Package 'snazzieR'

November 16, 2026

Type Package

Title Chic and Sleek Functions for Beautiful Statisticians

Version 0.1.3

Description Because your linear models deserve better than console output. A sleek color palette and kable styling to make your regression results look sharper than they are. Includes support for Partial Least Squares (PLS) regression via both the SVD and NIPALS algorithms, along with a unified interface for model fitting and fabulous LaTeX and console output formatting. Regression diagnostics include the standard residual plots, marginal model plots, a table of unusual observations and variance inflation factors. LaTeX tables also preview as HTML when R Markdown chunks are run interactively. See the package website at <https://finitesample.space/snazzier>.

License MIT + file LICENSE

URL https://finitesample.space/snazzier, https://detectivefierce.github.io/snazzieR/, https://github.com/DetectiveFierce/snazzieR

BugReports https://github.com/DetectiveFierce/snazzieR/issues

Depends R (>= 4.1.0)

Imports ggplot2 (>= 3.5.0), graphics, grDevices, grid, htmltools, kableExtra, knitr, stats

Suggests rmarkdown, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Aidan J. Wagner [aut, cre]

Maintainer Aidan J. Wagner <JesusButForGayPeople@proton.me>

Repository CRAN

Date/Publication 2026-11-16 20:20:02 UTC

Contents

ANOVA.summary.table . . . 2
color.ref . . . 3
colors . . . 4
create_kfold_splits . . . 6
diagnostics.plots . . . 6
eigen.summary . . . 7
format.pls . . . 8
kfold_cross_validation . . . 9
marginal.model.plots . . . 10
model.equation . . . 11
model.summary.table . . . 12
pls.regression . . . 13
pls.summary . . . 14
predict.ridge.model . . . 15
print.ridge.model . . . 16
ridge.regression . . . 16
ridge.summary . . . 18
snazzieR.theme . . . 19
snazzieR_table . . . 20
unusual.points.table . . . 21
VIF.table . . . 23

Index **26**

`ANOVA.summary.table` *Generate a Summary Table for ANOVA Results*

Description

This function creates a summary table for ANOVA results, including degrees of freedom, sum of squares, mean squares, F-values, and p-values. The table can be output as either LaTeX (for PDF reports) or plain text (for console viewing).

Usage

```
ANOVA.summary.table(model, title = "ANOVA Summary", latex = TRUE, caption)
```

Arguments

model	A model object for which ANOVA results are computed (e.g., output from ‘lm()‘ or ‘aov()‘).
title	A character string used as the table's title (its LaTeX caption). Defaults to "ANOVA Summary".
latex	Logical; if ‘TRUE‘, returns a LaTeX-formatted table using ‘kableExtra‘. If ‘FALSE‘, prints a plain-text version to the console.
caption	Deprecated alias for title, kept so existing code keeps working.

Value

If ‘latex = TRUE‘, a LaTeX-formatted table object (of class "snazzieR_table", which also previews when a chunk is run interactively; see snazzieR_table). If ‘latex = FALSE‘, prints the summary table and returns it (invisibly).

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Generate a plain-text ANOVA summary table
ANOVA.summary.table(model, title = "ANOVA Summary", latex = FALSE)
```

---

color.ref *Display a Color Reference Palette*

Description

This function generates a plot displaying a predefined color palette with color codes for easy reference. The palette includes shades of Red, Orange, Yellow, Green, Blue, Purple, and Grey.

Usage

```
color.ref()
```

Details



Value

Called for its side effect of drawing the color palette; returns `NULL` invisibly.

Examples

```
color.ref()
```

`colors` *SnazzieR Color Palette*

Description

A collection of named hex colors grouped by hue and tone. Each color is available as an exported object (e.g., `Red`, `Dark.Red`).

Usage

```
color.list
```

Format

Each color is a character string representing a hex code.

Details

This palette consists of named hex colors. Each color's name (e.g., `Dark.Red`) is available as an exported object.

Swatch images are embedded below (not selectable):

Color	Swatch	Color	Swatch	Color	Swatch
Deep.Red		Deep.Green		Deep.Grey	
Dark.Red		Dark.Green		Dark.Grey	
Red		Green		Grey	
Light.Red		Light.Green		Light.Grey	
Pale.Red		Pale.Green		Pale.Grey	
Deep.Orange		Deep.Blue			
Dark.Orange		Dark.Blue			
Orange		Blue			
Light.Orange		Light.Blue			
Pale.Orange		Pale.Blue			
Deep.Yellow		Deep.Purple			
Dark.Yellow		Dark.Purple			
Yellow		Purple			
Light.Yellow		Light.Purple			
Pale.Yellow		Pale.Purple			

For the full list and hex codes, use `names(color.list)` or see `?color.list`.

See Also

color.ref, snazzieR.theme

`create_kfold_splits` *Create K-Fold Cross Validation Splits*

Description

Create K-Fold Cross Validation Splits

Usage

