num.boot** an integer indicating the number of times to bootstrap the data.
name a character string giving the overall title for the plot. The default is "Boxplot Comparison of Signature AUCs".

pb.show logical for whether to show a progress bar while running code. Default is TRUE.

abline.col the color to be used for the dotted line at AUC = 0.5 (the chance line). The default is "red".

fill.col the color to be used to fill the boxplots. The default is "white".

outline.col the color to be used for the boxplot outlines. The default is "black".

rotateLabels If TRUE, rotate labels. Default is FALSE.

violinPlot logical. Setting violinPlot = TRUE creates violin plots in place of boxplots. The mean and +/- 1 standard deviation are added to the violin plot interior for each signature. The default is FALSE.

Value

A plot with side-by-side boxplots of bootstrapped AUC values for each specified signature.

Examples

```r
# Run signature profiling
choose_sigs <- COVIDsignatures[c("Zak_RISK_16", "Zhao_NANO_6")]
prof_indian <- runTBsigProfiler(TB_indian[seq_len(25), ],
    useAssay = "logcounts",
    algorithm = "ssGSEA",
    signatures = choose_sigs,
    parallel.sz = 1)

# Create boxplots
compareBoxplots(prof_indian, annotationColName = "label",
    signatureColNames = names(choose_sigs), rotateLabels = TRUE)
```

COVIDsignatures A list of published/pre-print COVID-19 signatures.

Description

A set of 47 COVID-19 gene signatures from various single-cell and bulk RNA-seq publications and preprint manuscripts from early- to mid-2020. This set of signatures uses gene symbols.

Usage

COVIDsignatures

Format

list
Details

Signature names are composed of the last name of the primary author, followed by the type of sequencing data from which the signature was derived, the tissue from which the signature was derived, and ending with a reference to the cell type, infection status, or disease to which the signature belongs, as defined in the original publication.

Note that in some cases signatures will be positive identifiers of COVID-19 as positive markers of immune cell clusters are often provided for single-cell RNA-seq data; this should be taken into account when creating ROC curves and computing any AUC or disease risk estimates.

Source


COVIDsignatures


- **Wei_sc_PBMCs_T_cells**: Wei et al. 2020. "Viral Invasion and Type I Interferon Response Characterize the Immunophenotypes during COVID-19 Infection." SSRN: https://dx.doi.org/10.2139/ssrn.3555695

- **Wei_sc_PBMC_B_cells**: Wei et al. 2020. "Viral Invasion and Type I Interferon Response Characterize the Immunophenotypes during COVID-19 Infection." SSRN: https://dx.doi.org/10.2139/ssrn.3555695


**Examples**

```r
data("COVIDsignatures")
cv_glmnet_OriginalModel
```

---

**cv_glmnet_OriginalModel**

*Train original model for gene signatures Leong_24, Leong_RISK_29, Zhao_NANO_6 using lasso logistic regression.*

---

**Description**

Train original model for gene signatures Leong_24, Leong_RISK_29, Zhao_NANO_6 using lasso logistic regression.

**Usage**

```r
cv_glmnet_OriginalModel(dat_list, dat_test_sig)
```

**Arguments**

- **dat_list**: A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.
- **dat_test_sig**: A data frame contains corresponding gene sets from the input.
**deseq2_norm_rle**

**Value**

The predicted score for each sample in the test study.

---

**distinctColors**

Generate a distinct palette for coloring different clusters.

**Description**

Create a distinct palette for coloring different heatmap clusters. The function returns colors for input into `ComplexHeatmap::Heatmap()`, `signatureGeneHeatmap()` and `signatureHeatmap()`.

**Usage**

```r
distinctColors(
  n,
  hues = c("red", "cyan", "orange", "blue", "yellow", "purple", "green", "magenta"),
  saturation.range = c(0.7, 1),
  value.range = c(0.7, 1)
)
```

---

**deseq2_norm_rle**

Normalize gene expression count data.

**Description**

Normalize gene expression count data.

**Usage**

```r
deseq2_norm_rle(inputData)
```

**Arguments**

- **inputData**: a `data.frame` or `matrix` of gene expression count data. Required.

**Value**

A `data.frame` or `matrix` of normalized count data.

**Examples**

```r
## Example using the counts assay from a SummarizedExperiment
data_in <- SummarizedExperiment::assay(TB_indian, "counts")
res <- deseq2_norm_rle(data_in)
```
evaluateOriginalModel

Arguments

- **n**: an integer describing the number of colors to generate. Required.
- **hues**: a vector of character strings indicating the R colors available from the colors() function. These will be used as the base colors for the clustering scheme. Different saturations and values (i.e. darkness) will be generated for each hue. Default is `c("red", "cyan", "orange", "blue", "yellow", "purple", "green", "magenta")`.
- **saturation.range**: a numeric vector of length 2 with values between 0 and 1 giving the range of saturation. The default is `c(0.25, 1)`.
- **value.range**: a numeric vector of length 2 with values between 0 and 1 giving the range of values. The default is `c(0.5, 1)`.

Value

A vector of distinct colors that have been converted to HEX from HSV.

Examples

```r
distinctColors(10)
```

Description

This function computes prediction for multiple TB signatures based on their training models/methods. To avoid naming issues, the gene names for both training data and input gene sets have been updated using the `checkGeneSymbols` function. TB signatures with available original models are: Anderson_42, Anderson_OD_51, Kaforou_27, Kaforou_OD_44, Kaforou_OD_53, Sweeney_OD_3, Maertzdorf_4, Verhagen_10, Jacobsen_3, Sambarey_HIV_10, Leong_24, Berry_OD_86, Berry_393, Bloom_OD_144, Suliman_RISK_4, Zak_RISK_16, Leong_RISK_29, and Zhao_NANO_6. The predicted score for each signature has been stored in the column data section of the input SummarizedExperiment study.

Usage

```r
evaluateOriginalModel(
  input,
  geneSignaturesName,
  useAssay = 1,
  adj = 0.001,
  BPPARAM = BiocParallel::SerialParam(progressbar = TRUE)
)
```
**Arguments**

- **input** A SummarizedExperiment object with gene symbols as the assay row names.
- **geneSignaturesName** A character string/vector specifying the signature of interest. If any(geneSignaturesName == "") == TRUE, it will run all available gene signatures’ original models.
- **useAssay** A character string or an integer specifying the assay in the input. Default is the first assay in the assay list. Used for the test SummarizedExperiment object. Default is 1, indicating the first assay in the input.
- **adj** A small positive real number used in ComBat to solve for genes with 0 counts (rare cases). Default is 1e-3.
- **BPPARAM** An instance inherited from bplapply.

**Value**

A SummarizedExperiment object with predicted scores for each sample obtained from the signature’s original model.

