Package: autocogs (via r-universe)

September 4, 2024

Title Automatic Cognostic Summaries

Version 0.1.4

Description Automatically calculates cognostic groups for plot objects and list column plot objects. Results are returned in a nested data frame.

Depends R (>= 3.4.0)

License MIT + file LICENSE

Encoding UTF-8

Imports dplyr, checkmate, MASS, broom, moments, diptest, mclust, ggplot2, tibble, utils, hexbin, progress

RoxygenNote 7.1.1

Collate 'autocog.R' 'known_cog_groups.R' 'add_cog_group.R' 'field_info.R' 'add_cog_group_.R' 'known_layer_cogs.R' 'add_layer_cogs.R' 'add_layer_cogs_.R' 'cog_desc.R' 'cog_spec.R' 'layer_count.R' 'layer_info.R' 'of_type.R' 'plot_class.R' 'plot_cogs.R'

Suggests testthat, covr

URL https://github.com/schloerke/autocogs

BugReports https://github.com/schloerke/autocogs/issues

Repository https://schloerke.r-universe.dev

RemoteUrl https://github.com/schloerke/autocogs

RemoteRef HEAD

RemoteSha bc07bd4b2c134f2cf788d1016d174cf5a209d40a

Contents

`add_cog_group` *Add a cognostic group*

Description

Add a new cognostic to be used when calculating automatic cognostics.

Usage

```
add_cog_group(name, fields, description = NA, fn, ...)
```

Arguments

`name`	Name of cognostic group
`fields`	`data.frame` of `'dimension'` and `'type'` columns. `dplyr::bind_rows()` of `field_info` outputs for convenience
`description`	Description of cognostic group
`fn`	function to calculate a cognostic group. May return a named list or a single row tibble. Each value of the return data should be the output of `cog_desc`
`...`	ignored

`add_layer_cogs` *Add plot layer cognostics*

Description

Add a new set of cognostic groups for a given plot layer. If the plot layer is found, the corresponding cognostic groups will be calculated.

Usage

```
add_layer_cogs(name, description, cog_groups, kind = "ggplot", ...)
```

Arguments

name	Name of plot layer. This should match the output of the "name" values of layer_info
description	Description of cognostic group
cog_groups	A data.frame (or tibble) containing the columns: "cog_group", "cols", "name". "cog_group" column should contain a string value of a known cognostic group. "cols" should be a single value or vector of column names to use from the data supplied by layer_info during calculations. "name" should contain the final storage name of the cognostic group.
kind	String value that will match the output of plot_class of the desired plot object
...	ignored

---

autocog *Auto cognostic function*

Description

Calculate an auto cognostic function given a name

Usage

```
autocog(.name, ..., .fn_only = FALSE)
```

Arguments

.name	name of a known cognostic
...	arguments passed onto the found function
.fn_only	boolean that determines if the function should be returned

Examples

```
autocog("univariate_continuous", iris$Sepal.Length)
fn <- autocog("univariate_continuous", .fn_only = TRUE)
fn(iris$Sepal.Length)
```

`autocog_` *Default Cognostic Group Functions*

Description

These set of functions comprise the default cognostic groups. Each function produces it's own cognostic information given the required pieces of data.

The functions' print method will display the description. `autocog_*` functions will take the `known_cog_groups()` functions and format the output into a single row tibble. Any new known cognostic group function, NAME, will create a function called autocog_NAME, which may be called.

Default Cognostic Group Functions:

- autocog_bivariate_continuous
- autocog_bivariate_counts
- autocog_bivariate_step
- autocog_boxplot
- autocog_density_2d_continuous
- autocog_density_continuous
- autocog_grouped_counts
- autocog_grouped_testing
- autocog_hex_counts
- autocog_histogram_counts
- autocog_linear_model
- autocog_loess_model
- autocog_pairwise_counts
- autocog_quantile_quantile
- autocog_scagnostics
- autocog_smooth_line
- autocog_square_counts
- autocog_univariate_continuous
- autocog_univariate_counts
- autocog_univariate_discrete

Arguments

`x`	data that should appear on an x axis
`y`	data that should appear on an y axis
`...`	ignored
`direction`	step direction. Defaults to `"hv"`

na.rm	should NA points be removed when performing calculations
h, n, bins, binwidth, clusters, bw, adjust, kernel, trim, group, groups, center, boundary, closed, pad, breaks, weights, formula, method_args, span, distribution, dparams, method, se, fullrange, xseq, level, origin, drop	parameters usually set by corresponding "geoms" to be used within ggplot2 Stat* methods

See Also

known_cog_groups()

Examples

```
autocog_bivariate_continuous
autocog_bivariate_continuous(iris$Sepal.Length, iris$Sepal.Width)
```

---

cog_desc *Cognostic*

Description

Add a description to a cognostic (subset metric)

Usage

```
cog_desc(x, desc = NULL)
```

Arguments

x	univariate scalar
desc	description of x

Examples

```
cog_desc(mean(1:10), "mean of 10 numbers")
```

cog_group *Cog group data frame*

Description

Make a cog group data frame to be passed into add_layer_cogs

Usage

```
cog_group(...)
```

Arguments

...	sets of three values to fill in 'cog_group', 'cols', and 'name'

