Package 'gtregression'

October 20, 2026

Type Package

Title Tools for Creating Publication-Ready Regression Tables

Version 1.2.0

Description Simplifies regression modelling in R by integrating modelling, diagnostics, visualisation, and summary tools into a cohesive, user-friendly interface. Designed to be accessible for researchers, particularly those in Low- and Middle-Income Countries (LMIC). Supports logistic, linear, count, matched case-control, ordinal, multinomial, multilevel, Cox, parametric survival, and competing-risk regression, and absolute risk differences with the number needed to treat. 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>), Firth penalized logistic regression (Firth 1993 <doi:10.1093/biomet/80.1.27>), Poisson and robust Poisson regression (Zou 2004 <doi:10.1093/aje/kwh090>), negative binomial regression (Hilbe 2011, ISBN:9780521179515), Cox proportional hazards regression, parametric survival regression, causal mediation analysis, risk differences (Miettinen and Nurminen 1985 <doi:10.1002/sim.4780040211>), 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

Language en-US

LazyData true

Depends R (>= 4.1.0)

Imports dplyr, risks, purrr, MASS, rlang, stats, lmtest, patchwork, ggplot2, utils, sandwich, tibble, broom, broom.helpers, gt (>= 0.6.0), officer, flextable, scales (>= 1.3.0), forestploter, logistf, survival, cmprsk, lme4, ordinal, nnet

VignetteBuilder knitr

Suggests haven, testthat (>= 3.1.5), knitr, rmarkdown, car, ggtext, htmltools, jsonlite, shiny, rstudioapi, DT, readxl, rio

Config/testthat/edition 3

Config/Needs/website pkgdown

URL https://gtregression.thinkdenominator.com/, https://github.com/ThinkDenominator/gtregression

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

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Rubeshkumar Polani [aut, cre] (ORCID: <https://orcid.org/0000-0002-0418-7592>), Salin K Eliyas [aut] (ORCID: <https://orcid.org/0000-0002-8020-5860>), Manikandanesan Sakthivel [aut] (ORCID: <https://orcid.org/0000-0002-5438-3970>), Mogan Kaviprawin [aut] (ORCID: <https://orcid.org/0000-0002-1659-059X>), Yuvaraj Krishnamoorthy [aut] (ORCID: <https://orcid.org/0000-0003-4688-510X>), Marie Gilbert Majella [aut] (ORCID: <https://orcid.org/0000-0003-4036-5162>)

Maintainer Rubeshkumar Polani <rubesh@thinkdenominator.com>

Repository CRAN

Date/Publication 2026-10-20 13:30:15 UTC

Contents

check_collinearity 4
check_convergence 5
check_ph 6
compare_models 8
competing_risk_reg 10
conditional_logit_reg 12
cox_reg 13
cuminc_plot 18
data_birthwt 20
data_competing_risk 21
data_diabetes_mediation 22
data_endometrial 23
data_epilepsy 23
data_gt_quin 24
data_infertility 25
data_lungcancer 25
data_SynthDiabetes 26
descriptive_table 27
dissect 28

forest_df 29
forest_reg 30
gtregression_addin 33
gtregression_app 34
identify_confounder 35
import_data 38
interaction_models 39
km_plot 41
km_risk_table 44
logrank_test 46
mediation_analysis 47
merge_tables 49
modify_table 51
multilevel_reg 53
multinomial_reg 55
multi_reg 56
ordinal_reg 59
plot_cox_survival 60
plot_mediation 62
plot_model_fit 62
plot_reg 64
plot_reg_combine 65
plot_surv_fit 67
print.gtregression 69
risk_difference 70
rmst_table 73
save_docx 75
save_forest 76
save_plot 78
save_table 79
select_models 80
stratified_multi_reg 82
stratified_uni_reg 84
survival_prob 86
survival_quantiles 88
survival_summary 90
surv_model_compare 91
surv_predict 93
surv_reg 94
uni_reg 98
$.gtregression 100

Index **101**

`check_collinearity` *Check collinearity using VIF for fitted models*

Description

Computes Variance Inflation Factors (VIF) for fitted multivariable models returned by `multi_reg()` or related functions.

Usage

```
check_collinearity(model, format = c("flextable", "gt", "tibble"))
```

Arguments

`model`	A fitted model object returned by `multi_reg()`, `stratified_multi_reg()`, or compatible `gtregression` functions. Univariable model objects are rejected because VIF is not applicable.
`format`	Output format. One of `"flextable"` (default), `"gt"`, or `"tibble"`. Use `format = "tibble"` to preserve the original tibble or nested-list output.

Details

If the **car** package is installed, `check_collinearity()` uses `car::vif()`. Otherwise, it computes VIF/GVIF from the fitted model matrix so that diagnostics remain available without an additional dependency.

Value

For multivariable models, a tibble if a single fitted model is present, or a named list of tibbles if multiple fitted models are present. With `format = "gt"` or `format = "flextable"`, leaf tibbles are converted to formatted tables while preserving any list nesting. The tibble contains:

`Variable` Model term.

`VIF` Variance inflation factor. For multi-degree-of-freedom terms, this is the adjusted GVIF: `GVIF^(1 / (2 * Df))`.

`Interpretation` Simple interpretation based on common cut points: no collinearity, moderate, or high.

For univariate models, an error is raised indicating that VIF is not applicable.

check_convergence *Check regression model convergence*

Description

Check whether requested models converge, either one exposure at a time or as a single multivariable model.

Usage

```
check_convergence(
  data,
  exposures,
  outcome,
  approach = "logit",
  multivariate = FALSE,
  format = c("flextable", "gt", "tibble")
)
```

Arguments

data	A data frame containing the dataset.
exposures	A character vector of predictor variable names. Quoted names are recommended in scripts, and bare names are also accepted. If multivariate = FALSE, each exposure is assessed separately. If multivariate = TRUE, exposures are included together.
outcome	A character string specifying the outcome variable. Quoted and bare names are accepted.
approach	A character string specifying the regression approach. One of: "logit", "logbinomial", "poisson", "robpoisson", or "negbin".
multivariate	Logical. If TRUE, checks convergence for a multivariable model; otherwise, performs checks for each univariate model.
format	Output format. One of "flextable" (default), "gt", or "tibble". Use format = "tibble" for the original data-frame style output.

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 is especially useful for "logbinomial" models, where convergence problems are common and robust Poisson may be a practical alternative.

Value

A data frame, `gt_tbl`, or `flextable` 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_SynthDiabetes, package = "gtregression")

  check_convergence(
    data = data_SynthDiabetes,
    exposures = c("age", "mass"),
    outcome = "diabetes",
    approach = "logit",
    format = "tibble"
  )

  check_convergence(
    data = data_SynthDiabetes,
    exposures = c("age", "mass"),
    outcome = "diabetes",
    approach = "logit",
    multivariate = TRUE,
    format = "gt"
  )
}
```

---

check_ph *Check proportional hazards assumption for Cox models*

Description

Test the proportional hazards assumption for fitted Cox proportional hazards models using Schoenfeld residuals.

Usage

```
check_ph(
  model,
  transform = c("km", "rank", "identity"),
  alpha = 0.05,
  format = c("flextable", "gt", "tibble")
)
```

Arguments

model	A fitted coxph model or a cox_reg() object.
transform	Time transformation passed to survival::cox.zph(). One of "km", "rank", or "identity".
alpha	Significance level used for the simple interpretation column.
format	Output format. One of "flextable" (default), "gt", or "tibble".

Details

check_ph() is a diagnostic aid for Cox models. A small p-value suggests possible evidence against the proportional hazards assumption for a term or for the global model test. This should be interpreted with plots, clinical knowledge, follow-up patterns, and modelling purpose.

The returned table retains the underlying cox.zph diagnostics. Use plot() to draw scaled Schoenfeld residual plots. When several models were checked, select one by name or position with the model argument; use term to draw a single model term.

Value

A tibble, gt_tbl, or flextable with Schoenfeld residual proportional hazards tests. The table contains:

Model Model name. For a direct coxph object, this is "cox_model".

Term Model term or "GLOBAL".

Chi.square Chi-square statistic.

df Degrees of freedom.

p.value Test p-value.

Interpretation Simple screening interpretation using alpha.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

cox_fit <- cox_reg(
  data = lung_data,
  time = time,
  event = status,
```

```
  exposures = c(trt, celltype, age)
)

check_ph(cox_fit)
check_ph(cox_fit, format = tibble)
ph <- check_ph(cox_fit)
plot(ph, model = "trt")
```

`compare_models` *Compare prespecified regression models*

Description

Compare gtregression candidate models side by side using model-fit statistics. This is intended for transparent model comparison after you have already fitted the candidate models with functions such as `multi_reg()`, `cox_reg()`, or `surv_reg()`.

Usage

```
compare_models(
  ...,
  model_names = NULL,
  nested = TRUE,
  primary_exposure = NULL,
  exponentiate = NULL,
  digits = 2,
  p_digits = 3,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`...`	Two or more gtregression model objects, or one list containing them. Inputs should be outputs from `multi_reg()`, `cox_reg()`, or `surv_reg()`.
`model_names`	Optional character vector of names to display. If omitted, names supplied in `...` are used; otherwise models are labelled `Model 1`, `Model 2`, etc.
`nested`	Logical. If `TRUE`, likelihood-ratio statistics are calculated sequentially by comparing each model with the previous model. Use this only when models are nested and supplied in the intended order.
`primary_exposure`	Optional exposure or exact coefficient name to track across models. Quoted and bare names are accepted; a bare name resolves to a model variable first, not to a workspace object of the same name. For Cox models this can be used to show the hazard ratio and its percentage change across candidate models. "Change from first" is `100 * (estimate - first) / |first|` computed on the displayed

	scale: the ratio (OR, RR, IRR, HR, time ratio) for exponentiated models and the coefficient for linear models. This is the same scale as the change-in-estimate reported by `identify_confounder()`.
`exponentiate`	Logical. If `NULL`, Cox, logistic, Poisson, negative-binomial, and parametric survival models are exponentiated by default, while linear models are not.
`digits`	Number of digits for model statistics and estimates.
`p_digits`	Number of digits for p-values.
`format`	Output format. Defaults to `"flextable"`.
`theme`	Table theme preset.

Details

`compare_models()` does not refit models and does not perform hidden complete-case filtering. It compares models already fitted by gtregression and extracts the single fitted model stored in each object's `models` element. The reported N, event counts, and fit statistics therefore come from the model already fitted by `multi_reg()`, `cox_reg()`, or `surv_reg()`. This keeps model comparison separate from model selection: compare candidate models first, then choose the final model using clinical, epidemiological, and statistical judgement. The guided builder in `gtregression_app()` can create the fitted gtregression objects before calling this function; the generated app code exposes every model formula and comparison call.

Likelihood-ratio p-values are meaningful only for nested models fitted to the same analysis sample. `compare_models()` checks whether the fitted models appear to use the same analysis sample using retained model row identifiers when available; otherwise it compares N and event counts. It also checks whether sequential model pairs appear to be nested when `nested = TRUE`. Rendered warnings are context-aware: no warning about different analysis samples is shown when the compared models use the same observations, and no nested-model warning is shown when sequential models appear nested. AIC, BIC, log-likelihood, and likelihood-ratio statistics remain visible when warnings are needed, but should then be interpreted with the displayed caution.

Value

A `gtregression` object with:

- `table`: publication-ready table
- `table_body`: raw comparison statistics
- `table_display`: formatted display data
- `models`: fitted models compared
- `comparison_status`: whether models appear to use the same analysis sample
- `comparison_warnings`: caution messages that are highlighted in rendered tables when interpretation needs extra care

See Also

`gtregression_app()`, `multi_reg()`, `cox_reg()`, `surv_reg()`, `select_models()`

Examples

```
data("data_lungcancer", package = "gtregression")

lung_data <- data_lungcancer

cox_1 <- cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = trt
)

cox_2 <- cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = trt,
  adjust_for = c(age, karno)
)

compare_models(
  cox_1,
  cox_2,
  primary_exposure = trt
)

compare_models(
  cox_1,
  cox_2,
  model_names = c("Treatment only", "Treatment + age + performance"),
  primary_exposure = trt
)
```

---

competing_risk_reg *Fine-Gray competing-risk regression*

Description

Fit proportional subdistribution hazards models and return publication-ready subdistribution hazard ratios.

Usage

```
competing_risk_reg(
  data,
  time,
  event,
  event_of_interest,
```

```
  exposures,
  censor_code = 0,
  adjust_for = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  format = c("flextable", "gt"),
  theme = c("minimal"),
  model_stats = FALSE,
  show_ref = TRUE
)
```

Arguments

data	A data frame containing follow-up time, event status, and model variables.
time	Numeric follow-up-time variable. Quoted and bare names work.
event	Event-status variable with one censoring code and distinct codes for the event of interest and competing events.
event_of_interest	Observed code or label for the event being modelled.
exposures	Variables whose effects are reported.
censor_code	Observed code or label for censoring. Defaults to ‘0‘.
adjust_for	Optional adjustment variables. One adjusted model is fitted per exposure and only that exposure is reported.
multivariable	Logical; fit one model containing and reporting all variables in ‘exposures‘.
multivariate	Optional alias for ‘multivariable‘.
format	Output table format: ‘"flextable"‘ or ‘"gt"‘.
theme	Table styling preset.
model_stats	Logical; return N, event counts, pseudo-log-likelihood, and convergence status.
show_ref	Logical; display categorical reference rows.

Details

Each model uses the rows that are complete for its own variables. The N column and the footnote report the observations and the events of interest used by the fitted models, with the same wording as `cox_reg()` and `surv_reg()`.

Fine-Gray regression models the subdistribution hazard for the selected event. An SHR above 1 indicates a higher cumulative incidence of that event over follow-up, accounting for the presence of competing events. It is not a cause-specific Cox hazard ratio and should not be interpreted as one.

Value

A ‘gtregression‘ object containing the publication table, numeric estimates, fitted ‘crr‘ models, model summaries, and analysis metadata.

Examples

```
competing_risk_reg(
  data_competing_risk,
  time = follow_up_months,
  event = outcome,
  event_of_interest = Relapse,
  censor_code = Censored,
  exposures = c(treatment, age, stage)
)
```

---

conditional_logit_reg *Conditional logistic regression for matched case-control studies*

Description

Fits conditional logistic regression models for individually or frequency matched case-control data and returns publication-ready matched odds ratios.

Usage

```
conditional_logit_reg(
  data,
  outcome,
  exposures,
  matched_set,
  adjust_for = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  model_stats = FALSE,
  format = c("flextable", "gt"),
  theme = "minimal",
  show_ref = TRUE
)
```

Arguments

`data`	A data frame containing the outcome, predictors, and matched-set identifier.
`outcome`	Binary case-control outcome. The second factor/character level is treated as the case; numeric outcomes must use 0/1 or 1/2 coding.
`exposures`	Predictor names. Quoted and bare names are accepted.
`matched_set`	Variable identifying each matched case-control set.
`adjust_for`	Optional covariates. In exposure-by-exposure mode, each exposure is fitted with these covariates.
`multivariable`	Logical. If `TRUE`, one model includes all exposures and optional adjustment variables.

multivariate	Optional alias for multivariable.
model_stats	Logical. If TRUE, return model N, case and control counts, matched-set count, AIC, and log-likelihood.
format	Output format: "flextable" (default) or "gt".
theme	Table theme preset.
show_ref	Logical. Display reference rows for categorical predictors.

Details

Matching is handled by conditioning on matched_set; no odds ratio is estimated for that variable. Matched sets without both a case and a control are not informative for conditional logistic regression and are excluded with a message. This function is intended for matched case-control studies, not complex survey analysis.

Value

A gtregression object containing the publication table, underlying estimates, fitted clogit models, summaries, model statistics, and matching metadata.

Examples

```
matched_data <- data.frame(
  set = rep(1:8, each = 3),
  case = rep(c(1, 0, 0), 8),
  exposed = factor(rep(c("Yes", "No", "No"), 8),
                   levels = c("No", "Yes")),
  age = rep(40:47, each = 3) + rep(c(0, -1, 1), 8)
)

conditional_logit_reg(
  matched_data,
  outcome = case,
  exposures = exposed,
  matched_set = set
)
```

---

cox_reg *Cox proportional hazards regression*

Description

Fit Cox proportional hazards models and report hazard ratios.

Usage