**Examples**

```r
re <- evaluateOriginalModel(input = TB_indian, geneSignaturesName = c("Anderson_42"), useAssay = "counts")
re$Anderson_42_OriginalModel
```

---

**knn_OriginalModel**

Train original model for gene signatures Berry_393 and Berry_OD_86.

**Description**

Train original model for gene signatures Berry_393 and Berry_OD_86.

**Usage**

```r
knn_OriginalModel(dat_list, dat_test_sig)
```

**Arguments**

- **dat_list** A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.
- **dat_test_sig** A data frame contains corresponding gene sets from the input.

**Value**

The predicted score for each sample in the test study.
**lda_OriginalModel**

Train original model for gene signatures Jacobsen_3 and Sambarey_HIV_10.

**Usage**

```r
lda_OriginalModel(dat_list, dat_test_sig)
```

**Arguments**

- `dat_list`: A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.
- `dat_test_sig`: A data frame contains corresponding gene sets from the input.

**Value**

The predicted score for each sample in the test study.

---

**LOOAUC_simple_multiple_noplot_one_df**

Perform Leave-one-out CV with Logistic Regression.

**Description**

Perform Leave-one-out CV with Logistic Regression.

**Usage**

```r
LOOAUC_simple_multiple_noplot_one_df(df, targetVec)
```

**Arguments**

- `targetVec`: A binary vector of the response variable. Should be the same number of samples as in `df`. Required.
mkAssay

Value
A list of length 3 with elements

- **auc**: The AUC from the LOOCV procedure.
- **byClass**: A vector containing the sensitivity, specificity, positive predictive value, negative predictive value, precision, recall, F1, prevalence, detection rate, detection prevalence and balanced accuracy.
- **prob**: A vector of the test prediction probabilities.

Description
Given a SummarizedExperiment input with a counts or CPM assay, this function creates additional assays for by computing the CPM, log, or both of the input assay to be used in further analysis.

Usage
```r
mkAssay(
  SE_obj,
  input_name = "counts",
  output_name = NULL,
  log = FALSE,
  counts_to_CPM = TRUE,
  prior_counts = 3
)
```

Arguments
- **SE_obj**: a SummarizedExperiment object containing count or CPM data. Required.
- **input_name**: a character string specifying the name of the assay to be referenced for creating additional assays. Default is "counts".
- **output_name**: a character string to use in place of the input_name. If NULL, then input_name will be substituted. Default is NULL. See Return details for how names are altered.
- **log**: logical. Indicate whether an assay returned should be the log of whichever assay is specified in "output_name". If counts_to_CPM = TRUE as well, then a log CPM assay will also be created. Default is FALSE.
- **counts_to_CPM**: logical. This argument only applies if the input_type is a counts assay. If TRUE, then the output assays will include a normalized CPM assay. If log = TRUE as well, then a log CPM assay will also be created. Default is TRUE.
- **prior_counts**: a small integer specifying the average count to be added to each observation to avoid taking the log of zero. Used only if log = TRUE. The default is 3.
ObtainSampleScore_OriginalModel

Value
This function returns a SummarizedExperiment object with up to 3 additional assay types attached to the original inputted object.

output_name_cpm  
Counts per million
log_output_name_cpm  
Log counts per million
log_output_name  
Log of original input assay.

Author(s)
Aubrey Odom-Mabey

Examples

# Create a log assay of the original assay input
# TB_hiv dataset already has counts data
log_only <- mkAssay(TB_hiv, log = TRUE, counts_to_CPM = FALSE)
log_only

# Create a CPM assay
CPM_only <- mkAssay(TB_hiv)
CPM_only

# Create a logCPM, logcounts, and CPM assay
all_assays <- mkAssay(TB_hiv, log = TRUE)
all_assays
ObtainDiagnosis, annotationColName, FUN, adj
)

Arguments

theObject_train
A SummarizedExperiment object that has been pre-stored in the data file: OriginalTrainingData.

useAssay
A character string or an integer specifying the assay in the input. Used for the test SummarizedExperiment object. Default is 1, indicating the first assay in the test SummarizedExperiment object.

gene_set
A character vector that includes gene symbols for selected gene signature.

input
A SummarizedExperiment object with gene symbols as the assay row names.

SigName
Optional. A character string that indicates the name for gene_set. SigName is used to provide information when gene signatures were missing in the test data.

obtainDiagnosis
Boolean. Used to create training data if TRUE. Default is FALSE

annotationColName
A character string specifying the column name of disease status. Only used when creating training data. Default is NULL.

FUN
A character string specifying the function name of the corresponding signature’s original model.

adj
A small real number used in combat to solve for genes with 0 counts in rare cases. Not required for most of cases.

Value

The predicted score for each sample in the test study using corresponding gene signature’s original model.

OriginalTrainingData

Discovery datasets for corresponding gene signatures.

Description

Discovery datasets for corresponding gene signatures.

Usage

OriginalTrainingData
**plotQuantitative**

**Format**

list

**Source**

See ?TBsignatures for reference information.

**Examples**

```r
data("OriginalTrainingData")
```

---

**Description**

This function takes as input a data.frame with genetic expression count data, and uses a bootstrapped leave-one-out cross validation procedure with logistic regression to allow for numeric and graphical comparison across any number of genetic signatures. It creates a boxplot of bootstrapped AUC values.

**Usage**

```r
plotQuantitative(
  df.input, 
  targetVec.num, 
  signature.list = NULL, 
  signature.name.vec = NULL, 
  num.boot = 100, 
  pb.show = TRUE, 
  name = "Signature Evaluation: Bootstrapped AUCs", 
  fill.col = "white", 
  outline.col = "black", 
  abline.col = "red", 
  rotateLabels = FALSE
)
```

**Arguments**

- `df.input` a data.frame of gene expression count data. Required.
- `targetVec.num` a numeric binary vector of the response variable. The vector should be the same number of rows as df. Required.
- `signature.list` a list of signatures to run with their associated genes. This list should be in the same format as TBsignatures, included in the TBSignatureProfiler package. If `signature.list = NULL`, the default set of signatures TBsignatures list is used. For details, run ?TBsignatures.
signature.name.vec
A vector specifying the names of the signatures to be compared. This should be the same length as signature.list. If signature.name.vec = NULL, the default set of signatures TBsignatures list is used.

num.boot
an integer specifying the number of bootstrap iterations.

pb.show
logical. If TRUE then a progress bar for the bootstrapping procedure will be displayed as output. The default is TRUE.

name
a character string giving a name for the outputted boxplot of bootstrapped AUCs. The default is "Signature Evaluation: Bootstrapped AUCs".

fill.col
the color to be used to fill the boxplots. The default is "white".

outline.col
the color to be used for the boxplot outlines. The default is "black".

abline.col
the color to be used for the dotted line at AUC = 0.5 (the chance line). The default is "red".

rotateLabels
logical. If TRUE, the x-axis labels will be rotated. The default is FALSE.

