# Package 'parcats'

December 10, 2023

**Title** Interactive Parallel Categories Diagrams for 'easyalluvial'

**Version** 0.0.5

**URL** https://erblast.github.io/parcats/

**BugReports** https://github.com/erblast/parcats/issues/

**Description** Complex graphical representations of data are best explored using interactive elements. 'parcats' adds interactive graphing capabilities to the 'easyalluvial' package. The 'plotly.js' parallel categories diagrams offer a good framework for creating interactive flow graphs that allow manual drag and drop sorting of dimensions and categories, highlighting single flows and displaying mouse over information. The 'plotly.js' dependency is quite heavy and therefore is outsourced into a separate package.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Depends** R (>= 3.0.0)

**Suggests** testthat, covr, randomForest, knitr, rmarkdown, spelling, plotly, shiny

**RoxygenNote** 7.2.3

**Imports** easyalluvial (>= 0.2.1.0), tidyr (>= 1.0.0), dplyr, purrr, forcats, magrittr, tibble, htmlwidgets, stringr

**Language** en-US

**NeedsCompilation** no

**Author** Bjoern Koneswarakantha [aut, cre] (<https://orcid.org/0000-0003-4585-7799>)

**Maintainer** Bjoern Koneswarakantha <datistics@gmail.com>

**Repository** CRAN

**Date/Publication** 2023-12-10 11:40:02 UTC

## R topics documented:

## parcats *create plotly parallel categories diagram from alluvial plot*

### Description

creates an interactive parallel categories diagram from an 'easyalluvial' plot using the 'plotly.js' library

### Usage

```
parcats(
  p,
  marginal_histograms = TRUE,
  data_input = NULL,
  imp = TRUE,
  width = NULL,
  height = NULL,
  elementId = NULL,
  hoveron = "color",
  hoverinfo = "count+probability",
  arrangement = "perpendicular",
  bundlecolors = TRUE,
  sortpaths = "forward",
  labelfont = list(size = 24, color = "black"),
  tickfont = NULL,
  offset_marginal_histograms = 0.7,
  offset_imp = 0.9
)
```

### Arguments

`p` alluvial plot

`marginal_histograms` logical, add marginal histograms, Default: TRUE

`data_input` dataframe, data used to create alluvial plot, Default: NULL

`imp` dataframe, with not more then two columns one of them numeric containing importance measures and one character or factor column containing corresponding variable names as found in training data.

`width` integer, htmlwidget width in pixels, Default: NULL

`height` integer, htmlwidget height in pixels, Default: NULL

`elementId` , htmlwidget elementid, Default: NULL `hoveron` character, one of c(’category’, ’color’, ’dimension’), Sets the hover interaction mode for the parcats diagram.’, ’If ‘category‘, hover interaction take place per category.’, ’If ‘color‘, hover interactions take place per color per category.’, ’If ‘dimension‘, hover interactions take place across all categories per dimension., Default: ’color’

`hoverinfo` character, one of c(’count’, ’probability’, ’count+probability’) set info displayed on mouse hover Default: ’count+probability’

`arrangement,` character, one of c(’perpendicular’, ’freeform’, ’fixed’) ’Sets the drag interaction mode for categories and dimensions.’, ’If ‘perpendicular‘, the categories can only move along a line perpendicular to the paths.’, ’If ‘freeform‘, the categories can freely move on the plane.’, ’If ‘fixed‘, the categories and dimensions are stationary.’, Default: ’perpendicular’

`bundlecolors` logical, ’Sort paths so that like colors are bundled together within each category.’, Default: TRUE

`sortpaths` character, one of c(’forward’, ’backward’), ’Sets the path sorting algorithm.’, ’If ‘forward‘, sort paths based on dimension categories from left to right.’, Default: ’forward’ ’If ‘backward‘, sort paths based on dimensions categories from right to left.’

`labelfont` list, ’Sets the font for the ‘dimension‘ labels.’, Default: list(size = 24, color = ’black’)

`tickfont` list, Sets the font for the ‘category‘ labels.’, Default: NULL

`offset_marginal_histograms`
double, height ratio reserved for parcats diagram, Default: 0.8

`offset_imp` double, width ratio reserved for parcats diagram, Default: 0.9

### Details

most parameters are best left at default values

### Value

htmlwidget

### See Also

`alluvial_wide`, `alluvial_long`, `alluvial_model_response`, `alluvial_model_response_caret`

### Examples

```
library(easyalluvial)

# alluvial wide ---------------------------------
p = alluvial_wide(mtcars2, max_variables = 5)

parcats(p, marginal_histograms = FALSE)
parcats(p, marginal_histograms = TRUE, data_input = mtcars2)

if(check_pkg_installed("randomForest", raise_error = FALSE)) {
  # alluvial for model response --------------------
  df = mtcars2[, ! names(mtcars2) %in% 'ids' ]
  m = randomForest::randomForest( disp ~ ., df)
  imp = m$importance
  dspace = get_data_space(df, imp, degree = 3)
  pred = predict(m, newdata = dspace)
  p = alluvial_model_response(pred, dspace, imp, degree = 3)

  parcats(p, marginal_histograms = TRUE, imp = TRUE, data_input = df)
}
```

## parcats-shiny *Shiny bindings for parcats*

### Description

Output and render functions for using parcats within Shiny applications and interactive Rmd documents.

### Usage

```
parcatsOutput(outputId, width = "100%", height = "100%", inline = FALSE)

render_parcats(expr, env = parent.frame(), quoted = FALSE)
```

### Arguments

| | |
|---|---|
| `outputId` | output variable to read from |
| `width, height` | Must be a valid CSS unit (like `'100%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended. |
| `inline,` | logical, Default: FALSE |
| `expr` | An expression that generates a parcats |
| `env` | The environment in which to evaluate `expr`. |
| `quoted` | Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable. |

### Value

No return value, called for side effects `parcats_demo` *run parcats shiny demo*

## Description

run parcats shiny demo

## Usage

```
parcats_demo()
```

## Value

No return value, called for side effects

## Examples

```
if (interactive()) {
  parcats_demo()
}
```

# Index