Package ‘autoplotly’

April 18, 2021

Type Package
Title Automatic Generation of Interactive Visualizations for Statistical Results
Version 0.1.4
Description Functionalities to automatically generate interactive visualizations for statistical results supported by ‘ggfortify’, such as time series, PCA, clustering and survival analysis, with ‘plotly.js’ <https://plotly.com/> and ‘ggplot2’ style. The generated visualizations can also be easily extended using ‘ggplot2’ and ‘plotly’ syntax while staying interactive.
License GPL-2
URL https://github.com/terrytangyuan/autoplotly
BugReports https://github.com/terrytangyuan/autoplotly/issues
Encoding UTF-8
Depends R (>= 3.1)
Imports methods, ggplot2, plotly, ggfortify
RoxygenNote 7.0.2
VignetteBuilder knitr
Suggests testthat, knitr, markdown
NeedsCompilation no
Author Yuan Tang [aut, cre] (<https://orcid.org/0000-0001-5243-233X>)
Maintainer Yuan Tang <terrytangyuan@gmail.com>
Repository CRAN
Date/Publication 2021-04-18 06:50:11 UTC

R topics documented:

autoplotly ........................................... 2

Index 4
autoplotly  

Automatic Visualization of Popular Statistical Results Using 'plotly.js' and 'ggplot2'

Description

This function provides functionality to automatically generate interactive plot for many popular statistical results supported by 'ggfortify' package using 'plotly.js' and 'ggplot2'.

This package provides functionalities to automatically generate interactive visualizations for many popular statistical results supported by ggfortify package with 'plotly.js' and 'ggplot2' style. The generated plot can also be easily extended using 'ggplot2' and 'plotly' syntax while staying interactive.

Usage

autoplotly(
  object,
  ...,  # Arguments passed to ggfortify::autoplot function for the applied object.
  width = NULL,
  height = NULL,
  tooltip = "all",
  dynamicTicks = FALSE,
  layerData = 1,
  originalData = TRUE,
  source = "A",
  widths = NULL,
  heights = NULL,
  margin = 0.02,
  shareX = TRUE,
  shareY = TRUE,
  titleX = shareX,
  titleY = shareY,
  which_layout = "merge"
)

Arguments

object  
The object that represents your statistical result, e.g. stats::prcomp(iris[-5]).

width  
Width of the plot in pixels (optional, defaults to automatic sizing).

height  
Height of the plot in pixels (optional, defaults to automatic sizing).

tooltip  
a character vector specifying which aesthetic mappings to show in the tooltip. The default, "all", means show all the aesthetic mappings (including the unofficial "text" aesthetic). The order of variables here will also control the order they
appear. For example, use `tooltip = c("y", "x", "colour")` if you want `y` first, `x` second, and `colour` last.

dynamicTicks should plotly.js dynamically generate axis tick labels? Dynamic ticks are useful for updating ticks in response to zoom/pan interactions; however, they can not always reproduce labels as they would appear in the static `ggplot2` image.

layerData data from which layer should be returned?

originalData should the "original" or "scaled" data be returned?

source a character string of length 1. Match the value of this string with the source argument in `event_data()` to retrieve the event data corresponding to a specific plot (shiny apps can have multiple plots).

widths relative width of each column on a 0-1 scale. By default all columns have an equal relative width.

heights relative height of each row on a 0-1 scale. By default all rows have an equal relative height.

margin either a single value or four values (all between 0 and 1). If four values are provided, the first is used as the left margin, the second is used as the right margin, the third is used as the top margin, and the fourth is used as the bottom margin. If a single value is provided, it will be used as all four margins.

shareX should the x-axis be shared amongst the subplots?

shareY should the y-axis be shared amongst the subplots?

titleX should x-axis titles be retained?

titleY should y-axis titles be retained?

which_layout adopt the layout of which plot? If the default value of "merge" is used, layout options found later in the sequence of plots will override options found earlier in the sequence. This argument also accepts a numeric vector specifying which plots to consider when merging.

Examples

# Automate generating interactive plot for results produced by `stats::prcomp`

```r
p <- autoplotly(prcomp(iris[1:4]), data = iris,
                 colour = 'Species', label = TRUE, label.size = 3, frame = TRUE)
```

# You can apply additional `ggplot2` elements to interactive plot built using `autoplotly`

```r
p +
  ggplot2::ggtitle("Principal Components Analysis") +
  ggplot2::labs(y = "Second Principal Components", x = "First Principal Components")
```

# Or apply additional `plotly` elements to the generated interactive plot

```r
p %>% plotly::layout(annotations = list(  
  text = "Example Text",
  font = list(  
    family = "Courier New, monospace",
    size = 18,
    color = "black"),
  x = 0,
  y = 0,
  showarrow = TRUE))
```
Index

autoplotly, 2

event_data(), 3