Package: gtregression (via r-universe)

March 27, 2026

Type Package

Title Tools for Creating Publication-Ready Regression Tables

Version 1.1.0

Description Simplifies regression modeling in R by integrating multiple modeling and summarization tools into a cohesive, user-friendly interface. Designed to be accessible for researchers, particularly those in Low- and Middle-Income Countries (LMIC). Built upon widely accepted statistical methods, including logistic regression (Hosmer et al. 2013, ISBN:9781118548429), log-binomial regression (Spiegelman and Hertzmark 2005 <doi:10.1093/aje/kwi188>), Poisson and robust Poisson regression (Zou 2004 <doi:10.1093/aje/kwh090>), negative binomial regression (Hilbe 2011, ISBN:9780521179515), and linear regression (Kutner et al. 2005, ISBN:9780071122214). Leverages multiple dependencies to ensure high-quality output and generate reproducible, publication-ready tables in alignment with best practices in epidemiology and applied statistics.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports dplyr, gtsummary, risks, purrr, MASS, rlang, stats, lmtest, patchwork, ggplot2, tidyr, utils, sandwich, tibble, broom, broom.helpers, gt, officer, flextable

VignetteBuilder knitr

Suggests testthat (>= 3.1.5), knitr, rmarkdown, mlbench, car, forcats, pkgdown, scales (>= 1.3.0), ggtext

RoxygenNote 7.3.3

Config/testthat/edition 3

Config/Needs/website pkgdown

URL `https://thinkdenominator.github.io/gtregression/`

BugReports https://github.com/ThinkDenominator/gtregression/issues

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev libnode-dev libx11-dev zlib1g-dev

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

Date/Publication 2025-09-28 15:40:30 UTC

RemoteUrl https://github.com/thinkdenominator/gtregression

RemoteRef HEAD

RemoteSha 34ec48812bb47991959437d40c0df8490959ab27

Contents

`check_assumptions` *Check model assumptions (beginner-friendly)*

Description

Fits a model under the hood (like ‘uni_reg()‘/‘multi_reg()‘) and runs assumption checks appropriate to the approach. Returns a layered object with a tidy summary table and plots for quick visual diagnosis.

Usage

```
check_assumptions(
  data,
  outcome,
  exposures = NULL,
 approach = c("auto", "linear", "logit", "log-binomial", "poisson", "robpoisson",
    "negbin"),
  multivariate = TRUE,
  confounders = NULL,
  weights = NULL,
  cluster = NULL,
  groups = 10,
  top_n = 5,
  explain = TRUE,
  output = c("both", "summary", "plots"),
  quiet = TRUE,
  ...
)
```

Arguments

`data`	A data frame.
`outcome`	Character. Outcome variable name.
`exposures`	Character vector of predictors. If ‘NULL‘, uses all columns except ‘outcome‘.
`approach`	One of ‘"auto","linear","logit","log-binomial","poisson", "robpoisson","negbin"‘. Default ‘"auto"‘.
`multivariate`	Logical. If ‘TRUE‘, fits one adjusted model with all ‘exposures‘ (and ‘confounders‘ if supplied). If ‘FALSE‘, screens each exposure with ‘outcome ~ exposure‘ and stacks the diagnostics.
`confounders`	Optional character vector (used when ‘multivariate = TRUE‘).
`weights`	Optional weights vector name (character).
`cluster`	Optional cluster id variable name for robust notes (placeholder).
`groups`	Integer. Number of bins for calibration curve (binary models).
`top_n`	Integer. How many influential points to list/mark.

`explain`	Logical. If ‘TRUE‘, adds plain-English suggestions to notes.
`output`	One of ‘"both","summary","plots"‘.
`quiet`	Logical. Suppress messages.
`...`	Reserved for future options.

Value

An object of class ‘gt_assumption_check‘ with: - ‘$summary‘: tibble of assumption results - ‘$plots‘: named list of ggplot objects (may be empty) - ‘$details‘: raw test objects to aid reproducibility (always includes ‘$fit‘) - ‘$meta‘: list with ‘approach‘, ‘formula‘, ‘multivariate‘, ‘n‘, ‘weights_used‘

Examples

```
# Logistic example
df <- mtcars; df$am <- as.integer(df$am)
ac <- check_assumptions(
  data = df, outcome = "am", exposures = c("wt","hp"),
  approach = "auto", multivariate = TRUE, explain = TRUE
)
ac$summary
if (interactive()) plot(ac)

# Poisson example
ac2 <- check_assumptions(
  data = warpbreaks, outcome = "breaks",
  exposures = c("wool","tension"), approach = "auto", multivariate = TRUE
)
ac2$summary
```

`check_collinearity` *Check Collinearity Using VIF for Fitted Models*

Description

Computes Variance Inflation Factors (VIF) for fitted models returned by uni_reg(), multi_reg(), uni_reg_nbin(), or multi_reg_nbin(). Returns one VIF table per model. For multivariate models only

Usage

```
check_collinearity(model)
```

Arguments

`model`	A fitted model object with class "uni_reg", "multi_reg", "uni_reg_nbin", or "multi_reg_nbin".

Value

A tibble containing VIF values and interpretation. For multivariable models, returns one tibble. For univariate models, an error is raised indicating VIF is not applicable.

