Package 'cassowaryr'

October 12, 2022

Title Compute Scagnostics on Pairs of Numeric Variables in a Data Set

Version 2.0.0

Description Computes a range of scatterplot diagnostics (scagnostics) on pairs of numerical variables in a data set. A range of scagnostics, including graph and association-based scagnostics described by Leland Wilkinson and Graham Wills (2008) <doi:10.1198/106186008X320465> and association-based scagnostics described by Katrin Grimm (2016,ISBN:978-3-8439-3092-5) can be computed. Summary and plotting functions are provided.

License GPL-3

Encoding UTF-8

LazyData true

URL https://github.com/numbats/cassowaryr

BugReports https://github.com/numbats/cassowaryr/issues

Depends R (>= 4.0.0)

Imports igraph, alphahull (>= 2.5), splancs, interp, energy, dplyr, ggplot2, magrittr, progress, tibble, stats, tidyselect

RoxygenNote 7.2.0

Suggests rmarkdown, knitr, mgcv, GGally, tidyr, testthat (>= 3.0.0), covr

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Harriet Mason [aut, cre], Stuart Lee [aut] (<https://orcid.org/0000-0003-1179-8436>), Ursula Laa [aut] (<https://orcid.org/0000-0002-0249-6439>), Di Cook [aut] (<https://orcid.org/0000-0002-3813-7155>)

Maintainer Harriet Mason <harriet.m.mason@gmail.com>

Repository CRAN

Date/Publication 2022-08-09 12:30:02 UTC

R topics documented:

`anscombe_tidy` *Data from Anscombe's famous example in tidy format*

Description

All variables and pairs of variables have same summary statistics but are very different data, as can be seen by visualisation.

Format

A tibble with 44 observations and 3 variables

set label of the data set, each set has 11 observations

x variable for horizontal axis

y variable for vertical axis

`calc_scags` *Compute selected scagnostics on subsets*

Description

Compute selected scagnostics on subsets

Usage

```
calc_scags(
  x,
  y,
  scags = c("outlying", "stringy", "striated", "striated2", "clumpy", "clumpy2",
    "sparse", "skewed", "convex", "skinny", "monotonic", "splines", "dcor"),
  out.rm = TRUE,
  euclid = FALSE
)
```

Arguments

`x`	numeric vector
`y`	numeric vector
`scags`	collection of strings matching names of scagnostics to calculate: outlying, stringy, striated, striated2, striped, clumpy, clumpy2, sparse, skewed, convex, skinny, monotonic, splines, dcor
`out.rm`	logical indicator to indicate if outliers should be removed before calculating non outlying measures
`euclid`	logical indicator to use Euclidean distance

Value

A data frame that gives the single plot's scagnostic score.

See Also

calc_scags_wide

Examples

```
# Calculate selected scagnostics on a single pair
calc_scags(anscombe$x1, anscombe$y1, scags=c("monotonic", "outlying"))

# Compute on long form data, or subsets
# defined by a categorical variable
require(dplyr)
datasaurus_dozen %>%
  group_by(dataset) %>%
  summarise(calc_scags(x,y, scags=c("monotonic", "outlying", "convex")))
```

`calc_scags_wide` *Compute scagnostics on all possible scatter plots for the given data*

Description

Compute scagnostics on all possible scatter plots for the given data

Usage

```
calc_scags_wide(
  all_data,
  scags = c("outlying", "stringy", "striated", "striated2", "clumpy", "clumpy2",
    "sparse", "skewed", "convex", "skinny", "monotonic", "splines", "dcor"),
  out.rm = TRUE,
  euclid = FALSE
)
```

Arguments

`all_data`	tibble of multivariate data on which to compute scagnostics
`scags`	collection of strings matching names of scagnostics to calculate: outlying, stringy, striated, striated2, striped, clumpy, clumpy2, sparse, skewed, convex, skinny, monotonic, splines, dcor
`out.rm`	logical indicator to indicate if outliers should be removed before calculating non outlying measures
`euclid`	logical indicator to use Euclidean distance

Value

A data frame that gives the data's scagnostic scores for each possible variable combination.

See Also

calc_scags

Examples

```
# Calculate selected scagnostics
data(pk)
calc_scags_wide(pk[,2:5], scags=c("outlying","monotonic"))
```

`datasaurus_dozen` *datasaurus_dozen data*

Description

From the datasauRus package. A modern update of Anscombe. All plots have same x and y mean, variance and correlation, but look different visually.

All variables and pairs of variables have same summary statistics but are very different data, as can be seen by visualisation.