```
create_kfold_splits(data, k = 5, seed = NULL)
```

Arguments

`data`	A data frame containing the dataset
`k`	Number of folds (default: 5)
`seed`	Random seed for reproducibility (optional)

Value

A list containing fold assignments for each observation

`diagnostics.plots` *Snazzy Regression Diagnostic Plots*

Description

Draws the four standard diagnostic plots for a linear model (the same panels as `plot(model)` for an `lm` fit) as a 2 x 2 grid of **ggplot2** plots styled with `snazzieR.theme()`.

Usage

```
diagnostics.plots(model, id.n = 3, cook.levels = c(0.5, 1))

## S3 method for class 'snazzieR_plot_grid'
print(x, ...)
```

Arguments

`model`	A fitted `lm` object.
`id.n`	Number of extreme points to label in each panel (default 3).
`cook.levels`	Cook's distance levels drawn as contours in the leverage panel (default `c(0.5, 1)`).
`x`	A `"snazzieR_plot_grid"` object.
`...`	Unused.

Details

The panels reproduce `plot.lm` with its default `which = c(1, 2, 3, 5)`:

1. **Residuals vs Fitted**: residuals against fitted values with a `lowess` smooth (`f = 2/3`, `iter = 3`) and a reference line at zero.
2. **Q-Q Residuals**: standardized residuals against theoretical normal quantiles (`ppoints`) with the line through the first and third quartiles, as drawn by `qqline`.
3. **Scale-Location**: $\sqrt{|\text{standardized residuals}|}$ against fitted values with a lowess smooth.
4. **Residuals vs Leverage**: standardized residuals against leverage (hat values), with Cook's distance contours at `cook.levels`.

As in `plot.lm`, the `id.n` most extreme observations in each panel are labelled with their observation numbers (placed so that labels do not overlap), observations with zero weight are dropped, and observations with leverage one are omitted from the standardized-residual panels.

Value

An object of class `"snazzieR_plot_grid"`: a list whose `plots` element holds the four `ggplot` objects (named `residuals.fitted`, `qq`, `scale.location` and `residuals.leverage`). Printing it draws the 2 x 2 grid; individual panels can be printed or modified on their own.

See Also

`plot.lm`, `marginal.model.plots`

Examples

```
model <- lm(mpg ~ wt + hp, data = mtcars)
diagnostics.plots(model)

# A single panel
diagnostics.plots(model)$plots$qq
```

`eigen.summary` *Summarize Eigenvalues and Eigenvectors of a Covariance Matrix*

Description

This function computes the eigenvalues and eigenvectors of a given covariance matrix, ensures sign consistency in the eigenvectors, and outputs either a LaTeX table or plaintext summary displaying the results.

Usage

```
eigen.summary(
  cov.matrix,
  title = "Eigenvectors of Covariance Matrix",
  latex = TRUE,
  caption
)
```

Arguments

cov.matrix	A square numeric matrix representing the covariance matrix.
title	A character string used as the table's title (default: "Eigenvectors of Covariance Matrix").
latex	A logical indicating whether to output LaTeX table (default: TRUE). If FALSE, prints as plain text.
caption	Deprecated alias for title, kept so existing code keeps working.

Value

A LaTeX formatted table of class "snazzieR_table" (if latex = TRUE; see snazzieR_table for interactive previews) or plaintext console output (if latex = FALSE).

Examples

```
cov_matrix <- matrix(c(4, 2, 2, 3), nrow = 2)
eigen.summary(cov_matrix, title = "Eigenvalues and Eigenvectors", latex = FALSE)
```

---

format.pls *Format PLS Model Output as LaTeX or Console Tables*

Description

Formats and displays Partial Least Squares (PLS) model output from pls.regression() as either LaTeX tables (for PDF rendering) or console-friendly output.

Usage