Examples

```
if (requireNamespace("gtregression", quietly = TRUE) &&
  requireNamespace("mlbench", quietly = TRUE) &&
  getRversion() >= "4.1.0") {
  data(PimaIndiansDiabetes2, package = "mlbench")
  pima <- PimaIndiansDiabetes2 |> dplyr::filter(!is.na(diabetes))
  pima$diabetes <- ifelse(pima$diabetes == "pos", 1, 0)
  fit <- multi_reg(pima,
    outcome = "diabetes",
    exposures = c("age", "mass", "glucose"),
    approach = "logit"
  )
  check_collinearity(fit)
}
```

`check_convergence` *Check Convergence for a Regression Model*

Description

Assesses model convergence and provides diagnostics for each exposure (in univariate mode) or for the full model (in multivariable mode), depending on the regression approach used.

Usage

```
check_convergence(
  data,
  exposures,
  outcome,
  approach = "logit",
  multivariate = FALSE
)
```

Arguments

`data`	A data frame containing the dataset.
`exposures`	A character vector of predictor variable names. If `multivariate = FALSE`, each exposure is assessed separately. If `multivariate = TRUE`, exposures are included together.
`outcome`	A character string specifying the outcome variable.
`approach`	A character string specifying the regression approach. One of: `"logit"`, `"log-binomial"`, `"poisson"`, `"robpoisson"`, or `"negbin"`.
`multivariate`	Logical. If `TRUE`, checks convergence for a multivariable model; otherwise, performs checks for each univariate model.

Details

For `robpoisson`, predicted probabilities (fitted values) may exceed 1, which is acceptable when estimating risk ratios but should not be interpreted as actual probabilities.

This function is useful for identifying convergence issues, especially for `"log-binomial"` models, which often fail to converge .

Value

A data frame summarizing convergence diagnostics, including:

`Exposure`	Name of the exposure variable.
`Model`	The regression approach used.
`Converged`	`TRUE` if the model converged successfully; `FALSE` otherwise.
`Max.prob`	Maximum predicted probability or fitted value in the dataset.

See Also

[identify_confounder()], [interaction_models()]

Examples

```
if (requireNamespace("gtregression", quietly = TRUE)) {
  data(data_PimaIndiansDiabetes, package = "gtregression")

  check_convergence(
    data = data_PimaIndiansDiabetes,
    exposures = c("age", "bmi"),
    outcome = "diabetes",
    approach = "logit"
  )

  check_convergence(
    data = data_PimaIndiansDiabetes,
    exposures = c("age", "bmi"),
    outcome = "diabetes",
    approach = "logit",
    multivariate = TRUE
  )
}
```

`data_birthwt` *Birth Weight Data*

Description

A dataset from the **MASS** package containing risk factors associated with low birth weight (LBW) in newborns. Originally collected at Baystate Medical Center, Springfield, Massachusetts, USA.

Usage

```
data_birthwt
```

Format

A data frame with 189 observations and 10 variables:

low Indicator for birth weight < 2500g (binary): `0 = normal`, `1 = low birth weight`

age Mother's age in years (numeric)

lwt Mother's weight in pounds at last menstrual period (numeric)

race Mother's race (factor): `1 = White`, `2 = Black`, `3 = Other`

smoke Smoking status during pregnancy (binary): `0 = No`, `1 = Yes`

ptl Number of previous premature labors (integer)

ht History of hypertension (binary): `0 = No`, `1 = Yes`

ui Presence of uterine irritability (binary): `0 = No`, `1 = Yes`

ftv no of physician visits during the 1st trimester (integer, 0–6)

bwt Birth weight in grams (numeric)

Details

The outcome variable is binary ('low'): birth weight < 2500g (yes = 1) or not (no = 0).

Source

Hosmer, D.W., Lemeshow, S. (1989). *Applied Logistic Regression.* New York: Wiley. Also available in **MASS** and described in detail in its documentation.

`data_epilepsy` *Epilepsy Treatment and Seizure Counts*

Description

RCT on the effect of a drug on the seizures in patients with epilepsy. Contains repeated measures data with treatment groups, baseline seizure counts, and follow-up counts.

Usage

```
data_epilepsy
```

Format

A data frame with 236 observations and 9 variables:

y Number of seizures in a 2-week period (count)

trt Treatment group (factor): `placebo` or `progabide`

base Seizure count during baseline period (numeric)

age Age of patient (numeric)

V4 Indicator for 4th visit (binary)

subject Patient ID (factor)

period Follow-up period number (integer)

lbase Log of baseline seizures (numeric)

lage Log of age (numeric)

Source

MASS package. Original data from Thall and Vail (1990)

`data_gt_quin` *Student Absenteeism in Rural Schools*

Description

This dataset contains observations on the number of days absent from school for children in rural Australia, along with student characteristics. It's commonly used to demonstrate count models such as Poisson and Negative Binomial regression.

Usage

```
data_gt_quin
```

Format

A data frame with 146 observations and 5 variables:

Eth Ethnicity (`"A"` = Aboriginal, `"N"` = Non-Aboriginal)

Sex Sex (`"F"` or `"M"`)

Age Age group (`"F0"`, `"F1"`, `"F2"`, `"F3"`)

Lrn Learner status (`"AL"` = average learner, `"SL"` = slow learner)

Days Number of days absent from school (count outcome)

Source

MASS package. See also Venables and Ripley (2002), *Modern Applied Statistics with S*.