Format

A tibble with 1,846 observations and 3 variables

dataset label of data set

x variable for horizontal axis

y variable for vertical axis

A tibble with 142 observations and 26 variables

away_x, away_y x and y variables for away data

bullseye_x, bullseye_y x and y variables for bullseye data

circle_x, circle_y x and y variables for circle data

dino_x, dino_y x and y variables for dino data

dots_x, dots_y x and y variables for dots data

h_lines_x, h_lines_y x and y variables for h_lines data

high_lines_x, high_lines_y x and y variables for high_lines data

slant_down_x, slant_down_y x and y variables for slant_down data

slant_up_x, slant_up_y x and y variables for slant_up data

star_x, star_y x and y variables for star data

v_lines_x, v_lines_y x and y variables for v_lines data

wide_lines_x, wide_lines_y x and y variables for wide_lines data

star_x, star_y x and y variables for star data

x_shape_x, x_shape_y x and y variables for x_shape data

draw_alphahull *Drawing the alphahull*

Description

This function will draw the alphahull for a scatterplot.

Usage

```
draw_alphahull(x, y, alpha = 0.5, clr = "black", fill = FALSE, out.rm = TRUE)
```

Arguments

x	numeric vector
y	numeric vector
alpha	transparency value of points
clr	optional colour of points and lines, default black
fill	Fill the polygon
out.rm	option to return the outlier removed alphahull

Value

A alphahull::ahull(del, alpha = alpha) "gg" object that draws the plot's alpha hull.

Examples

```
require(dplyr)
require(ggplot2)
require(alphahull)
data("features")
nl <- features %>% filter(feature == "clusters")
draw_alphahull(nl$x, nl$y)
```

draw_convexhull *Drawing the Convex Hull*

Description

This function will draw the Convex Hull for a scatterplot.

Usage

```
draw_convexhull(x, y, alpha = 0.5, clr = "black", fill = FALSE, out.rm = TRUE)
```

Arguments

x	numeric vector
y	numeric vector
alpha	transparency value of points
clr	optional colour of points and lines, default black
fill	Fill the polygon
out.rm	option to return the outlier removed convex hull

Value

A "gg" object that draws the plot's convex hull.

Examples

```
require(dplyr)
require(ggplot2)
data("features")
nl <- features %>% filter(feature == "clusters")
draw_convexhull(nl$x, nl$y, fill=TRUE, out.rm=FALSE)
```

---

draw_mst *Drawing the MST*

Description

This function will draw the MST for a scatterplot.

Usage

```
draw_mst(x, y, alpha = 0.5, out.rm = TRUE)
```

Arguments

x	numeric vector
y	numeric vector
alpha	The alpha value used to build the graph object. Larger values allow points further apart to be connected.
out.rm	option to return the outlier removed MST

Value

A "gg" object that draws the plot's MST.

Examples

```
require(dplyr)
require(ggplot2)
data("features")
nl <- features %>% filter(feature == "nonlinear2")
draw_mst(nl$x, nl$y)
```

---

features *Simulated data with special features*

Description

Simulated data with common features that might be seen in 2D data. Variable are feature, x, y.

Format

A tibble with 1,013 observations and 3 variables, and 15 different patterns

feature label of data set

x variable for horizontal axis

y variable for vertical axis

---

numbat *A toy data set with a numbat shape hidden among noise variables*

Description

There are 7 variables (x1-x7) and 2,100 observations. Variables 4 and 7 have the numbat. The rest are noise. Group A has the numbat, and group B is all noise.

---

pk *Parkinsons data from UCI machine learning archive*

Description

Biomedical voice measurements from 31 people, 23 with Parkinson's disease (PD). Each column in the table is a particular voice measure, and each row corresponds one of 195 voice recording from these individuals ("name" column). The main aim of the data is to discriminate healthy people from those with PD, according to "status" column which is set to 0 for healthy and 1 for PD.

Format

A tibble with 1,013 observations and 3 variables

name ASCII subject name and recording number

`MDVP:Fo(Hz)` Average vocal fundamental frequency

`MDVP:Fhi(Hz)` Maximum vocal fundamental frequency

`MDVP:Flo(Hz)` Minimum vocal fundamental frequency

`MDVP:Jitter`,`MDVP:Jitter(Abs)`,`MDVP:RAP`,`MDVP:PPQ`,`Jitter:DDP` Several measures of variation in fundamental frequency

`MDVP:Shimmer`,`MDVP:Shimmer(dB)`,`Shimmer:APQ3`,`Shimmer:APQ5`,`MDVP:APQ`,`Shimmer:DDA` Several measures of variation in amplitude

`NHR`,`HNR` Two measures of ratio of noise to tonal components in the voice

`status` Health status of the subject (one) - Parkinson's, (zero) - healthy

`RPDE`,`D2` Two nonlinear dynamical complexity measures

`DFA` Signal fractal scaling exponent

`spread1`,`spread2`,`PPE` Three nonlinear measures of fundamental frequency variation

Details

The data is available at The UCI Machine Learning Repository in ASCII CSV format. The rows of the CSV file contain an instance corresponding to one voice recording. There are around six recordings per patient, the name of the patient is identified in the first column.

