Package ‘ztils’

April 10, 2025

Type Package

Title Various Common Statistical Utilities

Version 1.0.0

Description Utilities for simplifying common statistical operations including
probability density functions, cumulative distribution functions,
Kolmogorov-Smirnov tests, principal component analysis plots, and prediction plots.

License MIT + file LICENSE

URL `https://zachpeagler.github.io/ztils/`,
`https://github.com/zachpeagler/ztils`

BugReports `https://github.com/zachpeagler/ztils/issues`

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.3.0)

Imports ggplot2, MASS, scico, vegan

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Zach Peagler [aut, cre, cph]

Maintainer Zach Peagler <`zachpeagler00@gmail.com`>

Repository CRAN

Date/Publication 2025-04-10 14:30:01 UTC

Contents

`glm_pseudor2` *glm_pseudoR2*

Description

A function for calculating the pseudo R^2 of a glm object

Usage

```
glm_pseudor2(mod)
```

Arguments

`mod`	The model for which to calculate the pseudo R^2

Value

The pseudo R^2 value of the model

Examples

```
gmod <- glm(Sepal.Length ~ Petal.Length + Species, data = iris)
glm_pseudor2(gmod)
```

`multicdf_cont` *Multiple Cumulative Distribution Functions for Continuous Variables*

Description

This function gets the cumulative distribution function for selected distributions against a continuous, non-negative input variable. Possible distributions include "normal", "lognormal", "gamma", "exponential", "cauchy", "t", "weibull", "logistic", and "all".

Usage

```
multicdf_cont(var, seq_length = 50, distributions = "all")
```

Arguments

`var`	The variable of which to get the CDF
`seq_length`	The length of sequence to fit the distribution to
`distributions`	The distributions to fit x against

Value

A dataframe with x, the real density, and the pdf of the desired distributions with length (nrows) equal to seq_length +1.

Examples

```
multicdf_cont(iris$Petal.Length)

multicdf_cont(iris$Sepal.Length,
              100,
              c("normal", "lognormal")
              )
```

`multicdf_plot` *multicdf_plot*

Description

This function extends 'multiCDF_cont' and gets the cumulative distribution functions (CDFs) for selected distributions against a continuous variable. Possible distributions include any combination of "normal", "lognormal", "gamma", "exponential", and "all" (which just uses all of the prior distributions). It then plots this using 'ggplot2' and a 'scico' palette, using var_name for the plot labeling, if specified. If not specified, it will use var instead.

Usage

```
multicdf_plot(
  var,
  seq_length = 50,
  distributions = "all",
  palette = "oslo",
  var_name = NULL
)
```

Arguments

`var`	The variable to for which to plot CDFs
`seq_length`	The number of points over which to fit x
`distributions`	The distributions to fit x against
`palette`	The color palette to use on the graph
`var_name`	The variable name to use for x

Value

A plot showing the CDF of the selected variable against the selected distributions over the selected sequence length

Examples

```
multicdf_plot(iris$Sepal.Length)

multicdf_plot(iris$Sepal.Length,
              seq_length = 100,
              distributions = c("normal", "lognormal", "gamma"),
              palette = "bilbao",
              var_name = "Sepal Length (cm)"
              )
```

`multiks_cont` *Multiple Kolmogorov-Smirnov Tests for Continuous Variables*

Description

This function gets the distance and p-value from a Kolmogorov-smirnov test for selected distributions against a continuous input variable. Possible distributions include "normal", "lognormal", "gamma", "exponential", and "all".

Usage

```
multiks_cont(var, distributions = "all")
```

Arguments

`var`	The variable to perform ks tests against
`distributions`	The distributions to test x against

Value

A dataframe with the distance and p value for each performed ks test

Examples

```
multiks_cont(iris$Sepal.Length)

multiks_cont(iris$Sepal.Length, c("normal", "lognormal"))
```

`multipdf_cont` *Multiple Proportional Density Functions for Continuous Variables*

Description

This function gets the proportional density functions for selected distributions against continuous, non-negative numbers. Possible distributions include "normal", "lognormal", "gamma", "exponential", and "all".

Usage

```
multipdf_cont(var, seq_length = 50, distributions = "all")
```

Arguments

`var`	The variable of which to get the PDF.
`seq_length`	The length of sequence to fit the distribution to
`distributions`	The distributions to fit x against

Value

A dataframe with x, the real density, and the pdf of the desired distributions with length (nrows) equal to seq_length +1.

Examples

```
multipdf_cont(iris$Petal.Length)

multipdf_cont(iris$Sepal.Length, 100, c("normal", "lognormal"))
```

`multipdf_plot` *multipdf_plot*

Description

This function extends 'multiPDF_cont' and gets the probability density functions (PDFs) for selected distributions against continuous variables. Possible distributions include any combination of "normal", "lognormal", "gamma", "exponential", and "all" (which just uses all of the prior distributions). It then plots this using 'ggplot2' and a 'scico' palette, using var_name for the plot labeling, if specified. If not specified, it will use var instead.