`data_infertility` *Infertility Matched Case-Control Study*

Description

investigating the relationship between infertility and abortions.

Usage

```
data_infertility
```

Format

A data frame with 248 observations and 8 variables:

education Education level (0 = 0–5 years, 1 = 6–11 years, 2 = 12+ years)

age Age in years

parity Number of prior pregnancies

induced Number of induced abortions

case Infertility case status (1 = case, 0 = control)

spontaneous Number of spontaneous abortions

stratum Matched set ID

pooled.stratum Pooled stratum ID used for conditional regression

Source

`https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/infert.html`

`data_lungcancer` *Lung Cancer Trial Data*

Description

Survival data from a clinical trial of lung cancer patients conducted by the Veteran's Administration.

Usage

```
data_lungcancer
```

Format

A data frame with 137 observations and 8 variables:

trt Treatment group (1 = standard, 2 = test)

celltype Cell type (squamous, smallcell, adeno, large)

time Survival time (in days)

status Censoring status (1 = died, 0 = censored)

karno Karnofsky performance score (higher = better)

diagtime Months from diagnosis to randomization

age Age in years

prior Prior therapy (0 = no, 10 = yes)

Source

https://CRAN.R-project.org/package=survival

References

Kalbfleisch JD and Prentice RL (1980). The Statistical Analysis of Failure Time Data.

`data_PimaIndiansDiabetes`

PimaIndians2 Diabetes Dataset

Description

A cleaned version of the original Pima Indians Diabetes dataset from the ‘mlbench‘ package. Useful for demonstrating regression approaches for binary outcomes.

Usage

```
data_PimaIndiansDiabetes
```

Format

A data frame with 768 observations and 9 variables:

pregnant Number of times pregnant

glucose Plasma glucose concentration (glucose tolerance test)

pressure Diastolic blood pressure (mm Hg)

triceps Triceps skin fold thickness (mm)

insulin 2-Hour serum insulin (mu U/ml)

mass Body mass index (BMI)

pedigree Diabetes pedigree function

age Age in years

diabetes Factor indicating diabetes status (pos/neg)

Source

https://www.openml.org/d/37

`descriptive_table` *Descriptive Summary Table (no gtsummary) using gt/flextable*

Usage

```
descriptive_table(
  data,
  exposures,
  by = NULL,
  percent = c("column", "row"),
  digits = 1,
  show_missing = c("ifany", "no"),
  show_dichotomous = c("all_levels", "single_row"),
  show_overall = c("no", "first", "last"),
  statistic = NULL,
  value = NULL,
  format = c("gt", "flextable"),
  theme = c("minimal")
)
```

Arguments

`data`	data.frame
`exposures`	character; variables to summarise
`by`	optional single grouping variable
`percent`	"column" (default) or "row"; aliases like "col"/"rows" accepted
`digits`	integer; decimals for \itemshow_missing"ifany" (default) or "no" \itemshow_dichotomous"all_levels" (default) or "single_row" \itemshow_overall"no" (default), "first", or "last" \itemstatisticoptional named vector per continuous var: values in "mean","median","mode","count" (default is "median" = Median (IQR)) \itemvalueoptional named list for single-row binaries (e.g., list(sex="Female")) \itemformat"gt" (default) or "flextable" \itemthemepreset or primitives

list with class c("gtregression","descriptive_table", <engine>):

- $table: gt_tbl or flextable
- $table_display: display-ready data
- $table_body: long audit data (var/level/type)

- metadata fields

Publication-ready summary of categorical and continuous variables (optionally stratified). Mimics the OG gtsummary style: * column headers include N, e.g. "Overall, N=200" * categorical rows shown as n (* continuous rows default to Median (IQR) (footnote reflects summary)

`dissect` *Dissect a Dataset Before Regression*

Description

Returns a tidy summary of each variable's structure, missingness, uniqueness, and suitability for use in regression models.

Usage

```
dissect(data)
```

Arguments

`data`	A data frame.

Value

A tibble with columns: Variable, Type, Missing (and Regression Hint.

Examples

```
dissect(data_birthwt)
```

`forest_reg` *Draw a Forest Plot (Publication-Ready)*

Description

Wrapper around 'forestploter::forest()' that works directly with 'forest_df()' output or with raw regression objects.

