Package 'tcgaViz'

June 7, 2022

Title Vizualisation Tool for The Cancer Genome Atlas Program (TCGA)

Version 0.6.0

Description Differential analysis of tumor tissue immune cell type abundance based on RNASeq gene-level expression from The Cancer Genome Atlas (TCGA) database.

License GPL-3

Depends R (>= 2.10)

Imports config,
data.table,
dplyr,
DT,
ggplot2,
ggpubr,
golem,
grDevices,
magrittr,
methods,
openxlsx,
plotly,
readr,
reshape2,
rlang,
rstatix,
shiny,
shinyFeedback,
shinyjs,
stats,
stringr,
tidyr,
tidyselect,
utils

Suggests BiocCheck,
covr,
knitr,

rmarkdown,
spelling,
testthat

VignetteBuilder knitr

biocViews Visualization, ImmunoOncology, Genetics, GeneExpression, MultipleComparison, DifferentialExpression, RNASeq

Encoding UTF-8

Language en-US

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

R topics documented:

`calculate_pvalue` *Corrected Wilcoxon tests*

Description

Displays stars for each cell type corresponding to the significance level of two mean comparison tests between expression levels (high or low) with multiple correction.

Usage

```
calculate_pvalue(
  x,
  method_test = "wilcox_test",
  method_adjust = "BH",
  p_threshold = 0.05
)
```

Arguments

x	object from convert2biodata() for a dataframe containing columns named high (logical), cell_type (factor) and value (float).
method_test	character for the choice of the statistical test among 't_test' or 'wilcox_test'.
method_adjust	character for the choice of the multiple correction test among 'BH', 'bonferroni', 'BY', 'fdr', 'hochberg', 'holm', 'hommel', 'none'
p_threshold	float for the significativity threshold of the P-value.

Value

rstatix_test object for a table with cell types in the row and P-values, corrections and other statistics in the column.

Examples

```
data(tcga)
(df <- convert2biodata(
    algorithm = "Cibersort_ABS",
    disease = "breast invasive carcinoma",
    tissue = "Primary Tumor",
    gene_x = "ICOS"
))
calculate_pvalue(df)
calculate_pvalue(
    df,
    method_test = "t_test",
    method_adjust = "bonferroni",
    p_threshold = 0.01
)
```

---

convert2biodata *Format biological data*

Description

Merges gene and cell datasets with the same TCGA sample identifiers, splits samples according to the expression levels of a selected gene into two categories (below or above average) and formats into a 3-column data frame: gene expression levels, cell types, and gene expression values.

Usage

```
convert2biodata(algorithm, disease, tissue, gene_x, stat = "mean", path = ".")
```

Arguments

algorithm	character for the algorithm used to estimate the distribution of cell type abundance among : 'Cibersort', 'Cibersort_ABS', 'EPIC', 'MCP_counter', 'Quantiseq', 'Timer', 'Xcell', 'Xcell (2)' and 'Xcell64'.
disease	character for the type of TCGA cancer (see the list in extdata/disease_names.csv).
tissue	character for the type of TCGA tissue among : 'Additional - New Primary', 'Additional Metastatic', 'Metastatic', 'Primary Blood Derived Cancer - Peripheral Blood', 'Primary Tumor', 'Recurrent Tumor', 'Solid Tissue Normal'
gene_x	character for the gene selected in the differential analysis (see the list in extdata/gene_names.csv).
stat	character for the statistic to be chosen among "mean", "median" or "quantile".
path	character for the path name of the tcga dataset.

Value

data frame with the following columns:

- high (logical): the expression levels of a selected gene, TRUE for below or FALSE for above average.
- cells (factor): cell types.
- value (float): the abundance estimation of the cell types.

Examples

```
data(tcga)
(convert2biodata(
    algorithm = "Cibersort_ABS",
    disease = "breast invasive carcinoma",
    tissue = "Primary Tumor",
    gene_x = "ICOS"
))
```

---

convert_biodata *Format biological data*

Description

Merges gene and cell datasets with the same TCGA sample identifiers, splits samples according to the expression levels of a selected gene into two categories (below or above average) and formats into a 3-column data frame: gene expression levels, cell types, and gene expression values.