```
cox_reg(
  data,
  time,
  event,
  exposures,
  adjust_for = NULL,
  stratifier = NULL,
  interaction = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  format = c("flextable", "gt"),
  theme = c("minimal"),
  show_sample = "both",
  model_stats = FALSE,
  show_ref = TRUE,
  baseline_strata = NULL,
  time_start = NULL,
  cluster = NULL,
  weights = NULL,
  robust = NULL,
  ties = c("efron", "breslow", "exact"),
  control = NULL
)
```

Arguments

`data`	A `data.frame` containing survival time, event status, and exposure variables.
`time`	Survival follow-up time. Quoted and bare names are accepted.
`event`	Event indicator. Quoted and bare names are accepted. Numeric `0/1`, numeric `1/2`, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
`exposures`	Character vector of exposure variable names. Quoted names are recommended in scripts, and bare names are also accepted.
`adjust_for`	Optional character vector of adjustment variables. When supplied, one adjusted Cox model is fitted per exposure.
`stratifier`	Optional single stratifying variable. When supplied, stratum-specific Cox tables are produced using the same crude, adjusted, or multivariable workflow requested by the other arguments. The stratifier cannot also be used as the time, event, exposure, adjustment, or interaction variable.
`interaction`	Optional character scalar specifying one interaction term using standard formula syntax, e.g. `"trt*prior"`. Quoted and bare interaction syntax are accepted. In exposure-by-exposure mode, supply a single exposure; in `multivariable = TRUE` mode, the interaction is added to the single multivariable model.
`multivariable`	Logical; if `FALSE` (default), the current exposure-by-exposure workflow is used. If `TRUE`, one multivariable Cox model is fitted using all variables in `exposures`, and all exposure coefficients are reported.

multivariate — Optional logical alias for multivariable. This is accepted for convenience; multivariable is used internally.

format — Output table format; one of "flextable" (default) or "gt".

theme — Table styling preset.

show_sample — For stratified Cox tables (stratifier supplied), controls which sample-size columns are shown in the publication table. One of "both" (default; N and events), "n", "events", or "none". Model statistics, when requested, still retain both N and event counts. This argument is ignored when stratifier is not supplied: non-stratified tables always show the model N column and report the number of events in a footnote.

model_stats — Logical; if TRUE, extract model-fit statistics including AIC, BIC, log-likelihood, concordance, number of events, and N.

show_ref — Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

baseline_strata — Optional character vector of categorical variables used to stratify the baseline hazard within one Cox model. Common hazard ratios are estimated while each stratum has a separate baseline hazard. No hazard ratio is estimated for these variables. This cannot be combined with stratifier, which fits separate models.

time_start — Optional numeric variable containing entry or interval-start times. When supplied, a counting-process response Surv(time_start, time, event) is fitted, enabling delayed entry and start-stop data.

cluster — Optional variable identifying correlated observations. Cox models use a clustered sandwich variance estimate for this variable.

weights — Optional numeric case-weight variable.

robust — Optional logical passed to survival::coxph(). The default NULL lets coxph() choose its standard behaviour; clustered models use robust variance automatically.

ties — Method for handling tied event times: "efron" (default), "breslow", or "exact".

control — Optional named list of arguments for survival::coxph.control(), or a completed control list returned by that function.

Details

By default, cox_reg() keeps the exposure-by-exposure workflow: without adjust_for, one crude Cox model is fitted per exposure; with adjust_for, one adjusted Cox model is fitted per exposure and only the exposure estimate is reported. This is useful for screening or for reporting several adjusted exposure effects.

With multivariable = TRUE, all variables in exposures are included in a single Cox model and all coefficients are reported. This mirrors the multivariable workflow used by multi_reg(). The adjust_for argument is not used in this mode; include every variable that belongs in the model inside exposures. Since these estimates are adjusted for the other variables in the same model, the table reports Adjusted HR (95% CI).

Interaction terms specified via `interaction` are included using standard formula expansion (for example, `trt*prior`). Interaction effects are displayed as additional rows beneath the corresponding exposure.

`baseline_strata` fits one Cox model with separate baseline hazards for the observed combinations of the selected categorical variables. It reports common hazard ratios for the predictors and no coefficient for the strata. This differs from `stratifier`, which fits separate models and reports stratum-specific results.

The proportional hazards assumption should be assessed separately, for example with `check_ph()`.

Stratified Cox tables show both N and event counts by default (`show_sample = "both"`). Use `show_sample = "n"`, `show_sample = "events"`, or `show_sample = "none"` to control the displayed sample columns. All counts are taken from the fitted model within each stratum, so rows dropped for missing values in any model variable (including `time_start`, `cluster`, and `weights`) are not counted. When models within a stratum use different numbers of observations, the footnote reports the range. Rows with a missing `stratifier` value are excluded and counted in a footnote. `show_sample` has no effect without `stratifier`.

Complete-case analysis is used: each model uses the rows that are complete for its own variables. Unused factor levels are dropped for each fitted model, so the reference category shown as `"Ref."` is the first level present in the data used by that model. Ordered factors are analysed with treatment (category-versus-reference) contrasts. Data in which every observation is an event are accepted.

If exposure variables have a `"label"` attribute, for example from `labelled::var_label()`, those labels are used automatically in the displayed table.

Value

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

table A `flextable` or `gt_tbl`.

table_body Data frame of hazard ratios and confidence intervals.

table_display Data frame used to render the publication table.

models List of fitted `coxph` models.

model_summaries Summary output for the fitted models.

model_stats Model-fit statistics when `model_stats = TRUE`; otherwise `NULL`.

variable_labels Named character vector of display labels.

footnotes Character vector used by the rendered table. Adjustment notes use the same display labels as the table.

time,event,approach,format,source,adjust_for,baseline_strata, exposures,interaction Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))
lung_data$prior <- factor(lung_data$prior, levels = c(0, 10),
                          labels = c("No", "Yes"))
```

```
cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c("trt", "celltype", "karno", "age")
)

cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, celltype, prior),
  adjust_for = c(age, karno)
)

# Interaction in an adjusted exposure model
cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = trt,
  adjust_for = c(age, karno),
  interaction = trt*prior
)

cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, celltype, prior, age, karno),
  multivariable = TRUE
)

# One model with separate baseline hazards by cell type
cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, age, karno),
  baseline_strata = celltype,
  multivariable = TRUE
)

# multivariate is accepted as an alias
cox_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, age, karno),
  multivariate = TRUE
)
```

cuminc_plot *Cumulative incidence plot for competing risks*

Description

Estimate and plot the cumulative incidence of one event in the presence of competing events, with optional group comparison using Gray's test.

Usage

```
cuminc_plot(
  data,
  time,
  event,
  event_of_interest,
  censor_code = 0,
  by = NULL,
  conf.int = TRUE,
  p_value = TRUE,
  p_value_position = NULL,
  break_time_by = NULL,
  xlim = NULL,
  ylim = NULL,
  xlab = "Time",
  ylab = "Cumulative incidence",
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  legend_title = NULL,
  legend_position = NULL,
  palette = NULL,
  y_percent = TRUE,
  theme = "classic",
  grid = FALSE,
  base_size = 13
)
```

Arguments

data	A data frame containing follow-up time and event status.
time	Follow-up time. Quoted and bare column names are accepted.
event	Event-status variable with one censoring code and distinct codes for each event type. Quoted and bare column names are accepted.
event_of_interest	Code or label for the event to plot. This must be an observed value of 'event' and must differ from 'censor_code'.

censor_code	Code or label identifying censored observations. Defaults to '0'.
by	Optional single grouping variable. When supplied, separate curves are drawn and Gray's test compares groups for the event of interest. Groups are ordered by factor level (as in km_plot()).
conf.int	Logical; show pointwise 95 percent confidence intervals.
p_value	Logical; display Gray's test p-value when 'by' is supplied.
p_value_position	Optional numeric vector of length two giving the x and y coordinates of the p-value annotation.
break_time_by	Optional positive interval for x-axis breaks.
xlim, ylim	Optional numeric vectors of length two, given as c(lower, upper), defining plot limits. The plot is zoomed to these limits; no data are removed.
xlab, ylab	Axis labels.
title, subtitle, caption	Optional plot text.
legend_title	Optional legend title. The display label of 'by' is used by default.
legend_position	One of '"bottom"', '"top"', '"right"', '"left"', or '"none"'.
palette	Optional character vector of curve colors.
y_percent	Logical; display cumulative incidence as percentages.
theme	One of '"classic"', '"minimal"', '"bw"', or '"light"'.
grid	Logical; show major grid lines.
base_size	Base font size.

Details

Cumulative incidence estimates the absolute probability of the selected event over time while correctly retaining competing events in the risk process. Treating competing events as ordinary censoring with a Kaplan-Meier curve generally overestimates event probability.

Gray's test compares cumulative-incidence functions between groups. It is a group comparison and not an adjusted effect estimate; use 'competing_risk_reg()' when a subdistribution hazard ratio is required.

Value

A 'ggplot2' object with attributes 'fit', 'plot_data', 'gray_test', 'event_of_interest', and 'censor_code'.

Examples

```
cuminc_plot(
  data_competing_risk,
  time = follow_up_months,
  event = outcome,
  event_of_interest = Relapse,
  censor_code = Censored,
```

```
  by = treatment
)
```

`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_competing_risk` *Synthetic Competing-Risk Oncology Dataset*

Description

A synthetic clinical follow-up dataset for teaching cumulative incidence and competing-risk regression. The primary event is cancer relapse, while death before relapse is a competing event because it prevents a later relapse from being observed.

Usage

```
data_competing_risk
```

Format

A data frame with 500 observations and 8 variables:

patient_id Synthetic patient identifier.

follow_up_months Observed follow-up time in months.

outcome First observed outcome: `Censored`, `Relapse`, or `Death without relapse`.

treatment Randomized treatment group: `Standard` or `Intervention`.

age Age at baseline in years.

sex Recorded sex: `Female` or `Male`.

stage Disease stage at baseline: `I`, `II`, or `III`.

performance_status Baseline performance score from 0 to 2, with higher values indicating greater functional limitation.

Details

These data are entirely synthetic and do not correspond to real people. Latent times to relapse and death were generated from separate exponential event processes and compared with an independently generated censoring time. The intervention lowers the simulated relapse rate, while age, disease stage, and performance status influence one or both event processes. The dataset is intended for teaching workflow and interpretation, not for clinical inference or benchmarking estimators.

Source

Generated deterministically for the gtregression package.

`data_diabetes_mediation`

Diabetes Mediation Teaching Dataset

Description

A health-related teaching dataset prepared from synthetic diabetes data. It is designed for the practical question: could part of the obesity and diabetes relationship operate through plasma glucose, after adjustment for basic clinical covariates?

Usage

```
data_diabetes_mediation
```

Format

A data frame with variables:

diabetes Diabetes status (`No`, `Yes`)

obesity Obesity status based on BMI >= 30 (`No`, `Yes`)

glucose Plasma glucose concentration

bmi Body mass index

age Age in years

blood_pressure Diastolic blood pressure

pregnancies Number of pregnancies

diabetes_pedigree Diabetes pedigree function

Details

This dataset is intended for practice and teaching. Mediation estimates from observational data should be interpreted causally only when the temporal ordering and no-unmeasured-confounding assumptions are justified by study design, DAGs, and subject-matter knowledge.

Source

Derived from `data_SynthDiabetes`, which is based on `mlbench::SynthDiabetes2` (**mlbench** version 2.1-11 or later).

data_endometrial *Endometrial Cancer Histology Grade Data*

Description

A classic endometrial cancer dataset used to demonstrate separation in logistic regression. The outcome is high histology grade. Neovascularization is completely absent among low-grade cases in this dataset, making it useful for teaching Firth penalized logistic regression.

Usage

```
data_endometrial
```

Format

A data frame with 79 observations and 4 variables:

NV Neovascularization status (0 = absent, 1 = present)

PI Pulsatility index of the uterine artery

EH Endometrium height

HG Histology grade (0 = low grade, 1 = high grade)

Source

brglm2 package. The packaged dataset was sourced from https://users.stat.ufl.edu/~aa/glm/data/, the data repository used in Agresti (2015). Originally analyzed in Heinze and Schemper (2002).

References

Agresti A (2015). *Foundations of Linear and Generalized Linear Models*. Wiley.

Heinze G, Schemper M (2002). A solution to the problem of separation in logistic regression. *Statistics in Medicine*, 21, 2409-2419. doi:10.1002/sim.1047

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_SynthDiabetes *Synthetic Diabetes Dataset*

Description

A synthetic replacement for the former diabetes dataset from the **mlbench** package. It has the same structure and number of observations but does not represent real people. Useful for demonstrating regression approaches for binary outcomes.

Usage

```
data_SynthDiabetes
```

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

`mlbench::SynthDiabetes2`, available in **mlbench** version 2.1-11 or later.

`descriptive_table` *Descriptive summary table*

Description

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)

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("flextable", "gt"),
  theme = c("minimal")
)
```

Arguments

`data`	data.frame
`exposures`	Character vector of variables to summarise. Quoted names are recommended in scripts, and bare names are also accepted.
`by`	Optional single grouping variable. Quoted and bare names are accepted. If it is also listed in 'exposures', it is left out of the summarised variables, with a message.
`percent`	"column" (default) or "row"; aliases like "col"/"rows" accepted
`digits`	integer; decimals for % and continuous stats (default 1)
`show_missing`	"ifany" (default) or "no". With "ifany", a "(Missing)" row (n and % of the column N) is added under every variable that has missing values, categorical or continuous.
`show_dichotomous`	"all_levels" (default) or "single_row"
`show_overall`	"no" (default), "first", or "last"

statistic	Optional summary type for numeric variables. Use a single value such as "mean" or mean for all numeric variables, or a named vector such as c(age = mean, ftv = categorical). Supported values are "mean", "median", "mode", "count", and "categorical". Use "categorical" for numeric ordinal variables that should be shown as n (%).
value	optional named list for single-row binaries (e.g., list(sex="Female")); formula entries like list(sex ~ "Female") are also accepted. The value must be one of the observed levels of the variable.
format	"flextable" (default) or "gt"
theme	preset or primitives

Details

If variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table. Internal matching still uses the original column names.

Variable types: factor, character and logical variables are shown as n (%) per level (logical: FALSE/TRUE); numeric variables with exactly two distinct values are shown as categories; numeric variables with a single distinct value are shown as one category (n, 100%) unless a numeric statistic is requested for them; other numeric variables are continuous. Levels of numeric variables shown as categories are sorted by value. Date and date-time variables are not summarised: they are excluded and named in a message.

Value

A list with class c("gtregression", "descriptive_table", ...) containing:

table A gt_tbl or flextable.

table_display Display-ready data.

table_body Long audit data with variable, level, and type.

variable_labels Named character vector of display labels used for variables.

metadata Additional metadata fields.

---

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, verbose = FALSE, format = c("flextable", "gt", "tibble"))
```

Arguments

data	A data frame.
verbose	Logical; if TRUE, print the summary and interpretation notes. The tibble is returned invisibly only when printed by the console.
format	Output format. One of "flextable" (default), "gt", or "tibble". Use format = "tibble" for pipeline-friendly raw output.

Value

A tibble, gt_tbl, or flextable, depending on format. The tibble has columns: Variable, Type, Missing (Levels, Compatibility, and Hint.

Examples

```
dissect(data_birthwt, format = "tibble")
dissect(data_birthwt, format = "gt")

# Print notes that help interpret the data structure
dissect(data_birthwt, verbose = TRUE, format = "tibble")
```

---

forest_df *Build a compatible data frame for forest plots*

Description

Creates the tabular input used by forest_reg() from uni_reg(), multi_reg(), and optionally descriptive_table() objects. This function is useful when users want to inspect, edit, or reuse the exact data that will be passed to the forest plot. Stratified regression objects are also supported when supplied one at a time; the returned data frame keeps the characteristic rows once and places strata side by side as separate effect columns for forest_reg().