```
## S3 method for class 'pls'
format(x, ..., include.scores = TRUE, latex = FALSE)
```

Arguments

x	A list returned by pls.regression() (class "pls") containing PLS model components.
...	Further arguments passed to or from methods (unused).
include.scores	Logical. Whether to include score matrices (T and U). Default is TRUE.
latex	Logical. If TRUE, produces LaTeX output (for PDF rendering). If FALSE, prints to console. Default is FALSE.

Value

When `latex = TRUE`, returns a `knitr::asis_output` object (LaTeX code) of class `"snazzieR_table"` (see `snazzieR_table` for interactive previews). When `FALSE`, prints formatted tables to console.

`kfold_cross_validation` *Perform K-Fold Cross Validation*

Description

Perform K-Fold Cross Validation

Usage

```
kfold_cross_validation(
  data,
  formula,
  model_function,
  predict_function = predict,
  metric_function = NULL,
  k = 5,
  seed = NULL,
  ...
)
```

Arguments

`data`	A data frame containing the dataset
`formula`	A formula specifying the model (e.g., y ~ x1 + x2)
`model_function`	Function to fit the model (e.g., lm, glm)
`predict_function`	Function to make predictions (default: predict)
`metric_function`	Function to calculate performance metric
`k`	Number of folds (default: 5)
`seed`	Random seed for reproducibility (optional)
`...`	Additional arguments passed to model_function

Value

A list containing fold results and summary statistics

`marginal.model.plots` *Snazzy Marginal Model Plots*

Description

Draws marginal model plots for a linear model, the same plots as `car::mmps(model)`, as a single faceted **ggplot2** plot styled with `snazzieR.theme()`. Only **ggplot2** is required; **car** is not needed.

Usage

```
marginal.model.plots(model, fitted = TRUE, ncol = NULL, title = NULL)
```

Arguments

`model`	A fitted `lm` object.
`fitted`	Logical; include the panel against the fitted values (default `TRUE`).
`ncol`	Number of panel columns. Defaults to **car**'s layout.
`title`	Plot title. Defaults to `"Marginal Model Plots"` (or `"Marginal Model Plot"` for a single panel), as in **car**.

Details

For each eligible term in the model, and for the fitted values, the response is plotted against that variable together with two smooths:

Data (solid) a loess smooth of the response on the variable;

Model (dashed) a loess smooth of the model's fitted values on the variable.

If the model is adequate the two smooths should agree. The smooths match those drawn by **car**: `loess` with `span = 2/3`, `degree = 1`, `family = "symmetric"` and four robustness iterations, evaluated at 50 equally spaced points.

As in `car::mmps()`, factors and interaction terms are skipped, and matrix terms such as `poly()` or spline bases are replaced by their fitted linear combination (`predict(model, type = "terms")`). The panel layout follows **car**'s defaults (e.g. 3 x 3 for 7 to 9 panels).

Value

A `ggplot` object. Its title is drawn as the legend title, so the Data/Model key sits to the right of it.

See Also

`diagnostics.plots`

Examples

```
model <- lm(mpg ~ wt + hp + qsec, data = mtcars)
marginal.model.plots(model)

# Label which model the plots belong to
marginal.model.plots(model, title = "Marginal Model Plots: Fuel Economy")
```

`model.equation` *Generate a Model Equation from a Linear Model*

Description

This function extracts and formats the equation from a linear model object. It includes an option to return the equation as a LaTeX-formatted string or print it to the console.

Usage

```
model.equation(model, latex = TRUE)
```

Arguments

`model`	A linear model object (e.g., output from ‘lm()‘).
`latex`	A logical value indicating whether to return a LaTeX-formatted equation (default: TRUE). If FALSE, the equation is printed to the console.

Value

If ‘latex‘ is TRUE, the equation is returned as LaTeX code using ‘knitr::asis_output()‘. If FALSE, the equation is printed to the console.

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Get LaTeX equation
model.equation(model)

# Print equation to console
model.equation(model, latex = FALSE)
```

`model.summary.table` *Generate a Summary Table for a Linear Model*

Description

This function creates a summary table for a linear model, including estimated coefficients, standard errors, p-values with significance codes, and model statistics such as MSE and R-squared. The table can be output as either LaTeX (for PDF reports) or plain text (for console viewing).

Usage

```
model.summary.table(model, title = "Model Summary", latex = TRUE, caption)
```

Arguments

`model`	A linear model object (typically the result of 'lm()').
`title`	A character string used as the table's title (its LaTeX caption). Defaults to `"Model Summary"`. The fitted model equation is typeset on its own centered line between the title and the table.
`latex`	Logical; if 'TRUE' (default), returns a LaTeX-formatted table using 'kableExtra'. If 'FALSE', prints plain-text summary tables to the console.
`caption`	Deprecated alias for `title`, kept so existing code keeps working.

