Package ‘gtsummary’

April 16, 2020

Title  Presentation-Ready Data Summary and Analytic Result Tables

Version 1.3.0

Description  Creates presentation-ready tables summarizing data sets, regression models, and more. The code to create the tables is concise and highly customizable. Data frames can be summarized with any function, e.g. mean(), median(), even user-written functions. Regression models are summarized and include the reference rows for categorical variables. Common regression models, such as logistic regression and Cox proportional hazards regression, are automatically identified and the tables are pre-filled with appropriate column headers.

License  MIT + file LICENSE

URL  https://github.com/ddsjoberg/gtsummary,
     http://www.danielsjoberg.com/gtsummary/

BugReports  https://github.com/ddsjoberg/gtsummary/issues

Depends  R (>= 3.4)

Imports  broom (>= 0.5.5),
          broom.mixed (>= 0.2.4),
          crayon (>= 1.3.4),
          dplyr (>= 0.8.5),
          forcats (>= 0.5.0),
          glue (>= 1.4.0),
          gt (>= 0.2.0.5),
          knitr (>= 1.28),
          lifecycle (>= 0.2.0),
          magrittr (>= 1.5),
          purrr (>= 0.3.3),
          rlang (>= 0.4.5),
          stringr (>= 1.4.0),
          survival,
          tibble (>= 3.0.0),
          tidyr (>= 1.0.2),
          tidyselect (>= 1.0.0),
          usethis (>= 1.5.1)

Suggests  car,
          covr,
R topics documented:

flextable,  
geepack,  
Hmisc,  
kableExtra,  
lme4,  
pkgdown,  
rmarkdown,  
scales,  
spelling,  
testthat

VignetteBuilder  knitr
RdMacros  lifecycle
Encoding  UTF-8
Language  en-US
LazyData  true
Roxygen  list(markdown = TRUE)
RoxygenNote  7.1.0

R topics documented:

add_global_p . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
add_global_p.tbl_regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
add_global_p.tbl_uvregression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
add_n . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
add_nevent . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
add_nevent.tbl_regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
add_nevent.tbl_uvregression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
add_overall . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
add_p . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
add_p.tbl_cross . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
add_p.tbl_summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
add_q . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
add_stat_label . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
as_flextable . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
as_gt . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
as_kable . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
as_kable_extra . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
as_tibble.gtsummary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
bold_italicize_labels_levels . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
bold_p . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
combine_terms . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
gtsummary_logo . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
inline_text . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
inline_text.tbl_cross . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
inline_text.tbl_regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
inline_text.tbl_summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
inline_text.tbl_survfit . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30
inline_text.tbl_uvregression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
modify_header . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
print_gtsummary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
add_global_p

select Helpers

sort_p

style_percent

style_pvalue

style_ratio

style_sigfig

tbl_cross

tbl_merge

tbl_regression

tbl_stack

tbl_summary

tbl_survfit

tbl_uvregression

Index

add_global_p

Adds the global p-value for a categorical variables

Description

This function uses car::Anova with argument type = "III" to calculate global p-values for categorical variables. Output from tbl_regression and tbl_uvregression objects supported.

Usage

add_global_p(x, ...)

Arguments

x

tbl_regression or tbl_uvregression object

...

Further arguments passed to or from other methods.

Note

If a needed class of model is not supported by car::Anova, please create a GitHub Issue to request support.

Author(s)

Daniel D. Sjoberg

See Also

add_global_p.tbl_regression, add_global_p.tbl_uvregression
add_global_p.tbl_regression

Adds the global p-value for categorical variables

Description

This function uses `car::Anova` with argument `type = "III"` to calculate global p-values for categorical variables.

Usage

```r
## S3 method for class 'tbl_regression'
add_global_p(
  x,
  include = x$table_body$variable[x$table_body$var_type %in% c("categorical", "interaction")],
  keep = FALSE,
  terms = NULL,
  ...
)
```

Arguments

- **x**: Object with class `tbl_regression` from the `tbl_regression` function
- **include**: Variables to calculate global p-value for. Input may be a vector of quoted or unquoted variable names. tidyselect and gtsummary select helper functions are also accepted. Default is `NULL`, which adds global p-values for all categorical and interaction terms.
- **keep**: Logical argument indicating whether to also retain the individual p-values in the table output for each level of the categorical variable. Default is `FALSE`.
- **terms**: DEPRECATED. Use `include=` argument instead.
- **...**: Additional arguments to be passed to `car::Anova`

Value

A `tbl_regression` object

Note

If a needed class of model is not supported by `car::Anova`, please create a GitHub Issue to request support.

Example Output

Author(s)

Daniel D. Sjoberg
See Also

Other tbl_regression tools: add_nevent.tbl_regression(), add_q(), bold_italicize_labels_levels, combine_terms(), inline_text.tbl_regression(), modify_header(), tbl_merge(), tbl_regression(), tbl_stack()

Examples

tbl_lm_global_ex1 <-
    lm(marker ~ age + grade, data) %>%
    tbl_regression() %>%
    add_global_p()

add_global_p.tbl_uvregression

  Adds the global p-value for categorical variables

Description

This function uses car::Anova with argument type = "III" to calculate global p-values for categorical variables.