The data are originally analysed in: Max A. Little, Patrick E. McSharry, Eric J. Hunter, Lorraine O. Ramig (2008), 'Suitability of dysphonia measurements for telemonitoring of Parkinson's disease', IEEE Transactions on Biomedical Engineering.

`scree` *Pre-processing to generate scagnostic measures*

Description

Pre-processing to generate scagnostic measures

Usage

```
scree(x, y, binner = NULL, ...)
```

Arguments

`x, y`	numeric vectors
`binner`	an optional function that bins the x and y vectors prior to triangulation
`...`	other args

Value

An object of class "scree" that consists of three elements:

- `del`: the Delauney-Voronoi tesselation from `alphahull::delvor()`
- `weights`: the lengths of each edge in the Delauney triangulation
- `alpha`: the radius or `alpha` value that will be used to generate the alphahull

Examples

```
x <- runif(100)
y <- runif(100)
scree(x,y)
```

`sc_clumpy` *Compute clumpy scagnostic measure using MST*

Description

Compute clumpy scagnostic measure using MST

Usage

```
sc_clumpy(x, y)

## Default S3 method:
sc_clumpy(x, y)

## S3 method for class 'scree'
sc_clumpy(x, y = NULL)

## S3 method for class 'igraph'
sc_clumpy(x, y)
```

Arguments

`x`	numeric vector of x values
`y`	numeric vector of y values

Value

A "numeric" object that gives the plot's clumpy score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(clumpy = sc_clumpy(x,y))
sc_clumpy(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

---

sc_clumpy2 *Compute adjusted clumpy measure using MST*

Description

Compute adjusted clumpy measure using MST

Usage

```
sc_clumpy2(x, y)

## Default S3 method:
sc_clumpy2(x, y)

## S3 method for class 'scree'
sc_clumpy2(x, y = NULL)

## S3 method for class 'igraph'
sc_clumpy2(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's clumpy2 score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(clumpy = sc_clumpy2(x,y))
sc_clumpy2(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

sc_clumpy_r *Compute robust clumpy scagnostic measure using MST*

Description

Compute robust clumpy scagnostic measure using MST

Usage

```
sc_clumpy_r(x, y)

## Default S3 method:
sc_clumpy_r(x, y)

## S3 method for class 'scree'
sc_clumpy_r(x, y = NULL)

## S3 method for class 'igraph'
sc_clumpy_r(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's robust clumpy score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(clumpy = sc_clumpy_r(x,y))
sc_clumpy_r(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

sc_convex *Compute convex scagnostic measure*

Description

Compute convex scagnostic measure

Usage

```
sc_convex(x, y)

## Default S3 method:
sc_convex(x, y)

## S3 method for class 'scree'
sc_convex(x, y = NULL)

## S3 method for class 'list'
sc_convex(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's convex score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(convex = sc_convex(x,y))
sc_convex(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

sc_dcor *Distance correlation index.*

Description

(Taken from tourr package) Computes the distance correlation based index on 2D projections of the data.

Usage

```
sc_dcor(x, y)
```

Arguments

x	numeric vector
y	numeric vector

Value

A "numeric" object that gives the plot's dcor score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe)
anscombe_tidy <- anscombe %>%
pivot_longer(cols = everything(),
  names_to = c(".value", "set"),
  names_pattern = "(.)(.)")
ggplot(anscombe_tidy, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_dcor(anscombe$x1, anscombe$y1)
sc_dcor(anscombe$x2, anscombe$y2)
sc_dcor(anscombe$x3, anscombe$y3)
sc_dcor(anscombe$x4, anscombe$y4)
```

sc_monotonic *Measure of Spearman Correlation*

Description

Measure of Spearman Correlation

Usage

```
sc_monotonic(x, y)
```

Arguments

x	numeric vector
y	numeric vector

Value

A "numeric" object that gives the plot's monotonic score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe)
anscombe_tidy <- anscombe %>%
pivot_longer(cols = everything(),
  names_to = c(".value", "set"),
  names_pattern = "(.)(.)")
ggplot(anscombe_tidy, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_monotonic(anscombe$x1, anscombe$y1)
sc_monotonic(anscombe$x2, anscombe$y2)
sc_monotonic(anscombe$x3, anscombe$y3)
sc_monotonic(anscombe$x4, anscombe$y4)
```

sc_outlying *Compute outlying scagnostic measure using MST*

Description

Compute outlying scagnostic measure using MST

Usage

