

Package ‘gg1d’

December 9, 2024

Title Exploratory Data Analysis using Tiled One-Dimensional Graphics

Version 0.1.0

Description Streamlines exploratory data analysis by providing a turnkey approach to visualising n-dimensional data which graphically reveals correlative or associative relationships between 2 or more features. Represents all dataset features as distinct, vertically aligned bar or tile plots, with plot types auto-selected based on whether variables are categorical or numeric.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

URL <https://github.com/selkamand/gg1d>,
<https://selkamand.github.io/gg1d/>

BugReports <https://github.com/selkamand/gg1d/issues>

Imports assertions (>= 0.2.0), cli, ggiraph, ggplot2, ggtext,
patchwork (>= 1.3.0), rank (>= 0.1.1), rlang

Suggests covr, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 2.10)

LazyData true

NeedsCompilation no

Author Sam El-Kamand [aut, cre] (<<https://orcid.org/0000-0003-2270-8088>>),
Children's Cancer Institute Australia [cph]

Maintainer Sam El-Kamand <sam.elkamand@gmail.com>

Repository CRAN

Date/Publication 2024-12-09 19:40:02 UTC

Contents

beautify	2
column_info_table	2
ggld	4
ggld_options	6
lazy_birdwatcher	9
sensible_2_breaks	10

Index	11
--------------	-----------

beautify	<i>Make strings prettier for printing</i>
----------	---

Description

Takes an input string and 'beautify' by converting underscores to spaces and

Usage

```
beautify(string, autodetect_units = TRUE)
```

Arguments

string	input string
autodetect_units	automatically detect units (e.g. mm, kg, etc) and wrap in brackets.

Value

string

column_info_table	<i>Parse a tibble and ensure it meets standards</i>
-------------------	---

Description

Parse a tibble and ensure it meets standards

Usage

```
column_info_table(
  data,
  maxlevels = 6,
  col_id = NULL,
  cols_to_plot,
  tooltip_column_suffix = "_tooltip",
  ignore_column_regex = "_ignore$",
  palettes,
  colours_default,
  colours_default_logical,
  verbose
)
```

Arguments

<code>data</code>	data.frame to autoplot (data.frame)
<code>maxlevels</code>	for categorical variables, what is the maximum number of distinct values to allow (too many will make it hard to find a palette that suits). (number)
<code>col_id</code>	name of column to use as an identifier. If null, artificial IDs will be created based on row-number.
<code>cols_to_plot</code>	names of columns in data that should be plotted. By default plots all valid columns (character)
<code>tooltip_column_suffix</code>	the suffix added to a column name that indicates column should be used as a tooltip (string)
<code>ignore_column_regex</code>	a regex string that, if matches a column name, will cause that column to be exclude from plotting (string) (default: "_ignore\$")
<code>palettes</code>	A list of named vectors. List names correspond to data column names (categorical only). Vector names to levels of columns. Vector values are colours, the vector names are used to map values in data to a colour.
<code>colours_default</code>	Default colors for categorical variables without a custom palette.
<code>colours_default_logical</code>	Colors for binary variables: a vector of three colors representing TRUE, FALSE, and NA respectively (character).
<code>verbose</code>	Numeric value indicating the verbosity level: <ul style="list-style-type: none"> • 2: Highly verbose, all messages. • 1: Key messages only. • 0: Silent, no messages.

Value

tibble with the following columns:

1. colnames
2. coltype (categorical/numeric/tooltip/invalid)
3. ndistinct (number of distinct values)
4. plottable (should this column be plotted)
5. tooltip_col (the name of the column to use as the tooltip) or NA if no obvious tooltip column found

gg1d

AutoPlot an entire data.frame

Description

Visualize all columns in a data frame with gg1d's vertically aligned plots and automatic plot selection based on variable type. Plots are fully interactive, and custom tooltips can be added.

Usage

```
gg1d(
  data,
  col_id = NULL,
  col_sort = NULL,
  order_matches_sort = TRUE,
  maxlevels = 6,
  verbose = 2,
  drop_unused_id_levels = FALSE,
  interactive = TRUE,
  return = c("plot", "column_info", "data"),
  palettes = NULL,
  sort_type = c("frequency", "alphabetical"),
  desc = TRUE,
  limit_plots = TRUE,
  max_plottable_cols = 15,
  cols_to_plot = NULL,
  tooltip_column_suffix = "_tooltip",
  ignore_column_regex = "_ignore$",
  convert_binary_numeric_to_factor = TRUE,
  options = gg1d_options(show_legend = !interactive)
)
```

Arguments

data	data.frame to autoplot (data.frame)
col_id	name of column to use as an identifier. If null, artificial IDs will be created based on row-number.
col_sort	name of columns to sort on. To do a hierarchical sort, supply a vector of column names in the order they should be sorted (character).