Usage

```
forest_reg(
  df = NULL,
  uni = NULL,
  multi = NULL,
  desc = NULL,
  theme = NULL,
  ci_col_width = 0.25,
  side = c("right", "left"),
  quiet = TRUE,
  ...
)
```

Arguments

`df`	Output of ‘forest_df()‘. If ‘NULL‘, will be built from (uni, multi, desc).
`uni, multi, desc`	Optional gtregression objects to pass through to ‘forest_df()‘.
`theme`	Optional ‘forestploter::forest_theme()‘. If ‘NULL‘, a sensible default is used. You may pass colors and styling either here (e.g., ‘ci_col‘, ‘refline_gp‘) or via ‘...‘.
`ci_col_width`	Numeric or length-2 numeric. Relative width of the CI column(s). A vector like ‘c(0.22, 0.26)‘ lets you tune uni vs adjusted columns separately.
`side`	Character. For each effect, position of the plot relative to the effect-size text: ‘"left"‘ = plot first then text; ‘"right"‘ = text first then plot. **Note:** The ‘Characteristic‘ column (and any descriptive/summary columns) always remains on the left.
`quiet`	Logical. Suppress forestploter warnings. Default = ‘TRUE‘.
`...`	Passed to ‘forestploter::forest()‘. Common options include: ‘ci_col‘, ‘point_col‘, ‘point_shape‘, ‘rowheight‘, ‘ticks_at‘, ‘title‘, ‘footnote‘.
`bold_headers`	Logical. Bold the exposure headers (non-indented rows) in the first column. Default ‘TRUE‘.

Value

A ‘gtregression_forest‘ object with elements: - ‘plot‘: the forest plot - ‘data‘: the input data frame (post-processed order, no ‘se_*‘ columns) - ‘meta‘: model metadata

---

`identify_confounder` *Identify Confounders Using the Change-in-Estimate Method*

Description

Identifies whether one or more variables are confounders by comparing the crude and adjusted effect estimates of a primary exposure on an outcome. A variable is flagged as a confounder if its inclusion changes the estimate by more than a specified threshold (default = 10

Usage

```
identify_confounder(
  data,
  outcome,
  exposure,
  potential_confounder,
  approach = "logit",
  threshold = 10
)
```

Arguments

`data`	A data frame containing the variables.
`outcome`	The name of the outcome variable (character string).
`exposure`	The primary exposure variable (character string).
`potential_confounder`	One or more variables to test as potential confounders.
`approach`	The regression modeling approach. One of: `"logit"`, `"log-binomial"`, `"poisson"`, `"negbin"`, `"robpoisson"`, or `"linear"`.
`threshold`	Numeric. Percent change threshold to define confounding (default = 10). If the absolute percent change exceeds this, the variable is flagged as a confounder.

Details

Supports logistic, log-binomial, Poisson, robust Poisson, negative binomial, and linear regression approaches.

This method does not evaluate effect modification. Use causal diagrams (e.g., DAGs) and subject-matter knowledge to supplement decisions.

Value

If one confounder is provided, prints crude and adjusted estimates with a confounding flag. If multiple are given, returns a tibble with:

covariate Name of potential confounder.

crude_est Crude effect estimate.

adjusted_est Adjusted estimate including the confounder.

pct_change Percent change from crude to adjusted.

is_confounder Logical: whether confounding threshold is exceeded.

See Also

[check_convergence()], [interaction_models()]

Examples

```
data <- data_PimaIndiansDiabetes
identify_confounder(
  data = data,
  outcome = "glucose",
  exposure = "insulin",
  potential_confounder = "age_cat",
  approach = "linear"
)
```

`interaction_models` *Compare Models With and Without Interaction Term*

Description

This function fits two models—one with and one without an interaction term between an exposure and a potential effect modifier— and compares them using either a likelihood ratio test (LRT) or Wald test. It is useful for assessing whether there is statistical evidence of interaction (effect modification).

Usage

```
interaction_models(
  data,
  outcome,
  exposure,
  covariates = NULL,
  effect_modifier,
  approach = "logit",
  test = c("LRT", "Wald"),
  verbose = TRUE
)
```

Arguments

`data`	A data frame containing all required variables.
`outcome`	The name of the outcome variable
`exposure`	The name of the main exposure variable.
`covariates`	character vector of additional covariates to adjust for
`effect_modifier`	The name of the variable to test for interaction
`approach`	The regression modeling approach to use. One of: `"logit"`, `"log-binomial"`, `"poisson"`, `"robpoisson"`, `"negbin"`, or `"linear"`.
`test`	Type of statistical test for model comparison. Either: `"LRT"` (likelihood ratio test, default) or `"Wald"`.
`verbose`	Logical; if `TRUE`, prints a basic interpretation of whether interaction is likely present (default = `FALSE`).

Value

A list with the following elements:

- `model_no_interaction`: The model without the interaction term.
- `model_with_interaction`: The model with the interaction term.
- `p_value`: The p-value for interaction (based on selected test).
- `interpretation`: A brief text interpretation if `verbose = TRUE`.

Examples

```
data <- data_PimaIndiansDiabetes
```

`merge_tables` *Merge tables (descriptive / uni / multi) and preserve look & notes*

Description

Merge tables (descriptive / uni / multi) and preserve look & notes

Usage

```
merge_tables(..., spanners = NULL, theme = "minimal")
```

Arguments

`...`	package tables with $table_display (same engine)
`spanners`	labels over each panel
`theme`	merge theme (preset or primitives)

`modify_table` *Modify Regression/Descriptive Tables (labels, headers, caption, notes)*

Usage

```
modify_table(
  gt_table,
  variable_labels = NULL,
  level_labels = NULL,
  header_labels = NULL,
  caption = NULL,
  bold_labels = FALSE,
  bold_levels = FALSE,
  remove_N = FALSE,
  remove_N_obs = FALSE,
  remove_abbreviations = FALSE,
  caveat = NULL
)
```