Usage

```
forest_df(uni = NULL, multi = NULL, desc = NULL, digits = 2)
```

Arguments

uni	A gtregression regression object, usually the crude results (for example uni_reg(), cox_reg(), conditional_logit_reg() or ordinal_reg()). If multi is supplied without uni, the adjusted object is plotted as a single effect column.
multi	Optional adjusted object of the same model family, for example multi_reg() or a model fitted with multivariable = TRUE. Exposure-specific adjusted results (adjust_for) and the single multivariable model can also be shown together; the second column is then labelled "Mutually adjusted".
desc	Optional descriptive table object from descriptive_table().
digits	Number of digits used when formatting confidence intervals.

Details

Effect columns use the same header as the source tables, for example "Matched OR (95% CI)" for `conditional_logit_reg()` and "Common OR (95% CI)" for `ordinal_reg()`, and the estimates and intervals are taken unchanged from each result's `table_body`.

Value

A data frame with display columns, formatted effect-size columns, standard-error columns, and plotting attributes used by `forest_reg()` to draw confidence intervals.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

uni_or <- uni_reg(
  birthwt_data,
  outcome = "low",
  exposures = c("age", "lwt", "smoke", "ht"),
  approach = "logit"
)
multi_or <- multi_reg(
  birthwt_data,
  outcome = "low",
  exposures = c("smoke", "ht"),
  adjust_for = c("age", "lwt"),
  approach = "logit"
)

forest_data <- forest_df(uni_or, multi_or)
head(forest_data)
```

---

forest_reg *Draw a forest table from regression outputs*

Description

Wrapper around `forestploter::forest()` that works directly with `forest_df()` output or with `gtregression` regression objects. It can show descriptive columns and one or more model effect columns in a table-style forest plot while leaving axis and interval drawing to **forestploter**.

Usage

```
forest_reg(
  df = NULL,
  uni = NULL,
  multi = NULL,
  desc = NULL,
  theme = NULL,
  ci_col_width = 20,
  side = c("right", "left"),
  quiet = TRUE,
  effects = NULL,
  ticks_at = NULL,
  ticks_digits = NULL,
  xlim = NULL,
  style_strata = TRUE,
  strata_fill = "#EAF2F1",
  ...
)
```

Arguments

df	Output of `forest_df()`. If `NULL`, it is built from `uni`, `multi`, and `desc`.
uni, multi, desc	Optional `gtregression` objects to pass through to `forest_df()`.
theme	Optional `forestploter::forest_style()` or `forestploter::forest_theme()`. If `NULL`, `forestploter` defaults are used. You may pass colors and styling either here or through `....`
ci_col_width	Numeric value, or one value per effect column, controlling the blank spacer width used by `forestploter` for the confidence-interval plot column(s). Values greater than 1 are interpreted as approximate character counts. Values between 0 and 1 are accepted for backward compatibility and converted to character counts.
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. The `Characteristic` column and descriptive columns remain on the left.
quiet	Logical. Suppress forestploter warnings. Default = 'TRUE'.
effects	Deprecated and ignored. This argument never affected the rendered forest plot and is retained temporarily for backward compatibility.
ticks_at	Optional numeric vector, or list of numeric vectors for multiple effect columns, specifying x-axis tick positions. If `NULL`, `forestploter::forest()` chooses the default ticks.
ticks_digits	Optional number of digits for x-axis tick labels.
xlim	Optional numeric vector of length 2, or list of length-2 numeric vectors for multiple effect columns, specifying x-axis limits. If `NULL`, `forestploter::forest()` chooses the default limits.
style_strata	Logical. Retained for older vertically grouped stratified forest data. Current `forest_df()` stratified output places strata side by side, so no stratum header rows are styled.

strata_fill	Character. Fill color used for older styled stratum header rows.
...	Passed to forestploter::forest(). Common options include title and footnote.

Value

A gtregression_forest object with elements:

plot The forest plot object.

data The plotting data sent to forestploter::forest().

input_data The original forest_df() data, including standard-error helper columns.

meta Model metadata, including reference line and x-axis transformation.

Examples

```
birthwt_data <- data_birthwt |>
  transform(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1),
                 labels = c("Normal BW", "Low BW"))
  )

uni_or <- uni_reg(
  birthwt_data,
  outcome = "low",
  exposures = c("age", "lwt", "smoke", "ht"),
  approach = "logit"
)
multi_or <- multi_reg(
  birthwt_data,
  outcome = "low",
  exposures = c("smoke", "ht"),
  adjust_for = c("age", "lwt"),
  approach = "logit"
)

forest_reg(uni = uni_or, multi = multi_or)

# If axis labels overlap, set x-axis limits and tick marks.
forest_reg(
  uni = uni_or,
  multi = multi_or,
  xlim = list(c(0.25, 8), c(0.25, 12)),
  ticks_at = list(c(0.5, 1, 2, 4, 8), c(0.5, 1, 2, 4, 8))
)

# If the forest plot panel is too narrow or too wide, tune ci_col_width.
forest_reg(
  uni = uni_or,
  multi = multi_or,
```

```
  ci_col_width = c(18, 22)
)
```

`gtregression_addin` *Launch the gtregression app from the RStudio Addins menu*

Description

The RStudio Addin (*Addins* > *Regression without code (gtregression app)*) opens the gtregression app in an RStudio dialog window. It is the same application as `gtregression_app()`, which opens it in your web browser.

Usage

```
gtregression_addin()
```

Details

There is no separate Addin interface: the interface, analyses, generated R code and access to data frames in your global environment are identical to `gtregression_app()`. Only the container differs (a 1200 x 800 RStudio dialog instead of a browser window). The **shiny** and **rstudioapi** packages are optional and are checked only when this function is called. Selected data frames are copied into the app as read-only snapshots; the source objects are not modified.

Value

Invisibly returns the result of `shiny::runGadget()`.

See Also

`gtregression_app`

Examples

```
if (
  interactive() &&
    requireNamespace("rstudioapi", quietly = TRUE) &&
    rstudioapi::isAvailable()
) {
  gtregression_addin()
}
```

`gtregression_app` *Launch the gtregression app*

Description

Open the gtregression app, the graphical interface for **gtregression**: data import and preparation, descriptive tables, regression tables, survival analysis, diagnostics, causal mediation, plots, exports and reproducible R code.

Usage

```
gtregression_app(..., launch.browser = NULL)
```

Arguments

`...`	Additional arguments passed to `shiny::runApp()`, for example `port`.
`launch.browser`	Passed to `shiny::runApp()`. The default opens the app in your web browser in interactive sessions. Supply a function such as `rstudioapi::viewer` to choose another viewer.

Details

`gtregression_app()` opens the gtregression app in your web browser. The RStudio Addin (*Addins > Regression without code (gtregression app)*) opens exactly the same application in an RStudio dialog window through `gtregression_addin()`; only the window differs.

The app runs in your local R session, so data frames in your global environment can be chosen from the Data workspace (*R Environment*). Selected data are copied into the app as read-only snapshots; the source objects are not modified. Files can also be imported, and example data are included.

The app is intentionally kept out of the core package startup path. Shiny and other interface packages are suggested dependencies and are loaded only when `gtregression_app()` is called.

The Advanced tab includes a guided candidate-model builder. Users can name two to six models, choose model-specific exposures and adjustment variables, optionally add an interaction, and track a primary exposure. The app fits compatible `multi_reg()`, `cox_reg()`, or `surv_reg()` objects before passing them to `compare_models()`. Generated code records every fitting call and the final comparison for reproducible use outside the app.

Value

Invisibly returns the result of `shiny::runApp()`.

See Also

`gtregression_addin`, `compare_models()`, `multi_reg()`, `cox_reg()`, `surv_reg()`

Examples

```
if (interactive()) {
  gtregression_app()
}
```

`identify_confounder` *Identify confounders and effect modifiers*

Description

Review whether one or more candidate variables may act as confounders or effect modifiers for one or more exposures.

Usage

```
identify_confounder(
  data,
  outcome = NULL,
  exposure,
  potential_confounder,
  approach = "logit",
  time = NULL,
  event = NULL,
  distribution = "weibull",
  method = "change",
  threshold = 10,
  emm_threshold = 10,
  emm_test = c("interaction", "both", "estimate"),
  interaction_alpha = 0.05,
  format = c("flextable", "gt"),
  theme = c("minimal")
)
```

Arguments

`data`	A data frame.
`outcome`	Outcome variable name. Quoted and bare names are accepted. Required for ordinary regression approaches. Leave unset for Cox and parametric survival approaches and supply `time` and `event` instead.
`exposure`	Exposure variable name(s). Can be a character scalar or vector. Quoted names are recommended in scripts, and bare names are also accepted.
`potential_confounder`	Candidate confounder/effect-modifier variable name(s). Can be a character scalar or vector. Quoted names are recommended in scripts, and bare names are also accepted.

approach	Regression approach. One of "logit", "logbinomial", "poisson", "robpoisson", "linear", "negbin", "cox", or "survreg".
time	Survival time variable name for approach = "cox" or approach = "survreg". Quoted and bare names are accepted.
event	Event indicator variable name for survival approaches. Values may be coded as 0/1, 1/2, logical, or a two-level factor/character variable.
distribution	Parametric survival distribution for approach = "survreg". One of "weibull", "exponential", "lognormal", or "loglogistic".
method	Confounding assessment method. One of "change", "mh", or "both". "change" compares crude and adjusted model estimates. "mh" compares crude and Mantel-Haenszel pooled estimates and is available for binary outcome, binary exposure, and categorical strata. "both" uses either method.
threshold	Percent change threshold for confounding assessment.
emm_threshold	Threshold for relative spread in stratum-specific estimates when using estimate-based effect-modification screening.
emm_test	One of "interaction", "both", or "estimate".
interaction_alpha	Alpha threshold for interaction p-values.
format	Output table format. One of "flextable" (default) or "gt".
theme	Table theme preset or primitives.

Details

The function first assesses possible effect modification using stratum-specific estimates and/or an interaction test. If no important effect modification is detected, it then assesses confounding using the selected method.

This is a screening aid for viewing and organising results. Confounding and effect modification should be interpreted using subject-matter knowledge, study design, and causal diagrams such as DAGs. Automated change-in-estimate and interaction checks should not be used as the sole basis for model adjustment.

For each exposure-candidate pair, the interaction screen fits `outcome ~ exposure + potential_confounder + exposure:potential_confounder`. It does not adjust for other covariates. Consequently, the reported interaction p-value is a candidate-only screening result, not a final adjusted interaction analysis. Use `interaction_models(covariates = ...)` when testing a planned, covariate-adjusted interaction. The interaction p-value is a likelihood-ratio test of the models with and without the interaction for every approach except "robpoisson", where a robust (sandwich HC0) Wald test is used because a likelihood-ratio test is not valid.

Percentage changes ("change-in-estimate") are computed on the displayed ratio scale (OR, RR, IRR, HR, time ratio) as `100 * |adjusted - crude| / crude`; for linear models the coefficients are compared directly.

A numeric candidate coded as at most 10 distinct whole numbers is treated as categorical in every step (adjusted model, interaction model, stratum-specific estimates and Mantel-Haenszel). Other numeric candidates are adjusted for as continuous and have no stratum-specific screen. The estimate-spread screen only uses strata with at least 20 observations and a stable estimate (converged model, finite standard error); the `stable` column of `stratum_estimates` shows which strata were used.

Use this function when you want to screen one or more candidate variables and organise crude, adjusted, Mantel-Haenszel, and effect-modification signals in one place. For a focused comparison of models with and without a planned exposure-by-modifier interaction term, use `interaction_models()`.

Value

If a single exposure-candidate pair is supplied, returns a detailed list.

If multiple combinations are supplied, returns a list with:

summary A tibble with one row per exposure-candidate combination.

details A named list of detailed results for each combination.

See Also

`interaction_models()` for focused model comparison of a planned interaction term.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW")),
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    race = factor(race, levels = c(1, 2, 3),
                  labels = c("White", "Black", "Other"))
  )

identify_confounder(
  data = birthwt_data,
  outcome = low,
  exposure = smoke,
  potential_confounder = race,
  approach = logit
)

lung_data <- data_lungcancer |>
  dplyr::mutate(
    trt = factor(trt, levels = c(1, 2),
                 labels = c("Standard treatment", "Test treatment")),
    prior = factor(prior, levels = c(0, 10), labels = c("No", "Yes"))
  )

identify_confounder(
  data = lung_data,
  time = time,
  event = status,
  exposure = trt,
  potential_confounder = prior,
  approach = cox
)
```

`import_data` *Import an analysis dataset*

Description

Read common tabular data formats through one consistent interface. Optional packages are required only for the formats that use them.

Usage

```
import_data(
  file,
  format = NULL,
  sheet = NULL,
  header = TRUE,
  sep = NULL,
  na = c("", "NA"),
  encoding = "UTF-8",
  name_repair = c("check_unique", "unique", "minimal"),
  ...
)
```

Arguments

`file`	Path to the data file.
`format`	Optional file format when it cannot be inferred from ‘file‘. Supply an extension without the leading dot, such as ‘"csv"‘ or ‘"dta"‘.
`sheet`	Excel sheet name or number. Used only for ‘.xls‘ and ‘.xlsx‘ files. The first sheet is used by default.
`header`	Logical; whether delimited text files contain column names.
`sep`	Optional delimiter for ‘.csv‘, ‘.tsv‘, and ‘.txt‘ files. Defaults to comma for CSV, tab for TSV, and whitespace for TXT.
`na`	Character vector of strings interpreted as missing values.
`encoding`	Character encoding used for delimited text files. With the default `"UTF-8"`, a file that is not valid UTF-8 (typically a Latin-1 or Windows-1252 export) is read as Latin-1 and a message says so; set `encoding` explicitly to override.
`name_repair`	How to handle column names: ‘"check_unique"‘ preserves names and errors on blank or duplicated names; ‘"unique"‘ repairs them with [make.names()]; ‘"minimal"‘ preserves names without checking.
`...`	Additional arguments passed to the format-specific reader.

Details

Supported extensions are CSV (‘.csv‘), tab-separated or text files (‘.tsv‘, ‘.txt‘), Excel (‘.xls‘, ‘.xlsx‘), R data (‘.rds‘), Stata (‘.dta‘), SPSS (‘.sav‘, ‘.zsav‘, ‘.por‘), and SAS (‘.sas7bdat‘, ‘.xpt‘).

‘readxl‘ is required for Excel files and ‘haven‘ is required for Stata, SPSS, and SAS files. RDS files must contain a data frame or tibble.

Value

A data frame. Tibbles and labelled columns returned by ‘readxl‘ or ‘haven‘ are retained.

Examples

```
csv_file <- tempfile(fileext = ".csv")
utils::write.csv(
  data.frame(outcome = c(0, 1), age = c(45, 61)),
  csv_file,
  row.names = FALSE
)
imported <- import_data(csv_file)
names(imported)
```

---

interaction_models *Compare models with and without an interaction term*

Description

Fits two models, one with and one without an interaction term between an exposure and a potential effect modifier. The models are compared using a likelihood ratio test or Wald test to assess statistical evidence of interaction.

Usage

```
interaction_models(
  data,
  outcome = NULL,
  exposure,
  covariates = NULL,
  effect_modifier,
  approach = "logit",
  time = NULL,
  event = NULL,
  distribution = "weibull",
  test = c("LRT", "Wald"),
  alpha = 0.05,
  verbose = FALSE,
  format = c("flextable", "gt", "tibble")
)
```

Arguments

data	A data frame containing all required variables.
outcome	Outcome variable name. Quoted and bare names are accepted. Required for ordinary regression approaches. Leave unset for Cox and parametric survival approaches and supply time and event instead.
exposure	Main exposure variable name. Quoted and bare names are accepted.
covariates	Optional character vector of adjustment covariates. They are included in both the model without and the model with the interaction term. Quoted names are recommended in scripts, and bare names are also accepted.
effect_modifier	Variable name for the potential effect modifier. Quoted and bare names are accepted.
approach	Regression approach. One of "logit", "logbinomial", "poisson", "robpoisson", "negbin", "linear", "cox", or "survreg".
time	Survival time variable name for approach = "cox" or approach = "survreg". Quoted and bare names are accepted.
event	Event indicator variable name for survival approaches.
distribution	Parametric survival distribution for approach = "survreg". One of "weibull", "exponential", "lognormal", or "loglogistic".
test	Statistical test for model comparison. One of "LRT" or "Wald". For approach = "robpoisson" a likelihood-ratio test is not valid, so the Wald test with the robust (sandwich HC0) variance is always returned and reported as such in $test.
alpha	Significance threshold used to classify the interaction result.
verbose	Logical; if TRUE, prints a short interpretation.
format	Output format for the viewing table. One of "flextable" (default), "gt", or "tibble". Use format = "tibble" to keep only the original list structure.