Value
a boxplot comparing the bootstrapped AUCs of inputted signatures

Examples

inputTest <- matrix(rnorm(1000), 100, 20,
  dimnames = list(paste0("gene", seq.int(1, 100)),
                 paste0("sample", seq.int(1, 20))))

inputTest <- as.data.frame(inputTest)
targetVec <- sample(c(0,1), replace = TRUE, size = 20)
signature.list <- list(sig1 = c("gene1", "gene2", "gene3"),
                        sig2 = c("gene4", "gene5", "gene6"))
signature.name.vec <- c("sig1", "sig2")
num.boot <- 5
plotQuantitative(inputTest, targetVec.num = targetVec,
                 signature.list = signature.list,
                 signature.name.vec = signature.name.vec,
                 num.boot = num.boot, rotateLabels = FALSE)

randomForest_OriginalModel

Train original model for gene signatures Maertzdorf_4, Maertzdorf_15, Verhagen_10, and LauxdaCosta_OD_3.

Description

Train original model for gene signatures Maertzdorf_4, Maertzdorf_15, Verhagen_10, and LauxdaCosta_OD_3.
Usage

```r
randomForest_OriginalModel(dat_list, dat_test.sig)
```

**Arguments**

- `dat_list` A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.
- `dat_test.sig` A data frame contains corresponding gene sets from the input.

**Value**

The predicted score for each sample in the test study.

---

**Description**

A function used to perform reference batch correction and imputation in the testing data for gene signatures that require retraining of the model. We used the k-nearest neighbors to impute the expression values for missing gene(s). The imputation operation is achieved using `impute.knn`. Since the computational time for the imputation step can be excessive for large number of missing genes. We made some constrains to prevent the overflow of imputation operation. The evaluation will not run if more than `geneMax*100%` percent of the genes are not found for the corresponding gene signature in the input study. By default `geneMax = 0.8`, so the evaluation will not run if more than 80% of the genes are missing when matching the input study to the reference data.

Usage

```r
refCombatImpute(
  theObject_train,
  useAssay, gene_set, input, SigName, adj, geneMax = 0.8
)
```

**Arguments**

- `theObject_train` A SummarizedExperiment object that has been pre-stored in the data file: OriginalTrainingData.
- `useAssay` A character string or an integer specifying the assay in the `input`. Used for the test SummarizedExperiment object. Default is 1, indicating the first assay in the test SummarizedExperiment object.
runTBsigProfiler

Description

Using some subset of the signatures listed in TBsignatures and specified scoring algorithms, this function runs gene signature profiling on an input gene expression dataset. It allows for scores to be computed for these signatures which can be compared using various visualization tools also provided in the TBSignatureProfiler package.

Usage

runTBsigProfiler(
  input, 
  useAssay = NULL, 
  signatures = NULL, 
  algorithm = c("GSVA", "ssGSEA", "ASSIGN", "PLAGE", "Zscore", "singscore"), 
  combineSigAndAlgorithm = FALSE, 
  assignDir = NULL, 
  outputFormat = NULL, 
  parallel.sz = 0, 
  ASSIGNiter = 1e+05, 
  ASSIGNburnin = 50000, 
  ssgsea_norm = TRUE, 
  update_genes = TRUE 
)

Arguments

input an input data object of the class SummarizedExperiment, data.frame, or matrix containing gene expression data. Required.

gene_set A character vector that includes gene symbols for selected gene signature.

input A SummarizedExperiment object with gene symbols as the assay row names.

SigName Optional. A character string that indicates the name for gene_set. SigName is used to provide information when gene signatures were missing in the test data.

adj A small real number used in combat to solve for genes with 0 counts in rare cases. Not required for most of cases.

geneMax A real number between 0 and 1. This is used to detect the maximum percent missing genes allowed in the evaluated signatures. See impute.knn for details. The default value is 0.8.

Value

Gene set subset

runTBsigProfiler Run TB gene signature profiling.
runTBsigProfiler

useAssay a character string specifying the assay to use for signature profiling when input is a SummarizedExperiment. Required only for input data of the class SummarizedExperiment. If null, the assay used will be "counts". The default is NULL.

signatures a list of signatures to run with their associated genes. This list should be in the same format as TBsignatures, included in the TBSignatureProfiler package. If signatures = NULL, the default set of signatures TBsignatures list is used. For details, run ?TBsignatures. If <2 genes in a signature are present in the sample, that signature will not be evaluated and will not be present in the resulting SE object. The default is NULL.

algorithm a vector of algorithms to run, or character string if only one is desired. The default is c("GSVA", "ssGSEA", "ASSIGN", "PLAGE", "Zscore", "singscore").

combineSigAndAlgorithm logical, if TRUE, output row names will be of the form <algorithm>_<signature>. It must be set to codeTRUE if the outputFormat will be a SummarizedExperiment and length(algorithm) > 1. It will always be FALSE if only one algorithm is selected. If FALSE, there will be a column named algorithm' that lists which algorithm is used, and a column named 'pathway' that lists the signature profiled. If NULL, and one algorithm was used, the algorithm will not be listed. The default is FALSE.

assignDir a character string naming a directory to save intermediate ASSIGN results if algorithm specifies "ASSIGN". The default is NULL, in which case intermediate results will not be saved.

outputFormat a character string specifying the output data format. Possible values are "SummarizedExperiment", "matrix", or "data.frame". The default is to return the same type as the input object.

parallel.sz an integer identifying the number of processors to use when running the calculations in parallel for the GSVA and ssGSEA algorithms. If parallel.sz = 0, all cores are used. The default is 0.

ASSIGNiter an integer indicating the number of iterations to use in the MCMC for the ASSIGN algorithm. The default is 100,000.

ASSIGNburnin an integer indicating the number of burn-in iterations to use in the MCMC for the ASSIGN algorithm. These iterations are discarded when computing the posterior means of the model parameters. The default is 50,000.

ssgsea_norm logical, passed to GSVA::gsva(). When parameter algorithm = "ssgsea", the profiler runs the SSGSEA method from Barbie et al. (2009) normalizing the scores by the absolute difference between the minimum and the maximum, as described in their paper. When ssgsea.norm = FALSE, this last normalization step is skipped. The default is TRUE.

update_genes logical, denotes whether gene names from signatures and the rownames of input should be checked for accuracy using HGNChelper::checkGeneSymbols(). The mapping assumes genes are from humans and will keep unmapped genes as the original input gene name. Default is TRUE.