Arguments

`gt_table`	Table object produced by this package (must contain ‘$table_display‘).
`variable_labels`	Named character vector: ‘c(old_var = "New label", ...)‘.
`level_labels`	Named list for factor levels: ‘list(var1 = c(old = "New", ...), var2 = c(...))‘.
`header_labels`	Named vector to rename visible headers, e.g. ‘c("OR (95 \itemcaptionOptional caption/title. \itembold_labelsLogical; bold variable (header) rows in the body. \itembold_levelsLogical; bold factor level rows in the body. \itemremove_NLogical; if ‘TRUE‘, drops the ‘N‘ column for univariate package tables. \itemremove_N_obsLogical; if ‘TRUE‘, suppresses multivariable complete-case footnote. \itemremove_abbreviationsLogical; if ‘TRUE‘, removes the Abbreviations footnote line. \itemcaveatOptional extra footnote.

The modified table object (same class as input).

Works with objects created by this package (class ‘"gtregression"‘: ‘uni_reg()‘, ‘multi_reg()‘, ‘descriptive_table()‘, ‘merge_tables()‘). No ‘gtsummary‘ dependency or fallback.

`multi_reg` *Multivariable regression*

Description

Create a publication-ready multivariable regression table using either **gt** or **flextable**, without a gtsummary dependency.

Usage

```
multi_reg(
  data,
  outcome,
  exposures,
  approach = "logit",
  format = c("gt", "flextable"),
  theme = c("minimal")
)
```

Arguments

`data`	data.frame
`outcome`	character scalar; outcome column name
`exposures`	character vector; exposure column names (all included in one model)
`approach`	one of `"logit"`, `"log-binomial"`, `"poisson"`, `"linear"`, `"robpoisson"`, or `"negbin"`
`format`	one of `"gt"` (default) or `"flextable"`
`theme`	preset name (e.g. `"minimal"`, `"striped"`, `"clinical"`, `"shaded"`, `"jama"`) or primitives `c("plain","zebra","lines","labels_bold","compact","header_shaded")`

Value

A list of class `c("gtregression","multi_reg", ...)` with elements:

table A `gt_tbl` (when `format="gt"`) or `flextable` (when `format="flextable"`).

table_body Data frame of adjusted estimates and CIs (per level).

table_display Data frame for display (headers + levels) without N column.

models List with the single multivariable model.

model_summaries `summary()` of the fitted model.

reg_check Diagnostics for linear model; message otherwise.

approach, format, source Metadata fields.

Examples

```
d <- mtcars
if (requireNamespace("gt", quietly = TRUE)) {
  multi_reg(d, "am", c("mpg","cyl","gear"), approach = "logit", format = "gt")$table
}
if (requireNamespace("flextable", quietly = TRUE)) {
 multi_reg(d, "am", c("mpg","cyl","gear"), approach = "logit", format = "flextable")$table
}
```

`plot_reg` *Visualize a Regression Model as a Forest Plot*

Description

Creates a forest plot from a ‘gtsummary‘-style object produced by ‘gtregression‘ functions (e.g., ‘uni_reg()‘, ‘multi_reg()‘). The function supports both univariate and multivariable models, renders hierarchical labels (variable headers vs. levels), and computes significance highlighting using either *p*-values (linear models) or CI-vs-reference rules (non-linear models).

Usage

```
plot_reg(
  tbl,
  title = NULL,
  order_y = NULL,
  log_x = FALSE,
  xlim = NULL,
  breaks = NULL,
  point_color = "#1F77B4",
  errorbar_color = "#4C4C4C",
  base_size = 14,
  show_ref = TRUE,
  sig_color = NULL,
  sig_errorbar_color = NULL,
  alpha = 0.05
)
```

Arguments

`tbl`	A ‘gtsummary‘-like object returned by ‘gtregression‘ (must contain ‘table_body‘ and attributes ‘source‘ and ‘approach‘).
`title`	Optional plot title (character).
`order_y`	Optional character vector to customize the y-axis header ordering.
`log_x`	Logical. If ‘TRUE‘, log x-axis (ignored for linear models).
`xlim`	Optional numeric vector of length 2 for x-axis limits.
`breaks`	Optional numeric vector for x-axis tick breaks (ignored if ‘log_x = TRUE‘).
`point_color`	Fill color for points (default ‘"#1F77B4"‘).
`errorbar_color`	Color for all error bars (default ‘"#4C4C4C"‘).
`base_size`	Base font size for ‘theme_minimal()‘ (default ‘14‘).
`show_ref`	Logical. If ‘TRUE‘, includes the reference level on the plot and labels it ‘(Ref.)‘.
`sig_color`	Optional fill color for **significant** points; if ‘NULL‘, significant points reuse ‘point_color‘.
`sig_errorbar_color`	Optional color for **significant** error bars; if ‘NULL‘, significant bars reuse ‘errorbar_color‘.
`alpha`	Significance level for linear models when ‘p.value‘ is available (default ‘0.05‘).

Details

Reference line: The vertical reference is fixed at ‘0‘ for linear models and ‘1‘ for all other approaches, inferred from ‘attr(tbl, "approach")‘.