Value

If 'latex = TRUE', returns a LaTeX-formatted 'kableExtra' table object (of class `"snazzieR_table"`, which also previews nicely when a chunk is run interactively; see `snazzieR_table`). If 'latex = FALSE', prints formatted summary tables to the console and returns the underlying data frame.

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Print a plain-text version to the console
model.summary.table(model, title = "Linear Model Summary", latex = FALSE)
```

`pls.regression` *Partial Least Squares (PLS) Regression Interface*

Description

Performs Partial Least Squares (PLS) regression using either the NIPALS or SVD algorithm for component extraction. This is the main user-facing function for computing PLS models. Internally, it delegates to either `NIPALS.pls()` or `SVD.pls()`.

Usage

```
pls.regression(x, y, n.components = NULL, calc.method = c("SVD", "NIPALS"))
```

Arguments

`x`	A numeric matrix or data frame of predictor variables (X), with dimensions n × p.
`y`	A numeric matrix or data frame of response variables (Y), with dimensions n × q.
`n.components`	Integer specifying the number of latent components (H) to extract. If NULL, defaults to the rank of `x`.
`calc.method`	Character string indicating the algorithm to use. Must be either `"SVD"` (default) or `"NIPALS"`.

Details

This function provides a unified interface for Partial Least Squares regression. Based on the value of `calc.method`, it computes latent variables using either:

- `"SVD"` — A direct method using the singular value decomposition of the cross-covariance matrix ($X^\top Y$).
- `"NIPALS"` — An iterative method that alternately estimates predictor and response scores until convergence.

The outputs from both methods include scores, weights, loadings, regression coefficients, and explained variance.

Value

A list (from either `SVD.pls()` or `NIPALS.pls()`) containing:

model.type Character string ("PLS Regression").

T, U Score matrices for X and Y.

W, C Weight matrices for X and Y.

P_loadings, Q_loadings Loading matrices.

B_vector Component-wise regression weights.

coefficients Final regression coefficient matrix (rescaled).

intercept Intercept vector (typically zero due to centering).

X_explained, Y_explained Variance explained by each component.

X_cum_explained, Y_cum_explained Cumulative variance explained.

References

Abdi, H., & Williams, L. J. (2013). Partial least squares methods: Partial least squares correlation and partial least square regression. *Methods in Molecular Biology (Clifton, N.J.)*, 930, 549–579. doi:10.1007/9781627030595_23

de Jong, S. (1993). SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3), 251–263. doi:10.1016/01697439(93)85002X

See Also

`SVD.pls`, `NIPALS.pls`

Examples

```
## Not run:
X <- matrix(rnorm(100 * 10), 100, 10)
Y <- matrix(rnorm(100 * 2), 100, 2)

# Using SVD (default)
model1 <- pls.regression(X, Y, n.components = 3)

# Using NIPALS
model2 <- pls.regression(X, Y, n.components = 3, calc.method = "NIPALS")

## End(Not run)
```

`pls.summary` *Format PLS Model Output as LaTeX Tables*

Description

Formats and displays Partial Least Squares (PLS) model output from `pls.regression()` as LaTeX tables for PDF rendering.

Usage

```
pls.summary(x, ..., include.scores = TRUE)
```

Arguments

x	A list returned by pls.regression() (class "pls") containing PLS model components.
...	Further arguments passed to or from methods (unused).
include.scores	Logical. Whether to include score matrices (T and U). Default is TRUE.

Value

Returns a knitr::asis_output object (LaTeX code) for PDF rendering It has class "snazzieR_table", so it also previews when a chunk is run interactively (see snazzieR_table).

Examples

```
# Load example data
data(mtcars)

# Prepare data for PLS regression
X <- mtcars[, c("wt", "hp", "disp")]
Y <- mtcars[, "mpg", drop = FALSE]

# Fit PLS model with 2 components
pls.fit <- pls.regression(X, Y, n.components = 2)

# Print a LaTeX-formatted summary
pls.summary(pls.fit, include.scores = FALSE)
```

---

predict.ridge.model *Predict Method for Ridge Model Objects*

Description

Predicts response values for new data using a fitted ridge model.

Usage

```
## S3 method for class 'ridge.model'
predict(object, newdata = NULL, ...)
```

Arguments

object	A ‘ridge.model‘ object
newdata	A data frame or matrix containing new predictor values
...	Additional arguments (not used)

Value

A numeric vector of predicted values

print.ridge.model *Print Method for Ridge Model Objects*

Description

Prints a summary of the ridge model fit.

Usage