Usage

```
convert_biodata(
  genes,
  cells,
  select = colnames(genes)[3],
  stat = "mean",
  disease = NULL,
  tissue = NULL
)
```

Arguments

genes	data frame whose first two columns contain identifiers and the others float values.
cells	data frame whose first two columns contain identifiers and the others float values.
select	character for a column name in genes.
stat	character for the statistic to be chosen among "mean", "median" or "quantile".
disease	character for the type of TCGA cancer (see the list in extdata/disease_names.csv).
tissue	character for the type of TCGA tissue among : 'Additional - New Primary', 'Additional Metastatic', 'Metastatic', 'Primary Blood Derived Cancer - Peripheral Blood', 'Primary Tumor', 'Recurrent Tumor', 'Solid Tissue Normal'

Details

`disease` and `tissue` arguments should be displayed in the title of `plot.biodata()` only if the `genes` argument does not already have them in its attributes.

Value

data frame with the following columns:

- `high` (logical): the expression levels of a selected gene, TRUE for below or FALSE for above average.
- `cells` (factor): cell types.
- `value` (float): the abundance estimation of the cell types.

Examples

```
data(tcga)
(df_formatted <- convert_biodata(tcga$genes, tcga$cells$Cibersort, "ICOS"))
```

plot.biodata *Distribution plot*

Description

Distribution plot of cell subtypes according to the expression level (high or low) of a selected gene.

Usage

```
## S3 method for class 'biodata'
plot(
  x,
  type = "violin",
  dots = FALSE,
  title = NULL,
  xlab = NULL,
  ylab = NULL,
  stats = NULL,
  draw = TRUE,
  axis.text.x = element_text(size = 10),
  axis.text.y = element_text(size = 8),
  cex.lab = 12,
  cex.main = 16,
  col = (scales::hue_pal())(length(unique(x$cell_type))),
 axis.title.x = element_text(size = cex.lab, face = "bold.italic", vjust = -0.5),
 axis.title.y = element_text(size = cex.lab, face = "bold.italic", vjust = -0.5),
 plot.title = element_text(size = cex.main, face = "bold", vjust = 1, hjust = 0.5),
  plot.margin = unit(c(0, 0, 0, -0.5), "cm"),
  ...
)
```

Arguments

x	object from convert2biodata() for a dataframe containing columns named high (logical), cell_type (factor) and value (float).
type	character for the type of plot to be chosen among "violin" or "boxplot".
dots	boolean to add all points to the graph.
title	character for the title of the plot.
xlab	character for the name of the X axis label.
ylab	character for the name of the Y axis label.
stats	object from calculate_pvalue().
draw	bolean to plot the graph.

axis.text.x	tick labels along axes (element_text()). Specify all axis tick labels (axis.text), tick labels by plane (using axis.text.x or axis.text.y), or individually for each axis (using axis.text.x.bottom, axis.text.x.top, axis.text.y.left, axis.text.y.right). axis.text.*.* inherits from axis.text.* which inherits from axis.text, which in turn inherits from text
axis.text.y	tick labels along axes (element_text()). Specify all axis tick labels (axis.text), tick labels by plane (using axis.text.x or axis.text.y), or individually for each axis (using axis.text.x.bottom, axis.text.x.top, axis.text.y.left, axis.text.y.right). axis.text.*.* inherits from axis.text.* which inherits from axis.text, which in turn inherits from text
cex.lab	numerical value giving the amount by which x and y plotting labels should be magnified relative to the default.
cex.main	numerical value giving the amount by which main plotting title should be magnified relative to the default.
col	character for the specification for the default plotting color. See section 'Color Specification' in graphics::par().
axis.title.x	labels of axes (element_text()). Specify all axes' labels (axis.title), labels by plane (using axis.title.x or axis.title.y), or individually for each axis (using axis.title.x.bottom, axis.title.x.top, axis.title.y.left, axis.title.y.right). axis.title.*.* inherits from axis.title.* which inherits from axis.title, which in turn inherits from text
axis.title.y	labels of axes (element_text()). Specify all axes' labels (axis.title), labels by plane (using axis.title.x or axis.title.y), or individually for each axis (using axis.title.x.bottom, axis.title.x.top, axis.title.y.left, axis.title.y.right). axis.title.*.* inherits from axis.title.* which inherits from axis.title, which in turn inherits from text
plot.title	plot title (text appearance) (element_text(); inherits from title) left-aligned by default
plot.margin	margin around entire plot (unit with the sizes of the top, right, bottom, and left margins)
...	arguments to pass to ggplot2::theme().

