Package ‘cofeatureR’

October 12, 2022

Title Generate Cofeature Matrices

Version 1.1.1

Description Generate cofeature (feature by sample) matrices. The package utilizes ggplot2::geom_tile() to generate the matrix allowing for easy additions from the base matrix.

Depends R (>= 3.1.0)

Imports ggplot2 (>= 1.0.0), dplyr (>= 0.4.3), lazyeval (>= 0.1.10), tibble

URL https://github.com/tinyheero/cofeatureR

BugReports https://github.com/tinyheero/cofeatureR/issues

License GPL-3

LazyData true

RoxygenNote 6.0.1

Suggests testthat

NeedsCompilation no

Author Fong Chun Chan [aut, cre]

Maintainer Fong Chun Chan <fongchunchan@gmail.com>

Repository CRAN

Date/Publication 2018-06-24 15:09:05 UTC

R topics documented:

add_tiles ................................................................. 2
cofeatureR ............................................................. 2
plot_cofeature_mat .................................................... 2

Index 5
add_tiles

Add tiles to the ggplot2

Description

Add tiles to the ggplot2

Usage

add_tiles(p1, in.df, tile.col, missing.fill.col, tile.border.size)

Arguments

- **p1**: Existing ggplot2
- **in.df**: A 3 column (feature, sampleID, type) data.frame object
- **tile.col**: Border color of each cell. If not set, no border color is used.
- **missing.fill.col**: Color of the cell that has missing values
- **tile.border.size**: Integer to indicate the size of the tile borders.

cofeatureR

cofeatureR: Generate Cofeature Matrices

Description

Generate cofeature (feature by sample) matrices. The package utilizes ggplot2::geom_tile to generate the matrix allowing for easy customization of additions from the base matrix.

plot_cofeature_mat

Plot a Cofeature Matrix

Description

Generates a ggplot2::geom_tile plot of features by sample. It is able to deal with multiple types affecting the same sample.

Usage

plot_cofeature_mat(in.df, feature.order, sample.id.order, fill.colors, type.display.mode = c("multiple", "single"), type.order, tile.col = NA, rotate.x.labels, missing.fill.col, dot.flag = FALSE, dot.size, tile.flag = TRUE, drop.x = FALSE, tile.border.size = 1)
**Arguments**

- `in.df`: A 3 column (feature, sampleID, type) `data.frame` object.
- `feature.order`: Character vector indicating the order of the features in the final plot on the y-axis. If not set, the function will set it automatically.
- `sample.id.order`: Character vector indicating the order of the samples in the final plot on the x-axis. If not set, the function will set it automatically.
- `fill.colors`: Character vector indicating the colors of the different "types". The names should be the types with the value being the color.
- `type.display.mode`: Specify whether multiple or a single feature type can appear in the same feature/sample cell.
- `type.order`: Specify the "priority" of the feature types. This only has an effect when `type.display.mode` is set to single.
- `tile.col`: Border color of each cell. If not set, no border color is used.
- `rotate.x.labels`: Rotate the x-axes labels by a certain degree.
- `missing.fill.col`: Color of the cell that has missing values.
- `dot.flag`: Boolean to turn on/off dots (dot.flag).
- `dot.size`: Column name indicating the size of the dots. Only takes effect if `dot.flag` is TRUE.
- `tile.flag`: Boolean to turn on/off tiles (tile.flag).
- `drop.x`: Boolean to drop levels (from a factor) in the x dimension.
- `tile.border.size`: Integer to indicate the size of the tile borders.

**Examples**

```r
# Not run:
v1 <- c("RCOR1", "NCOR1", "LCOR", "RCOR1", "RCOR1", "RCOR1")
v2 <- c("sampleA", "sampleC", "sampleB", "sampleC", "sampleA", "sampleC")
v3 <- c("Deletion", "Deletion", "SNV", "Rearrangement", "SNV", "Rearrangement", "SNV")
v4 <- c(0.05, 0.5, 0.25, 0.01, 0.03, 0.24, 0.89)
v5 <- c(1, 2, 1, 2, 2, 1)
feature.order <- c("RCOR1", "NCOR1", "LCOR")
sample.id.order <- c("sampleA", "sampleB", "sampleC")
in.df <- dplyr::data_frame(feature = v1, sampleID = v2, type = v3,
                           p_value = -log10(v4), dir_flag = v5)
fill.colors <- c("Deletion" = "Blue", "Rearrangement" = "Green", "SNV" = "Red")
plot_cofeature_mat(in.df)

# With black tile color
plot_cofeature_mat(in.df, tile.col = "black")
```
# Fill in missing values with a lightgrey color
plot_cofeature_mat(in.df, tile.col = "black", missing.fill.col = "lightgrey")

# Rotate x-axes labels by 90 degrees
plot_cofeature_mat(in.df, rotate.x.labels = 90)

# Specify order of features, samples, and colors
plot_cofeature_mat(in.df, feature.order, sample.id.order, fill.colors = fill.colors)

# Specify each cell can only have one "feature type"
plot_cofeature_mat(in.df, feature.order, sample.id.order, fill.colors = fill.colors, type.display.mode = "single")

# Specify the specific priority of the "feature type" for cells with multiple features
plot_cofeature_mat(in.df, feature.order, sample.id.order, fill.colors = fill.colors, type.display.mode = "single", type.order = c("Rearrangement", "SNV", "Deletion"))

# Add dots to tiles for an additional layer of information
plot_cofeature_mat(in.df, dot.size = "p_value")

# Only display dots
plot_cofeature_mat(in.df, dot.flag = TRUE, dot.size = "p_value", tile.flag = FALSE)

# Samples will not be dropped
sample.id.order.new <- c("sampleA", "sampleB", "sampleC", "sampleD")
plot_cofeature_mat(in.df, tile.col = "black", sample.id.order = sample.id.order.new)

# Samples can be dropped by setting drop.x = TRUE
plot_cofeature_mat(in.df, tile.col = "black", sample.id.order = sample.id.order.new, drop.x = TRUE)

## End(Not run)
Index

add_tiles, 2

cofeatureR, 2
cofeatureR-package (cofeatureR), 2

plot_cofeature_mat, 2