<code>order_matches_sort</code>	should the column plots be stacked top-to-bottom in the order they appear in <code>col_sort</code> (flag)
<code>maxlevels</code>	for categorical variables, what is the maximum number of distinct values to allow (too many will make it hard to find a palette that suits). (number)
<code>verbose</code>	Numeric value indicating the verbosity level: <ul style="list-style-type: none"> • 2: Highly verbose, all messages. • 1: Key messages only. • 0: Silent, no messages.
<code>drop_unused_id_levels</code>	if <code>col_id</code> is a factor with unused levels, should these be dropped or included in visualisation
<code>interactive</code>	produce interactive ggiraph visualisation (flag)
<code>return</code>	a string describing what this function should return. Options include: <ul style="list-style-type: none"> • plot: Return the gg1d visualisation (default) • column_info: Return a data.frame describing the columns the dataset. • data: Return the processed dataset used for plotting.
<code>palettes</code>	A list of named vectors. List names correspond to data column names (categorical only). Vector names to levels of columns. Vector values are colours, the vector names are used to map values in data to a colour.
<code>sort_type</code>	controls how categorical variables are sorted. Numerical variables are always sorted in numerical order irrespective of the value given here. Options are alphabetical or frequency
<code>desc</code>	sort in descending order (flag)
<code>limit_plots</code>	throw an error when there are > <code>max_plottable_cols</code> in dataset (flag)
<code>max_plottable_cols</code>	maximum number of columns that can be plotted (default: 15) (number)
<code>cols_to_plot</code>	names of columns in data that should be plotted. By default plots all valid columns (character)
<code>tooltip_column_suffix</code>	the suffix added to a column name that indicates column should be used as a tooltip (string)
<code>ignore_column_regex</code>	a regex string that, if matches a column name, will cause that column to be exclude from plotting (string) (default: <code>"_ignore\$"</code>)
<code>convert_binary_numeric_to_factor</code>	If a numeric column contains only values 0, 1, & NA, then automatically convert to a factor.
<code>options</code>	a list of additional visual parameters created by calling <code>gg1d_options()</code> . See gg1d_options for details.

Value

ggiraph interactive visualisation

Examples

```

path_gg1d <- system.file("example.csv", package = "gg1d")
df <- read.csv(path_gg1d, header = TRUE, na.strings = "")

# Create Basic Plot
gg1d(df, col_id = "ID", col_sort = "Glasses")

# Configure plot gg1d_options()
gg1d(
  lazy_birdwatcher,
  col_sort = "Magpies",
  palettes = list(
    Birdwatcher = c(Robert = "#E69F00", Catherine = "#999999"),
    Day = c(Weekday = "#999999", Weekend = "#009E73")
  ),
  options = gg1d_options(
    show_legend = TRUE,
    fontsize_barplot_y_numbers = 12,
    legend_text_size = 16,
    legend_key_size = 1,
    legend_nrow = 1,
  )
)

```

gg1d_options

Visual Parameters for gg1d Plots

Description

Configures aesthetic and layout settings for plots generated by gg1d.

Usage

```

gg1d_options(
  colours_default = c("#66C2A5", "#FC8D62", "#8DA0CB", "#E78AC3", "#A6D854", "#FFD92F",
    "#E5C494"),
  colours_default_logical = c(`TRUE` = "#648fff", `FALSE` = "#dc267f"),
  colours_missing = "grey90",
  show_legend_titles = FALSE,
  legend_title_position = c("top", "bottom", "left", "right"),
  legend_nrow = 4,
  legend_ncol = NULL,
  legend_title_size = NULL,
  legend_text_size = NULL,
  legend_key_size = 0.3,
  legend_orientation_heatmap = c("horizontal", "vertical"),
  show_legend = TRUE,

```

```

legend_position = c("right", "left", "bottom", "top"),
na_marker = "!",
na_marker_size = 8,
na_marker_colour = "black",
show_na_marker_categorical = FALSE,
show_na_marker_heatmap = FALSE,
colours_heatmap_low = "purple",
colours_heatmap_high = "seagreen",
transform_heatmap = c("identity", "log10", "log2"),
fontsize_values_heatmap = 3,
show_values_heatmap = FALSE,
colours_values_heatmap = "white",
vertical_spacing = 0,
numeric_plot_type = c("bar", "heatmap"),
y_axis_position = c("left", "right"),
width = 0.9,
relative_height_numeric = 4,
cli_header = "Running gg1d",
interactive_svg_width = NULL,
interactive_svg_height = NULL,
fontsize_barplot_y_numbers = 8,
max_digits_barplot_y_numbers = 3,
fontsize_y_title = 12,
beautify_text = TRUE
)

```

Arguments

`colours_default` Default colors for categorical variables without a custom palette.

