Package 'plotluck'

June 26, 2019

Title 'ggplot2' Version of ``I'm Feeling Lucky!''

Version 1.1.1

Description Examines the characteristics of a data frame and a formula to automatically choose the most suitable type of plot out of the following supported options: scatter, violin, box, bar, density, hexagon bin, spine plot, and heat map. The aim of the package is to let the user focus on what to plot, rather than on the ``how'' during exploratory data analysis. It also automates handling of observation weights, logarithmic axis scaling, reordering of factor levels, and overlaying smoothing curves and median lines. Plots are drawn using 'ggplot2'.

Depends R (>= 3.1)

Imports grid, Hmisc (>= 3.17.4), quantreg (>= 5.26), scales (>= 0.4.1), plyr (>= 1.8.4), hexbin (>= 1.27.1), RColorBrewer (>= 1.1.2), ggplot2 (>= 2.2.0)

License MIT + file LICENSE

URL https://github.com/stefan-schroedl/plotluck

BugReports https://github.com/stefan-schroedl/plotluck/issues

LazyData true

Suggests testthat, ggplot2movies, mgcv, nlme, knitr, rmarkdown, gapminder

RoxygenNote 6.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Stefan Schroedl [aut, cre]

Maintainer Stefan Schroedl <stefan.schroedl@gmx.de>

Repository CRAN

Date/Publication 2019-06-26 11:50:03 UTC

R topics documented:

`plotluck` *"I'm feeling lucky" for ggplot*

Description

The purpose of `plotluck` is to let the user focus on *what* to plot, and automate the *how*. Given a dependency formula with up to three variables, it tries to choose the most suitable type of plot. It also automates sampling large datasets, correct handling of observation weights, logarithmic axis scaling, ordering and pruning of factor levels, and overlaying smoothing curves or median lines.

Usage

```
plotluck(data, formula, weights, opts = plotluck.options(), ...)
```

Arguments

`data`	a data frame.
`formula`	an object of class `formula`: a symbolic description of the relationship of up to three variables.

Formula	**Meaning**	**Plot types**
`y~1`	Distribution of single variable	Density, histogram, scatter, dot, bar
`y~x`	One explanatory variable	Scatter, hex, violin, box, spine, heat
`y~x+z`	Two explanatory variables	heat, spine
`y~1|z` or `y~x|z`	One conditional variable	Represented through coloring or facetting
`y~1|x+z`	Two conditional variables	Represented through facetting

In addition to these base plot types, the dot symbol `"."` can also be used, and denotes all variables in the data frame. This gives rise to a lattice or series of plots (use with caution, can be slow).

Formula	**Meaning**
`.~1`	Distribution of each variable in the data frame, separately
`y~.`	Plot y against each variable in the data frame
`.~x`	Plot each variable in the data frame against x
`.~.`	Plot each variable in the data frame against each other.

See also section "Generating multiple plots at once" below.

weights	observation weights or frequencies (optional).
opts	a named list of options (optional); See also plotluck.options.
...	additional parameters to be passed to the respective ggplot2 geom objects.

Value

a ggplot object, or a plotluck.multi object if the dot symbol was used.

Determining the type of plot

Besides the shape of the formula, the algorithm takes into account the type of variables as either numeric, ordered, or unordered factors. Often, it makes sense to treat ordered factors similarly as numeric types.

One-variable numeric (resp. factor) distributions are usually represented by density (resp. Cleveland dot) charts, but can be overridden to histograms or bar plots using the geom option. Density plots come with an overlaid vertical median line.

For two numerical variables, by default a scatter plot is produced, but for high numbers of points a hexbin is preferred (option min.points.hex). These plots come with a smoothing line and standard deviation.

The relation between two factor variables can be depicted best by spine (a.k.a., mosaic) plots, unless they have too many levels (options max.factor.levels.spine.x, max.factor.levels.spine.y, max.factor.levels.spine.z). Otherwise, a heat map is produced.