Usage

## S3 method for class 'tbl_uvregression'
add_global_p(x, ...)

Arguments

  x

    Object with class tbl_uvregression from the tbl_uvregression function

  ...      Additional arguments to be passed to car::Anova.

Value

A tbl_uvregression object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_uvregression tools: add_nevent.tbl_uvregression(), add_q(), bold_italicize_labels_levels, inline_text.tbl_uvregression(), modify_header(), tbl_merge(), tbl_stack(), tbl_uvregression()
Examples

tbl_uv_global_ex2 <-
  trial[c("response", "trt", "age", "grade")] %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>%
  add_global_p()

add_n

Add column with N

Description

For each variable in a tbl_summary table, the add_n function adds a column with the total number of non-missing (or missing) observations

Usage

add_n(x,
  statistic = "{n}",
  col_label = "**N**",
  footnote = FALSE,
  last = FALSE,
  missing = NULL
)

Arguments

x Object with class tbl_summary from the tbl_summary function

statistic String indicating the statistic to report. Default is the number of non-missing observation for each variable, statistic = "{n}". Other statistics available to report include:
  • "{N}" total number of observations,
  • "{n}" number of non-missing observations,
  • "{n_miss}" number of missing observations,
  • "{p}" percent non-missing data,
  • "{p_miss}" percent missing data The argument uses glue::glue syntax and multiple statistics may be reported, e.g. statistic = "{n} / {N} ({p}%)"

col_label String indicating the column label. Default is "**N**"

footnote Logical argument indicating whether to print a footnote clarifying the statistics presented. Default is FALSE

last Logical indicator to include N column last in table. Default is FALSE, which will display N column first.

missing DEPRECATED. Logical argument indicating whether to print N (missing = FALSE), or N missing (missing = TRUE). Default is FALSE
Value

A `tbl_summary` object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_summary` tools: `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Examples

tbl_n_ex <-
  trial[, c("trt", "age", "grade", "response")] %>%
  tbl_summary(by = trt) %>%
  add_n()

Description

Adds a column of the number of events to tables created with `tbl_regression` or `tbl_uvregression`. Supported model types include GLMs with binomial distribution family (e.g. `stats::glm`, `lme4::glmer`, and `geepack::geeglm`) and Cox Proportion Hazards regression models (`survival::coxph`).

Usage

`add_nevent(x, ...)`

Arguments

- `x` `tbl_regression` or `tbl_uvregression` object
- `...` Additional arguments passed to or from other methods.

Author(s)

Daniel D. Sjoberg

See Also

`add_nevent.tbl_regression`, `add_nevent.tbl_uvregression`, `tbl_regression`, `tbl_uvregression`
Description

This function adds a column of the number of events to tables created with tbl_regression. Supported model types include GLMs with binomial distribution family (e.g. stats::glm, lme4::glmer, and geepack::geeglm) and Cox Proportion Hazards regression models (survival::coxph).

The number of events is added to the internal .$table_body tibble, and not printed in the default output table (similar to N). The number of events is accessible via the inline_text function for printing in a report.

Usage

```r
## S3 method for class 'tbl_regression'
add_nevent(x, ...)
```

Arguments

- `x` tbl_regression object
- `...` Not used

Value

A tbl_regression object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_regression tools: add_global_p.tbl_regression(), add_q(), bold_italicize_labels_levels(), combine_terms(), inline_text.tbl_regression(), modify_header(), tbl_merge(), tbl_regression(), tbl_stack()
Description

Adds a column of the number of events to tables created with tbl_uvregression. Supported model types include GLMs with binomial distribution family (e.g. stats::glm, lme4::glmer, and geepack::geeglm) and Cox Proportion Hazards regression models (survival::coxph).

Usage

```r
## S3 method for class 'tbl_uvregression'
add_nevent(x, ...)
```

Arguments

- `x`: tbl_uvregression object
- `...`: Not used

Value

A tbl_uvregression object

Reporting Event N

The number of events is added to the internal `.table_body` tibble, and printed to the right of the N column. The number of events is also accessible via the `inline_text` function for printing in a report.

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_uvregression tools: `add_global_p.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

Examples

```r
tbl_uv_nevent_ex <-
trial[c("response", "trt", "age", "grade")]
  %>%
tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial)
  ) %>%
  add_nevent()
```
add_overall

Add column with overall summary statistics

Description

Adds a column with overall summary statistics to tables created by tbl_summary.

Usage

add_overall(x, last = FALSE)

Arguments

x  
Object with class tbl_summary from the tbl_summary function

last  
Logical indicator to display overall column last in table. Default is FALSE, which will display overall column first.

Value

A tbl_summary object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_summary tools: add_n(), add_p.tbl_summary(), add_q(), add_stat_label(), bold_italicize_labels_levels(), inline_text.tbl_summary(), inline_text.tbl_survfit(), modify_header(), tbl_merge(), tbl_stack(), tbl_summary()

Examples

tbl_overall_ex <-
  trial[c("age", "response", "grade", "trt")]
  tbl_summary(by = trt)
  add_overall()
add_p

Adds p-values to gtsummary table

Description

Adds p-values to gtsummary table

Usage

add_p(x, ...)

Arguments

x  Object created from a gtsummary function
...

...  Additional arguments passed to other methods.