Examples

```
library("ggplot2")
data(tcga)
(df <- convert2biodata(
    algorithm = "Cibersort_ABS",
    disease = "breast invasive carcinoma",
    tissue = "Primary Tumor",
    gene_x = "ICOS"
))
plot(df)
stats <- calculate_pvalue(df)
plot(
    df,
    stats = stats,
```

```
    type = "boxplot",
    dots = TRUE,
    xlab = "Expression level of the 'ICOS' gene by cell type",
    ylab = "Percent of relative abundance\n(from the Cibersort_ABS algorithm)",
    title = "Differential analysis of tumor tissue immune cell type abundance
    based on RNASeq gene-level expression from The Cancer Genome Atlas
    (TCGA) database",
    axis.text.y = element_text(size = 8, hjust = 0.5),
    plot.title =  element_text(face = "bold", hjust = 0.5)
)
```

---

run_app *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function that will be called before the app is actually run. This is only needed for `shinyAppObj`, since in the `shinyAppDir` case, a `global.R` file can be used for this purpose.
options	Named options that should be passed to the `runApp` call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify `width` and `height` parameters which provide a hint to the embedding environment about the ideal height/width for the app.
enableBookmarking	Can be one of `"url"`, `"server"`, or `"disable"`. The default value, `NULL`, will respect the setting from any previous calls to `enableBookmarking()`. See `enableBookmarking()` for more information on bookmarking your app.
uiPattern	A regular expression that will be applied to each `GET` request to determine whether the `ui` should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
...	arguments to pass to golem_opts. See `?golem::get_golem_options` for more details.

tcga *Biological data*

Description

A list of biological data: RNASeq data, phenotypic metadata and cell abundance.

Usage

```
data(tcga)
```

Details

- `genes`: RNASeq from The Cancer Genome Atlas (TCGA) database.
- `phenotypes`: Metadata from the TCGA database containing sample ID, sample type ID, sample type and primary disease.
- `cells`: Abundance estimates of cell types

Note

Subset of invasive breast carcinoma data from primary tumor tissue. The cell type data are from a subset generated by the Cibersort_ABS algorithm (https://cibersort.stanford.edu/). For the complete dataset, please use:

```
path <- system.file("extdata", package = "tcgaViz")
load(file.path(path, "tcga.rda"))
```

Source

- dataset: gene expression RNAseq - Batch effects normalized mRNA data
- hub: https://pancanatlas.xenahubs.net
- cohort: TCGA Pan-Cancer (PANCAN)
- dataset ID: EB++AdjustPANCAN_IlluminaHiSeq_RNASeqV2.geneExp.xena
- download: https://tcga-pancan-atlas-hub.s3.us-east-1.amazonaws.com/download/EB%2B%2BAdjustPANCAN_IlluminaHiSeq_RNASeqV2.geneExp.xena.gz (full metadata)
- samples: 11060
- version: 2016-12-29
- type of data: gene expression RNAseq
- unit: log2(norm_value+1)
- raw data: https://www.synapse.org/#!Synapse:syn4976369.3
- input data format: ROWs (identifiers) x COLUMNs (samples) (i.e. genomicMatrix)

Examples

```
data(tcga)
(df <- convert2biodata(
    algorithm = "Cibersort_ABS",
    disease = "breast invasive carcinoma",
    tissue = "Primary Tumor",
    gene_x = "ICOS"
))
(stats <- calculate_pvalue(df))
plot(df, stats = stats)
```

Index