Examples

```
cog_group(
  "univariate_discrete", "x", "_x",
  "univariate_counts", "x", "_n"
)
cog_group(
  "univariate_continuous", "x", "_x",
  "univariate_continuous", "y", "_y",
  "bivariate_continuous", c("x", "y"), "_bivar",
  "scagnostics", c("x", "y"), "_scagnostic",
  "bivariate_counts", c("x", "y"), "_n"
)
```

cog_spec *Cognostic Specification*

Description

Cognostic Specification

Usage

```
cog_spec(
  bivariate_continuous = TRUE,
  bivariate_counts = TRUE,
  bivariate_step = TRUE,
  boxplot = TRUE,
  density_2d_continuous = TRUE,
  density_continuous = TRUE,
  grouped_counts = TRUE,
```

```
  grouped_testing = TRUE,
  hex_counts = TRUE,
  histogram_counts = TRUE,
  linear_model = TRUE,
  loess_model = TRUE,
  pairwise_counts = TRUE,
  quantile_quantile = TRUE,
  scagnostics = TRUE,
  smooth_line = TRUE,
  square_counts = TRUE,
  univariate_continuous = TRUE,
  univariate_counts = TRUE,
  univariate_discrete = TRUE,
  ...,
  .keep_layer = TRUE
)

as_cog_specs(p, specs)
```

Arguments

bivariate_continuous, bivariate_counts, bivariate_step, boxplot, density_2d_continuous, density_continuous, grouped_counts, grouped_testing, hex_counts, histogram_counts, linear_model, loess_model, pairwise_counts, quantile_quantile, scagnostics, smooth_line, square_counts, univariate_continuous, univariate_counts, univariate_discrete	names of cognostic groups to calculate. The boolean value (TRUE) supplied to each argument determines if the value should be displayed if possible or removed if possible.
...	ignored. Will cause error if any are supplied
.keep_layer	boolean (TRUE) that determines if the layer should be kept at all
p	plot object in question
specs	list of cog_spec outputs for each layer of the plot object

Value

cognostic specification that determines which cogs are added or removed if possible

Examples

```
# example cog specifications
# display like normal
cog_spec(); TRUE
# remove scagnostics
cog_spec(scagnostics = FALSE)
# remove layer
cog_spec(.keep_layer = FALSE); FALSE
```

```
# set up data
p <- ggplot2::qplot(Sepal.Length, Sepal.Width, data = iris, geom = c("point", "smooth"))
dt <- tibble::tibble(panel = list(p))

# compute cognostics like normal
add_panel_cogs(dt)

# do not compute scagnostics for geom_point cognostics
# compute geom_smooth cognostics
add_panel_cogs(dt, spec = list(cog_spec(scagnostics = FALSE), TRUE))

# do not compute scagnostics for geom_point cognostics
# do not compute geom_smooth cognostics
add_panel_cogs(dt, spec = list(cog_spec(scagnostics = FALSE), FALSE))
```

`field_info` *Field Type Information*

Description

Field Type Information

Usage

```
field_info(
  dimension = c("x", "y", "z", "group", "any"),
  type = c("continuous", "discrete", "date", "any")
)
```

Arguments

`dimension`	field name. Use one of the listed options provided
`type`	field type. Use one of the listed options provided

`known_cog_groups` *Cognostic Group information*

Description

To add more cognostic groups, please see `add_cog_group()`

Usage

```
known_cog_groups()
```

Examples

```
known_cog_groups()
```

known_layer_cogs *Layer Cognostic groups*

Description

Display all layer cognostic information to be paired with information from `known_cog_groups()`.

Usage

```
known_layer_cogs()
```

Examples

```
known_layer_cogs()
```

layer_count *Plot layer count*

Description

Retrieves the number of layers in a given plot

Usage

```
layer_count(p)

## Default S3 method:
layer_count(p)

## S3 method for class 'ggplot'
layer_count(p)
```

Arguments

p	plot object

Value

number

Examples

```
library(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) + geom_point()
layer_count(p) # 1
layer_count(p + geom_smooth(method = "lm") + geom_density_2d()) # 3
```

layer_info *Data List*

Description

Data List

Usage

```
layer_info(p, keep = TRUE, ...)

## Default S3 method:
layer_info(p, keep = TRUE, ...)

## S3 method for class 'ggplot'
layer_info(p, keep = TRUE, ...)
```

Arguments

p	plot object
keep	boolean vector (size = 1 or length(plot$layers)). Determines if that layer should have cognostics calculated
...	parameters passed on to corresponding layer_info

Examples

```
require(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point(data = mpg, mapping = aes(cty, hwy))
layer_info(p)
```

panel_cogs *Panel cognostics*

Description

Return or concatenate panel cognostics. For each panel (plot) in the panel column, cognostics will be calculated for each panel. The result will be returned in a nested tibble.

Usage

```
panel_cogs(dt, panel_col = "panel", ...)

add_panel_cogs(dt, panel_col = "panel", ...)
```

Arguments

dt	data to be used
panel_col	panel column to be used in dt
...	parameters passed to layer_info

---

plot_class *Plot class*

Description

First class of the plot object. Exception is ggplot2 as many objects are of class 'gg'

Usage

```
plot_class(p)

## Default S3 method:
plot_class(p)

## S3 method for class 'gg'
plot_class(p)

## S3 method for class 'ggplot'
plot_class(p)
```

Arguments

p	plot object to retrieve class from

Examples

```
library(ggplot2)
p <- qplot(Sepal.Length, Sepal.Width, data = iris)
plot_class(p)
```

Index