Author(s)

Daniel D. Sjoberg

See Also

add_p.tbl_summary, add_p.tbl_cross

add_p.tbl_cross

Adds p-value to crosstab table

Description

Experimental  Calculate and add a p-value comparing the two variables in the cross table.

Usage

## S3 method for class 'tbl_cross'
add_p(x, test = NULL, pvalue_fun = NULL, source_note = FALSE, ...)

Arguments

x  Object with class tbl_cross from the tbl_cross function
test  A string specifying statistical test to perform. Default is "chisq.test" when expected cell counts >=5 and "fisher.test" when expected cell counts <5.
pvalue_fun  Function to round and format p-value. Default is style_pvalue, except when source_note = TRUE when the default is style_pvalue(x, prepend_p = TRUE)
source_note  Logical value indicating whether to show p-value in the {gt} table source notes rather than a column.
...

...  Not used
Example Output

Author(s)

Karissa Whiting

See Also

Other tbl_cross tools: inline_text.tbl_cross(), tbl_cross()

Examples

```r
code
add_p_cross_ex1 <-
  trial %>%
  tbl_cross(row = stage, col = trt) %>%
  add_p()

add_p_cross_ex2 <-
  trial %>%
  tbl_cross(row = stage, col = trt) %>%
  add_p(source_note = TRUE)
```

---

**add_p.tbl_summary**

 Adds p-values to summary tables

---

**Description**

Adds p-values to tables created by tbl_summary by comparing values across groups.

**Usage**

```r
code
## S3 method for class 'tbl_summary'
add_p(
  x,
  test = NULL,
  pvalue_fun = NULL,
  group = NULL,
  include = everything(),
  exclude = NULL,
  ...
)
```

**Arguments**

- `x` Object with class tbl_summary from the tbl_summary function
- `test` List of formulas specifying statistical tests to perform, e.g. `list(all_continuous() ~ "t.test", all_categorical() ~ "fisher.test")`. Options include
  - "t.test" for a t-test,
  - "aov" for a one-way ANOVA test,
• "wilcox.test" for a Wilcoxon rank-sum test,
• "kruskal.test" for a Kruskal-Wallis rank-sum test,
• "chisq.test" for a chi-squared test of independence,
• "chisq.test.no.correct" for a chi-squared test of independence without continuity correction,
• "fisher.test" for a Fisher's exact test,
• "lme4" for a random intercept logistic regression model to account for clustered data, lme4::glmer(by ~ variable + (1 | group), family = binomial).

The by argument must be binary for this option.

Tests default to "kruskal.test" for continuous variables, "chisq.test" for categorical variables with all expected cell counts >=5, and "fisher.test" for categorical variables with any expected cell count <5. A custom test function can be added for all or some variables. See below for an example.

pvalue_fun Function to round and format p-values. Default is style_pvalue. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. pvalue_fun = function(x) style_pvalue(x,digits = 2) or equivalently, purrr::partial(style_pvalue,digits = 2)).

group Column name (unquoted or quoted) of an ID or grouping variable. The column can be used to calculate p-values with correlated data (e.g. when the test argument is "lme4"). Default is NULL. If specified, the row associated with this variable is omitted from the summary table.

include Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or tidyselect select helper functions. Default is everything().

exclude DEPRECATED

... Not used

Value

A tbl_summary object

Setting Defaults

If you like to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, `.Rprofile`. The default confidence level can also be set. Please note the default option for the estimate is the same as it is for tbl_regression().

• options(gtsummary.pvalue_fun = new_function)

Example Output

Author(s)

Emily C. Zabor, Daniel D. Sjoberg
See Also

See tbl_summary vignette for detailed examples

Other tbl_summary tools: add_n(), add_overall(), add_q(), add_stat_label(), bold_italicize_labels_levels(), inline_text.tbl_summary(), inline_text.tbl_survfit(), modify_header(), tbl_merge(), tbl_stack(), tbl_summary()

Examples

```r
add_p_ex1 <-
  trial[c("age", "grade", "response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p()
# Conduct a custom McNemar test for response,
# Function must return a named list of the p-value and the
# test name: list(p = 0.123, test = "McNemar's test")
# The '...' must be included as input
# This feature is experimental, and the API may change in the future
my_mcnemar <- function(data, variable, by, ...) {
  result <- list()
  result$p <- stats::mcnemar.test(data[[variable]], data[[by]])$p.value
  resultitest <- "McNemar's test"
  result
}
add_p_ex2 <-
  trial[c("response", "trt")]) %>%
  tbl_summary(by = trt) %>%
  add_p(test = response ~ "my_mcnemar")
```

**add_q**

*Add a column of q-values to account for multiple comparisons*

**Description**

Adjustments to p-values are performed with stats::p.adjust.