```
## S3 method for class 'ridge.model'
print(x, ...)
```

Arguments

x	A ‘ridge.model‘ object
...	Additional arguments (not used)

ridge.regression *Ridge Regression with Automatic Lambda Selection*

Description

Performs ridge regression with automatic selection of the optimal regularization parameter ‘lambda‘ by minimizing k-fold cross-validated mean squared error (CV-MSE) using Brent's method. Supports both formula and matrix interfaces.

Usage

```
ridge.regression(
  formula = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  lambda.range = c(1e-04, 100),
  folds = 5,
  ...
)
```

Arguments

formula	A model formula like ‘y ~ x1 + x2‘. Mutually exclusive with ‘x‘/‘y‘.
data	A data frame containing all variables used in the formula.
x	A numeric matrix of predictor variables (n × p). Used when formula is not provided.

`y`	A numeric vector of response variables (n × 1). Used when formula is not provided.
`lambda.range`	A numeric vector of length 2 specifying the interval for lambda optimization. Default: 'c(1e-4, 100)'.
`folds`	An integer specifying the number of cross-validation folds. Default: '5'.
`...`	Additional arguments passed to internal methods.

Details

This function implements ridge regression with automatic hyperparameter tuning. The algorithm:

- Standardizes predictor variables (centers and scales)
- Centers the response variable
- Uses k-fold cross-validation to find the optimal lambda
- Fits the final model with the optimal lambda
- Returns a structured object for prediction and analysis

The ridge regression solution is computed using the closed-form formula: $\hat{\beta} = (X^T X + \lambda I)^{-1} X^T y$

Value

A 'ridge.model' object containing:

coefficients Final ridge coefficients (no intercept)

std_errors Standard errors of the coefficients

intercept Intercept term (from y centering)

optimal.lambda Best lambda minimizing CV-MSE

cv.ms Minimum CV-MSE achieved

cv.results Data frame with lambda and CV-MSE pairs

x.scale Standardization info: mean and sd for each predictor

y.center Centering constant for y

fitted.values Final model predictions on training data

residuals Training residuals (y - fitted)

call Matched call (for debugging)

method Always "ridge"

folds Number of CV folds used

formula Stored if formula interface used

terms Stored if formula interface used

References

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55-67.

See Also

predict.ridge.model

Examples

```
## Not run:
# Formula interface
model1 <- ridge.regression(mpg ~ wt + hp + disp, data = mtcars)

# Matrix interface
X <- as.matrix(mtcars[, c("wt", "hp", "disp")])
y <- mtcars$mpg
model2 <- ridge.regression(x = X, y = y)

# Custom lambda range and folds
model3 <- ridge.regression(mpg ~ .,
    data = mtcars,
    lambda.range = c(0.1, 10), folds = 10
)

## End(Not run)
```

---

ridge.summary *Format Ridge Model Output as LaTeX Tables*

Description

Formats and displays ridge regression model output from `ridge.regression()` as LaTeX tables for PDF rendering or plain text for console viewing.

Usage

```
ridge.summary(x, ..., include.cv.trace = TRUE, latex = TRUE)
```

Arguments

`x`	A ridge model object returned by `ridge.regression()` (class `"ridge.model"`).
`...`	Further arguments passed to or from methods (unused).
`include.cv.trace`	Logical. Whether to include cross-validation trace information. Default is `TRUE`.
`latex`	Logical; if `TRUE` (default), returns LaTeX-formatted tables using `kableExtra`. If `FALSE`, prints plain-text summary tables to the console.

Value

If `latex = TRUE`, returns a `knitr::asis_output` object (LaTeX code) for PDF rendering It has class `"snazzieR_table"`, so it also previews when a chunk is run interactively (see `snazzieR_table`). If `latex = FALSE`, prints formatted summary tables to the console and returns the underlying data frames.

Examples

```
# Load example data
data(mtcars)

# Fit ridge regression model
ridge.fit <- ridge.regression(mpg ~ wt + hp + disp, data = mtcars)

# Print a LaTeX-formatted summary
ridge.summary(ridge.fit, include.cv.trace = FALSE)

# Print a plain-text summary
ridge.summary(ridge.fit, include.cv.trace = FALSE, latex = FALSE)
```

---

snazzieR.theme *A Custom ggplot2 Theme for Publication-Ready Plots*

Description

This theme provides a clean, polished look for ggplot2 plots, with a focus on readability and aesthetics. It includes a custom color palette and formatting for titles, axes, and legends.

Usage