Header / data detection: Variable headers are recognized via ‘row_type == "label"‘ together with ‘header_row‘ or missing CI; categorical levels use ‘row_type == "level"‘; continuous predictors appear as ‘row_type == "label"‘ **with** CIs and are treated as data rows.

Significance highlighting: - For ‘approach == "linear"‘ with available ‘p.value‘, rows are significant when ‘p.value < alpha‘. - Otherwise, rows are significant when the CI does not cross the reference (‘0‘ or ‘1‘ as above). Use ‘sig_color‘ / ‘sig_errorbar_color‘ to customize the appearance.

Value

A ‘ggplot2‘ object representing the forest plot.

See Also

`uni_reg`, `multi_reg`, `plot_reg_combine`

Examples

```
if (requireNamespace("mlbench", quietly = TRUE) &&
    requireNamespace("gtregression", quietly = TRUE)) {
  data("PimaIndiansDiabetes2", package = "mlbench")
  pima <- PimaIndiansDiabetes2
  pima$diabetes <- ifelse(pima$diabetes == "pos", 1, 0)
  pima$bmi_cat <- cut(
    pima$mass,
    breaks = c(-Inf, 18.5, 24.9, 29.9, Inf),
    labels = c("Underweight", "Normal", "Overweight", "Obese")
  )

  # Univariate logistic regression table via gtregression
  tbl_uni <- gtregression::uni_reg(
    data = pima,
    outcome = "diabetes",
    exposures = c("age", "bmi_cat"),
    approach = "logit"
  )

  p <- plot_reg(tbl_uni, title = "Univariate (logit)", sig_color = "#D55E00")
  print(p)
}
```

`plot_reg_combine` *Side-by-Side Forest Plots: Univariate vs Multivariable*

Description

Creates two aligned forest plots (univariate and multivariable) from ‘gtsummary‘-style objects returned by ‘gtregression‘ functions (e.g., ‘uni_reg()‘, ‘multi_reg()‘).

Usage

```
plot_reg_combine(
  tbl_uni,
  tbl_multi,
  title_uni = NULL,
  title_multi = NULL,
  ref_line = NULL,
  order_y = NULL,
  log_x = FALSE,
  point_color = "#1F77B4",
  errorbar_color = "#4C4C4C",
  base_size = 14,
  show_ref = TRUE,
  sig_color = NULL,
  sig_errorbar_color = NULL,
  xlim_uni = NULL,
  breaks_uni = NULL,
  xlim_multi = NULL,
  breaks_multi = NULL,
  alpha = 0.05
)
```

Arguments

`tbl_uni`	Univariate ‘gtsummary‘-like table.
`tbl_multi`	Multivariable ‘gtsummary‘-like table.
`title_uni`, `title_multi`	Optional panel titles.
`ref_line`	Optional numeric reference line (defaults to 0 for linear, 1 otherwise, inferred per panel).
`order_y`	Optional character vector to customize header ordering.
`log_x`	Logical. If ‘TRUE‘, use log x-axis (ignored for linear models).
`point_color`, `errorbar_color`	Base colors for non-significant rows.
`base_size`	Base font size for ‘theme_minimal()‘.
`show_ref`	Logical; if ‘TRUE‘, include and tag reference levels ‘(Ref.)‘.
`sig_color`, `sig_errorbar_color`	Optional colors for significant rows; if ‘NULL‘, they reuse the base colors.
`xlim_uni`, `breaks_uni`	Optional x-limits and breaks for the univariate panel.
`xlim_multi`, `breaks_multi`	Optional x-limits and breaks for the multivariable panel.
`alpha`	Significance level for linear models when ‘p.value‘ is available.

Details

The y-axis rows are aligned by a unique ‘(variable, level)‘ key so each estimate appears exactly once per panel. Label styling is plain text by default (CRAN-safe). To render bold headers / grey refs in vignettes, pair

Value

A ‘patchwork‘ object with two ‘ggplot2‘ panels.

Examples

```
if (requireNamespace("mlbench", quietly = TRUE) &&
    requireNamespace("gtregression", quietly = TRUE)) {
  data("PimaIndiansDiabetes2", package = "mlbench")
  d <- PimaIndiansDiabetes2
  d$diabetes <- ifelse(d$diabetes == "pos", 1, 0)

  tbl_u <- gtregression::uni_reg(d, outcome = "diabetes",
                               exposures = c("age","glucose"), approach = "logit")
  tbl_m <- gtregression::multi_reg(d, outcome = "diabetes",
                                 exposures = c("age","glucose"), approach = "logit")
  plot_reg_combine(tbl_u, tbl_m,
                   title_uni = "Univariate", title_multi = "Adjusted")
}
```

`print.gtregression` *Print gtregression objects (unified)*

Description

Prints the rendered table for any object produced by this package (objects that include class `"gtregression"`), regardless of subtype (`uni_reg`, `multi_reg`, `stratified_*`, `merged_table`, `descriptive_table`, ...). If no rendered table is found, a compact structure of the object (or its display data) is shown.

Usage

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

Arguments

`x`	An object with class `"gtregression"`.
`...`	Ignored. Present for compatibility with the generic.

`save_docx` *Save Multiple Tables and Plots to a Word Document*

Description

Saves a collection of gtsummary tables and ggplot2 plots into a .docx file.