**Usage**

add_q(x, method = "fdr", pvalue_fun = NULL)

**Arguments**

- **x**: a gtsummary object
- **method**: String indicating method to be used for p-value adjustment. Methods from stats::p.adjust are accepted. Default is method = "fdr".
- **pvalue_fun**: Function to round and format p-values. Default is style_pvalue. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. pvalue_fun = function(x) style_pvalue(x,digits = 2) or equivalently, purrr::partial(style_pvalue,digits = 2)).
**add_stat_label**

**Example Output**

**Author(s)**

Esther Drill, Daniel D. Sjoberg

**See Also**

Other tbl_summary tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_stat_label()`, `bold_italicize_labels_levels`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Other tbl_regression tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `bold_italicize_labels_levels`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`

Other tbl_uvregression tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `bold_italicize_labels_levels`, `inline_text.tbl_uvregression()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

**Examples**

```r
tbl_sum_q_ex1 <-
  trial[c("trt", "age", "grade", "response")]
  %>%
  tbl_summary(by = trt) %>
  add_p() %>
  add_q()

tbl_uv_q_ex2 <-
  trial[c("trt", "age", "grade", "response")]
  %>%
  tbl_uvregression(
    y = response,
    method = glm,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>
  add_global_p() %>
  add_q()
```

**Description**

Adds a column with labels describing the summary statistics presented for each variable in the `tbl_summary` table.

**Usage**

`add_stat_label(x)`

**Arguments**

- `x` Object with class tbl_summary from the tbl_summary function
Value

A tbl_summary object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_summary tools: add_n(), add_overall(), add_p.tbl_summary(), add_q(), bold_italicize_labels_levels(), inline_text.tbl_summary(), inline_text.tbl_survfit(), modify_header(), tbl_merge(), tbl_stack(), tbl_summary()

Examples

tbl_stat_ex <-
  trial[cb("trt", "age", "grade", "response")]
  tbl_summary() %>%
  add_stat_label()

Description

Experimental Function converts a gtsummary object to a flextable object. A user can use this function if they wish to add customized formatting available via the flextable functions. The flextable output is particularly useful when combined with R markdown with Word output, since the gt package does not support Word.

Usage

as_flextale(x, ...)

## S3 method for class 'gtsummary'
as_flextale(
x,
  include = everything(),
  return_calls = FALSE,
  strip_md_bold = TRUE,
  ...
)
as_flextable

Arguments

- **x**: Object created by a function from the gtsummary package (e.g. tbl_summary or tbl_regression)
- **...**: Not used
- **include**: Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything().
- **return_calls**: Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
- **strip_md_bold**: When TRUE, all double asterisk (markdown language for bold weight) in column labels and spanning headers are removed. Default is TRUE

Value

A flextable object

Details

The as_flextable() function takes the data frame that will be printed and converts it to a flextable and formats the table with the following flextable functions.

1. flextable::flextable()
2. flextable::set_header_labels() to set column labels
3. flextable::add_header_row(), if applicable, to set spanning column header
4. flextable::align() to set column alignment
5. flextable::padding() to indent variable levels
6. flextable::autofit() to estimate the column widths
7. flextable::footnote() to add table footnotes and source notes
8. flextable::bold() to bold cells in data frame
9. flextable::italic() to italicize cells in data frame

Any one of these commands may be omitted using the include= argument.

Pro tip: Use the flextable::width() function for exacting control over column width after calling as_flextable().

Author(s)

Daniel D. Sjoberg

See Also

Other gtsummary output types: as_gt(), as_kable_extra(), as_kable(), as_tibble.gtsummary()

Examples

```r
trial %>%
dplyr::select(trt, age, grade) %>%
tbl_summary(by = trt) %>%
add_p() %>%
as_flextable()
```
Convert gsummary object to a gt object

Description

Function converts a gsummary object to a gt_tbl object. Function is used in the background when the results are printed or knit. A user can use this function if they wish to add customized formatting available via the gt package.

Review the tbl_summary vignette or tbl_regression vignette for detailed examples in the 'Advanced Customization' section.

Usage

\[
\text{as_gt}(x, \text{include} = \text{everything}(), \text{return_calls} = \text{FALSE}, \text{exclude} = \text{NULL}, \text{omit} = \text{NULL})
\]

Arguments

\begin{itemize}
  \item \texttt{x} Object created by a function from the gsummary package (e.g. tbl_summary or tbl_regression)
  \item \texttt{include} Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gsummary select helper functions are also accepted. Default is everything().
  \item \texttt{return_calls} Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
  \item \texttt{exclude} DEPRECATED.
  \item \texttt{omit} DEPRECATED.
\end{itemize}

Value

A gt_tbl object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other gsummary output types: as_flextable(), as_kable_extra(), as_kable(), as_tibble.gsummary()
as_kable

Examples

```r
as_gt.ex <-
trial[c("trt", "age", "response", "grade")]
  tbl_summary(by = trt)
as_gt()
```

<table>
<thead>
<tr>
<th>as_kable</th>
<th>Convert gtsummary object to a kable object</th>
</tr>
</thead>
</table>

Description

Function converts a gtsummary object to a knitr_kable object. This function is used in the background when the results are printed or knit. A user can use this function if they wish to add customized formatting available via knitr::kable.

Output from knitr::kable is less full featured compared to summary tables produced with gt. For example, kable summary tables do not include indentation, footnotes, or spanning header rows.

Usage

```r
as_kable(x, include = everything(), return_calls = FALSE, exclude = NULL, ...)
```

Arguments

- `x` Object created by a function from the gtsummary package (e.g. tbl_summary or tbl_regression)
- `include` Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is everything(), which includes all commands in x$kable_calls.
- `return_calls` Logical. Default is FALSE. If TRUE, the calls are returned as a list of expressions.
- `exclude` DEPRECATED
- `...` Additional arguments passed to knitr::kable

Details

Tip: To better distinguish variable labels and level labels when indenting is not supported, try bold_labels() or italicize_levels().