For a mixed-type (factor/numeric) pair of variables, violin (overridable to box) plots are generated. However, if the resulting graph would contain too many (more than max.factor.levels.violin) violin plots in a row, the algorithm switches automatically. The number of bins of a histogram can be customized with n.breaks.histogram. The default setting, NA, applies a heuristic estimate.

The case of a response two dependent variables (‘y~x+z‘) is covered by either a spine plot (if all are factors) or a heat map.

In many cases with few points for one of the aggregate plots, a scatter looks better (options min.points.density, min.points.violin, min.points.hex).

If each factor combination occurs only once in the data set, we resort to bar plots.

Conditional variables

Conditional variables are represented by either trying to fit into the same graph using coloring (max.factor.levels.color), or by facetting (preferred dimensions facet.num.wrap (resp. facet.num.grid) for one resp. two variables). Numeric vectors are discretized accordingly. Facets are laid out horizontally or vertically according to the plot type, up to maximum dimensions of facet.max.rows and facet.max.cols.

Reordering of factor levels

To better illustrate the relation between an independent factor variable and a dependent numerical variable (or an ordered factor), levels are reordered according to the value of the dependent variable. If no other numeric or ordered variable exists, we sort by frequency.

Instance weights

Argument `weights` allows to specify weights or frequency counts for each row of data. All plots and summary statistics take weights into account when supplied. In scatter and heat maps, weights are indicated either by a shaded disk with proportional area (default) or by jittering (option `dedupe.scatter`), if the number of duplicated points exceeds `min.points.jitter`. The amount of jittering can be controlled with `jitter.x` and `jitter.y`.

Axis scaling

`plotluck` supports logarithmic and log-modulus axis scaling. log-modulus is considered if values are both positive and negative; in this case, the transform function is `f(x) = sign(x) * log(1+abs(x))`.

The heuristic to apply scaling is based on the proportion of total display range that is occupied by the 'core' region of the distribution between the lower and upper quartiles; namely, the fact whether the transform could magnify this region by a factor of at least `trans.log.thresh`.

Missing values

By default, missing (`NA` or `NaN`) values in factors are are shown as a special factor level code"?". They can be removed by setting `na.rm=TRUE`. Conventionally, missing numeric values are not shown.

Sampling

For very large data sets, plots can take a very long time (or even crash R). `plotluck` has a built-in stop-gap: If the data comprises more than `sample.max.rows`, it will be sampled down to that size (taking into account `weights`, if supplied).

Factor preprocessing

Character (resp. logical) vectors are converted to unordered (resp. ordered) factors.

Frequently, when numeric variables have very few values despite sufficient data size, it helps to treat these values as the levels of a factor; this is governed by option `few.unique.as.factor`.

If an unordered factor has too many levels, plots can get messy. In this case, only the `max.factor.levels` most frequent ones are retained, while the rest are merged into a default level `".other."`.

Coloring

If `color` or `fill` aesthetics are used to distinguish different levels or ranges of a variable, the color scheme adjusts to the type. Preferably, a sequential (resp. qualitative) palette is chosen for a numeric/ordered (unordered) factor (`palette.brewer.seq`, `palette.brewer.qual`); see also RColorBrewer.

Generating multiple plots at once

If `formula` contains a dot (`"."`) symbol, the function creates a number of 1D or 2D plots by calling `plotluck` repeatedly. As described above, this allows either single distribution, one-vs-all and all-vs-all variable plots. To save space, rendering is minimal without axis labels.

In the all-vs-all case, the diagonal contains 1D distribution plots, analogous to the behavior of the default plot method for data frames, see `plot.data.frame`.

With setting `in.grid=FALSE`, plots are produced in a sequence, otherwise together on one or multiple pages, if necessary (default). Page size is controlled by `multi.max.rows` and `multi.max.cols`.

With `entropy.order=TRUE`, plots are sorted by an estimate of empirical conditional entropy, with the goal of prioritizing the more predictive variables. Set `verbose=TRUE` if you want to see the actual values. For large data sets the calculation can be time consuming; entropy calculation can be suppressed by setting `multi.entropy.order=FALSE`.