Details

Use this function when the interaction is planned or clinically/causally motivated and you want a focused model comparison. Mantel-Haenszel estimation is not used here because this function tests an explicit interaction term in a regression model. For broader screening of candidate confounders or effect modifiers, including Mantel-Haenszel-supported checks when appropriate, use identify_confounder().

With covariates = c(age, sex), the two fitted models are outcome ~ exposure + effect_modifier + age + sex and outcome ~ exposure + effect_modifier + age + sex + exposure:effect_modifier. Both models use the same complete-case analysis data, so covariate adjustment is applied consistently to the interaction comparison.

Value

A list with model objects, formulas, p-value, decision, and a one-row summary tibble. When format is "gt" or "flextable", the list also includes table.

See Also

identify_confounder() for broader confounding and effect-modification screening.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW")),
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    race = factor(race, levels = c(1, 2, 3),
                  labels = c("White", "Black", "Other"))
  )

interaction_models(
  data = birthwt_data,
  outcome = low,
  exposure = smoke,
  effect_modifier = race,
  covariates = c(age, lwt),
  approach = logit
)

lung_data <- data_lungcancer |>
  dplyr::mutate(
    trt = factor(trt, levels = c(1, 2),
                 labels = c("Standard treatment", "Test treatment")),
    prior = factor(prior, levels = c(0, 10), labels = c("No", "Yes"))
  )

interaction_models(
  data = lung_data,
  time = time,
  event = status,
  exposure = trt,
  effect_modifier = prior,
  covariates = c(age, karno),
  approach = cox
)
```

---

km_plot *Kaplan-Meier survival plot*

Description

Plot observed survival over time, with optional confidence intervals, censoring marks, a log-rank p-value, and a number-at-risk table.

Usage

```
km_plot(
  data,
  time,
  event,
  by = NULL,
  conf.int = TRUE,
  risk_table = TRUE,
  p_value = TRUE,
  p_value_position = NULL,
  censor = TRUE,
  break_time_by = NULL,
  xlim = NULL,
  ylim = NULL,
  xlab = "Time",
  ylab = "Survival probability",
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  title_size = NULL,
  title_face = "bold",
  legend_title = NULL,
  legend_position = NULL,
  palette = NULL,
  y_percent = TRUE,
  theme = "classic",
  grid = FALSE,
  base_size = 13
)
```

Arguments

`data`	A `data.frame` containing survival time and event status.
`time`	Survival follow-up time. Quoted and bare names are accepted.
`event`	Event indicator. Quoted and bare names are accepted. Numeric `0/1`, numeric `1/2`, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
`by`	Optional grouping variable for separate Kaplan-Meier curves. Quoted and bare names are accepted.
`conf.int`	Logical; if `TRUE`, show confidence limits.
`risk_table`	Logical; if `TRUE`, add a number-at-risk table below the curve.
`p_value`	Logical; if `TRUE`, show the log-rank p-value when `by` is supplied.
`p_value_position`	Optional numeric vector of length 2 giving the `x` and `y` coordinates for the log-rank p-value inside the plotting panel. If `NULL`, a lower-left position is chosen automatically.

censor	Logical; if TRUE, show censoring marks.
break_time_by	Optional numeric interval for x-axis and risk-table time breaks. If NULL, breaks are chosen automatically.
xlim	Optional numeric vector of length 2 specifying x-axis limits.
ylim	Optional numeric vector of length 2 specifying y-axis limits. Values may be supplied on the survival-probability scale (for example c(0.5, 1)) or, when y_percent = TRUE, on the percentage scale (for example c(50, 100)).
xlab, ylab	Axis labels.
title	Optional plot title.
subtitle	Optional plot subtitle.
caption	Optional plot caption.
title_size	Optional numeric title font size. If NULL, ggplot2's theme default is used.
title_face	Font face for the title. One of "plain", "bold", "italic", or "bold.italic". Quoted and bare values are accepted.
legend_title	Optional legend title. If NULL, the labelled by variable name is used.
legend_position	Legend position. One of "bottom", "top", "right", "left", or "none". If NULL, grouped plots use "bottom" and ungrouped plots hide the legend. Quoted and bare values are accepted.
palette	Optional character vector of colors for grouped curves.
y_percent	Logical; if TRUE, display survival probability as percentages. If FALSE, display the raw 0 to 1 probability scale.
theme	Plot theme. One of "classic", "minimal", "bw", "light", or "none". Quoted and bare values are accepted.
grid	Logical; if TRUE, show major grid lines. The default is FALSE for a cleaner publication-style Kaplan-Meier plot.
base_size	Base font size.

Value

A ggplot2 object when risk_table = FALSE; otherwise a patchwork object combining the survival curve and risk table. The returned object has attributes fit, plot_data, risk_table, and logrank_p.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

km_plot(
  data = lung_data,
  time = time,
  event = status
)
```

```
km_plot(
  data = lung_data,
  time = time,
  event = status,
  by = trt,
  break_time_by = 200,
  ylim = c(50, 100),
  title = "A. Treatment group",
  title_size = 11,
  legend_position = "none"
)
```

`km_risk_table` *Kaplan-Meier risk table*

Description

Tabulate the number at risk at selected follow-up times.

Usage

```
km_risk_table(
  data,
  time,
  event,
  by = NULL,
  times,
  digits = 1,
  extend = TRUE,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`data`	A `data.frame` containing survival time and event status.
`time`	Survival follow-up time. Quoted and bare names are accepted.
`event`	Event indicator. Quoted and bare names are accepted. Numeric `0/1`, numeric `1/2`, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
`by`	Optional grouping variable for separate Kaplan-Meier risk tables. Quoted and bare names are accepted.
`times`	Numeric vector of follow-up times for the risk table.
`digits`	Number of digits for displayed follow-up times.

extend	Logical. If TRUE, requested times beyond the observed follow-up range are retained using the last available Kaplan-Meier risk set.
format	Output format. One of "flextable" (default), "gt", or "tibble".
theme	Table styling preset.

Details

The At risk column gives the number still under observation at each requested time. The Events and Censored columns are interval counts up to each requested time point as returned by summary.survfit().

Value

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

table A flextable, gt_tbl, or NULL when format = "tibble".

table_body Tibble with Kaplan-Meier risk table counts.

table_display Display data frame used to render the table.

fit Fitted survfit object.

time,event,by,times,format,source Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

km_risk_table(
  data = lung_data,
  time = time,
  event = status,
  by = trt,
  times = c(0, 90, 180, 365)
)

km_risk_table(
  data = lung_data,
  time = "time",
  event = "status",
  times = c(0, 90, 180),
  format = tibble
)
```

`logrank_test` *Log-rank test for Kaplan-Meier survival curves*

Description

Compare survival curves between groups using the log-rank test.

Usage

```
logrank_test(
  data,
  time,
  event,
  by,
  digits = 2,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`data`	A `data.frame` containing survival time, event status, and grouping variable.
`time`	Survival follow-up time. Quoted and bare names are accepted.
`event`	Event indicator. Quoted and bare names are accepted. Numeric `0/1`, numeric `1/2`, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
`by`	Grouping variable used to compare survival curves. Quoted and bare names are accepted.
`digits`	Number of digits for the chi-square statistic and expected events.
`format`	Output format. One of `"flextable"` (default), `"gt"`, or `"tibble"`.
`theme`	Table styling preset.

Details

The log-rank test compares the observed number of events with the expected number of events in each group under the null hypothesis that the survival curves are the same. It is a group comparison, not an effect-size estimate; use `cox_reg()` when a hazard ratio is needed.

Value

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

`table` A `flextable`, `gt_tbl`, or `NULL` when `format = "tibble"`.

`table_body` Tibble with group-level log-rank components and overall test statistics.

`test` One-row tibble with chi-square statistic, degrees of freedom, and p-value.

`fit` The `survdiff` object.

`time,event,by,format,source` Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

logrank_test(
  data = lung_data,
  time = time,
  event = status,
  by = trt
)

logrank_test(
  data = lung_data,
  time = "time",
  event = "status",
  by = "trt",
  format = tibble
)
```

`mediation_analysis` *Causal mediation analysis*

Description

Estimate direct, indirect, total, and proportion mediated effects from regression models.

Usage

```
mediation_analysis(
  data,
  exposure,
  mediator,
  outcome,
  covariates = NULL,
  mediator_approach = "linear",
  outcome_approach = "linear",
  exposure_value = NULL,
  reference_value = NULL,
  sims = 1000,
  conf_level = 0.95,
  seed = NULL,
  format = c("flextable", "gt"),
  theme = c("minimal")
)
```

Arguments

`data`	A data frame.
`exposure`	Exposure/treatment variable. Quoted and bare names are accepted.
`mediator`	Mediator variable. Quoted and bare names are accepted.
`outcome`	Outcome variable. Quoted and bare names are accepted.
`covariates`	Optional character vector of covariate names. Quoted names are recommended in scripts, and bare names are also accepted.
`mediator_approach`	Mediator model. Currently `"linear"`.
`outcome_approach`	Outcome model. One of `"linear"` or `"logit"`. Logistic mediation effects are reported on the predicted probability-difference scale.
`exposure_value`	Exposure value used as the treatment level. If `NULL`, the second factor level, value 1 for 0/1 variables, or the 75th percentile for continuous exposures is used.
`reference_value`	Exposure value used as the reference level. If `NULL`, the first factor level, value 0 for 0/1 variables, or the 25th percentile for continuous exposures is used.
`sims`	Number of non-parametric bootstrap replicates used for confidence intervals.
`conf_level`	Confidence level for intervals.
`seed`	Optional random seed for reproducible bootstrap intervals.
`format`	One of `"flextable"` (default) or `"gt"`.
`theme`	Table theme: a preset (`"minimal"`, `"clinical"`, `"striped"`, `"shaded"`, `"jama"`) or theme primitives.

Details

This function is for planned mediation questions, not automatic causal discovery. Causal interpretation requires the usual mediation assumptions, including no unmeasured exposure-outcome, exposure-mediator, or mediator-outcome confounding, correct temporal order, and suitable model specification. Use a directed acyclic graph (DAG) and subject-matter knowledge before interpreting the estimates causally.

For `outcome_approach = "linear"`, effects are mean differences. For `outcome_approach = "logit"`, effects are predicted probability differences, not odds ratios. With a logit outcome the effects are evaluated with the mediator set to its predicted mean under each exposure value (the mean is plugged into the outcome model). Because the logistic function is non-linear this is an approximation to the effects obtained by averaging over the mediator distribution; the table footnote states this.

Bootstrap replicates in which a model cannot be fitted are excluded from the percentile intervals and p-values; a warning and the table footnote report how many failed (also stored as `boot_failed`). The interval column is labelled with `conf_level`.

Value

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

table A formatted `flextable` or `gt_tbl`.

table_body Data frame of mediation effect estimates.

table_display Formatted data frame used to build the table.

models Fitted mediator and outcome models.

boot Bootstrap replicate estimates.

values Reference and exposure values used.

variable_labels Named character vector of display labels.

call Matched function call.

Examples

```
med <- mediation_analysis(
  data = data_diabetes_mediation,
  exposure = obesity,
  mediator = glucose,
  outcome = diabetes,
  covariates = c(age, blood_pressure, pregnancies, diabetes_pedigree),
  outcome_approach = logit,
  sims = 50,
  seed = 123
)

med$table
med$table_body

# HTML-first output
mediation_analysis(
  data = data_diabetes_mediation,
  exposure = obesity,
  mediator = glucose,
  outcome = diabetes,
  covariates = c(age, blood_pressure, pregnancies, diabetes_pedigree),
  outcome_approach = logit,
  format = gt,
  sims = 50,
  seed = 123
)$table
```

---

merge_tables *Merge two or more tables side by side*

Description

Merge two or more tables side by side

Usage

```
merge_tables(
  ...,
  spanners = NULL,
  theme = "minimal",
  format = c("flextable", "gt")
)
```

Arguments

`...`	Two or more `gtregression` objects containing `$table_display`.
`spanners`	Character vector of spanner labels, one per table. If `NULL`, defaults to `"Table 1"`, `"Table 2"`, etc.
`theme`	Merge theme preset or vector of primitives.
`format`	Output table format. One of `"flextable"` (default) or `"gt"`. The merged display is rebuilt in this format, independently of the formats used by the input tables.

Details

Binary variables should use the same row display across all input tables. For a descriptive, univariable, and multivariable merge, the clearest publication layout is usually `show_dichotomous = "all_levels"` in `descriptive_table()` and `show_ref = TRUE` in each regression function. Mixing these settings can create additional binary rows; in that situation `merge_tables()` issues a warning with the compatible settings.

Footnotes created by each input table are retained unchanged. Exact duplicate notes are shown once in the merged table.

Tables of the same type can be merged directly. This is useful for sensitivity analyses such as complete-case, best-case, and worst-case models. Supply one table per scenario and use `spanners` to identify each analysis; rows are aligned by their underlying variable and level.

Value

A merged table object of class `c("gtregression", "merged_table", ...)`.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    race = factor(race, levels = c(1, 2, 3),
                  labels = c("White", "Black", "Other")),
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

uni_tbl <- uni_reg(
  data = birthwt_data,
```

```
  outcome = "low",
  exposures = c("age", "lwt", "smoke", "ht"),
  approach = "logit"
)

multi_tbl <- multi_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("smoke", "ht"),
  adjust_for = c("age", "lwt"),
  approach = "logit"
)

merge_tables(
  uni_tbl,
  multi_tbl,
  spanners = c("Univariable", "Adjusted")
)

merge_tables(
  multi_tbl,
  multi_tbl,
  spanners = c("Primary analysis", "Sensitivity analysis")
)
```

`modify_table` *Modify table labels, headers, captions and notes*

Description

Works with objects created by this package (class `"gtregression"`): `uni_reg()`, `multi_reg()`, `cox_reg()`, `surv_reg()`, stratified regression outputs, `descriptive_table()`, and `merge_tables()`. No **gtsummary** dependency or fallback.

Usage

```
modify_table(
  gt_table,
  variable_labels = NULL,
  level_labels = NULL,
  header_labels = NULL,
  caption = NULL,
  bold_labels = TRUE,
  bold_levels = FALSE,
  italic_labels = FALSE,
  italic_levels = FALSE,
  remove_N = FALSE,
  remove_N_obs = FALSE,
  remove_abbreviations = FALSE,
```

```
  remove_adjustment_note = FALSE,
  caveat = NULL
)
```

Arguments

gt_table	Table object produced by this package (must contain $table_display).
variable_labels	Named character vector, for example c(old_var = "New label", ...).
level_labels	Named list for factor levels: list(var1 = c(old = "New", ...), var2 = c(...)).
header_labels	Named character vector to rename visible headers, e.g. c("OR (95% CI)" = "Crude OR", "p-value" = "P"). Common aliases such as estimate, p.value, and N are also accepted.
caption	Optional caption/title.
bold_labels	Logical; bold variable (header) rows in the body. Defaults to TRUE to preserve the package table hierarchy.
bold_levels	Logical; bold factor level rows in the body.
italic_labels	Logical; italicize variable (header) rows in the body.
italic_levels	Logical; italicize factor level rows in the body.
remove_N	Logical; if TRUE, drops displayed N columns from univariable and stratified package tables. For stratified survival outputs, event columns are retained unless the original table was created with show_sample = "none". Footnotes that only explain the removed column are dropped as well; the complete-observation footnote (N = ... complete observations included in the model.) is kept. Use remove_N_obs to remove it.
remove_N_obs	Logical; if TRUE, suppresses the complete-observation (complete-case) footnote, in every wording used by the package tables.
remove_abbreviations	Logical; if TRUE, removes the Abbreviations footnote line.
remove_adjustment_note	Logical; if TRUE, removes the automatic Adjusted for ... footnote. Use caveat to add customised wording.
caveat	Optional extra footnote.

Details

For risk_difference() tables only the generic operations are available: caption, caveat, variable_labels (outcome, exposure and adjustment variables), level_labels (the two groups and the event), header_labels and remove_abbreviations. The other arguments are refused with an error and nothing is changed.

Value

The modified table object (same class as input).

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

tbl <- uni_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("age", "smoke", "ht"),
  approach = "logit",
  format = "gt"
)

modify_table(
  tbl,
  variable_labels = c(age = "Maternal age", smoke = "Smoking"),
  level_labels = list(smoke = c(Yes = "Smoker")),
  header_labels = c(estimate = "Crude OR", p.value = "P"),
  caption = "Univariable regression for low birth weight"
)$table
```

---

multilevel_reg *Random-intercept multilevel regression*

Description

Fit a linear or logistic mixed-effects model with one grouping variable and return publication-ready fixed-effect estimates.