Usage

```
save_docx(tables = NULL, plots = NULL, filename = "report.docx", titles = NULL)
```

Arguments

`tables`	A list of gtsummary tables.
`plots`	A list of ggplot2 plot objects.
`filename`	File name for the output (with or without .docx extension).
`titles`	Optional. A character vector of titles.

Value

A Word document saved to a temporary directory (if no path is given). No object is returned.

Examples

```
library(gtsummary)
library(ggplot2)
tbl <- tbl_regression(glm(mpg ~ hp + wt, data = mtcars))
p <- ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
save_docx(
  tables = list(tbl),
  plots = list(p),
  filename = file.path(tempdir(), "report.docx"),
  titles = c("Table 1: Regression", "Figure 1: Scatterplot")
)
```

`save_plot` *Save a Single Plot*

Description

Saves a ggplot2 plot to a file in PNG, PDF, or JPG format.

Usage

```
save_plot(
  plot,
  filename = "plot",
  format = c("png", "pdf", "jpg"),
  width = 8,
  height = 6,
  dpi = 300
)
```

Arguments

`plot`	A ggplot2 object.
`filename`	Name of the file to save, with or without extension.
`format`	Output format. One of "png", "pdf", or "jpg".
`width`	Width of the saved plot in inches.
`height`	Height of the saved plot in inches.
`dpi`	Resolution of the plot in dots per inch (default is 300).

Value

Saves the file to a temporary directory (if no path is given).

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
save_plot(p, filename = file.path(tempdir(), "scatterplot"), format = "png")
```

`save_table` *Save a Single Regression Table*

Description

Saves a gtsummary table as a Word, PDF, or HTML file

Usage

```
save_table(tbl, filename = "table", format = c("docx", "pdf", "html"))
```

Arguments

`tbl`	A gtsummary object (e.g., tbl_regression(), tbl_summary()).
`filename`	File name to save the output. Extension is optional.
`format`	Output format. One of "docx", "pdf", or "html".

Value

Saves the file to a temporary directory (if no path is given). Does not return an object.

Examples

```
model <- glm(mpg ~ hp + wt, data = mtcars)
tbl <- gtsummary::tbl_regression(model)
save_table(tbl, filename = file.path(tempdir(), "regression_table"), format = "docx")
```

`select_models` *Stepwise Model Selection with Evaluation Metrics*

Description

Performs stepwise model selection using forward, backward, or both directions across different regression approaches. Returns a summary table with evaluation metrics (AIC, BIC, log-likelihood, deviance) and the best model.

Usage

```
select_models(
  data,
  outcome,
  exposures,
  approach = "logit",
  direction = "forward"
)
```

Arguments

`data`	A data frame containing the outcome and predictor variables.
`outcome`	A character string indicating the outcome variable.
`exposures`	vector of predictor variables to consider in the model.
`approach`	Regression method. One of: `"logit"`, `"log-binomial"`, `"poisson"`, `"robpoisson"`, `"negbin"`, or `"linear"`.
`direction`	Stepwise selection direction. One of: `"forward"` (default), `"backward"`, or `"both"`.

Value

A list with the following components:

- `results_table`: A tibble summarising each tested model's metric (AIC, BIC, deviance, log-likelihood, adjusted R² if applicable).
- `best_model`: The best-fitting model object based on low AIC.
- `all_models`: A named list of all fitted models.

Examples

```
data <- data_PimaIndiansDiabetes
stepwise <- select_models(
  data = data,
  outcome = "glucose",
  exposures = c("age", "pregnant", "mass"),
  approach = "linear",
  direction = "forward"
)
summary(stepwise)
stepwise$results_table
stepwise$best_model
```

`stratified_multi_reg` *Stratified multivariable regression (wide, adjusted; no gtsummary)*

Description

Fits one multivariable model per stratum and returns a unified wide table: a single "Characteristic" column and, under bold spanners for each stratum, two columns: "Adjusted <effect>" and "p-value".

Usage

```
stratified_multi_reg(
  data,
  outcome,
  exposures,
  stratifier,
  approach = "logit",
  format = c("gt", "flextable"),
  theme = c("minimal")
)
```

Arguments

`data`	data.frame
`outcome`	character scalar; outcome column name
`exposures`	character vector; predictors included in each model
`stratifier`	character scalar; stratifying variable
`approach`	"logit","log-binomial","poisson","linear","robpoisson","negbin"
`format`	"gt" (default) or "flextable"
`theme`	preset (e.g. "minimal","striped","clinical","shaded","jama") or primitives c("plain","zebra","lines","labels

Details

The footer shows two lines: 1) Abbreviations (from ‘.abbrev_note()‘), 2) Per-stratum complete-case N used in the multivariable model.

Value

A list of class `c("gtregression","stratified_multi_reg", ...)` with:

table	A `gt_tbl` (format="gt") or `flextable` (format="flextable").
table_display	Wide data.frame used to build the table.
per_stratum	Named list of per-stratum results (models/summaries/diagnostics).
models, model_summaries, reg_check	Named lists by stratum.
by, levels, approach, format, source	Metadata fields.

Examples