```
snazzieR.theme()
```

Value

A ggplot2 theme object.

See Also

`color.list`, `color.ref`

Examples

```
library(ggplot2)
set.seed(123)
chains.df <- data.frame(
  Iteration = 1:500,
  alpha.1 = cumsum(rnorm(500, mean = 0.01, sd = 0.2)) + rnorm(1, 5, 0.2),
  alpha.2 = cumsum(rnorm(500, mean = 0.005, sd = 0.2)) + rnorm(1, 5, 0.2),
  alpha.3 = cumsum(rnorm(500, mean = 0.000, sd = 0.2)) + rnorm(1, 5, 0.2),
```

```
  alpha.4 = cumsum(rnorm(500, mean = -0.005, sd = 0.2)) + rnorm(1, 5, 0.2),
  alpha.5 = cumsum(rnorm(500, mean = -0.01, sd = 0.2)) + rnorm(1, 5, 0.2)
)
chain.colors <- c("Chain 1" = Red, "Chain 2" = Orange, "Chain 3" = Yellow,
                  "Chain 4" = Green, "Chain 5" = Blue)
ggplot(chains.df, aes(x = Iteration)) +
  geom_line(aes(y = alpha.1, color = "Chain 1"), linewidth = 1.2) +
  geom_line(aes(y = alpha.2, color = "Chain 2"), linewidth = 1.2) +
  geom_line(aes(y = alpha.3, color = "Chain 3"), linewidth = 1.2) +
  geom_line(aes(y = alpha.4, color = "Chain 4"), linewidth = 1.2) +
  geom_line(aes(y = alpha.5, color = "Chain 5"), linewidth = 1.2) +
  labs(x = "Iteration", y = expression(alpha),
       title = expression("Traceplot for " ~ alpha)) +
  scale_color_manual(values = chain.colors, name = "Chains") +
  snazzieR.theme()
```

`snazzieR_table` *Interactive Previews for snazzieR Tables*

Description

The LaTeX tables returned by `model.summary.table`, `ANOVA.summary.table`, `eigen.summary`, `pls.summary`, `ridge.summary` and `format.pls` carry the class `"snazzieR_table"`. This lets them show a readable preview when a chunk is run interactively (for example with the play button in the RStudio or Positron editor), while knitting produces exactly the same LaTeX as before.

Usage

```
## S3 method for class 'snazzieR_table'
print(x, ...)

## S3 method for class 'snazzieR_table'
knit_print(x, options = list(), ...)
```

Arguments

`x`	A `"snazzieR_table"` object.
`...`	Further arguments passed to the underlying print method.
`options`	Chunk options, passed on by **knitr**.

Details

When a document is being knitted, printing is handed straight to the underlying **knitr/kableExtra** method, so compiled output does not change. Outside of knitting, the preview style is controlled by the option `snazzieR.preview`:

`"auto"` (default) An HTML preview in RStudio or Positron, a plain-text preview in other interactive sessions, and the raw LaTeX in non-interactive sessions.

`"html"` Always render an HTML preview.

`"text"` Always print a plain-text preview.

`"latex"` Always print the raw LaTeX source.

Value

`x`, invisibly.

Examples

```
model <- lm(mpg ~ wt + hp, data = mtcars)
tbl <- model.summary.table(model, title = "Fuel Economy")

# Force a plain-text preview
old <- options(snazzieR.preview = "text")
print(tbl)
options(old)
```

---

unusual.points.table *Table of Unusual Observations in a Regression*

Description

Identifies the observations of a fitted regression that should be investigated – high-leverage points, outliers, influential points and bad leverage points – and returns them in a table, as LaTeX (for PDF reports) or plain text (for the console), like the other snazzieR summary tables.

Usage