Value

A knitr_kable object

Author(s)

Daniel D. Sjoberg

See Also

Other gtsummary output types: as_flextable(), as_gt(), as_kable_extra(), as_tibble.gtsummary()
Examples

```r
trial %>%
tbl_summary(by = trt) %>%
bold_labels() %>%
as_kable()
```

\[as\_kable\_extra\]

Convert gtsummary object to a kableExtra object

\[Description\]

**Experimental** Function converts a gtsummary object to a knitr_kable + kableExtra object. A user can use this function if they wish to add customized formatting available via `knitr::kable` and kableExtra. Note that gtsummary uses the standard markdown ** to bold headers, and they may need to be changed manually with kableExtra output.

\[Usage\]

```r
as_kable_extra(
  x,
  include = everything(),
  return_calls = FALSE,
  strip_md_bold = TRUE,
  ...)
```

\[Arguments\]

- `x` Object created by a function from the gtsummary package (e.g. `tbl_summary` or `tbl_regression`)
- `include` Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is `everything()`, which includes all commands in `x$kable_calls`.
- `return_calls` Logical. Default is `FALSE`. If `TRUE`, the calls are returned as a list of expressions.
- `strip_md_bold` When `TRUE`, all double asterisk (markdown language for bold weight) in column labels and spanning headers are removed. Default is `TRUE`.
- `...` Additional arguments passed to `knitr::kable`

\[Value\]

A kableExtra object

\[Author(s)\]

Daniel D. Sjoberg

\[See Also\]

Other gtsummary output types: `as_flextable()`, `as_gt()`, `as_kable()`, `as_tibble.gtsummary()`
Examples

```r
tbl <-
    trial %>%
    tbl_summary(by = trt) %>%
    as_kable_extra()
```

Description

Function converts gtsummary objects to tibbles. The formatting stored in `x$kable_calls` is applied.

Usage

```r
## S3 method for class 'gtsummary'
as_tibble(
x, 
include = everything(),
col_labels = TRUE,
return_calls = FALSE,
exclude = NULL,
...)
```

Arguments

- `x` Object created by a function from the gtsummary package (e.g. `tbl_summary` or `tbl_regression`)
- `include` Commands to include in output. Input may be a vector of quoted or unquoted names. tidyselect and gtsummary select helper functions are also accepted. Default is `everything()`, which includes all commands in `x$kable_calls`.
- `col_labels` Logical argument adding column labels to output tibble. Default is `TRUE`.
- `return_calls` Logical. Default is `FALSE`. If `TRUE`, the calls are returned as a list of expressions.
- `exclude` DEPRECATED
- `...` Not used

Value

a tibble

Author(s)

Daniel D. Sjoberg

See Also

Other gtsummary output types: `as_flextable()`, `as_gt()`, `as_kable_extra()`, `as_kable()`
Examples

tbl <-
trial %>%
dplyr::select(trt, age, grade, response) %>%
tbl_summary(by = trt)

as_tibble(tbl)

# without column labels
as_tibble(tbl, col_labels = FALSE)

Description

Bold or Italicize labels or levels in gtsummary tables

Usage

boldLabels(x)

bold_levels(x)

italicize_labels(x)

italicize_levels(x)

Arguments

x Object created using gtsummary functions

Value

Functions return the same class of gtsummary object supplied

Functions

• bold_labels: Bold labels in gtsummary tables
• bold_levels: Bold levels in gtsummary tables
• italicize_labels: Italicize labels in gtsummary tables
• italicize_levels: Italicize levels in gtsummary tables

Example Output

Author(s)

Daniel D. Sjoberg
**bold_p**

**Description**

Bold values below a chosen threshold (e.g. <0.05) in a gtsummary tables.

**Usage**

```r
bold_p(x, t = 0.05, q = FALSE)
```

**Arguments**

- `x` Object created using gtsummary functions
- `t` Threshold below which values will be bold. Default is 0.05.
- `q` Logical argument. When TRUE will bold the q-value column rather than the p-values. Default is FALSE.

**Example Output**

```r
tbl_bold_ital_ex <- trial[c("trt", "age", "grade")] %>%
tbl_summary() %>%
bold_labels() %>%
bold_levels() %>%
italicize_labels() %>%
italicize_levels()
```

**Author(s)**

Daniel D. Sjoeborg, Esther Drill
Examples

```r
tbl_sum_bold_p_ex <-
  trial[c("age", "grade", "response", "trt")]
  tbl_summary(by = trt)
  add_p()
  bold_p(t = 0.65)

tbl_lm_bold_p_ex <-
  glm(response ~ trt + grade, trial, family = binomial(link = "logit"))
  tbl_regression(exponentiate = TRUE)
  bold_p(t = 0.65)
```

combine_terms

Combine terms in a regression model

Description

**Experimental** The function combines terms from a regression model, and replaces the terms with a single row in the output table. The p-value is calculated using `stats::anova()`.