Usage

```
multilevel_reg(
  data,
  outcome,
  exposures,
  group,
  adjust_for = NULL,
  approach = c("linear", "logit"),
  format = c("flextable", "gt"),
  theme = c("minimal"),
  model_stats = FALSE,
  show_ref = TRUE
)
```

Arguments

data	A data frame containing the outcome, predictors, and group.
outcome	Continuous outcome for ‘approach = "linear"‘ or binary outcome for ‘approach = "logit"‘.
exposures	Predictor variables to report.
group	Grouping variable for the random intercept.
adjust_for	Optional covariates included in the model but not reported.
approach	‘"linear"‘ or ‘"logit"‘.
format	Output table format: ‘"flextable"‘ or ‘"gt"‘.
theme	Table styling preset.
model_stats	Logical; return model fit, clustering, random-effect, and ICC statistics.
show_ref	Logical; display categorical reference rows.

Details

This intentionally focused workflow supports one random intercept. Random slopes, crossed effects, and nested random-effects formulas are outside this function’s scope and should be fitted directly with specialist software.

For linear models, the ICC is the random-intercept variance divided by total random-intercept plus residual variance. For logistic models, a latent-scale ICC is calculated using residual variance pi-squared over three.

Value

A ‘gtregression‘ object containing the publication table, numeric estimates, fitted mixed model, model statistics, and analysis metadata.

Examples

```
set.seed(71)
multilevel_data <- data.frame(
  clinic = factor(rep(seq_len(20), each = 12)),
  treatment = factor(rep(c("Control", "Intervention"), 120)),
  age = rnorm(240, 58, 9)
)
clinic_effect <- rnorm(20, 0, 3)
multilevel_data$score <- 50 + 2 * (multilevel_data$treatment == "Intervention") +
  0.2 * multilevel_data$age + clinic_effect[multilevel_data$clinic] + rnorm(240)

multilevel_reg(
  multilevel_data,
  outcome = score,
  exposures = c(treatment, age),
  group = clinic,
  approach = linear,
  model_stats = TRUE
)
```

`multinomial_reg` *Multinomial logistic regression*

Description

Fit crude, exposure-specific adjusted, or joint multivariable multinomial logistic models for an unordered outcome with three or more categories.

Usage

```
multinomial_reg(
  data,
  outcome,
  exposures,
  adjust_for = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  model_stats = FALSE,
  format = c("flextable", "gt"),
  theme = "minimal",
  show_ref = TRUE
)
```

Arguments

`data`	A data frame containing the outcome and predictors.
`outcome`	An unordered factor outcome with at least three levels. The first level is the reference outcome category.
`exposures`	Predictor variables to report.
`adjust_for`	Optional covariates included but not reported.
`multivariable`	Logical; fit all exposures in one model.
`multivariate`	Optional alias for ‘multivariable‘.
`model_stats`	Logical; return model-fit and sample-size statistics.
`format`	Output table format: ‘"flextable"‘ or ‘"gt"‘.
`theme`	Table styling preset.
`show_ref`	Logical; display categorical predictor reference rows.

Details

Each odds ratio compares one non-reference outcome category with the first outcome level. Outcome categories are nominal: their order has no numeric or severity interpretation. Use ‘relevel()‘ before fitting to choose another reference category.

Value

A ‘gtregression‘ object containing a side-by-side table of outcome comparisons, fitted models, statistics, and one standard regression object per comparison in ‘$per_outcome‘.

Examples

```
set.seed(1501)
example_data <- data.frame(
  destination = factor(sample(c("Home", "Rehabilitation", "Hospital"), 120,
                              replace = TRUE),
                       levels = c("Home", "Rehabilitation", "Hospital")),
  treatment = factor(sample(c("Control", "Intervention"), 120, TRUE)),
  age = rnorm(120, 65, 10)
)
multinomial_reg(
  example_data,
  outcome = destination,
  exposures = treatment,
  adjust_for = age
)
```

---

multi_reg *Multivariable regression*

Description

Fit adjusted or multivariable regression models and return a manuscript-style table in **flextable** or **gt** format.

Usage

```
multi_reg(
  data,
  outcome,
  exposures,
  adjust_for = NULL,
  interaction = NULL,
  approach = "logit",
  format = c("flextable", "gt"),
  theme = c("minimal"),
  model_stats = FALSE,
  show_ref = TRUE
)
```

Arguments

data	A data.frame containing the variables of interest.
outcome	Character scalar; name of the outcome variable. Quoted and bare names are accepted. For binary-outcome approaches ("logit", "firth", "logbinomial", "robpoisson") the outcome may be numeric 0/1 or 1/2 (2 is the event; integer or double), logical (TRUE is the event), or a two-level factor or character variable (the second level is the event, which is reported in a message). The same factor, character and logical coding is accepted by "poisson".
exposures	Character vector; exposure variable(s) to report. Quoted names are recommended in scripts, and bare names are also accepted. If adjust_for = NULL, all exposures are included in a single multivariable model. If adjust_for is supplied, one adjusted model is fitted per exposure and only exposure-specific adjusted estimate(s) are displayed.
adjust_for	Optional character vector of adjustment variables. Quoted and bare names are accepted. Must not overlap with exposures.
interaction	Optional character scalar specifying one interaction term using standard formula syntax, e.g. "bmi*sex". When used with adjust_for, only a single exposure should be supplied.
approach	Character scalar specifying the regression approach. One of "logit", "firth", "logbinomial", "poisson", "linear", "robpoisson", or "negbin". Use "firth" for Firth penalized logistic regression, especially with sparse cells or separation.
format	Output table format; one of "flextable" (default) or "gt".
theme	Table styling preset (e.g. "minimal", "striped", "clinical", "shaded", "jama") or a character vector of primitives such as c("plain","zebra","lines","labels_bold","compact",""
model_stats	Logical; if TRUE, extract model-fit statistics such as AIC, BIC, log-likelihood, deviance, pseudo R-squared for non-linear models, and R-squared for linear models. Statistics are stored in the returned object's model_stats element and are not added to the publication table.
show_ref	Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

Details

Use the default mode when you want all exposures in one model. Use adjust_for when you want one adjusted estimate per reported exposure, with the same adjustment set used repeatedly.

If exposure variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table and plots. Internal matching still uses the original column names.

Confidence intervals are Wald intervals; for approach = "linear" they use the t distribution with the residual degrees of freedom (as confint() for lm). Factor exposures are always fitted with treatment contrasts. Log-binomial models that fail with the default starting values are refitted from the intercept log(mean(y)); if the model still cannot be fitted the error names the approach and suggests approach = "robpoisson".

In default mode (adjust_for = NULL), all exposures are included in a single multivariable model. In adjusted mode, one model is fitted per exposure, adjusting for the variables specified in adjust_for.

Interaction terms specified via interaction are included in the model using standard formula expansion (e.g. bmi*sex). Interaction effects are displayed as additional rows beneath the corresponding exposure.

Value

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

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

table_body A data frame of adjusted estimates and confidence intervals for each exposure and level.

table_display A data frame used for rendering the final table, including header and level rows.

models A list of fitted model(s).

model_summaries summary() output for the fitted model(s).

model_stats Model-fit statistics when model_stats = TRUE; otherwise NULL.

variable_labels Named character vector of display labels used for exposure variables.

reg_check Regression diagnostics for linear models. Printing this element renders a publication-ready table; individual diagnostic rows remain available for programmatic use. Other approaches return a message.

approach The regression approach used.

format The output format used.

source Function identifier ("multi_reg").

adjusted_mode Whether one adjusted model per exposure was fitted.

adjust_for Adjustment variables supplied by the user, if any.

exposures Exposure variables supplied by the user.

interaction Interaction term supplied by the user, if any.

Examples

```
endometrial_data <- data_endometrial
endometrial_data$HG <- factor(endometrial_data$HG, levels = c(0, 1))
endometrial_data$NV <- factor(endometrial_data$NV, levels = c(0, 1))
multi_reg(endometrial_data, HG, c(NV, PI, EH), approach = firth, format = gt)$table
```

`ordinal_reg` *Proportional-odds ordinal logistic regression*

Description

Fit crude, exposure-specific adjusted, or joint multivariable proportional-odds models and return publication-ready common odds ratios.

Usage

```
ordinal_reg(
  data,
  outcome,
  exposures,
  adjust_for = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  assumption_test = TRUE,
  model_stats = FALSE,
  format = c("flextable", "gt"),
  theme = "minimal",
  show_ref = TRUE
)
```

Arguments

`data`	A data frame containing the outcome and predictors.
`outcome`	An ordered-factor outcome. Its level order defines movement from lower to higher categories.
`exposures`	Predictor variables to report.
`adjust_for`	Optional covariates included but not reported.
`multivariable`	Logical; fit all exposures in one model.
`multivariate`	Optional alias for ‘multivariable‘.
`assumption_test`	Logical; perform term-level likelihood-ratio tests of the proportional-odds assumption.
`model_stats`	Logical; return model-fit and sample-size statistics.
`format`	Output table format: ‘"flextable"‘ or ‘"gt"‘.
`theme`	Table styling preset.
`show_ref`	Logical; display categorical reference rows.

Details

An odds ratio above one indicates greater odds of being in a higher outcome category, assuming the proportional-odds assumption holds. The outcome must be an ordered factor so category direction is never inferred silently.

Value

A ‘gtregression‘ object containing the publication table, fitted models, common odds ratios, proportional-odds tests, and metadata.

Examples

```
ordinal_data <- data.frame(
  recovery = ordered(
    c("Poor", "Fair", "Good", "Excellent", "Fair", "Good"),
    levels = c("Poor", "Fair", "Good", "Excellent")
  ),
  treatment = factor(c("Control", "Control", "New", "New", "Control", "New")),
  age = c(70, 64, 58, 51, 62, 55)
)
ordinal_reg(
  ordinal_data,
  outcome = recovery,
  exposures = treatment,
  adjust_for = age,
  assumption_test = FALSE
)
```

---

plot_cox_survival *Plot adjusted survival curves from a Cox model*

Description

Plot model-based survival curves from a fitted Cox proportional hazards model at user-supplied or representative covariate profiles.

Usage

```
plot_cox_survival(
  model,
  model_name = NULL,
  newdata = NULL,
  by = NULL,
  conf.int = TRUE,
  break_time_by = NULL,
  xlim = NULL,
  xlab = "Time",
  ylab = "Adjusted survival probability",
  title = "Adjusted Cox survival curves",
  legend_title = "Profile",
  palette = NULL,
  base_size = 13
)
```

Arguments

model	A fitted coxph model or a cox_reg() result fitted without stratifier. Models with a stratified baseline hazard (strata() terms, or cox_reg(baseline_strata =)) are supported; see Details.
model_name	Model name or position when a cox_reg() result contains more than one fitted model.
newdata	Optional data frame containing one covariate profile per curve. Row names are used as curve labels when available.
by	Optional model variable used to generate one curve per observed level. Other model variables are held at representative values. Ignored when newdata is supplied.
conf.int	Logical; display pointwise confidence bands.
break_time_by	Optional numeric interval for x-axis breaks.
xlim	Optional numeric vector of length 2 specifying x-axis limits.
xlab, ylab	Axis labels.
title	Optional plot title.
legend_title	Legend title.
palette	Optional character vector of curve colours.
base_size	Base font size.

Details

For a model with a stratified baseline hazard, each covariate profile has a different survival curve in every baseline stratum. One curve is therefore drawn per profile and stratum: with a single profile the strata are distinguished by colour; with several profiles, colour identifies the profile and line type identifies the stratum. The stratum variables are not covariates and cannot be used in by. If newdata contains the stratum variables, each row is drawn only for its own stratum.

Value

A ggplot2 object with attributes survfit, plot_data, and prediction_data. plot_data has one row per time point and curve, with columns profile and stratum identifying the curve.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))
fit <- cox_reg(
  lung_data, time, status,
  exposures = c(trt, age, karno),
  multivariable = TRUE
)
plot_cox_survival(fit, by = trt)
```

plot_mediation *Plot mediation paths*

Description

Draw the exposure, mediator, and outcome path from a `mediation_analysis()` result.

Usage

```
plot_mediation(mediation_object, show_estimates = TRUE, base_size = 13)
```

Arguments

`mediation_object`	Object returned by `mediation_analysis()`.
`show_estimates`	Logical; if `TRUE`, show direct and indirect effect estimates on the plot.
`base_size`	Base font size.

Value

A `ggplot2` object.

Examples

```
med <- mediation_analysis(
  data = data_diabetes_mediation,
  exposure = obesity,
  mediator = glucose,
  outcome = diabetes,
  covariates = c(age, blood_pressure, pregnancies, diabetes_pedigree),
  outcome_approach = logit,
  sims = 50,
  seed = 123
)
plot_mediation(med)
plot_mediation(med, show_estimates = FALSE)
```

plot_model_fit *Plot model-fit diagnostics*

Description

Visualise model fit for fitted regression models and models stored inside `uni_reg()` or `multi_reg()` results.

Usage

```
plot_model_fit(
  model,
  model_name = NULL,
  type = c("auto", "all", "residual", "qq", "scale_location", "cooks",
    "observed_predicted", "calibration"),
  bins = 10,
  base_size = 13
)
```

Arguments

model	A fitted lm or glm model, or a uni_reg() / multi_reg() result.
model_name	Optional model name to select when model contains multiple fitted models. Quoted and bare names are accepted.
type	Plot type. One of "auto", "all", "residual", "qq", "scale_location", "cooks", "observed_predicted", or "calibration". Quoted and bare values are accepted.
bins	Number of groups used for binomial calibration plots.
base_size	Base font size for the plot theme.

Details

plot_model_fit() is a visual check of how a fitted model behaves, not a formal model-selection rule. For survival models, use check_ph() for Cox proportional hazards diagnostics and plot_surv_fit() for parametric survival model fit.

For binomial models, type = "calibration" compares grouped predicted probabilities with observed event proportions. This is most informative for multivariable models, where predictions vary across many patients. A univariable binary predictor may produce only two calibration points; that is expected and simply reflects the two fitted probabilities in the model. Logistic residual plots often show two bands because the outcome is coded as event/non-event.

Value

A ggplot2 object for a single plot, or a patchwork object when multiple diagnostics are requested.

Examples

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

fit_glm <- glm(am ~ mpg + wt, data = mtcars, family = binomial())
plot_model_fit(fit_glm, type = calibration, bins = 4)

uni_fit <- uni_reg(mtcars, am, c(mpg, wt), approach = logit)
plot_model_fit(uni_fit, model_name = mpg, type = residual)
```

plot_reg *Plot regression estimates*

Description

Draw a forest-style plot from a fitted gtregression object produced by functions such as uni_reg(), multi_reg(), cox_reg(), or surv_reg(). Stratified regression outputs are displayed as one faceted quick-view plot; use forest_df() and forest_reg() for publication-style stratified forest tables.