Value

A SummarizedExperiment object, data.frame, or matrix of signature profiling results. The returned object will be of the format specified in outputFormat. If input is a SummarizedExperiment
and outputFormat = "SummarizedExperiment", then the output will retain any input information stored in the input colData. In general, if outputFormat = "SummarizedExperiment" then columns in the colData will include the scores for each desired signature with samples on the rows. If input is a data.frame or matrix, then the returned object will have signatures on the rows and samples on the columns.

**Source**

Profiling for the Z-Score, PLAGE, GSVA, ssGSEA algorithms are all conducted with the Bioconductor GSVA package. Profiling for the singscore algorithm is conducted with the Bioconductor singscore package.

**References**


**Examples**

```r
## Using a data.frame input/output
# Create some toy data to test Zak_RISK_16 signature, using 5 samples with low
# expression & five samples with high expression of the signatures genes.
df_testdata <- as.data.frame(rbind(matrix(c(rnorm(80), rnorm(80) + 5), 16, 10,
dimnames = list(TBsignatures$Zak_RISK_16,
paste0("sample", seq_len(10)))),
matrix(rnorm(1000), 100, 10,
dimnames = list(paste0("gene", seq_len(100)),
paste0("sample", seq_len(10))))))
res <- runTBsigProfiler(input = df_testdata,
signatures = TBsignatures$"Zak_RISK_16",
algorithm = c("GSVA", "ssGSEA"),
combineSigAndAlgorithm = FALSE,
parallel.sz = 1)
subset(res, res$pathway == "Zak_RISK_16")
```

```r
## Using a SummarizedExperiment input/output
# The TB_indian SummarizedExperiment data is included in the package.
GSVA_res <- runTBsigProfiler(input = TB_indian,
```
useAssay = "logcounts",
signatures = TBsignatures["Zak_RISK_16"],
algorithm = c("GSVA"),
combineSigAndAlgorithm = FALSE,
parallel.sz = 1)

GSVA_res$Zak_RISK_16

## sigAnnotData

**Annotation information for published TB signatures.**

### Description

A data frame of annotation information for published tuberculosis signatures. Currently, this table includes two variables, `disease` and `tissue type`.

### Usage

`sigAnnotData`

### Format

data.frame

### Details

The `disease` variable indicates whether the signature was developed to distinguish TB from LTBI ("Disease"), TB from some combination of other diseases and possibly LTBI ("OD"), TB from Human Immunodeficiency Virus ("HIV"), TB from pneumonia ("PNA"), or identify risk of progression to TB ("RISK"), risk of TB treatment failure ("FAIL"), or classify treatment responses (i.e., failures from cures, "RES").

The `tissue type` variable denotes whether the signature was developed using samples of either whole blood/paxgene or peripheral blood mononuclear cells (PBMCs). Due to the manipulation of cells inherently required to obtain PBMCs, many scientists prefer to use only whole blood samples for analysis.

### Source

See `?TBsignatures` for reference information.

### Examples

data("sigAnnotData")
signatureBoxplot

Plot a boxplot of signature genes.

Description

Plot a boxplot of signature genes.

Usage

signatureBoxplot(
  inputData,
  annotationData,
  signatureColNames,
  annotationColName,
  name = "Signatures",
  scale = FALSE,
  violinPlot = FALSE,
  includePoints = TRUE,
  notch = FALSE,
  rotateLabels = FALSE,
  nrow = NULL,
  ncol = NULL,
  fill_colors = NULL
)

Arguments

inputData an input data object. It should either be of the class SummarizedExperiment and contain the profiled signature data and annotation data as columns in the colData, or alternatively be of the classes data.frame or matrix and contain only the gene expression data. Required.

annotationData a data.frame or matrix of annotation data, with one column. Only required if inputData is a data.frame or matrix of signature data.

signatureColNames a vector of the column names in colData that contain the signature data. Only required if inputData is a SummarizedExperiment object.

annotationColName a character string naming the column name in the colData that contains the annotation data to be used in making the boxplot. Only required if inputData is a SummarizedExperiment object.

name a character string giving the title of the boxplot. The default is "Signatures".

scale logical. Setting scale = TRUE scales the signature data. The default is FALSE.

violinPlot logical. Setting violinPlot = TRUE creates violin plots in place of boxplots. The default is FALSE.

includePoints logical. If TRUE, points will be included over the boxplots. The default is TRUE.
signatureGeneHeatmap

notch logical. Notches are used to compare groups; if the notches of two boxes do not overlap, this suggests that the medians are significantly different. If TRUE, the boxplot will be notched. The default is FALSE.

rotateLabels logical. If TRUE, the x-axis labels will be rotated. The default is FALSE.
nrow integer giving the number of rows in the resulting array.
ncol integer giving the number of columns in the resulting array.
fill_colors a vector of color names to be used as the fill colors for the boxplot. If NULL, colors will be supplied via RColorBrewer. The default is fill_colors = NULL.

Value
A ggplot2 boxplot of the signature data using the provided annotation information.

Examples
library(SummarizedExperiment)

# Generate some artificial data that shows a difference in Zak_RISK_16
mat_testdata <- rbind(matrix(c(rnorm(80), rnorm(80) + 5), 16, 10,
          dimnames = list(TBsignatures$Zak_RISK_16,
          paste0("sample", seq_len(10)))),
          matrix(rnorm(1000), 100, 10,
          dimnames = list(paste0("gene", seq_len(100)),
          paste0("sample", seq_len(10)))))

# Create a SummarizedExperiment object that contains the data
testdataSE <- SummarizedExperiment(assays = SimpleList(data = mat_testdata),
colData = DataFrame(sample =
c(rep("down", 5),
  rep("up", 5))))

# Run profiler using GSVA and ssGSEA on Zak_RISK_16 signature
res <- runTBsigProfiler(testdataSE, useAssay = "data",
signatures = TBsignatures["Zak_RISK_16"],
algorithm = c("GSVA", "ssGSEA"), parallel.sz = 1,
combineSigAndAlgorithm = TRUE)
signatureBoxplot(res, signatureColNames = c("GSVA_Zak_RISK_16",
  "ssGSEA_Zak_RISK_16"),
  annotationColName = "sample", name = "Zak_RISK_16 Signature")

signatureGeneHeatmap

Plot a heatmap of a single signature score with individual gene expression levels.