Usage

```r
combine_terms(x, formula_update, label = NULL, ...)
```

Arguments

- **x** a `tbl_regression` object
- **formula_update** formula update passed to the `stats::update`. This updated formula is used to construct a reduced model, and is subsequently passed to `stats::anova()` to calculate the p-value for the group of removed terms. See the `stats::update` help file for proper syntax. function's formula.= argument
- **label** Option string argument labeling the combined rows
- **...** Additional arguments passed to `stats::anova`

Value

`tbl_regression` object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels`, `inline_text.tbl_regression()`, `modify_header()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`
Examples

# fit model with nonlinear terms for marker
nlmod1 <- lm(
  age ~ marker + I(marker^2) + grade,
  trial[c("age", "marker", "grade")]) %>% na.omit() # keep complete cases only!
)

combine_terms_ex1 <-
  tbl_regression(nlmod1, label = grade ~ "Grade") %>%
  # collapse non-linear terms to a single row in output using anova
  combine_terms(
    formula_update = . ~ . - marker - I(marker^2),
    label = "Marker (non-linear terms)"
  )

# Example with Cubic Splines
library(Hmisc)
mod2 <- lm(
  age ~ rcspline.eval(marker, inclx = TRUE) + grade,
  trial[c("age", "marker", "grade")]) %>% na.omit() # keep complete cases only!
)

combine_terms_ex2 <-
  tbl_regression(mod2, label = grade ~ "Grade") %>%
  combine_terms(
    formula_update = . ~ . - rcspline.eval(marker, inclx = TRUE),
    label = "Marker (non-linear terms)"
  )

# Logistic Regression Example, LRT p-value
combine_terms_ex3 <-
  glm(
    response ~ marker + I(marker^2) + grade,
    trial[c("response", "marker", "grade")]) %>% na.omit(), # keep complete cases only!
    family = binomial)

  tbl_regression(label = grade ~ "Grade", exponentiate = TRUE) %>%
  # collapse non-linear terms to a single row in output using anova
  combine_terms(
    formula_update = . ~ . - marker - I(marker^2),
    label = "Marker (non-linear terms)",
    test = "LRT"
  )

---

gtsummary_logo

The gtsummary logo, using ASCII or Unicode characters

Description

Use crayon::strip_style() to get rid of the colors.

Usage

gtsummary_logo(unicode = l10n_info()$'UTF-8')
**inline_text.tbl_cross**

**Arguments**

- `unicode`: Whether to use Unicode symbols. Default is `TRUE` on UTF-8 platforms.

**Examples**

```r
gtsummary_logo()
```

---

**inline_text**  
*Report statistics from `gtsummary` tables inline*

**Description**

Report statistics from `gtsummary` tables inline

**Usage**

```r
inline_text(x, ...)
```

**Arguments**

- `x`: Object created from a `gtsummary` function
- `...`: Additional arguments passed to other methods.

**Value**

A string reporting results from a `gtsummary` table

**Author(s)**

Daniel D. Sjoberg

**See Also**

- `inline_text.tbl_summary`, `inline_text.tbl_regression`, `inline_text.tbl_uvregression`, `inline_text.tbl_survfit`

---

**inline_text.tbl_cross**  
*Report statistics from cross table inline*

**Description**

**Experimental** Extracts and returns statistics from a `tbl_cross` object for inline reporting in an R markdown document. Detailed examples in the `inline_text vignette`
Usage

## S3 method for class 'tbl_cross'
inline_text(
  x,
  col_level,
  row_level = NULL,
  pattern = NULL,
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE),
  ...
)

Arguments

x a tbl_cross object
col_level Level of the column variable to display. Can also specify "p.value" for the p-value and "stat_0" for Total column.
row_level Level of the row variable to display. Can also specify the 'Unknown' row. Default is NULL
pattern String indicating the statistics to return. Uses glue::glue formatting. Default is pattern shown in tbl_cross() output
pvalue_fun Function to round and format p-values. Default is style_pvalue. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. pvalue_fun = function(x) style_pvalue(x,digits = 2) or equivalently, purrr::partial(style_pvalue,digits = 2)).
... Not used

Value

A string reporting results from a gtsummary table

See Also

Other tbl_cross tools: add_p.tbl_cross(), tbl_cross()

Examples

tbl_cross <-
tbl_cross(trial, row = trt, col = response) %>%
  add_p()

inline_text(tbl_cross, row_level = "Drug A", col_level = "1")
inline_text(tbl_cross, row_level = "Total", col_level = "1")
inline_text(tbl_cross, col_level = "p.value")
Description

Takes an object with class tbl_regression, and the location of the statistic to report and returns statistics for reporting inline in an R markdown document. Detailed examples in the inline_text vignette.

Usage

```r
## S3 method for class 'tbl_regression'
inline_text(
  x,
  variable,
  level = NULL,
  pattern = "{estimate} ({conf.level*100}% CI {conf.low}, {conf.high}; {p.value})",
  estimate_fun = x$fmt_fun$estimate,
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE),
  ...
)
```

Arguments

- **x**: Object created from tbl_regression
- **variable**: Variable name of statistics to present
- **level**: Level of the variable to display for categorical variables. Default is NULL, returning the top row in the table for the variable.
- **pattern**: String indicating the statistics to return. Uses glue::glue formatting. Default is "{estimate} {(conf.level*100)% CI {conf.low}, {conf.high}; {p.value})". All columns from x$table_body are available to print as well as the confidence level (conf.level). See below for details.
- **estimate_fun**: function to style model coefficient estimates. Columns 'estimate', 'conf.low', and 'conf.high' are formatted. Default is x$input$estimate_fun
- **pvalue_fun**: function to style p-values and/or q-values. Default is function(x) style_pvalue(x, prepend_p = TRUE)
- **...**: Not used

Value

A string reporting results from a gtsummary table

Pattern argument

The following items are available to print. Use print(x$table_body) to print the table the estimates are extracted from.