Usage

```
plot_reg(
  tbl,
  title = NULL,
  caption = NULL,
  ref_line = NULL,
  order_y = NULL,
  log_x = FALSE,
  xlim = NULL,
  breaks = NULL,
  point_color = "#6B7280",
  errorbar_color = "#6B7280",
  point_size = 2.8,
  point_stroke = 0.55,
  ci_linewidth = 0.55,
  base_size = 12,
  show_ref = NULL,
  sig_color = "#0072B2",
  sig_errorbar_color = "#0072B2",
  alpha = 0.05,
  show_adjustment_note = TRUE
)
```

Arguments

tbl	A fitted gtregression object.
title	Optional plot title.
caption	Optional plot caption. If NULL, notes are added automatically for adjusted multi_reg() objects and displayed reference categories.
ref_line	Optional numeric value for the reference line. Defaults to 0 for linear models and 1 otherwise.
order_y	Optional character vector specifying exposure order.
log_x	Logical. If TRUE, use a log-scaled x-axis for ratio measures. The default FALSE uses a linear x-axis.

xlim	Optional numeric vector of length 2 specifying x-axis limits.
breaks	Optional numeric vector of x-axis tick breaks.
point_color	Fill color for non-significant estimate points.
errorbar_color	Color for non-significant confidence intervals.
point_size	Diameter of estimate points in millimetres.
point_stroke	Outline width of estimate points.
ci_linewidth	Line width of confidence intervals.
base_size	Base font size. The default is designed for a clear report or slide figure; increase it for a single large plot.
show_ref	Logical or NULL. The default NULL inherits the reference-row choice used to create tbl. If TRUE, reference rows are shown. If FALSE, binary exposures are shown as compact rows for the estimated non-reference category; affirmative levels such as Yes or 1 are displayed using the variable name.
sig_color	Optional fill color for significant points.
sig_errorbar_color	Optional color for significant error bars.
alpha	Significance level for linear models when p.value is available.
show_adjustment_note	Logical; if TRUE, add a default caption describing adjust_for variables when available.

Value

A ggplot2 object.

`plot_reg_combine` *Plot crude and adjusted estimates side by side*

Description

Creates two aligned forest plots from compatible gtregression regression objects such as crude and adjusted outputs from uni_reg(), multi_reg(), survival regression, or other supported model families.

Usage

```
plot_reg_combine(
  tbl_uni,
  tbl_multi,
  title_uni = NULL,
  title_multi = NULL,
  caption = NULL,
  ref_line = NULL,
  order_y = NULL,
```

```
  log_x = FALSE,
  point_color = "#6B7280",
  errorbar_color = "#6B7280",
  point_size = 2.8,
  point_stroke = 0.55,
  ci_linewidth = 0.55,
  base_size = 12,
  show_ref = NULL,
  sig_color = "#0072B2",
  sig_errorbar_color = "#0072B2",
  xlim_uni = NULL,
  breaks_uni = NULL,
  xlim_multi = NULL,
  breaks_multi = NULL,
  alpha = 0.05,
  show_adjustment_note = TRUE
)
```

Arguments

tbl_uni	The first gtregression object: crude results (for example uni_reg(), or cox_reg() without adjustment), or exposure-specific adjusted results (adjust_for without multivariable = TRUE).
tbl_multi	The second gtregression object from the same model family: the adjusted or single multivariable model. When both objects are adjusted, the second panel is labelled "Mutually Adjusted".
title_uni, title_multi	Optional panel titles.
caption	Optional combined plot caption. If NULL, notes are added automatically for adjusted multi_reg() objects and displayed reference categories.
ref_line	Optional numeric reference line. If NULL, uses 0 for linear models and 1 otherwise.
order_y	Optional character vector to customize exposure ordering.
log_x	Logical. If TRUE, use a log-scaled x-axis for ratio measures. The default FALSE uses a linear x-axis.
point_color, errorbar_color	Base colors for non-significant rows.
point_size	Diameter of estimate points in millimetres.
point_stroke	Outline width of estimate points.
ci_linewidth	Line width of confidence intervals.
base_size	Base font size for the plot theme.
show_ref	Logical or NULL. The default NULL inherits the setting used to create the supplied tables. If TRUE, include reference levels as (Ref.). If FALSE, binary exposures are shown as compact rows for the estimated non-reference category; affirmative levels such as Yes or 1 are displayed using the variable name.

sig_color, sig_errorbar_color
: Optional colors for significant rows. If NULL, base colors are reused.

xlim_uni, breaks_uni
: Optional x-axis limits and breaks for the univariate panel.

xlim_multi, breaks_multi
: Optional x-axis limits and breaks for the multivariable panel.

alpha
: Significance level for linear models when p.value is available.

show_adjustment_note
: Logical; if TRUE, add a default caption describing adjust_for variables from tbl_multi when available.

Value

A patchwork object with two ggplot2 panels.

---

plot_surv_fit *Plot observed and fitted parametric survival curves*

Description

Compare the observed Kaplan-Meier survival curve with fitted parametric survival curves from survival::survreg().

Usage

```
plot_surv_fit(
  data,
  time,
  event,
  by = NULL,
  adjust_for = NULL,
  distributions = c("weibull", "exponential", "lognormal", "loglogistic"),
  break_time_by = NULL,
  xlim = NULL,
  xlab = "Time",
  ylab = "Survival probability",
  title = NULL,
  legend_title = NULL,
  palette = NULL,
  base_size = 13,
  n_points = 200
)
```

Arguments

data	A data.frame containing survival time, event status, and optional grouping or adjustment variables.
time	Survival follow-up time. Quoted and bare names are accepted.
event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
by	Optional grouping variable for observed and fitted curves. Quoted and bare names are accepted.
adjust_for	Optional character vector of adjustment variables included in the fitted parametric model. Fitted curves are predicted at typical adjustment values.
distributions	Parametric survival distributions to overlay. One or more of "weibull", "exponential", "lognormal", or "loglogistic". Quoted and bare values are accepted. Common spellings such as "log-normal" and "log-logistic" are also accepted.
break_time_by	Optional numeric interval for x-axis breaks. If NULL, breaks are chosen automatically.
xlim	Optional numeric vector of length 2 specifying x-axis limits.
xlab, ylab	Axis labels.
title	Optional plot title.
legend_title	Optional legend title. If NULL, the labelled by variable name is used.
palette	Optional character vector of colors for observed groups.
base_size	Base font size.
n_points	Number of points used to draw each fitted curve.

Details

plot_surv_fit() is a visual diagnostic for parametric survival modelling. It is useful after surv_model_compare() and before treating a final surv_reg() model as the preferred model. It is not a Cox-model diagnostic; use check_ph() for Cox proportional hazards assumptions.

When adjust_for is supplied, fitted curves are predicted at typical adjustment values: medians for numeric variables and the most common level for categorical variables. Use this as a model-fit screen, not as a replacement for clinical or subject-matter judgement.

Value

A ggplot2 object with attributes km_fit, model_fits, observed_data, fitted_data, and prediction_data.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

plot_surv_fit(
  data = lung_data,
```

```
  time = time,
  event = status,
  by = trt,
  distributions = c(weibull, lognormal),
  break_time_by = 200
)

plot_surv_fit(
  data = lung_data,
  time = "time",
  event = "status",
  by = "trt",
  adjust_for = c(age, karno),
  distributions = "log-logistic"
)
```

`print.gtregression` *Print gtregression objects*

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.

Value

`x`, invisibly. Called for its side effect of printing the rendered table.

`risk_difference` *Risk difference between two groups*

Description

Reports the absolute risk of a binary outcome in each of two groups and the risk difference between them, with a confidence interval and p-value, as a publication-ready table. The crude analysis uses the Miettinen-Nurminen score method; the adjusted analysis standardises the risks from a logistic regression model. Optionally the number needed to treat (NNTB / NNTH) is added.

Usage

```
risk_difference(
  data,
  outcome,
  exposure,
  adjust_for = NULL,
  reference = NULL,
  event = NULL,
  event_type = NULL,
  conf_level = 0.95,
  format = "gt",
  theme = NULL
)
```

Arguments

`data`	A data frame.
`outcome`	Binary outcome column. Quoted and bare names are accepted. Logical, numeric 0/1 or 1/2, two-level factor, two-value character and value-labelled (**haven**) variables are supported.
`exposure`	Column defining exactly two groups (for example treatment arm or exposure status). Quoted and bare names are accepted.
`adjust_for`	Optional adjustment variables (quoted or bare names). `NULL` (default) gives the crude analysis; otherwise the adjusted (standardised) analysis is reported.
`reference`	Group used as the reference. The risk difference is the other (comparison) group minus the reference group. `NULL` uses the first factor level, `FALSE`, `0`, or the first value in sorted order.
`event`	Value of `outcome` that counts as the event. `NULL` uses `TRUE`, `1` (for 0/1 coding), `2` (for 1/2 coding), the second factor level, or the second value in sorted order.
`event_type`	`NULL` (default), `"harmful"` or `"beneficial"`. When given, a column with the number needed to treat is added (see Details). The event direction is needed to tell benefit from harm, so no NNT is shown without it.

conf_level	Confidence level of the interval; default 0.95. It changes the confidence interval only. The p-value is always the two-sided test of a risk difference of zero and does not depend on conf_level, so, for example, a 99% interval can include zero while the p-value is below 0.05.
format	"gt" (default) or "flextable".
theme	Table theme: a preset name ("minimal", "clinical", "striped", "shaded", "jama") or a vector of primitives, as in the regression functions. NULL (default) uses the package default theme ("minimal").

Details

Absolute risk and risk difference. The risk in a group is the proportion of its members with the event. The risk difference (RD) is the risk in the comparison group minus the risk in the reference group. It is stored as a difference of proportions (for example -0.08) and shown in percentage points (-8.0). A negative RD means that the event is less frequent in the comparison group. For a harmful event (for example death) a negative RD is an absolute risk reduction and a positive RD an absolute risk increase; for a beneficial event (for example cure) the reading is reversed. These are readings of the same number, not separate estimates, and the table reports the signed RD only. The resolved event, reference and comparison are stored in the result, are always written in the table footnote, and are announced in a message whenever they were inferred from anything other than logical or 0/1 coding.

Analysis population. One complete-case population is used: observations with a missing outcome or exposure (and, for the adjusted analysis, any missing adjustment variable) are excluded. The numbers supplied, analysed and excluded are stored and the footnote reports them.

Crude analysis (adjust_for = NULL). Risks are the observed proportions. The confidence interval is the Miettinen-Nurminen score interval (restricted maximum-likelihood variance with the $N/(N-1)$ factor, no continuity correction) and the p-value comes from the same score statistic at a difference of zero (equal to Pearson's chi-square multiplied by $(N-1)/N$). The method is defined when a group has no events or only events; when no one, or everyone, has the event the p-value is 1.

Adjusted analysis. A main-effects logistic regression of the outcome on the exposure and the adjustment variables is fitted. Each person's risk is predicted twice from their own covariates, once as a member of each group; the predictions are averaged over the analysis population to give the two adjusted (standardised) risks, and their difference is the adjusted RD. The table shows the observed events and group sizes next to the adjusted risks. The standard error is the influence-function standard error of the stacked estimating equations (logistic score equations and the two averages), which accounts for the estimation of the model and for the sampling of the covariates and does not rely on the logistic model being correct. The confidence interval is a Wald interval, with limits restricted to the range -1 to 1, and the p-value is two-sided normal. No small-sample correction is applied; in simulations with about 80 to 100 participants coverage of the 95% interval was about 94%. The analysis stops with an error, and returns no table, when the model does not converge, when fitted probabilities reach 0 or 1 (separation, including a group with no events or only events), when the exposure or an adjustment variable is aliased, or when the outcome is constant.

Interpretation. The function describes a calculation and makes no causal claim. In an individually randomised trial the adjusted RD is a covariate-adjusted estimate of the marginal treatment difference, and it remains valid when the logistic model is not exactly right (see the FDA 2023 guidance on covariate adjustment). In a non-randomised study it is a standardised risk difference; it can be

read causally only if there is no unmeasured confounding, the outcome model is correct and the groups overlap in their covariates.

NNTB and NNTH. With `event_type`, the number needed to treat is the reciprocal of the absolute value of the unrounded RD, shown rounded up to a whole person. It is labelled NNTB (number needed to treat for one additional person to benefit) when the comparison group does better and NNTH (for one additional person to be harmed) when it does worse: for a harmful event a negative RD gives NNTB and a positive RD NNTH, and the reverse for a beneficial event. When the confidence interval of the RD includes zero, the interval of the NNT runs from a finite NNTB through infinity to a finite NNTH and is shown and stored in that form (Altman 1998), never as a pair of finite limits. The NNT applies to the follow-up period of the study and assumes that the RD reflects what the intervention does.

Limits of this version. Binary outcome at a fixed follow-up; exactly two groups; independent observations; main-effects logistic model for adjustment. Not supported: more than two groups, continuous exposures, stratified (Mantel-Haenszel) or clustered designs, non-inferiority margins, time-to-event outcomes. The result can be printed, saved with `save_table` and relabelled with `modify_table` (caption, caveat, labels and headers only); it is not accepted by `merge_tables` or the forest-plot functions.

Value

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

table The rendered `gt_tbl` or `flextable`.

table_display Data frame of the formatted cells as shown.

table_body Data frame of numeric results, one row per group (reference first): `group`, `events`, `n`, `risk`, `adjusted_risk`, `adjusted_risk_se`, `rd`, `rd_se`, `rd_low`, `rd_high`, `p_value`. Risks and differences are proportions. `rd_se` is the influence-function standard error and is `NA` for the crude analysis, whose score interval is not based on a single standard error.

footnotes Character vector of table footnotes.

analysis `"crude"` or `"adjusted"`.

outcome, exposure, adjust_for Variable names.

event, reference, comparison Resolved event value and groups.

event_type, conf_level As supplied.

method List naming the estimate, interval, test and variance methods.

n List with the numbers of rows `supplied`, `analysed` and `excluded`.

nnt `NULL`, or a list with `type` (`"NNTB"`, `"NNTH"` or `NA`), `estimate` (unrounded, may be `Inf`), `crosses_infinity` and `segments` (data frame with `type`, `from`, `to`; two rows ending at `Inf` when the interval runs through infinity).

model The fitted logistic model (adjusted analysis), else `NULL`. In the model the outcome is `.rd_y` (1 = event) and the exposure is `.rd_a` (1 = comparison group).

variable_labels, level_labels, theme Labels and theme used for display.

source, format `"risk_difference"`; `"gt"` or `"flextable"`.

References

Miettinen O, Nurminen M (1985). Comparative analysis of two rates. *Statistics in Medicine*, 4(2), 213-226.

Newcombe RG (1998). Interval estimation for the difference between independent proportions: comparison of eleven methods. *Statistics in Medicine*, 17(8), 873-890.

Altman DG (1998). Confidence intervals for the number needed to treat. *BMJ*, 317(7168), 1309-1312.

U.S. Food and Drug Administration (2023). Adjusting for Covariates in Randomized Clinical Trials for Drugs and Biological Products: Guidance for Industry.

See Also

uni_reg and multi_reg for relative measures (odds ratios, risk ratios).

Examples

```
# Crude: low birth weight by maternal smoking
rd <- risk_difference(data_birthwt, low, smoke)
rd$table_display
rd$footnotes

# Adjusted (standardised), with the number needed to treat
adj <- risk_difference(
  data_birthwt, "low", "smoke",
  adjust_for = c("age", "lwt"),
  event_type = "harmful",
  format = "flextable"
)
adj$table_display
adj$table_body
adj$nnt

# Factor coding: name the event and the reference explicitly
bw <- data_birthwt
bw$smoking <- factor(bw$smoke, levels = c(0, 1), labels = c("No", "Yes"))
bw$weight <- factor(bw$low, levels = c(0, 1), labels = c("Normal", "Low"))
risk_difference(bw, weight, smoking, event = "Low", reference = "No",
                conf_level = 0.90)$table_display
```

---

rmst_table *Restricted mean survival time table*

Description

Estimate restricted mean survival time (RMST) up to a user-specified follow-up time. RMST is the average survival time observed within a fixed time window, such as 365 days.

Usage

```
rmst_table(
  data,
  time,
  event,
  by = NULL,
  tau,
  digits = 1,
  conf.level = 0.95,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

data	A data.frame containing survival time and event status.
time	Survival follow-up time. Quoted and bare names are accepted.
event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
by	Optional grouping variable for separate RMST estimates. Quoted and bare names are accepted.
tau	Restriction time for RMST. For example, tau = 365 reports mean survival time restricted to the first 365 days of follow-up. If tau is beyond the last follow-up time of a group, the estimate extrapolates the survival curve; a warning is raised and a footnote is added to the table.
digits	Number of digits for displayed survival time summaries.
conf.level	Confidence level for RMST confidence intervals. The column headers show the level used, for example "RMST (90% CI)".
format	Output format. One of "flextable" (default), "gt", or "tibble".
theme	Table styling preset.

Details

RMST is estimated from survival::survfit() using Kaplan-Meier methods and summary.survfit(rmean = tau). When by has exactly two groups, the table includes the RMST difference as the second group minus the first group, with a Wald confidence interval and p-value based on the reported RMST standard errors.