```
sc_outlying(x, y)

## Default S3 method:
sc_outlying(x, y)

## S3 method for class 'scree'
sc_outlying(x, y = NULL)

## S3 method for class 'igraph'
sc_outlying(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's outlying score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
ggplot(datasaurus_dozen, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~dataset, ncol=3, scales = "free")
sc_outlying(datasaurus_dozen_wide$dino_x, datasaurus_dozen_wide$dino_y)
sc_outlying(datasaurus_dozen_wide$dots_x, datasaurus_dozen_wide$dots_y)
sc_outlying(datasaurus_dozen_wide$h_lines_x, datasaurus_dozen_wide$h_lines_y)
```

---

sc_skewed *Compute skewed scagnostic measure using MST*

Description

Compute skewed scagnostic measure using MST

Usage

```
sc_skewed(x, y)

## Default S3 method:
sc_skewed(x, y)
```

```
## S3 method for class 'scree'
sc_skewed(x, y = NULL)

## S3 method for class 'igraph'
sc_skewed(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's skewed score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe_tidy)
ggplot(datasaurus_dozen, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~dataset, ncol=3, scales = "free")
sc_skewed(datasaurus_dozen_wide$dots_x, datasaurus_dozen_wide$dots_y)
sc_skewed(datasaurus_dozen_wide$h_lines_x, datasaurus_dozen_wide$h_lines_y)
sc_skewed(datasaurus_dozen_wide$x_shape_x, datasaurus_dozen_wide$x_shape_y)
```

---

sc_skinny *Compute skinny scagnostic measure*

Description

Compute skinny scagnostic measure

Usage

```
sc_skinny(x, y)

## Default S3 method:
sc_skinny(x, y)

## S3 method for class 'scree'
sc_skinny(x, y = NULL)

## S3 method for class 'list'
sc_skinny(x, y = NULL)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's skinny score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(skinny = sc_skinny(x,y))
sc_skinny(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

---

sc_sparse *Compute sparse scagnostic measure using MST*

Description

Compute sparse scagnostic measure using MST

Usage

```
sc_sparse(x, y)

## Default S3 method:
sc_sparse(x, y)

## S3 method for class 'scree'
sc_sparse(x, y = NULL)

## S3 method for class 'igraph'
sc_sparse(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's sparse score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
ggplot(datasaurus_dozen, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~dataset, ncol=3, scales = "free")
sc_sparse(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
sc_sparse(datasaurus_dozen_wide$circle_x, datasaurus_dozen_wide$circle_y)
sc_sparse(datasaurus_dozen_wide$dino_x, datasaurus_dozen_wide$dino_y)
```

`sc_sparse2` *Compute adjusted sparse measure using the alpha hull*

Description

Compute adjusted sparse measure using the alpha hull

Usage

```
sc_sparse2(x, y)

## Default S3 method:
sc_sparse2(x, y)

## S3 method for class 'scree'
sc_sparse2(x, y = NULL)

## S3 method for class 'list'
sc_sparse2(x, y = NULL)
```

Arguments

`x`	numeric vector of x values
`y`	numeric vector of y values

Value

A "numeric" object that gives the plot's sparse2 score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe_tidy)
ggplot(anscombe_tidy, aes(x=x, y=y)) +
```

```
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_sparse2(anscombe$x1, anscombe$y1)
```

`sc_splines` *Spline based index.*

Description

(Taken from tourr git repo) Compares the variance in residuals of a fitted spline model to the overall variance to find functional dependence in 2D projections of the data.

Usage

```
sc_splines(x, y)
```

Arguments

x	numeric vector
y	numeric vector

Value

A "numeric" object that gives the plot's spines score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe)
anscombe_tidy <- anscombe %>%
pivot_longer(cols = everything(),
  names_to = c(".value", "set"),
  names_pattern = "(.)(.)")
ggplot(anscombe_tidy, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_splines(anscombe$x1, anscombe$y1)
sc_splines(anscombe$x2, anscombe$y2)
sc_splines(anscombe$x3, anscombe$y3)
```

`sc_striated` *Compute striated scagnostic measure using MST*

Description

Compute striated scagnostic measure using MST

Usage

```
sc_striated(x, y)

## Default S3 method:
sc_striated(x, y)

## S3 method for class 'scree'
sc_striated(x, y = NULL)

## S3 method for class 'igraph'
sc_striated(x, y)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's striated score.

Examples