- `{estimate}` coefficient estimate formatted with `estimate_fun`
• \{conf.low\} lower limit of confidence interval formatted with 'estimate_fun'
• \{conf.high\} upper limit of confidence interval formatted with 'estimate_fun'
• \{ci\} confidence interval formatted with x$estimate_fun
• \{p.value\} p-value formatted with 'pvalue_fun'
• \{N\} number of observations in model
• \{label\} variable/variable level label

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_regression tools: add_global_p.tbl_regression(), add_nevent.tbl_regression(), add_q(), bold_italicize_labels_levels, combine_terms(), modify_header(), tbl_merge(), tbl_regression(), tbl_stack()

Examples

inline_text.ex1 <-
glm(response ~ age + grade, trial, family = binomial(link = "logit")) %>%
tbl_regression(exponentiate = TRUE)

inline_text(inline_text.ex1, variable = age)
inline_text(inline_text.ex1, variable = grade, level = "III")

inline_text.tbl_summary
  Report statistics from summary tables inline

Description

Extracts and returns statistics from a tbl_summary object for inline reporting in an R markdown document. Detailed examples in the inline_text vignette

Usage

## S3 method for class 'tbl_summary'
inline_text(
  x,
  variable,
  column = NULL,
  level = NULL,
  pattern = NULL,
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE),
  ...
)
inline_text.tbl_survfit

Description

Experimental Extracts and returns statistics from a tbl_survfit object for inline reporting in an R markdown document. Detailed examples in the inline_text vignette.
## S3 method for class 'tbl_survfit'
inline_text(
  x,
  time = NULL,
  prob = NULL,
  level = NULL,
  estimate_fun = NULL,
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE),
  ...
)

### Arguments

- **x**: Object created from `tbl_survfit`
- **time**: time for which to return survival probabilities.
- **prob**: probability with values in (0,1)
- **level**: Level of the variable to display for categorical variables. Can also specify the 'Unknown' row. Default is `NULL`
- **estimate_fun**: Function to round and format coefficient estimates. Default is `style_sigfig` when the coefficients are not transformed, and `style_ratio` when the coefficients have been exponentiated.
- **pvalue_fun**: Function to round and format p-values. Default is `style_pvalue`. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. `pvalue_fun = function(x) style_pvalue(x, digits = 2)` or equivalently, `purrr::partial(style_pvalue, digits = 2)`).

### Value
A string reporting results from a gtsummary table

### Author(s)
Daniel D. Sjoberg

### See Also
Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels()`, `inline_text.tbl_summary()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

### Examples
```r
library(survival)
# fit survfit
fit1 <- survfit(Surv(ttdeath, death) ~ trt, trial)
fit2 <- survfit(Surv(ttdeath, death) ~ 1, trial)

# sumarize survfit objects
tbl1 <- tbl_survfit(}

```r
fit1,
times = c(12, 24),
label = "Treatment",
label_header = "**{time} Month**"
)

tbl2 <- tbl_survfit(
fit2,
probs = 0.5,
label_header = "**Median Survival**"
)

# report results inline
inline_text(tbl1, time = 24, level = "Drug B")
inline_text(tbl2, prob = 0.5)
```

**Description**

Extracts and returns statistics from a table created by the `tbl_uvregression` function for inline reporting in an R markdown document. Detailed examples in the `inline_text` vignette

**Usage**

```r
## S3 method for class 'tbl_uvregression'
inline_text(
  x,
  variable,
  level = NULL,
  pattern = "{estimate} ({conf.level*100}% CI {conf.low}, {conf.high}; {p.value})",
  estimate_fun = x$inputs$estimate_fun,
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE),
  ... 
)
```

**Arguments**

- `x` Object created from `tbl_uvregression`
- `variable` Variable name of statistics to present
- `level` Level of the variable to display for categorical variables. Default is `NULL`, returning the top row in the table for the variable.
- `pattern` String indicating the statistics to return. Uses `glue::glue` formatting. Default is "{estimate} ({conf.level*100}% CI {conf.low}, {conf.high}; {p.value})". All columns from `x$table_body` are available to print as well as the confidence level (`conf.level`). See below for details.
- `estimate_fun` function to style model coefficient estimates. Columns 'estimate', 'conf.low', and 'conf.high' are formatted. Default is `x$inputs$estimate_fun`
- `pvalue_fun` function to style p-values and/or q-values. Default is `function(x) style_pvalue(x, prepend_p = TRUE)`
- `...` Not used
modify_header

Modify column headers in gtsummary tables

Description

Column labels can be modified to include calculated statistics; e.g., the N can be dynamically included by wrapping it in curly brackets (following glue::glue syntax).

Usage

modify_header(x, stat_by = NULL, ..., text_interpret = c("md", "html"))

Value

A string reporting results from a gtsummary table

pattern argument

The following items are available to print. Use print(x$table_body) to print the table the estimates are extracted from.

- {estimate} coefficient estimate formatted with 'estimate_fun'
- {conf.low} lower limit of confidence interval formatted with 'estimate_fun'
- {conf.high} upper limit of confidence interval formatted with 'estimate_fun'
- {ci} confidence interval formatted with x$estimate_fun
- {p.value} p-value formatted with 'pvalue_fun'
- {N} number of observations in model
- {label} variable/variable level label

Author(s)

Daniel D. Sjoberg

See Also

Other tbl_uvregression tools: add_global_p.tbl_uvregression(), add_nevent.tbl_uvregression(), add_q(), bold_italicize_labels_levels, modify_header(), tbl_merge(), tbl_stack(), tbl_uvregression()

Examples