`colours_default_logical` Colors for binary variables: a vector of three colors representing TRUE, FALSE, and NA respectively (character).

`colours_missing` Color for missing (NA) values in categorical plots (string).

`show_legend_titles` Display titles for legends (flag).

`legend_title_position` Position of the legend title ("top", "bottom", "left", "right").

`legend_nrow` Number of rows in the legend (number).

`legend_ncol` Number of columns in the legend. If set, `legend_nrow` should be NULL (number).

`legend_title_size` Size of the legend title text (number).

`legend_text_size` Size of the text within the legend (number).

<code>legend_key_size</code>	Size of the legend key symbols (number).
<code>legend_orientation_heatmap</code>	should legend orientation be "horizontal" or "vertical".
<code>show_legend</code>	Display the legend on the plot (flag).
<code>legend_position</code>	Position of the legend ("right", "left", "bottom", "top").
<code>na_marker</code>	Text used to mark NA values in numeric plots (string).
<code>na_marker_size</code>	Size of the text marker for NA values (number).
<code>na_marker_colour</code>	Color of the NA text marker (string).
<code>show_na_marker_categorical</code>	Show a marker for NA values on categorical tiles (flag).
<code>show_na_marker_heatmap</code>	Show a marker for NA values on heatmap tiles (flag).
<code>colours_heatmap_low</code>	Color for the lowest value in heatmaps (string).
<code>colours_heatmap_high</code>	Color for the highest value in heatmaps (string).
<code>transform_heatmap</code>	Transformation to apply before visualizing heatmap values ("identity", "log10", "log2").
<code>fontsize_values_heatmap</code>	Font size for heatmap values (number).
<code>show_values_heatmap</code>	Display numerical values on heatmap tiles (flag).
<code>colours_values_heatmap</code>	Color for heatmap values (string).
<code>vertical_spacing</code>	Space between each data row in points (number).
<code>numeric_plot_type</code>	Type of visualization for numeric data: "bar" or "heatmap".
<code>y_axis_position</code>	Position of the y-axis ("left" or "right").
<code>width</code>	controls how much space is present between bars and tiles within each plot. Can be 0-1 where values of 1 makes bars/tiles take up 100% of available space (no gaps between bars).
<code>relative_height_numeric</code>	how many times taller should numeric plots be relative to categorical tile plots. Only taken into account if <code>numeric_plot_type == "bar"</code> (number)
<code>cli_header</code>	Text used for h1 header. Included so it can be tweaked by packages that use ggld, so they can customise how the info messages appear.
<code>interactive_svg_width, interactive_svg_height</code>	width and height of the interactive graphic region (in inches). Only used when <code>interactive = TRUE</code> .

fontSize_barplot_y_numbers
fontSize of the text describing numeric barplot max & min values (number).

max_digits_barplot_y_numbers
Number of digits to round the numeric barplot max and min values to (number).

fontSize_y_title
fontSize of the y axis titles (a.k.a the data.frame column names) (number).

beautify_text
Beautify y-axis text and legend titles by capitalizing words and adding spaces (flag).

Value

A list of visualization parameters for gg1d.

Examples

```
path_gg1d <- system.file("example.csv", package = "gg1d")
df <- read.csv(path_gg1d, header = TRUE, na.strings = "")

# Create Basic Plot
gg1d(df, col_id = "ID", col_sort = "Glasses")

# Configure plot gg1d_options()
gg1d(
  lazy_birdwatcher,
  col_sort = "Magpies",
  palettes = list(
    Birdwatcher = c(Robert = "#E69F00", Catherine = "#999999"),
    Day = c(Weekday = "#999999", Weekend = "#009E73")
  ),
  options = gg1d_options(
    show_legend = TRUE,
    fontSize_barplot_y_numbers = 12,
    legend_text_size = 16,
    legend_key_size = 1,
    legend_nrow = 1,
  )
)
```

lazy_birdwatcher

Lazy Birdwatcher Dataset

Description

A simulated dataset describing the number of magpies observed by two birdwatchers.

Usage

```
lazy_birdwatcher
```

Format

lazy_birdwatcher:

A data frame with 45 rows and 3 columns:

Magpies Number of magpies observed

Day Was the day of observation a weekday or a weekend?

Birdwatcher Name of the birdwatcher

sensible_2_breaks *GGplot breaks*

Description

Find sensible values to add 2 breaks at for a ggplot2 axis

Usage

```
sensible_2_breaks(vector)
```

Arguments

vector vector fed into ggplot axis you want to define sensible breaks for

Value

vector of length 2. first element describes upper break position, lower describes lower break

Index

* datasets

lazy_birdwatcher, 9

beautify, 2

column_info_table, 2

gg1d, 4

gg1d_options, 5, 6

gg1d_options(), 5

lazy_birdwatcher, 9

sensible_2_breaks, 10