@note The return value is an object of class `plotluck_multi`. This class does not have any functionality; its sole purpose is to make this function work in the same way as `ggplot` and `plotluck`, namely, do the actual drawing if and only if the return value is not assigned.

Debugging

With the option `verbose=TRUE` turned on, the function will print out information about the chosen and applicable plot types, ordering, log scaling, etc.

Column name matching

Variable names can be abbreviated if they match a column name uniquely by prefix.

Remarks on supported plot types

By default, `plotluck` uses violin and density plots in place of the more traditional box-and-whisker plots and histograms; these modern graph types convey the shape of a distribution better. In the former case, summary statistics like mean and quantiles are less useful if the distribution is not unimodal; a wrong choice of the number of bins of a histogram can create misleading artifacts.

Following Cleveland's advice, factors are plotted on the y-axis to make labels most readable and compact at the same time. This direction can be controlled using option `prefer.factors.vert`.

Due to their well-documented problematic aspects, pie charts and stacked bar graphs are not supported.

With real-world data (as opposed to smooth mathematical functions), three-dimensional scatter, surface, or contour plots can often be hard to read if the shape of the distribution is not suitable, data coverage is uneven, or if the perspective is not carefully chosen depending on the data. Since they usually require manual tweaking, we have refrained from incorporating them.

Remarks on the use of options

For completeness, we have included the description of option parameters in the current help page. However, the tenet of this function is to be usable "out-of-the-box", with no or very little manual tweaking required. If you find yourself needing to change option values repeatedly or find the presets to be suboptimal, please contact the author.

Limitations

`plotluck` is designed for generic out-of-the-box plotting, and not suitable to produce more specialized types of plots that arise in specific application domains (e.g., association, stem-and-leaf, star

plots, geographic maps, etc). It is restricted to at most three variables. Parallel plots with variables on different scales (such as time series of multiple related signals) are not supported.

See Also

plotluck.options, sample.plotluck, ggplot

Examples

```
# Single-variable density
data(diamonds, package='ggplot2')
plotluck(diamonds, price~1)
invisible(readline(prompt="Press [enter] to continue"))

# Violin plot
data(iris)
plotluck(iris, Species~Petal.Length)
invisible(readline(prompt="Press [enter] to continue"))

# Scatter plot
data(mpg, package='ggplot2')
plotluck(mpg, cty~model)
invisible(readline(prompt="Press [enter] to continue"))

# Spine plot
data(Titanic)
plotluck(as.data.frame(Titanic), Survived~Class+Sex, weights=Freq)
invisible(readline(prompt="Press [enter] to continue"))

# Facetting
data(msleep, package='ggplot2')
plotluck(msleep, sleep_total~bodywt|vore)
invisible(readline(prompt="Press [enter] to continue"))

# Heat map
plotluck(diamonds, price~cut+color)


# Multi plots
# All 1D distributions
plotluck(iris, .~1)

# 2D dependencies with one fixed variable on vertical axis
plotluck(iris, Species~.)

# See also tests/testthat/test_plotluck.R for more examples!
```

`plotluck.options` *Create options structure for* `plotluck`

Description

Create options structure for `plotluck`

Usage

```
plotluck.options(opts, ...)
```

Arguments

`opts`	An (optional) named list to start with. Anything not specified in ... will be inherited from opts.
`...`	Parameters to override default settings

Value

A named list of options, usable as argument to function `plotluck`.

`plotluck` accepts a list of options to modify its behavior. Calling `plotluck.options` without arguments produces a list with the default values. Specifying any number of attribute/value pairs overrides them selectively.