```
unusual.points.table(
  model,
  title = "Unusual Observations",
  latex = TRUE,
  id = NULL,
  data = NULL,
  leverage.cutoff = NULL,
  residual.cutoff = 2,
  cook.cutoff = NULL,
  flag = c("leverage", "outlier", "influential"),
  n.max = NULL,
  digits = 4
)
```

Arguments

`model`	A fitted `lm` or `glm` object.
`title`	Table title (its LaTeX caption). Defaults to `"Unusual Observations"`.
`latex`	Logical; if `TRUE` (default) returns a LaTeX table. If `FALSE`, prints a plain-text table and returns the underlying data frame invisibly.
`id`	Optional labels to identify observations: either the name of a column in `data` (e.g. `"Name"`) or a vector with one label per row of the data the model was fitted to.
`data`	The data frame the model was fitted to. Only needed when `id` is a column name and the data cannot be recovered from the model's call.
`leverage.cutoff`	Leverage threshold. Defaults to $2(p+1)/n$.
`residual.cutoff`	Threshold for |standardized residual|. Defaults to 2.
`cook.cutoff`	Cook's distance threshold. Defaults to $4/(n-2)$.
`flag`	Which criteria make an observation unusual: any of `"leverage"`, `"outlier"` and `"influential"` (default: all three). For example, `flag = c("leverage", "outlier")` leaves out points that are only flagged by Cook's distance.
`n.max`	Maximum number of observations to list (default `NULL`, no limit). When more are flagged, the `n.max` with the largest Cook's distance are kept and the legend says how many were left out.
`digits`	Number of significant digits shown. Defaults to 4.

Details

Diagnostics are computed with `hatvalues`, `rstandard` and `cooks.distance`, so the function works for any `lm` fit (including weighted fits, factors, transformed variables and missing values) and for `glm` fits (using deviance residuals). Observations with zero weight are excluded, as in `plot.lm`.

An observation is listed if it meets at least one of these criteria (defaults follow Sheather, *A Modern Approach to Regression with R*):

High leverage $h_{ii} > 2(p+1)/n$, twice the average leverage, where $p+1$ is the number of estimated coefficients.

Outlier $|r_i| > 2$, where r_i is the standardized residual. (For large data sets a cutoff of 4 may be more appropriate.)

Influential Cook's distance $D_i > 4/(n-2)$.

Bad leverage both high leverage and an outlier.

For each flagged point the table reports its observed and fitted response, leverage, standardized residual, Cook's distance and flags. A legend under the table explains the flags and compares the number of outliers with the number expected by chance.

Value

If `latex = TRUE`, a LaTeX table of class `"snazzieR_table"` (see `snazzieR_table` for interactive previews). If `latex = FALSE`, the table is printed and a data frame with one row per flagged observation (sorted by Cook's distance) is returned invisibly. Its `"cutoffs"` attribute holds the thresholds used.

See Also

`diagnostics.plots`, `model.summary.table`

Examples

```
model <- lm(mpg ~ wt + hp, data = mtcars)

# Plain-text table, labelling cars by name
unusual.points.table(model, latex = FALSE, id = rownames(mtcars))

# Only the 3 most influential, ignoring points flagged by Cook's D alone
unusual.points.table(model, latex = FALSE, id = rownames(mtcars),
                     flag = c("leverage", "outlier"), n.max = 3)

# LaTeX table for a PDF report
tbl <- unusual.points.table(model, title = "Cars to Investigate")
```

VIF.table *Variance Inflation Factor Table*

Description

Computes variance inflation factors (VIFs) for the terms of a regression to diagnose multicollinearity, and returns them as a LaTeX table (for PDF reports) or plain text (for the console), like the other snazzieR summary tables. **car** is not required.

Usage

```
VIF.table(
  model,
  title = "Variance Inflation Factors",
  latex = TRUE,
  cols = "all",
  moderate = 5,
  severe = 10,
  digits = 4
)
```

Arguments

model	A fitted lm or glm object with at least two terms.
title	Table title (its LaTeX caption). Defaults to "Variance Inflation Factors".
latex	Logical; if TRUE (default) returns a LaTeX table. If FALSE, prints a plain-text table and returns the underlying data frame invisibly.
cols	Columns to show: "all" (default) or any combination of "VIF", "1/VIF", "GVIF" and "adj. GVIF". Term and Flag are always included.
moderate, severe	VIF cutoffs for flagging moderate and severe multicollinearity. Default 5 and 10.
digits	Number of significant digits shown. Defaults to 4.

Details

VIFs are computed from the correlation matrix of the estimated coefficients, exactly as car::vif() does, so they are valid for weighted lm fits and for glm fits. For a numeric predictor, $\mathrm{VIF}_j = 1/(1 - R_j^2)$, where R_j^2 is from regressing that predictor on all the others.

The available columns are:

VIF The variance inflation factor. Terms with more than one coefficient (factors, poly(), splines) have no ordinary VIF, so their VIF is reported as $(\text{adj. GVIF})^2$, which is on the same scale and equals the ordinary VIF for one-df terms.

1/VIF The tolerance, $1 - R_j^2$.

GVIF The generalized VIF of Fox and Monette (1992); equal to the VIF for one-df terms.

adj. GVIF $\mathrm{GVIF}^{1/(2\,\mathrm{Df})}$, comparable across terms with different degrees of freedom. Each cell also shows the term's degrees of freedom, e.g. 1.547 (df = 2).

The Term and Flag columns are always shown. Terms are flagged *moderate* when the VIF exceeds moderate (default 5) and *severe* when it exceeds severe (default 10). Rows are sorted from the highest VIF to the lowest. Aliased (perfectly collinear) coefficients are dropped and listed in the notes under the table.

Value

If latex = TRUE, a LaTeX table of class "snazzieR_table" (see snazzieR_table for interactive previews). If latex = FALSE, the table is printed and a data frame of the shown columns, one row per term, is returned invisibly.

References

Fox, J. and Monette, G. (1992). Generalized collinearity diagnostics. *Journal of the American Statistical Association*, 87, 178–183.

See Also

unusual.points.table, model.summary.table

Examples