Value

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

table A flextable, gt_tbl, or NULL when format = "tibble".

table_body Tibble with numeric RMST summaries.

table_display Display data frame used to render the table.

`fit` Fitted `survfit` object.

`time,event,by,tau,format,source` Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

rmst_table(
  data = lung_data,
  time = time,
  event = status,
  by = trt,
  tau = 365
)

rmst_table(
  data = lung_data,
  time = "time",
  event = "status",
  tau = 180,
  format = tibble
)
```

---

save_docx *Save multiple tables and plots to a Word document*

Description

Saves a collection of `gtregression` tables, merged tables, `gt_tbl`/`flextable` objects, and `ggplot2` plots into a single Word document.

Usage

```
save_docx(
  tables = NULL,
  plots = NULL,
  filename = "report.docx",
  titles = NULL,
  table_width = 6.5,
  plot_width = 6,
  plot_height = 5
)
```

Arguments

tables	A list of tables. Each element may be a gtregression object, merged_table object, gt_tbl, or flextable.
plots	A list of ggplot2 plot objects.
filename	File name for the output, with or without .docx. If no directory is supplied, the file is saved in tempdir().
titles	Optional character vector of titles for tables and plots in the order they are added.
table_width	Maximum table width in inches for Word export. The default 6.5 fits a standard portrait Word page with common margins. Use NULL to keep the original flextable widths.
plot_width	Width of inserted plots in inches.
plot_height	Height of inserted plots in inches.

Value

Saves the Word document to disk. Invisibly returns the normalized file path.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

tbl <- uni_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("age", "smoke"),
  approach = "logit",
  format = "flextable"
)

save_docx(tables = tbl, filename = tempfile("report"))
```

---

save_forest *Save a forest_reg() output*

Description

Saves a forest_reg() output, or a compatible **forestploter**/grid object, to a fixed graphics device. This is useful when the RStudio Viewer or operating-system graphics device crops wide forest plots or compresses forest columns.

Usage

```
save_forest(
  forest,
  filename = "forest",
  format = c("pdf", "png", "tiff", "jpg"),
  width = NULL,
  height = NULL,
  scale = 1,
  auto_size = TRUE,
  padding = 0.25,
  dpi = 300
)
```

Arguments

forest	A gtregression_forest object returned by forest_reg(), or a compatible **forestploter**/grid object.
filename	File name for the output, with or without extension. If no directory is supplied, the file is saved in tempdir().
format	Output format. One of "pdf", "png", "tiff", or "jpg".
width, height	Optional export width and height in inches. If either is NULL and auto_size = TRUE, the size is measured from the intrinsic **forestploter** gtable dimensions. If measurement is not possible, a practical fallback is estimated from the number of rows and columns in the forest_reg() output.
scale	Positive multiplier applied to the export width and height. This is a quick way to make a large forest plot roomier.
auto_size	Logical. If TRUE, measure the intrinsic forest plot size when width or height is not supplied. If FALSE, use the fallback row/column-based sizing.
padding	White space around the forest plot in inches.
dpi	Resolution for raster formats.

Value

Saves the file to disk. Invisibly returns the normalized file path.

Examples

```
birthwt_data <- data_birthwt |>
  transform(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

uni_or <- uni_reg(
  birthwt_data,
  outcome = "low",
```

```
  exposures = c("age", "smoke", "ht"),
  approach = "logit"
)

forest <- forest_reg(uni = uni_or)
save_forest(forest, filename = tempfile("forest"), format = "pdf")

# For large forest plots, increase width, height, scale, or padding.
save_forest(
  forest,
  filename = tempfile("forest-wide"),
  format = "png",
  scale = 1.2,
  padding = 0.35,
  dpi = 300
)
```

`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. If no directory is supplied, the file is saved in `tempdir()`.
`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.

Value

Saves the file to disk. Invisibly returns the normalized file path.

Examples

```
p <- ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) +
  ggplot2::geom_point()

save_plot(p, filename = tempfile("plot"), format = "png")
```

`save_table` *Save a single regression or summary table*

Description

Saves a `gtregression` table, merged table, `gt_tbl`, or `flextable` as a Word, RTF, PDF, or HTML file.

Usage

```
save_table(
  tbl,
  filename = "table",
  format = c("docx", "rtf", "pdf", "html"),
  orientation = c("auto", "portrait", "landscape"),
  fit_width = TRUE,
  font_size = 9,
  min_font_size = 8
)
```

Arguments

`tbl`	A `gtregression` object, `merged_table` object, `gt_tbl`, or `flextable`.
`filename`	File name for the output. Extension is optional. If no directory is supplied, the file is saved in `tempdir()`.
`format`	Output format. One of `"docx"`, `"rtf"`, `"pdf"`, or `"html"`. Flextable objects can be saved as `"docx"`, `"rtf"`, or `"html"`.
`orientation`	Word page orientation for DOCX output. One of `"auto"`, `"portrait"`, or `"landscape"`. With `"auto"`, wide tables are saved in landscape orientation before any font-size reduction is attempted.
`fit_width`	Logical. If `TRUE`, try to fit flextable DOCX output within the selected Word page width. If `FALSE`, keep the natural autofit table width.
`font_size`	Requested font size for flextable DOCX output.
`min_font_size`	Smallest font size allowed when fitting wide flextable DOCX output. The font size is never reduced below this value.

Value

Saves the file to disk. Invisibly returns the normalized file path.

Examples

```
birthwt_data <- data_birthwt |>
  dplyr::mutate(
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1), labels = c("Normal BW", "Low BW"))
  )

tbl <- uni_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("age", "smoke"),
  approach = "logit"
)

save_table(tbl, filename = tempfile("table"), format = "html")

# Wide Word tables can be saved in landscape orientation.

save_table(
  tbl,
  filename = tempfile("table-wide"),
  format = "docx",
  orientation = "auto",
  fit_width = TRUE,
  font_size = 9,
  min_font_size = 8
)
```

---

select_models *Stepwise model selection with fit metrics*

Description

Compare candidate models added or removed stepwise, then return the accepted steps, model-fit metrics, and the lowest-AIC model.

Usage

```
select_models(
  data,
  outcome,
  exposures,
  approach = "logit",
  time = NULL,
  event = NULL,
  distribution = "weibull",
  direction = "forward",
  format = c("flextable", "gt", "tibble")
)
```

Arguments

data	A data frame containing the outcome and predictor variables.
outcome	A single character string indicating the outcome variable. Quoted and bare names are accepted. Not used for survival approaches when time and event are supplied.
exposures	Character vector of predictor variables to consider. Quoted names are recommended in scripts, and bare names are also accepted.
approach	Regression method. One of: "logit", "logbinomial", "poisson", "robpoisson", "negbin", "linear", "cox", or "survreg".
time, event	Survival time and event indicator for approach = "cox" or approach = "survreg". Quoted and bare names are accepted.
distribution	Parametric survival distribution for approach = "survreg". One of "weibull", "exponential", "lognormal", or "loglogistic". Common spellings such as "log-normal" and "log-logistic" are also accepted.
direction	Stepwise selection direction. One of: "forward" (default), "backward", or "both".
format	Output format for the viewing table. One of "flextable" (default), "gt", or "tibble". Use format = "tibble" to keep only the original list structure.

Details

Treat stepwise selection as a screening tool. It is best used alongside the study question, clinical judgement, and model diagnostics rather than as an automatic final-model rule.

Information criteria are only comparable between models fitted to the same observations. select_models() therefore restricts the data to complete cases across the outcome (or time and event) and all candidate variables before selection, reports the N used and the number of rows excluded in a message and in the table note, and stores them as n_used and n_excluded. Backward (and stepwise) selection can remove the last variable and so reach the intercept-only model.

Value

A list with the following components:

- results_table: A tibble summarising each accepted step's model metrics (AIC, BIC, deviance, log-likelihood, and adjusted R-squared for linear models).
- best_model: The best-fitting model object based on lowest AIC.
- n_used, n_excluded: number of complete cases used for every candidate model and number of rows excluded because of missing values in the outcome or any candidate variable.
- all_models: A named list of the accepted stepwise models.
- direction: Stepwise selection direction used.
- table: A formatted gt_tbl or flextable when format is "gt" or "flextable".

Examples

```
data <- data_birthwt
stepwise <- select_models(
  data = data,
  outcome = "bwt",
  exposures = c("age", "lwt", "smoke"),
  approach = "linear",
  direction = "forward"
)
stepwise$results_table
stepwise$best_model
```

---

stratified_multi_reg *Stratified multivariable regression*

Description

Fits multivariable regression models within each stratum and returns a unified wide table with one "Characteristic" column and, under bold spanners for each stratum, model N, "Adjusted <effect>", and "p-value".

Usage

```
stratified_multi_reg(
  data,
  outcome,
  exposures,
  stratifier,
  adjust_for = NULL,
  interaction = NULL,
  approach = "logit",
  format = c("flextable", "gt"),
  theme = c("minimal"),
  show_ref = TRUE
)
```

Arguments

data	A data frame containing the variables.
outcome	Character scalar; name of the outcome variable. Quoted and bare names are accepted. For binary-outcome approaches ("logit", "firth", "logbinomial", "robpoisson") the outcome may be numeric 0/1 or 1/2 (2 is the event; integer or double), logical (TRUE is the event), or a two-level factor or character variable (the second level is the event, which is reported in a message). The same factor, character and logical coding is accepted by "poisson". The coding is decided once on the full data, before stratification.

exposures	Character vector of exposure variables to report. Quoted names are recommended in scripts, and bare names are also accepted.
stratifier	Character scalar; name of the stratifying variable. Quoted and bare names are accepted.
adjust_for	Optional character vector of adjustment variables. Quoted and bare names are accepted. This argument works the same way as in multi_reg().
interaction	Optional character scalar specifying one interaction term using standard formula syntax, e.g. "bmi*sex"
approach	One of "logit", "firth", "logbinomial", "poisson", "linear", "robpoisson", or "negbin"
format	One of "flextable" (default) or "gt".
theme	Preset name (e.g. "minimal", "striped", "clinical", "shaded", "jama") or primitives c("plain","zebra","lines","labels_bold","compact","header_shaded")
show_ref	Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

Details

If adjust_for = NULL, all exposures are included in one multivariable model within each stratum. If adjust_for is supplied, one adjusted model is fitted per exposure within each stratum.

If exposure variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table. Internal matching still uses the original column names.

Value

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

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

table_display Wide data frame used to build the table.

per_stratum Named list of per-stratum regression results.

models Named list of fitted models by stratum.

model_summaries Named list of model summaries by stratum.

variable_labels Named character vector of display labels used for exposure variables.

reg_check Named list of diagnostics by stratum.

by, levels, approach, format, source Metadata fields.

Examples

```
birthwt_data <- data_birthwt |>
  transform(
    race = factor(race, levels = c(1, 2, 3),
                  labels = c("White", "Black", "Other")),
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
```

```
    ht = factor(ht, levels = c(0, 1), labels = c("No", "Yes")),
    ui = factor(ui, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1),
                 labels = c("Normal BW", "Low BW"))
  )

stratified_multi <- stratified_multi_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("age", "lwt", "smoke", "ht"),
  stratifier = "race",
  approach = "logit",
  format = "gt"
)

stratified_adjusted <- stratified_multi_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("smoke", "ht", "ui"),
  stratifier = "race",
  adjust_for = c("age", "lwt"),
  approach = "logit",
  format = "gt"
)
```

`stratified_uni_reg` *Stratified univariable regression*

Description

Fit univariable regression models within each level of a stratifier and combine the results into a stratified table rendered with **gt** or **flextable**.

Usage

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

Arguments

`data`	A data frame containing the variables.

outcome	Character scalar; name of the outcome variable. Quoted and bare names are accepted. For binary-outcome approaches ("logit", "firth", "logbinomial", "robpoisson") the outcome may be numeric 0/1 or 1/2 (2 is the event; integer or double), logical (TRUE is the event), or a two-level factor or character variable (the second level is the event, which is reported in a message). The same factor, character and logical coding is accepted by "poisson". The coding is decided once on the full data, before stratification.
exposures	Character vector of exposure variables to model. Quoted names are recommended in scripts, and bare names are also accepted.
stratifier	Character scalar; name of the stratifying variable. Quoted and bare names are accepted.
approach	Modeling approach. One of "logit", "firth", "logbinomial", "poisson", "robpoisson", "linear", or "negbin".
format	Output table format; one of "flextable" (default) or "gt".
theme	Table styling preset or theme primitives.
show_ref	Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

Details

If exposure variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table. Internal matching still uses the original column names.

Value

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

table A rendered gt_tbl or flextable.

table_display Display-ready wide stratified results.

per_stratum List of complete uni_reg() results by stratum.

models List of fitted model objects by stratum.

model_summaries List of model summaries by stratum.

variable_labels Named character vector of display labels used for exposure variables.

reg_check Regression diagnostics by stratum.

by The stratifier variable.

levels Strata included in the analysis.

approach The regression approach used.

format The output format used.

source Function identifier ("stratified_uni_reg").

Accessors

`$table` Rendered stratified regression table.

`$table_display` Wide display data used to build the table.

`$per_stratum` Full `uni_reg()` result objects by stratum.

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

`$model_summaries` List of model summaries.

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

See Also

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

Examples

```
birthwt_data <- data_birthwt |>
  transform(
    race = factor(race, levels = c(1, 2, 3),
                  labels = c("White", "Black", "Other")),
    smoke = factor(smoke, levels = c(0, 1), labels = c("No", "Yes")),
    low = factor(low, levels = c(0, 1),
                 labels = c("Normal BW", "Low BW"))
  )

stratified_uni <- stratified_uni_reg(
  data = birthwt_data,
  outcome = "low",
  exposures = c("age", "lwt", "smoke"),
  stratifier = "race",
  approach = "logit",
  format = "gt"
)

stratified_uni$table
```

---

survival_prob *Kaplan-Meier survival probability table*

Description

Estimate Kaplan-Meier survival probabilities at user-specified follow-up times, such as 30-day, 6-month, or 1-year survival.

Usage

```
survival_prob(
  data,
  time,
  event,
  by = NULL,
  times,
  digits = 1,
  extend = TRUE,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

data	A data.frame containing survival time and event status.
time	Survival follow-up time. Quoted and bare names are accepted.
event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
by	Optional grouping variable for separate Kaplan-Meier survival probabilities. Quoted and bare names are accepted.
times	Numeric vector of follow-up times at which survival probability should be estimated.
digits	Number of digits for percentages and survival probabilities.
extend	Logical. If TRUE, requested times beyond the observed follow-up range are retained using the last available Kaplan-Meier estimate.
format	Output format. One of "flextable" (default), "gt", or "tibble".
theme	Table styling preset.

Details

Survival probabilities are estimated from survival::survfit() at the requested follow-up times. Events and censored counts are interval counts up to each requested time point as returned by summary.survfit().

Value

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

table A flextable, gt_tbl, or NULL when format = "tibble".

table_body Tibble with Kaplan-Meier survival probabilities.

table_display Display data frame used to render the table.

fit Fitted survfit object.

time,event,by,times,format,source Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

survival_prob(
  data = lung_data,
  time = time,
  event = status,
  by = trt,
  times = c(90, 180, 365)
)

survival_prob(
  data = lung_data,
  time = "time",
  event = "status",
  times = c(90, 180),
  format = tibble
)
```

---

survival_quantiles *Kaplan-Meier survival quantile table*

Description

Estimate Kaplan-Meier survival time quantiles, such as the 25th percentile, median, and 75th percentile survival times.