Option	Default	Comment
`na.rm`	`FALSE`	Do not show missing factor values as separate level.
`geom`	`"auto"`	Override type of plot; the available types for a given formula and variable
`sample.max.rows`	`100000`	If the data set has more rows than that, sample it down.
`trans.log.thresh`	`2`	Threshold for logarithmic axis scaling. Visible magnification factor of th
`n.breaks.histogram`	`NA`	Override the number of histogram breaks.
`min.points.hex`	`5000`	Minimum data points required to display a hexbin plot.
`min.points.density`	`20`	Minimum data points required to display a density or histogram plot.
`min.points.violin`	`20`	Minimum data points required to display a violin or box plot.
`resolution.heat`	`30`	Grid spacing for heat maps.
`dedupe.scatter`	`'area'`	To represent multiple instances of the same coordinates in scatter plot: sc
`min.points.jitter`	`3`	Minimum number of coordinate duplicates to start jittering points.
`jitter.x`	`0.4`	Amount of jitter to apply in horizontal direction, as a fraction of resolutio
`jitter.y`	`0.4`	Amount of jitter to apply in vertical direction, as a fraction of resolution.
`few.unique.as.factor`	`5`	If a numeric variable has less than that many unique values, make it an or
`max.factor.levels`	`30`	For factors with more than that many levels, least frequent ones will be p
`max.factor.levels.color`	`3`	Maximum number of factor levels that can be represented as colors in the
`max.factor.levels.violin`	`20`	Maximum number of levels to plot violins; rather switch to box plot.
`max.factor.levels.spine.x`	`20`	Maximum number of levels to plot in x-direction in a spine plot.
`max.factor.levels.spine.y`	`10`	Maximum number of levels to plot in y-direction in a spine plot.
`max.factor.levels.spine.z`	`5`	Maximum number of levels to represent as colors in a spine plot.
`spine.plot.margin.x`	`0.05`	Horizontal gap between rectangles in a spine plot.
`spine.plot.margin.y`	`0.02`	Vertical gap between rectangles in a spine plot.

facet.max.cols	10	Maximum number of facet columns for conditional variables.
facet.max.rows	10	Maximum number of facet rows for conditional variables.
facet.num.wrap	6	Preferred number of facets for single conditional variable.
facet.num.grid	3	Preferred number of facets for each of two conditional variables.
prefer.factors.vert	TRUE	In mixed numeric/factor plots, use vertical axis for the factor.
fill.default	"deepskyblue"	Default fill color for density and histogram plots.
palette.brewer.seq	"YlGn"	Sequential brewer palette name.
palette.brewer.qual	"Set1"	Qualitative brewer palette name.
multi.entropy.order	TRUE	Use estimated conditional entropy to order multi-plots.
multi.max.rows	6	Maximum number of rows for multi-plots.
multi.max.cols	6	Maximum number of columns for multi-plots.
multi.in.grid	TRUE	In multi-plots, make a page with multiple plots, or generate each one sep
verbose	FALSE	Print information about plot types, ordering, scaling, etc.

Note

plotluck's aim is to provide a function that is usable "out-of-the-box", with no or very little manual tweaking. If you find yourself needing to change option values repeatedly or find the presets to be suboptimal, please contact the author.

See Also

plotluck

Examples

```
# list all default options
plotluck.options()

data(iris)
# default with violin plot
plotluck(iris, Petal.Length~Species)

# use box-and-whiskers plot instead
plotluck(iris, Petal.Length~Species, opts=plotluck.options(geom='box'))
```

---

sample.plotluck *Run* plotluck *for a randomly generated formula.*

Description

sample.plotluck samples a formula as follows:

- Uniformly draw the number of variables (1-3).
- For each variable, uniformly choose one of the existing variable types from the data set (numeric, ordered or unordered factor).
- Uniformly select one of the data frame columns of that type.

Usage

```
sample.plotluck(data, ...)
```

Arguments

`data`	a data frame
`...`	additional parameters to be passed to `plotluck`, such as `weights` and `opts`.

Value

a ggplot2 object.

See Also

`plotluck`

Examples

```
set.seed(42)
data(iris)
sample.plotluck(iris)
```

Index