```
model <- lm(mpg ~ wt + hp + disp + qsec, data = mtcars)
VIF.table(model, latex = FALSE)

# Factors get generalized VIFs
VIF.table(lm(mpg ~ wt + hp + factor(cyl), data = mtcars), latex = FALSE)

# Only some columns
VIF.table(model, latex = FALSE, cols = c("VIF", "1/VIF"))

# LaTeX table for a PDF report
tbl <- VIF.table(model, title = "Collinearity Check")
```

Index

∗ **datasets**
colors, 4

ANOVA.summary.table, 2, *20*

Blue *(*colors*)*, 4

color.list, *19*
color.list *(*colors*)*, 4
color.ref, 3, *6*, *19*
colors, 4
cooks.distance, *22*
create_kfold_splits, 6

Dark.Blue *(*colors*)*, 4
Dark.Green *(*colors*)*, 4
Dark.Grey *(*colors*)*, 4
Dark.Orange *(*colors*)*, 4
Dark.Purple *(*colors*)*, 4
Dark.Red *(*colors*)*, 4
Dark.Yellow *(*colors*)*, 4
Deep.Blue *(*colors*)*, 4
Deep.Green *(*colors*)*, 4
Deep.Grey *(*colors*)*, 4
Deep.Orange *(*colors*)*, 4
Deep.Purple *(*colors*)*, 4
Deep.Red *(*colors*)*, 4
Deep.Yellow *(*colors*)*, 4
diagnostics.plots, 6, *10*, *23*

eigen.summary, 7, *20*

format.pls, 8, *20*

Green *(*colors*)*, 4
Grey *(*colors*)*, 4

hatvalues, *22*

kfold_cross_validation, 9
knit_print.snazzieR_table
*(*snazzieR_table*)*, 20

Light.Blue *(*colors*)*, 4
Light.Green *(*colors*)*, 4
Light.Grey *(*colors*)*, 4
Light.Orange *(*colors*)*, 4
Light.Purple *(*colors*)*, 4
Light.Red *(*colors*)*, 4
Light.Yellow *(*colors*)*, 4
loess, *10*
lowess, *7*

marginal.model.plots, *7*, 10
model.equation, 11
model.summary.table, 12, *20*, *23*, *24*

NIPALS.pls, *14*

Orange *(*colors*)*, 4

Pale.Blue *(*colors*)*, 4
Pale.Green *(*colors*)*, 4
Pale.Grey *(*colors*)*, 4
Pale.Orange *(*colors*)*, 4
Pale.Purple *(*colors*)*, 4
Pale.Red *(*colors*)*, 4
Pale.Yellow *(*colors*)*, 4
plot.lm, *7*
pls.regression, *8*, 13, *15*
pls.summary, 14, *20*
ppoints, *7*
predict.ridge.model, 15, *18*
print.ridge.model, 16
print.snazzieR_plot_grid
*(*diagnostics.plots*)*, 6
print.snazzieR_table *(*snazzieR_table*)*,
20
Purple *(*colors*)*, 4

qqline, *7*

Red *(*colors*)*, 4
ridge.regression, 16, *18*

ridge.summary, 18, *20*
rstandard, *22*

snazzieR.theme, *6*, *10*, 19
snazzieR_table, *3*, *8*, *9*, *12*, *15*, *19*, 20, *23*, *24*
SVD.pls, *14*

unusual.points.table, 21, *24*

VIF.table, 23

Yellow *(*colors*)*, 4