Description
This function takes the profiled gene expression data for a single signature and creates a heatmap based on the expression scores.
signatureGeneHeatmap

Usage

signatureGeneHeatmap(
  inputData,
  useAssay,
  sigGenes,
  name = "Signature",
  signatureColNames = NULL,
  annotationColNames = NULL,
  scale = TRUE,
  showColumnNames = TRUE,
  showRowNames = TRUE,
  colList = list(),
  choose_color = c("blue", "gray95", "red"),
  column_order = NULL,
  ...
)

Arguments

inputData a SummarizedExperiment object containing the profiled signature data and annotation data as columns in the colData. Required.

useAssay a character string specifying the assay to use for the gene expression data. Required.

sigGenes a vector identifying the genes in the signature to use in the heatmap. For inbuilt signatures, you can use TBsignatures (e.g., TBsignatures[['ACS_COR']]). Required.

name a character string with the plot title of the heatmap. The default is "Signatures".

signatureColNames a vector of the column names in the colData that contain the signature data. Required.

annotationColNames a vector of the column names in the colData that contain the annotation data. If NULL, no annotation bar besides those of the scoring algorithms will be drawn on the heatmap. The default is NULL.

scale logical. Setting scale = TRUE scales the signature data. The default is TRUE.

showColumnNames logical. Setting showColumnNames = TRUE will show the column names (i.e. sample names) on the heatmap. The default is TRUE.

showRowNames logical. Setting showRowNames = TRUE will show the row names (i.e. signature names) on the heatmap. The default is TRUE.

colList a named list of named vectors specifying custom color information to pass to ComplexHeatmap::Heatmap(). The list should have as many elements as there are annotation columns and gene signatures (i.e. sigGenes), and each
element name should correspond exactly with the name of each annotation column/signature. The colors in the vector elements should be named according to the levels of the factor in that column’s annotation data if the annotation is discrete, or it should be produced with circlize::colorRamp2 if the annotation/gene is continuous. By default, ColorBrewer color sets will be used. See the parameter colorSets for additional details.

**colorSets**
- a vector of names listing the color sets in the order that they should be used in creating the heatmap. By default, this function will use the color sets in the order listed in Usage for annotation information. You may replace the default with the same collection of sets in order that you want to use them, or provide custom color sets with the colList parameter.

**choose_color**
- a vector of color names to be interpolated for the heatmap gradient, or a colorRamp function produced by circlize::colorRamp2. The default is c("blue", "gray95", "red").

**column_order**
- a vector of character strings indicating the order in which to manually arrange the heatmap columns. Default is NULL, such that column order is automatically determined via clustering.

**Value**
- A ComplexHeatmap plot.

**Examples**

```r
library(SummarizedExperiment)
# Generate some artificial data that shows a difference in Zak_RISK_16
mat_testdata <- rbind(matrix(c(rnorm(80), rnorm(80) + 5), 16, 10,
                            dimnames = list(TBsignatures$Zak_RISK_16,
                                            paste0("sample", seq_len(10)))),
                        matrix(rnorm(1000), 100, 10,
                                dimnames = list(paste0("gene", seq_len(100)),
                                                paste0("sample", seq_len(10)))))
# Create a SummarizedExperiment object that contains the data
testdataSE <- SummarizedExperiment(assays = SimpleList(data = mat_testdata),
                                 colData = DataFrame(sample =
                                               c(rep("down", 5),
                                                rep("up", 5))))
# Run profiler using GSVA and ssGSEA on Zak_RISK_16
res <- runTBsigProfiler(testdataSE, useAssay = "data",
                        signatures = TBsignatures["Zak_RISK_16"],
                        algorithm = c("GSVA", "ssGSEA"), parallel.sz = 1,
                        combineSigAndAlgorithm = TRUE)
# Plot a heatmap of signature genes and pathway predictions
signatureGeneHeatmap(res, useAssay = "data",
                      sigGenes = TBsignatures["Zak_RISK_16"],
                      signatureColNames = c("GSVA_Zak_RISK_16",
                                            "ssGSEA_Zak_RISK_16"))
```

signatureHeatmap

Plot a heatmap of signature scores.

Description

This function takes a dataset of scored gene expression data as an input and returns a ComplexHeatmap plot for visual comparison of signature performance. The function takes arguments listed here as well as any others to be passed on to ComplexHeatmap::Heatmap().

Usage

signatureHeatmap(
  inputData,
  annotationData = NULL,
  name = "Signatures",
  signatureColNames,
  annotationColNames = NULL,
  colList = list(),
  scale = FALSE,
  showColumnNames = TRUE,
  showRowNames = TRUE,
  choose_color = c("blue", "gray95", "red"),
  split_heatmap = "none",
  annotationSignature = sigAnnotData,
  column_order = NULL,
  ...
)

Arguments

inputData an input data object. It should either be of the class SummarizedExperiment and contain the profiled signature data and annotation data as columns in the colData, or alternatively be of the classes data.frame or matrix and contain only the gene expression data. Required.

annotationData a data.frame or matrix of annotation data, with one column. Only required if inputData is a data.frame or matrix of signature data. The row names must equal those of the inputData column names. Default is NULL.

name a character string with the plot title of the heatmap. The default is "Signatures".

signatureColNames a vector of the column names in colData that contain the signature data. Only required if inputData is a SummarizedExperiment object.
**signatureHeatmap**

- **annotationColNames**: a vector of the column names in colData that contain the annotation data. Only required if inputData is a SummarizedExperiment. Default is NULL.

- **colList**: a named list of named vectors specifying custom color information to pass to ComplexHeatmap::Heatmap(). The list should have as many elements as there are annotation columns, and each element name should correspond exactly with the name of each annotation column. The colors in the vector elements should be named according to the levels of the factor in that column's annotation data if the annotation is discrete, or it should be produced with circlize::colorRamp2 if the annotation is continuous. By default, ColorBrewer color sets will be used. See the the parameter colorSets for additional details.

- **scale**: logical. Setting scale = TRUE scales the signature data. The default is FALSE.

- **showColumnNames**: logical. Setting showColumnNames = TRUE will show the column names (i.e. sample names) on the heatmap. The default is TRUE.

- **showRowNames**: logical. Setting showColumnNames = TRUE will show the row names (i.e. signature names) on the heatmap. The default is TRUE.

- **colorSets**: a vector of names listing the color sets in the order that they should be used in creating the heatmap. By default, this function will use the color sets in the order listed in Usage for annotation information. You may replace the default with the same collection of sets in order that you want to use them, or provide custom color sets with the colList parameter.

- **choose_color**: a vector of color names to be interpolated for the heatmap gradient, or a colorRamp function produced by circlize::colorRamp2. The default is c("blue", "gray95", "red").

- **split_heatmap**: a character string either giving the column title of annotationSignature containing annotation data for which to split the heatmap rows (i.e., signatures), or "none" if no split is desired. To split based on the type of signature, set split_heatmap = "disease". The default is "none".

- **annotationSignature**: a data.frame or matrix with information to be used in splitting the heatmap. The first column should signature names. The column of annotation information should be specified in split_heatmap. Other columns will be ignored. The default is sigAnnotData.