Usage

```
multipdf_plot(
  var,
  seq_length = 50,
  distributions = "all",
  palette = "oslo",
  var_name = NULL
)
```

Arguments

`var`	The variable to for which to plot PDFs
`seq_length`	The number of points over which to fit x
`distributions`	The distributions to fit x against
`palette`	The color palette to use on the graph
`var_name`	The variable name to use for x. If no name is provided, the function will grab the column name provided in x

Value

A plot showing the PDF of the selected variable against the selected distributions over the selected sequence length

Examples

```
multipdf_plot(iris$Sepal.Length)

multipdf_plot(iris$Sepal.Length,
             seq_length = 100,
             distributions = c("normal", "lognormal", "gamma"),
             palette = "bilbao",
             var_name = "Sepal Length (cm)"
             )
```

`no_extremes` *No extremes*

Description

This function returns a dataframe subsetted to not include observations that are beyond the extremes of the specified variable. Extremes are defined by the quantiles +- 3 times the interquartile range.

Usage

```
no_extremes(data, var)
```

Arguments

`data`	The data to subset
`var`	The variable to subset by.

Value

A dataframe without entries containing extremes in the selected variable.

Examples

```
no_extremes(iris, Sepal.Length)
```

`no_outliers` *No outliers*

Description

This function returns a dataframe subsetted to not include observations that are beyond the outliers of the specified variable. Outliers are defined by the quantiles +- 1.5 times the interquartile range.

Usage

```
no_outliers(data, var)
```

Arguments

`data`	The data to subset
`var`	The variable to subset by

Value

A dataframe without entries containing outliers in the selected variable.

Examples

```
no_outliers(iris, Sepal.Length)
```

`pca_data` *Principal Component Analysis Data*

Description

This function uses a dataframe, PCA variables, and a scaled boolean to generate a dataframe with principal components as columns.

Usage

```
pca_data(data, pcavars, scaled = FALSE)
```

Arguments

`data`	The dataframe to add principal components to.
`pcavars`	The variables to include in the principle component analysis
`scaled`	A boolean (TRUE or FALSE) indicating if the pcavars are already scaled

Value

A plot showing PC1 on the x axis, PC2 on the y axis, colored by group, with vectors and labels showing the individual pca variables.

Examples

```
pca_data(iris, iris[,c(1:4)], FALSE)
```

`pca_plot` *Principal Component Analysis Plot*

Description

This function uses a group, PCA variables, and a scaled boolean to generate a biplot.using 'ggplot2' and 'scico'.

Usage

```
pca_plot(group, pcavars, scaled = FALSE, palette = "oslo")
```

Arguments

`group`	The group variable (column)
`pcavars`	The variables to include in the principle component analysis
`scaled`	A boolean (TRUE or FALSE) indicating if the pcavars are already scaled
`palette`	A color palette to use on the plot, with each group assigned to a color.

Value

A plot showing PC1 on the x axis, PC2 on the y axis, colored by group, with vectors and labels showing the individual pca variables.

Examples

```
pca_plot(iris$Species, iris[,c(1:4)])

pca_plot(iris$Species, iris[,c(1:4)], FALSE, "bilbao")
```

`predict_plot` *Prediction Plot*

Description

This function uses a model, dataframe, and supplied predictor, response, and group variables to make predictions based off the model over a user-defined length with options to predict over the confidence or prediction interval and to apply a mathematical correction. It then graphs both the real data and the specified interval using 'ggplot2'. You can also choose the color palette from 'scico' palettes.

Usage

```
predict_plot(
  mod,
  data,
  rvar,
  pvar,
  group = NULL,
  length = 50,
  interval = "confidence",
  correction = "normal",
  palette = "oslo"
)
```

Arguments

`mod`	the model used for predictions
`data`	the data used to render the "real" points on the graph and for aggregating groups to determine prediction limits (should be the same as the data used in the model)
`rvar`	the response variable (y variable / variable the model is predicting)
`pvar`	the predictor variable (x variable / variable the model will predict against)
`group`	the group; should be a factor; one response curve will be made for each group
`length`	the length of the variable over which to predict (higher = more resolution, essentially)

`interval`	the type of interval to predict ("confidence" or "prediction")
`correction`	the type of correction to apply to the prediction ("normal", "exponential", or "logit")
`palette`	the color palette used to color the graph, with each group corresponding to a color

Value

A plot showing the real data and the model's predicted 95% CI or PI over a number of groups, with optional corrections.

Examples

```
## Example 1
mod1 <- lm(Sepal.Length ~ Petal.Length + Species, data = iris)
predict_plot(mod1, iris, Sepal.Length, Petal.Length, Species)
```

Index