```
if (requireNamespace("mlbench", quietly = TRUE) &&
  requireNamespace("dplyr", quietly = TRUE)) {
  data(PimaIndiansDiabetes2, package = "mlbench")
  pima <- dplyr::mutate(
    PimaIndiansDiabetes2,
    diabetes = ifelse(diabetes == "pos", 1, 0),
    glucose_cat = dplyr::case_when(
      glucose < 140 ~ "Normal",
      glucose >= 140 ~ "High"
    )
  )
  stratified_multi <- stratified_multi_reg(
    data = pima,
    outcome = "diabetes",
    exposures = c("age", "mass"),
    stratifier = "glucose_cat",
    approach = "logit"
  )
  stratified_multi$table
}
```

`stratified_uni_reg` *Performs univariate regression for each exposure on a binary, count, or continuous outcome, stratified by a specified variable. Produces a stacked ‘gtsummary‘ table with one column per stratum, along with underlying models and diagnostics.*

Description

Performs univariate regression for each exposure on a binary, count, or continuous outcome, stratified by a specified variable. Produces a stacked ‘gtsummary‘ table with one column per stratum, along with underlying models and diagnostics.

Usage

```
stratified_uni_reg(
  data,
  outcome,
  exposures,
  stratifier,
  approach = "logit",
  format = c("gt", "flextable"),
  theme = c("minimal")
)
```

Arguments

`data`	A data frame containing the variables.
`outcome`	name of the outcome variable.
`exposures`	A vector specifying the predictor (exposure) variables.
`stratifier`	A character string specifying the stratifier
`approach`	Modeling approach to use. One of: ‘"logit"‘ (Odds Ratios), ‘"log-binomial"‘ (Risk Ratios), ‘"poisson"‘ (Incidence Rate Ratios), ‘"robpoisson"‘ (Robust RR), ‘"linear"‘ (Beta coefficients), ‘"negbin"‘ (Incidence Rate Ratios),.

Value

An object of class ‘stratified_uni_reg‘, which includes: - ‘table‘: A ‘gtsummary::tbl_stack‘ object with stratified results, - ‘models‘: A list of fitted models for each stratum, - ‘model_summaries‘: A tidy list of model summaries, - ‘reg_check‘: A tibble of regression diagnostics (when available).

Accessors

`$table` Stacked stratified regression table.

`$models` List of fitted model objects for each stratum.

`$model_summaries` List of tidy model summaries.

`$reg_check` Diagnostic check results (when applicable).

See Also

[multi_reg()], [plot_reg()], [identify_confounder()]

Examples

```
if (requireNamespace("mlbench", quietly = TRUE) &&
  requireNamespace("dplyr", quietly = TRUE)) {
  data(PimaIndiansDiabetes2, package = "mlbench")
  pima <- dplyr::mutate(
    PimaIndiansDiabetes2,
    diabetes = ifelse(diabetes == "pos", 1, 0),
    glucose_cat = dplyr::case_when(
```

```
      glucose < 140 ~ "Normal",
      glucose >= 140 ~ "High"
    )
  )
  stratified_uni <- stratified_uni_reg(
    data = pima,
    outcome = "diabetes",
    exposures = c("age", "mass"),
    stratifier = "glucose_cat",
    approach = "logit"
  )
  stratified_uni$table
}
```

`uni_reg` *Univariate regression*

Description

Create a publication-ready univariate regression table using either **gt** or **flextable**

Usage

```
uni_reg(
  data,
  outcome,
  exposures,
  approach = "logit",
  format = c("gt", "flextable"),
  theme = c("minimal")
)
```

Arguments

`data`	data.frame
`outcome`	character scalar; outcome column name
`exposures`	character vector; exposure column names
`approach`	one of `"logit"`, `"log-binomial"`, `"poisson"`, `"linear"`
`format`	one of `"gt"` (default) or `"flextable"`
`theme`	preset name (e.g. `"minimal"`, `"striped"`, `"clinical"`, `"shaded"`, `"jama"`) or primitives `c("plain","zebra","lines","labels_bold","compact","header_shaded")`

Value

A list of class `c("gtregression","uni_reg", ...)` with elements:

table A `gt_tbl` (when `format="gt"`) or `flextable` (when `format="flextable"`).

table_body Data frame of numeric estimates and CIs.

table_display Data frame for display (headers + levels).

models List of fitted univariate models.

model_summaries Per-model `summary()` results.

reg_check Diagnostics for linear models; message otherwise.

approach, format, source Metadata fields.

Examples

```
d <- mtcars
if (requireNamespace("gt", quietly = TRUE)) {
  uni_reg(d, "am", c("mpg","cyl"), approach = "logit", format = "gt")$table
}
if (requireNamespace("flextable", quietly = TRUE)) {
  uni_reg(d, "am", c("mpg","cyl"), approach = "logit", format = "flextable")$table
}
```

`$.gtregression` *Access fields on gtregression objects with ‘$‘*

Description

Works for any object from this package, since they all carry class ‘"gtregression"‘. Returns NULL (quietly) if the field isn’t present.

Usage

```
## S3 method for class 'gtregression'
x$name
```

Details

Common fields: - table, table_display, table_body - models, model_summaries, reg_check - approach, format (or engine), source - parts, spanners (for merged tables) - by, levels (for descriptive tables)

Index