- **column_order**: a vector of character strings indicating the order in which to manually arrange the heatmap columns. Default is NULL, such that column order is automatically determined via clustering.

- **...**: Additional arguments to be passed to ComplexHeatmap::Heatmap().

**Details**

If both annotationData = NULL and annotationColNames = NULL, no annotation bar will be drawn on the heatmap.

**Value**

A ComplexHeatmap plot.
Examples

```r
library(SummarizedExperiment)
# Generate some artificial data that shows a difference in Zak_RISK_16
mat_testdata <- rbind(matrix(c(rnorm(80), rnorm(80) + 5), 16, 10,
    dimnames = list(TBsignatures$Zak_RISK_16,
        paste0("sample", seq_len(10))))),
    matrix(rnorm(1000), 100, 10,
        dimnames = list(paste0("gene", seq_len(100)),
        paste0("sample", seq_len(10)))))
# Create a SummarizedExperiment object that contains the data
testdataSE <- SummarizedExperiment(assays = SimpleList(data = mat_testdata),
    colData = DataFrame(sample =
        c(rep("down", 5),
        rep("up", 5))))
res <- runTBsigProfiler(testdataSE, useAssay = "data",
    signatures = TBsignatures["Zak_RISK_16"],
    algorithm = c("GSVA", "ssGSEA"), parallel.sz = 1,
    combineSigAndAlgorithm = TRUE)
signatureHeatmap(res, signatureColNames = c("GSVA_Zak_RISK_16",
    "ssGSEA_Zak_RISK_16"),
    annotationColNames = "sample", scale = TRUE,
    showColumnNames = FALSE, split_heatmap = "none")
# Example using custom colors for the annotation information
color2 <- stats::setNames(c("purple", "black"), c("down", "up"))
color.list <- list("sample" = color2)
signatureHeatmap(res, signatureColNames = c("GSVA_Zak_RISK_16",
    "ssGSEA_Zak_RISK_16"),
    annotationColNames = "sample", scale = TRUE,
    showColumnNames = FALSE, colList = color.list, split_heatmap = "none")
```

Description

This function takes as input a data.frame with genetic expression count data, and uses a bootstrapped leave-one-out cross validation procedure with logistic regression to allow for numeric and graphical comparison across any number of genetic signatures.

Usage

```r
SignatureQuantitative(
    df.input,
    targetVec.num,
    signature.list = NULL,
```

SignatureQuantitative  *Use logistic regression and bootstrap LOOCV to evaluate signatures.*
signature.name.vec = NULL,
num.boot = 100,
pb.show = TRUE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df.input</td>
<td>a data.frame of gene expression count data. Required.</td>
</tr>
<tr>
<td>targetVec.num</td>
<td>a numeric binary vector of the response variable. The vector should be the same number of rows as df. Required.</td>
</tr>
<tr>
<td>signature.list</td>
<td>a list of signatures to run with their associated genes. This list should be in the same format as TBsignatures, included in the TBSignatureProfiler package.</td>
</tr>
<tr>
<td>signature.name.vec</td>
<td>A vector specifying the names of the signatures to be compared. This should be the same length as signature.list. If signature.name.vec = NULL, the default set of signatures TBsignatures list is used.</td>
</tr>
<tr>
<td>num.boot</td>
<td>an integer specifying the number of bootstrap iterations.</td>
</tr>
<tr>
<td>pb.show</td>
<td>logical. If TRUE then a progress bar for the bootstrapping procedure will be displayed as output. The default is TRUE.</td>
</tr>
<tr>
<td>name</td>
<td>a character string giving a name for the outputted boxplot of bootstrapped AUCs. The default is &quot;Quantitative Evaluation of Signatures via Bootstrapped AUCs&quot;.</td>
</tr>
</tbody>
</table>

Value

the AUC, sensitivity and specificity

Examples

```r
inputTest <- matrix(rnorm(1000), 100, 20,
dimnames = list(paste0("gene", seq.int(1, 100)),
               paste0("sample", seq.int(1, 20))))
inputTest <- as.data.frame(inputTest)
targetVec <- sample(c(0,1), replace = TRUE, size = 20)
signature.list <- list(sig1 = c("gene1", "gene2", "gene3"),
                        sig2 = c("gene4", "gene5", "gene6"))
signature.name.vec <- c("sig1", "sig2")
num.boot <- 2
SignatureQuantitative(inputTest, targetVec.num = targetVec,
                      signature.list = signature.list,
                      signature.name.vec = signature.name.vec,
                      num.boot = num.boot)
```
signatureROCplot

Create an array of ROC plots to compare signatures.

Description

Create an array of ROC plots to compare signatures.

Usage

signatureROCplot(
  inputData, 
  annotationData, 
  signatureColNames, 
  annotationColName, 
  scale = FALSE, 
  choose_colors = c("cornflowerblue", "gray24"), 
  name = "Signatures", 
  nrow = NULL, 
  ncol = NULL 
)

Arguments

inputData an input data object. It should either be of the class SummarizedExperiment and contain the profiled signature data and annotation data as columns in the colData, or alternatively be of the classes data.frame or matrix and contain only the gene expression data. Required.

annotationData a data.frame or matrix of annotation data, with one column. Only required if inputData is a data.frame or matrix of signature data.

signatureColNames a vector of the column names of inputData that contain the signature data. If inputData is a SummarizedExperiment object, these are the column names of the object colData.

annotationColName a character string naming the column name in the colData that contains the annotation data to be used in making the boxplot. Only required if inputData is a SummarizedExperiment object.

scale logical. Setting scale = TRUE scales the signature data. The default is FALSE.

choose_colors a vector of length 2 defining the colors to be used in the ROC plots. The default is c("cornflowerblue", "gray24").

name a character string giving the title of the boxplot. The default is "Signatures".

nrow integer giving the number of rows in the resulting array.

ncol integer giving the number of columns in the resulting array.
signatureROCplot_CI

Create an array of ROC plots with confidence interval bands to compare signatures.