```
require(ggplot2)
require(dplyr)
data(anscombe_tidy)
ggplot(anscombe_tidy, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_striated(anscombe$x1, anscombe$y1)
sc_striated(anscombe$x2, anscombe$y2)
```

`sc_striated2` *Compute angle adjusted striated measure using MST*

Description

Compute angle adjusted striated measure using MST

Usage

```
sc_striated2(x, y)

## Default S3 method:
sc_striated2(x, y)

## S3 method for class 'scree'
sc_striated2(x, y = NULL)

## S3 method for class 'igraph'
sc_striated2(x, y)
```

Arguments

x	numeric vector of x values, or an MST object
y	numeric vector of y values, or a scree object

Value

A "numeric" object that gives the plot's striated2 score.

Examples

```
require(ggplot2)
require(dplyr)
ggplot(features, aes(x=x, y=y)) +
   geom_point() +
   facet_wrap(~feature, ncol = 5, scales = "free")
features %>% group_by(feature) %>% summarise(striated = sc_striated2(x,y))
sc_striated2(datasaurus_dozen_wide$away_x, datasaurus_dozen_wide$away_y)
```

`sc_stringy` *Compute stringy scagnostic measure using MST*

Description

Compute stringy scagnostic measure using MST

Usage

```
sc_stringy(x, y)

## Default S3 method:
sc_stringy(x, y)

## S3 method for class 'scree'
sc_stringy(x, y = NULL)

## S3 method for class 'igraph'
sc_stringy(x, y = NULL)
```

Arguments

x	numeric vector of x values
y	numeric vector of y values

Value

A "numeric" object that gives the plot's stringy score.

Examples

```
require(ggplot2)
require(tidyr)
require(dplyr)
data(anscombe_tidy)
ggplot(anscombe_tidy, aes(x=x, y=y)) +
  geom_point() +
  facet_wrap(~set, ncol=2, scales = "free")
sc_stringy(anscombe$x1, anscombe$y1)
sc_stringy(anscombe$x2, anscombe$y2)
sc_stringy(anscombe$x3, anscombe$y3)
sc_stringy(anscombe$x4, anscombe$y4)
```

sc_striped *Measure of Discreteness*

Description

This metric computes the 1-(ratio between the number of unique values to total data values) on number of rotations of the data, and returns the smallest value. If this value is large it means that there are only a few unique data values, and hence the distribution is discrete

Usage

```
sc_striped(x, y)
```

Arguments

x	numeric vector
y	numeric vector

Value

double

Examples

```
data("datasaurus_dozen_wide")
sc_striped(datasaurus_dozen_wide$v_lines_x,
           datasaurus_dozen_wide$v_lines_y)
sc_striped(datasaurus_dozen_wide$dino_x,
           datasaurus_dozen_wide$dino_y)
```

top_pairs *Calculate the top scagnostic for each pair of variables*

Description

Calculate the top scagnostic for each pair of variables

Usage

```
top_pairs(scags_data)
```

Arguments

scags_data	A dataset of scagnostic values that was returned by calc_scags or calc_scags_wide

Value

A data frame where each row is a each scatter plot, its highest valued scagnostic, and its respective value

See Also

calc_scags calc_scags_wide top_scags

Examples

```
#an example using calc_scags
require(dplyr)
datasaurus_dozen %>%
  group_by(dataset) %>%
  summarise(calc_scags(x,y, scags=c("monotonic", "outlying", "convex"))) %>%
  top_pairs()
 #an example using calc_scags_wide
 data(pk)
 scags_data <- calc_scags_wide(pk[,2:5], scags=c("outlying","clumpy","monotonic"))
 top_pairs(scags_data)
```

---

top_scags *Calculate the top pair of variables or group for each scagnostic*

Description

Calculate the top pair of variables or group for each scagnostic

Usage

```
top_scags(scags_data)
```

Arguments

scags_data	A dataset of scagnostic values that was returned by calc_scags or calc_scags_wide

Value

A data frame where each row is a scagnostic with its highest pair and the associated value

See Also

calc_scags calc_scags_wide top_pairs

Examples

```
#an example using calc_scags
require(dplyr)
datasaurus_dozen %>%
  group_by(dataset) %>%
  summarise(calc_scags(x,y, scags=c("monotonic", "outlying", "convex"))) %>%
  top_scags()
 #an example using calc_scags_wide
 data(pk)
 scags_data <- calc_scags_wide(pk[,2:5], scags=c("outlying","clumpy","monotonic"))
 top_scags(scags_data)
```

Index