Usage

```
survival_quantiles(
  data,
  time,
  event,
  by = NULL,
  probs = c(0.25, 0.5, 0.75),
  digits = 1,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

data	A data.frame containing survival time and event status.
time	Survival follow-up time. Quoted and bare names are accepted.

event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
by	Optional grouping variable for separate Kaplan-Meier quantiles. Quoted and bare names are accepted.
probs	Numeric vector of event-time quantiles to estimate. The default c(0.25, 0.5, 0.75) reports the 25th percentile, median, and 75th percentile event times.
digits	Number of digits for survival time summaries.
format	Output format. One of "flextable" (default), "gt", or "tibble".
theme	Table styling preset.

Details

A probability of 0.50 is the median event time: the estimated time by which 50 A probability of 0.25 is the time by which 25 corresponding to 75 follow-up, the display table shows "Not reached".

Value

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

table A flextable, gt_tbl, or NULL when format = "tibble".

table_body Tibble with Kaplan-Meier quantiles.

table_display Display data frame used to render the table.

fit Fitted survfit object.

time,event,by,probs,format,source Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

survival_quantiles(
  data = lung_data,
  time = time,
  event = status,
  by = trt
)

survival_quantiles(
  data = lung_data,
  time = "time",
  event = "status",
  probs = c(0.25, 0.5),
  format = tibble
)
```

`survival_summary` *Kaplan-Meier survival summary table*

Description

Summarise Kaplan-Meier follow-up with total N, events, censoring, and median survival with a 95

Usage

```
survival_summary(
  data,
  time,
  event,
  by = NULL,
  digits = 1,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`data`	A `data.frame` containing survival time and event status.
`time`	Survival follow-up time. Quoted and bare names are accepted.
`event`	Event indicator. Quoted and bare names are accepted. Numeric `0/1`, numeric `1/2`, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
`by`	Optional grouping variable for separate Kaplan-Meier summaries. Quoted and bare names are accepted.
`digits`	Number of digits for survival time summaries.
`format`	Output format. One of `"flextable"` (default), `"gt"`, or `"tibble"`.
`theme`	Table styling preset.

Details

Median survival is estimated from `survival::survfit()`. If the median survival time is not reached during follow-up, the display table shows `"Not reached"`.

Value

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

`table` A `flextable`, `gt_tbl`, or `NULL` when `format = "tibble"`.

`table_body` Tibble with numeric Kaplan-Meier summaries.

`table_display` Display data frame used to render the table.

`fit` Fitted `survfit` object.

`time,event,by,format,source` Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

survival_summary(
  data = lung_data,
  time = time,
  event = status,
  by = trt
)

survival_summary(
  data = lung_data,
  time = "time",
  event = "status",
  format = tibble
)
```

`surv_model_compare` *Compare parametric survival model distributions*

Description

Fit the same parametric survival regression model using multiple distributions and compare model-fit statistics such as AIC and BIC.

Usage

```
surv_model_compare(
  data,
  time,
  event,
  exposures,
  adjust_for = NULL,
  distributions = c("weibull", "exponential", "lognormal", "loglogistic"),
  digits = 2,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`data`	A `data.frame` containing survival time, event status, and predictor variables.
`time`	Survival follow-up time. Quoted and bare names are accepted.

event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
exposures	Character vector of main exposure variable names. Quoted names are recommended in scripts, and bare names are also accepted.
adjust_for	Optional character vector of adjustment variables. These variables are included in every candidate model.
distributions	Parametric survival distributions to compare. Defaults to c("weibull", "exponential", "lognormal", "loglogistic"). Quoted and bare values are accepted. Common spellings such as "log-normal" and "log-logistic" are also accepted.
digits	Number of digits for displayed model statistics.
format	Output format. One of "flextable" (default), "gt", or "tibble".
theme	Table styling preset.

Details

The same model formula is fitted for each candidate distribution using survival::survreg(). Lower AIC or BIC values indicate better relative model fit among the compared distributions. These statistics should be used with clinical judgment and visual checks; they do not prove that a distribution is scientifically correct.

Value

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

table A flextable, gt_tbl, or NULL when format = "tibble".

table_body Tibble with model-fit statistics.

table_display Display data frame used to render the table.

models Named list of fitted survreg models.

time,event,exposures,adjust_for,distributions,format,source Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))
lung_data$prior <- factor(lung_data$prior, levels = c(0, 10),
                          labels = c("No", "Yes"))

surv_model_compare(
  data = lung_data,
  time = time,
  event = status,
  exposures = c("trt", "celltype", "prior"),
  adjust_for = c("age", "karno")
)

surv_model_compare(
```

```
  data = lung_data,
  time = "time",
  event = "status",
  exposures = c(trt, prior),
  distributions = c(weibull, "log-logistic"),
  format = tibble
)
```

---

surv_predict *Predict survival probabilities from a parametric survival model*

Description

Estimate model-based survival probabilities at user-specified follow-up times from a fitted `survival::survreg()` model.

Usage

```
surv_predict(
  model,
  newdata = NULL,
  times,
  digits = 1,
  format = c("flextable", "gt", "tibble"),
  theme = c("minimal")
)
```

Arguments

`model`	A fitted `survreg` model, or a `surv_reg()` object containing exactly one fitted model.
`newdata`	Optional `data.frame` of profiles for prediction. If `NULL`, one typical profile is built from the model data using medians for numeric variables and the most common level for categorical variables.
`times`	Numeric vector of follow-up times at which survival probability should be predicted.
`digits`	Number of digits for displayed follow-up times and probabilities.
`format`	Output format. One of `"flextable"` (default), `"gt"`, or `"tibble"`.
`theme`	Table styling preset.

Details

`surv_predict()` is for parametric survival regression models fitted by `surv_reg()` or `survival::survreg()`. It is not a Kaplan-Meier estimate and it is not a Cox prediction helper.

Supported distributions are `"weibull"`, `"exponential"`, `"lognormal"`, and `"loglogistic"`, matching `surv_reg()`. Predictions are conditional on the supplied profile and the chosen parametric distribution.

Value

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

`table` A `flextable`, `gt_tbl`, or `NULL` when `format = "tibble"`.

`table_body` Tibble with predicted survival probabilities.

`table_display` Display data frame used to render the table.

`model` The fitted `survreg` model used for prediction.

`newdata,times,distribution,format,source` Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))

fit <- surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = "trt",
  adjust_for = c("age", "karno"),
  distribution = weibull
)

surv_predict(
  model = fit$models$trt,
  newdata = data.frame(
    trt = factor("Test", levels = levels(lung_data$trt)),
    age = 60,
    karno = 70
  ),
  times = c(90, 180, 365)
)

surv_predict(
  model = fit,
  times = c(90, 180),
  format = tibble
)
```

---

surv_reg *Parametric survival regression*

Description

Fit parametric survival models and report time ratios.

Usage

```
surv_reg(
  data,
  time,
  event,
  exposures,
  adjust_for = NULL,
  stratifier = NULL,
  interaction = NULL,
  multivariable = FALSE,
  multivariate = NULL,
  distribution = "weibull",
  format = c("flextable", "gt"),
  theme = c("minimal"),
  show_sample = "both",
  model_stats = FALSE,
  show_ref = TRUE
)
```

Arguments

data	A data.frame containing survival time, event status, and exposure variables.
time	Survival follow-up time. Quoted and bare names are accepted.
event	Event indicator. Quoted and bare names are accepted. Numeric 0/1, numeric 1/2, logical, character, and factor variables are accepted. For two-level character or factor variables, the second level is treated as the event.
exposures	Character vector of exposure variable names. Quoted names are recommended in scripts, and bare names are also accepted.
adjust_for	Optional character vector of adjustment variables. When supplied, one adjusted model is fitted per exposure.
stratifier	Optional single stratifying variable. When supplied, stratum-specific parametric survival tables are produced using the same crude, adjusted, or multivariable workflow requested by the other arguments. The stratifier cannot also be used as the time, event, exposure, adjustment, or interaction variable.
interaction	Optional character scalar specifying one interaction term using standard formula syntax, e.g. "trt*prior". Quoted and bare interaction syntax are accepted. In exposure-by-exposure mode, supply a single exposure; in multivariable = TRUE mode, the interaction is added to the single multivariable model.
multivariable	Logical; if FALSE (default), the current exposure-by-exposure workflow is used. If TRUE, one multivariable parametric survival model is fitted using all variables in exposures, and all exposure coefficients are reported.
multivariate	Optional logical alias for multivariable. This is accepted for convenience; multivariable is used internally.
distribution	Parametric survival distribution. One of "weibull", "exponential", "lognormal", or "loglogistic". Quoted and bare values are accepted. Common spellings such as "log-normal" and "log-logistic" are also accepted.

format	Output table format; one of "flextable" (default) or "gt".
theme	Table styling preset.
show_sample	For stratified parametric survival tables (stratifier supplied), controls which sample-size columns are shown in the publication table. One of "both" (default; N and events), "n", "events", or "none". Model statistics, when requested, still retain both N and event counts. This argument is ignored when stratifier is not supplied: non-stratified tables always show the model N column and report the number of events in a footnote.
model_stats	Logical; if TRUE, extract model-fit statistics including AIC, BIC, log-likelihood, scale, number of events, and N.
show_ref	Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

Details

surv_reg() fits accelerated failure time style parametric survival models using survival::survreg(). The exponentiated coefficient is displayed as a time ratio. A time ratio above 1 suggests longer survival time; a time ratio below 1 suggests shorter survival time, conditional on the chosen distribution.

By default, surv_reg() keeps the exposure-by-exposure workflow: without adjust_for, one crude model is fitted per exposure; with adjust_for, one adjusted model is fitted per exposure and only the exposure estimate is reported.

With multivariable = TRUE, all variables in exposures are included in one parametric survival model and all coefficients are reported. Since these estimates are adjusted for the other variables in the same model, the table reports Adjusted Time Ratio (95% CI). The adjust_for argument is not used in this mode; include every variable that belongs in the model inside exposures.

Interaction terms specified via interaction are included using standard formula expansion (for example, trt*prior). Interaction effects are displayed as additional rows beneath the corresponding exposure.

Stratified parametric survival tables show both N and event counts by default (show_sample = "both"). Use show_sample = "n", show_sample = "events", or show_sample = "none" to control the displayed sample columns. All counts are taken from the fitted model within each stratum. When models within a stratum use different numbers of observations, the footnote reports the range. Rows with a missing stratifier value are excluded and counted in a footnote. show_sample has no effect without stratifier.

Complete-case analysis is used: each model uses the rows that are complete for its own variables. Unused factor levels are dropped for each fitted model, so the reference category shown as "Ref." is the first level present in the data used by that model. Ordered factors are analysed with treatment (category-versus-reference) contrasts. Data in which every observation is an event are accepted; follow-up times must be greater than zero.

If exposure variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table.

Value

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

table A `flextable` or `gt_tbl`.

table_body Data frame of time ratios and confidence intervals.

table_display Data frame used to render the publication table.

models List of fitted `survreg` models.

model_summaries Summary output for the fitted models.

model_stats Model-fit statistics when `model_stats = TRUE`; otherwise `NULL`.

variable_labels Named character vector of display labels.

footnotes Character vector used by the rendered table. Adjustment notes use the same display labels as the table.

time,event,distribution,approach,format,source,adjust_for,exposures,interaction Metadata fields.

Examples

```
lung_data <- data_lungcancer
lung_data$trt <- factor(lung_data$trt, levels = c(1, 2),
                        labels = c("Standard", "Test"))
lung_data$prior <- factor(lung_data$prior, levels = c(0, 10),
                          labels = c("No", "Yes"))

surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c("trt", "celltype", "karno", "age")
)

surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, celltype, prior),
  adjust_for = c(age, karno),
  distribution = lognormal
)

# Interaction in an adjusted exposure model
surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = trt,
  adjust_for = c(age, karno),
  interaction = trt*prior,
  distribution = weibull
)
```

```
surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, celltype, prior, age, karno),
  distribution = weibull,
  multivariable = TRUE
)

# multivariate is accepted as an alias
surv_reg(
  data = lung_data,
  time = time,
  event = status,
  exposures = c(trt, age, karno),
  multivariate = TRUE
)
```

`uni_reg` *Univariable regression*

Description

Fit one model per exposure and return a clean regression table in **flextable** or **gt** format.

Usage

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

Arguments

`data`	A data frame containing the outcome and exposure variables.
`outcome`	Character scalar; outcome column name. Quoted and bare names are accepted. For binary-outcome approaches (`"logit"`, `"firth"`, `"logbinomial"`, `"robpoisson"`) the outcome may be numeric 0/1 or 1/2 (2 is the event; integer or double), logical (`TRUE` is the event), or a two-level factor or character variable (the second level is the event, which is reported in a message). The same factor, character and logical coding is accepted by `"poisson"`.

exposures	Character vector; exposure column names. Quoted names are recommended in scripts, and bare names are also accepted.
approach	Regression approach. One of "logit", "firth", "logbinomial", "poisson", "robpoisson", "linear", or "negbin". Use "firth" for Firth penalized logistic regression, especially with sparse cells or separation.
format	One of "flextable" (default) or "gt".
theme	Preset name (e.g. "minimal", "striped", "clinical", "shaded", "jama") or primitives c("plain","zebra","lines","labels_bold","compact","header_shaded")
model_stats	Logical; if TRUE, extract model-fit statistics such as AIC, BIC, log-likelihood, deviance, pseudo R-squared for non-linear models, and R-squared for linear models. Statistics are stored in the returned object's model_stats element and are not added to the publication table.
show_ref	Logical; if TRUE (default), display reference-category rows as "Ref.". If FALSE, hide reference rows; a message reminds users to use show_ref = TRUE when reference rows are needed.

Details

Use this when you want a quick crude association table before building an adjusted model. The fitted models are kept in the returned object, so the formatted table does not hide the underlying analysis.

If exposure variables have a "label" attribute, for example from labelled::var_label(), those labels are used automatically in the displayed table and plots. Internal matching still uses the original column names.

Confidence intervals are Wald intervals; for approach = "linear" they use the t distribution with the residual degrees of freedom (as confint() for lm), consistent with the t-based p-values. Factor exposures are always fitted with treatment contrasts (each level versus the reference), whatever options(contrasts =) or a contrasts attribute says. An exposure whose model cannot be fitted is kept in the table with a dash and named in a footnote and in failed_exposures.

Value

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

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

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.

model_stats Model-fit statistics when model_stats = TRUE; otherwise NULL.

variable_labels Named character vector of display labels used for exposure variables.

reg_check Diagnostics for linear models. Printing this element renders a publication-ready table; individual diagnostic rows remain available for programmatic use. Other approaches return a message.

approach, format, source Metadata fields.

Examples

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

endometrial_data <- data_endometrial
endometrial_data$HG <- factor(endometrial_data$HG, levels = c(0, 1))
endometrial_data$NV <- factor(endometrial_data$NV, levels = c(0, 1))
uni_reg(endometrial_data, HG, c(NV, PI, EH), approach = firth, format = gt)$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 is not present.

Usage

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

Arguments

x	A gtregression object.
name	Field name to access.

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)

Value

The requested element of x, or NULL if x has no element of that name. engine and format are treated as aliases of each other.

Index

∗ **datasets**
    data_birthwt, 20
    data_competing_risk, 21
    data_diabetes_mediation, 22
    data_endometrial, 23
    data_epilepsy, 23
    data_gt_quin, 24
    data_infertility, 25
    data_lungcancer, 25
    data_SynthDiabetes, 26
$.gtregression, 100

check_collinearity, 4
check_convergence, 5
check_ph, 6
compare_models, 8
competing_risk_reg, 10
conditional_logit_reg, 12
cox_reg, 13
cuminc_plot, 18

data_birthwt, 20
data_competing_risk, 21
data_diabetes_mediation, 22
data_endometrial, 23
data_epilepsy, 23
data_gt_quin, 24
data_infertility, 25
data_lungcancer, 25
data_SynthDiabetes, 26
descriptive_table, 27
dissect, 28

forest_df, 29
forest_reg, 30

gtregression_addin, 33, *34*
gtregression_app, *33*, 34

identify_confounder, 35, *40, 41*
import_data, 38
interaction_models, *37*, 39

km_plot, 41
km_risk_table, 44

logrank_test, 46

mediation_analysis, 47
merge_tables, 49, *72*
modify_table, 51, *72*
multi_reg, 56, *73*
multilevel_reg, 53
multinomial_reg, 55

ordinal_reg, 59

plot_cox_survival, 60
plot_mediation, 62
plot_model_fit, 62
plot_reg, 64
plot_reg_combine, 65
plot_surv_fit, 67
print.gtregression, 69

risk_difference, 70
rmst_table, 73

save_docx, 75
save_forest, 76
save_plot, 78
save_table, *72*, 79
select_models, 80
stratified_multi_reg, 82
stratified_uni_reg, 84
surv_model_compare, 91
surv_predict, 93
surv_reg, 94
survival_prob, 86
survival_quantiles, 88
survival_summary, 90

uni_reg, *73*, 98