Usage

```r
signatureROCplot_CI(
  inputData, annotationData, signatureColNames, annotationColName, scale = FALSE,
  choose_colors = c("cornflowerblue", "gray50", "gray79"), name = NULL,
  nrow = NULL, ncol = NULL, ci.lev = 0.95, pb.show = TRUE
)
```

Arguments

- **inputData**: an input data object. It should either be of the class `SummarizedExperiment` and contain the profiled signature data and annotation data as columns in the `colData`, or alternatively be of the classes `data.frame` or `matrix` and contain only the gene expression data. Required.
annotationData  a data.frame or matrix of annotation data, with one column. Only required if inputData is a data.frame or matrix of signature data.

signatureColNames  a vector of the column names of inputData that contain the signature data. If inputData is a SummarizedExperiment object, these are the column names of the object colData.

annotationColName  a character string naming the column name in the colData that contains the annotation data to be used in making the boxplot. Only required if inputData is a SummarizedExperiment object.

scale  logical. Setting scale = TRUE scales the signature data. The default is FALSE.

choose_colors  a vector of length 3 defining the colors to be used in the ROC plots. The default is c("cornflowerblue","gray50", "gray79").

name  a character string giving the title of the ROC plot. If NULL, the plot title will be "ROC Plots for Gene Signatures, <ci.lev>% Confidence". The default is NULL.

nrow  integer giving the number of rows in the resulting array.

ncol  integer giving the number of columns in the resulting array.

ci.lev  a number between 0 and 1 giving the desired level of confidence for computing ROC curve estimations.

pb.show  logical for whether to show a progress bar while running code. The default is TRUE.

Value  An array of ROC plots.

Examples  
# Run signature profiling
choose_sigs <- TBsignatures[c(1, 2)]
prof_indian <- runTBsigProfiler(TB_indian, useAssay = "logcounts",
algorithm = "Zscore",
signatures = choose_sigs,
parallel.sz = 1)

# Create ROC plots with confidence intervals
signatureROCplot_CI(prof_indian, signatureColNames = names(choose_sigs),
annotationColName = "label")
subsetGeneSet Filter gene expression value matrix based on certain gene sets.

Description
A function used to subset gene expression value matrix based on certain gene sets.

Usage
subsetGeneSet(
  theObject, gene_set, useAssay,
  obtainDiagnosis = FALSE,
  annotationColName = NULL
)

Arguments
theObject A SummarizedExperiment object that has been pre-stored in OriginalTraining-Data.RDA
gene_set A character vector that includes gene symbols for gene signatures.
useAssay A character string or an integer specifying the assay in the theObject that will be selected.
obtainDiagnosis Boolean. Usually used to create training data if TRUE. Default is FALSE
annotationColName A character string specifying the column name of disease status. Only used when creating training data. Default is NULL.

Value
A matrix with selected gene expression value if obtainDiagnosis == FALSE. If obtainDiagnosis == TRUE, return a list contains the selected gene expression value and diagnosis results for each sample.

SulimanOriginalModel Train original model gene signature Suliman_RISK_4.

Description
Train original model gene signature Suliman_RISK_4.
**svm_OriginalModel**

**Usage**

```r
SulimanOriginalModel(dat_list, dat_test_sig)
```

**Arguments**

- **dat_list**
  A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.

- **dat_test_sig**
  A data frame contains corresponding gene sets from the input.

**Value**

The predicted score for each sample in the test study.

---

**Description**

Train original model for gene signatures Bloom_OD_144 and Zak_RISK_16.

**Usage**

```r
svm_OriginalModel(dat_list, dat_test_sig)
```

**Arguments**

- **dat_list**
  A list contains training data and disease status outcomes from the discovery data of corresponding gene signatures.

- **dat_test_sig**
  A data frame contains corresponding gene sets from the input.

**Value**

The predicted score for each sample in the test study.
Create a table of results for t-tests and bootstrapped AUCs for multiple scored signatures.

This function collects the results of bootstrapping and t-tests for a scored gene expression dataset and presents them using a JavaScript table with an R interface, or as a `data.frame`.

Usage

```r
tableAUC(
  SE_scored,
  annotationColName,
  signatureColNames,
  num.boot = 100,
  pb.show = TRUE,
  output = "DataTable"
)
```

Arguments

- `SE_scored` a `SummarizedExperiment` object with genes as the row features and signature scores in the `colData`. There should also be a column of annotation data. Required.
- `annotationColName` a character string giving the column name in `colData` that contains the annotation data. Required.
- `signatureColNames` a vector of column names in the `colData` that contain the signature score data. Required.
- `num.boot` integer. The number of times to bootstrap the data. The default is 100.
- `pb.show` logical for whether to show a progress bar while running code. The default is `TRUE`.
- `output` a character string indicating the table output format. Possible values are `DataTable` and `data.frame`. The default is `DataTable`.

Value

A JavaScript table with an R interface using the `DT` package.

Examples

```r
# Run signature profiling
choose_sigs <- TBsignatures[c(1, 2)]
prof_indian <- runTBsigProfiler(TB_indian, useAssay = "logcounts",
```
TBcommon

A list of published TB signatures, using author-given names.

Description

A set of Tuberculosis gene signatures from various publications. This set of signatures uses gene symbols. Attempts have been made to use updated gene symbols and remove symbols that did not match the most recent annotation. Additional sets for Entrez IDs and Ensembl IDs are forthcoming.

Usage

TBcommon

Format

list

Details

This list differs from TBsignatures in that signatures with names specified in their originating publication (or that of a peer) are given that common name rather than using the TBSignatureProfiler naming system. Otherwise, signature names are composed of the last name of the primary author, followed by a possible context for the signature, and ending with either the number of gene transcripts or genes in the signature with respect to however it was described in the original publication.

Possible signature contexts:

- OD: Other diseases
- HIV: Human Immunodeficiency Virus
- PNA: Pneumonia
- RISK: Risk of developing active TB
- RES: Response to TB treatment
• FAIL: Failure of TB treatment

Note that in some cases signatures will be positive identifiers of TB whereas others are negative identifiers; this should be taken into account when creating ROC curves and computing any AUC estimates.

Source


• Chen_5: Chen L, Hua J, He X. Coexpression Network Analysis-Based Identification of Critical Genes Differentiating between Latent and Active Tuberculosis. Dis Markers. 2022;2022:2090560. 10.1155/2022/2090560


• **Gjoen_10**: Gjøen, J.E., Jenum, S., Sivakumaran, D. et al. 'Novel transcriptional signatures for sputum-independent diagnostics of tuberculosis in children.' Sci Rep 7, 5839 (2017). 10.1038/s41598-017-05057-x

• **Gjoen_7**: Gjøen, J.E., Jenum, S., Sivakumaran, D. et al. 'Novel transcriptional signatures for sputum-independent diagnostics of tuberculosis in children.' Sci Rep 7, 5839 (2017). 10.1038/s41598-017-05057-x


• **Huang_13**: Huang, Hai-Hui et al. 'Identification of 13 Blood-based Gene Expression Signatures to Accurately Distinguish Tuberculosis from Other Pulmonary Diseases and Healthy Controls'. 1 Jan. 2015 : S1837 - S1843. 10.3233/BME-151486


• **Roe_3**: Roe, Jennifer, Venturini, Cristina, Gupta, Rishi K., Gurry, Celine, Chain, Benjamin M., Sun, Yuxin, Southern, Jo, Jackson, Charlotte, Lipman, Marc, C., Miller, Robert F., Martinneau, Adrian R., Abubakar, Ibrahim, Noursadeghi, Mahdad. 2019 "T1 Blood transcriptomic stratification of short-term risk in contacts of tuberculosis": . 10.1093/cid/ciz252


Examples

```r
data("TBcommon")```
Description

A set of Tuberculosis gene signatures compiled from various publications. This set of signatures uses gene symbols. Attempts have been made to use updated gene symbols and remove symbols that did not match the most recent annotation. Additional sets for Entrez IDs and Ensembl IDs are forthcoming.

Usage

TBsignatures

Format

list

Details

Signature names are composed of the last name of the primary author, followed by a possible context for the signature, and ending with either the number of gene transcripts or genes in the signature, with respect to however it was described in the signature’s original publication.

Possible signature contexts:

- <blank>: TB vs LTBI or Healthy Controls
- OD: Other diseases
- HIV: Human Immunodeficiency Virus
- PNA: Pneumonia
- RISK: Risk of developing active TB
- RES: Response to TB treatment
- FAIL: Failure of TB treatment

Note that in some cases signatures will be positive identifiers of TB whereas others are negative identifiers; this should be taken into account when creating ROC curves and computing any AUC estimates.

Source


• **Chen_5**: Chen L, Hua J, He X. Coexpression Network Analysis-Based Identification of Critical Genes Differentiating between Latent and Active Tuberculosis. Dis Markers. 2022;2022:2090560. 10.1155/2022/2090560


• **Huang_13**: Huang, Hai-Hui et al. 'Identification of 13 Blood-based Gene Expression Signatures to Accurately Distinguish Tuberculosis from Other Pulmonary Diseases and Healthy Controls'. 1 Jan. 2015 : S1837 - S1843.10.3233/BME-151486


• **Kaforou_27**: Kaforou, Myrsini, Victoria J. Wright, Tolu Oni, Neil French, Suzanne T. Anderson, Nonzwakazi Bangani, Claire M. Banwell, et al. 2013. "Detection of Tuberculosis in HIV-
Infected and Uninfected African Adults Using Whole Blood RNA Expression Signatures: A Case-Control Study." PLoS Medicine 10 (10): e1001538. 10.1371/journal.pmed.1001538


• **Maertzdorf_4**: Maertzdorf, Jeroen, Gayle McEwen, January Weiner 3rd, Song Tian, Eric Lader, Ulrich Schriek, Harriet Mayanja-Kizza, Martin Ota, John Kenneth, and Stefan He...
Kaufmann. 2016. "Concise Gene Signature for Point-of-Care Classification of Tuberculosis." EMBO Molecular Medicine 8 (2): 86-95. 10.15252/emmm.201505790


• **Zhao_NANO_6**: To be available when published To be available when published

**Examples**

```r
data("TBsignatures")
```

**TBsignaturesSplit**  
*Up/Down-regulated genes information for selected TB signatures.*

**Description**

Up/Down-regulated genes information for selected TB signatures.

**Usage**

`TBsignaturesSplit`
**TBSapp**

Format

list

Source

See ?TBsignatures for reference information.

Examples

data("TBsignaturesSplit")

---

**TBSapp**  
*Run the TBSignatureProfiler Shiny application.*

---

**Description**

Use this function to run the TBSignatureProfiler application.

**Usage**

TBSapp()

**Value**

The Shiny application will open.

**Examples**

# Upload data through the app
if (interactive()) {
  TBSapp()
}

---

**TB_hiv**  
*An example TB dataset with TB/HIV data.*

---

**Description**

An example dataset containing the gene expression and metadata in a SummarizedExperiment object for 31 subjects with HIV and/or Tuberculosis diseases. Information on subject infection status can be accessed with TB_hiv$Disease. Samples with both TB and HIV contamination are marked as tb_hiv, while samples with HIV and no TB are marked as hiv_only.

**Usage**

TB_hiv
This dataset was published as part of a study to assess whether gene expression signatures and cytokine levels would distinguish active TB in advanced HIV in a cohort residing in Sub-Saharan Africa (Verma et. al 2018). Participants were severely immunosuppressed TB-HIV patients who had not yet received TB treatment or anti-retroviral therapy (ART). The dataset included in this package has been lightly edited from the originally published dataset due to the removal of one participant who was HIV positive, on ART and developed TB during follow-up. Whole blood RNA-Seq analysis was performed on all 31 participants.


An example TB dataset with Indian population data.

An example dataset containing the gene expression and metadata in a SummarizedExperiment object for an Indian population. Active TB contamination of the 44 subjects is denoted for each as a "1" (active) or "0" (latent/not present), and can be accessed via `TB_indian$label`. The SummarizedExperiment object contains 2 assays (counts and log(counts)), and the column names give the unique subject identification number along with the subject’s gender.
References


Examples

data("TB_indian")
Index

* datasets
  common_sigAnnotData, 8
  COVIDsignatures, 12
  OriginalTrainingData, 23
  sigAnnotData, 30
  TB_hiv, 61
  TB_indian, 62
  TBcommon, 45
  TBsignatures, 53
  TBsignaturesSplit, 60
  .OriginalModel_NoRetraining, 3
  .OriginalModel_Retraining, 4
  addTBsignature, 5
  Bootstrap_LOOCV_LR_AUC, 7
  bootstrapAUC, 6
  bplapply, 3, 4, 19
  checkGeneSymbols, 18
  ComBat, 4, 19
  common_sigAnnotData, 8
  compareAlgs, 9
  compareBoxplots, 11
  COVIDsignatures, 12
  cv_glmnet_OriginalModel, 16
  deseq2_norm_rle, 17
  distinctColors, 17
  evaluateOriginalModel, 18
  impute.knn, 26, 27
  knn, 4
  knn_OriginalModel, 19
  lda, 4
  lda_OriginalModel, 20
  LOOAUC_simple_multiple_noplot_one_df, 20
  mkAssay, 21
  ObtainSampleScore_OriginalModel, 22
  OriginalTrainingData, 23
  plotQuantitative, 24
  randomForest, 4
  randomForest_OriginalModel, 25
  ref_combat_impute, 26
  runTBsigProfiler, 27
  sigAnnotData, 30
  signatureBoxplot, 31
  signatureGeneHeatmap, 32
  signatureHeatmap, 35
  SignatureQuantitative, 37
  signatureROCplot, 39
  signatureROCplot_CI, 40
  subsetGeneSet, 42
  SulimanOriginalModel, 42
  svm, 4
  svm_OriginalModel, 43
  tableAUC, 44
  TB_hiv, 61
  TB_indian, 62
  TBcommon, 45
  TBsignatures, 53
  TBsignaturesSplit, 60
  TBSPapp, 61