```r
inline_text_ex1 <-
  trial[c("response", "age", "grade")] %>%
tbl_uvregression(
  method = glm,
  method.args = list(family = binomial),
  y = response,
  exponentiate = TRUE
)
inline_text(inline_text_ex1, variable = age)
inline_text(inline_text_ex1, variable = grade, level = "III")
```
modify_header

Arguments

- `x`: `gtsummary` object, e.g. `tbl_summary` or `tbl_regression`
- `stat_by`: String specifying text to include above the summary statistics stratified by a variable. Only use with stratified `tbl_summary` objects. The following fields are available for use in the headers:
  - `{n}` number of observations in each group,
  - `{N}` total number of observations,
  - `{p}` percentage in each group,
  - `{level}` the 'by' variable level,
  - "fisher.test" for a Fisher's exact test,

Syntax follows `glue::glue`, e.g. `stat_by = "**{level}**,N = {n} {{style_percent(p)}%}"`. The `by` argument from the parent `tbl_summary()` cannot be NULL.

- `text_interpret`: Specifies column label of any other column in `.table_body`. Argument is the column name, and the value is the new column header (e.g. `p.value = "Model P-values"`). Use `print(x$table_body)` to see columns available.

- `text_interpret`: indicates whether text will be interpreted as markdown ("md") or HTML ("html"). The text is interpreted with the `gt` package's `md()` or `html()` functions. The default is "md", and is ignored when the print engine is not `gt`.

Value

Function return the same class of `gtsummary` object supplied

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_summary` tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels()`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `tbl_merge()`, `tbl_stack()`, `tbl_summary()`

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels()`, `combine_terms()`, `inline_text.tbl_regression()`, `tbl_merge()`, `tbl_regression()`, `tbl_stack()`, `tbl_summary()

Other `tbl_uvregression` tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels()`, `inline_text.tbl_uvregression()`, `tbl_merge()`, `tbl_stack()`, `tbl_uvregression()`

Other `tbl_survival` tools: `inline_text.tbl_survival()`, `tbl_survival.survfit()`

Examples

```r
tbl_col_ex1 <-
trial[,c("age", "grade", "response")]
%>%
tbl_summary() %>%
modify_header(stat_0 = "**All Patients**, N = (N)")
```
tbl_col_ex2 <-
trial[c("age", "grade", "response", "trt")]
%>%
tbl_summary(by = trt)
%>%
modify_header(
  stat_by = "**{level}**, N = {n} ({style_percent(p, symbol = TRUE))}" 
)

print_gtsummary

print and knit_print methods for gtsummary objects

Description

print and knit_print methods for gtsummary objects

Usage

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

## S3 method for class 'gtsummary'
knit_print(x, ...)

Arguments

x
An object created using gtsummary functions

... Not used

Author(s)

Daniel D. Sjoberg

See Also

tbl_summary tbl_regression tbl_uvregression tbl_merge tbl_stack

select_helpers

Select helper functions

Description

Set of functions to supplement the tidyselect set of functions for selecting columns of data frames. all_continuous(), all_categorical(), and all_dichotomous() may only be used with tbl_summary(), where each variable has been classified into one of these three groups. All other helpers are available throughout the package.
**Usage**

```r
all_continuous()

all_categorical(dichotomous = TRUE)

all_dichotomous()

all_numeric()

all_character()

all_integer()

all_double()

all_logical()

all_factor()
```

**Arguments**

- `dichotomous` Logical indicating whether to include dichotomous variables. Default is TRUE

**Value**

A character vector of column names selected

**Examples**

```r
select_ex1 <-
  trial %>%
  dplyr::select(age, response, grade) %>%
  tbl_summary(
    statistic = all_continuous() ~ "{mean} ({sd})",
    type = all_dichotomous() ~ "categorical"
  )
```

---

**sort_p**

Sort variables in table by ascending p-values

**Description**

Sort tables created by gtsummary by p-values

**Usage**

```r
sort_p(x, q = FALSE)
```

**Arguments**

- `x` An object created using gtsummary functions
- `q` Logical argument. When TRUE will sort by the q-value column
Example Output

Author(s)
Karissa Whiting

Examples

tbl_sum_sort_p_ex <-
  trial[c("age", "grade", "response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  sort_p()

tbl_lm_sort_p_ex <-
  glm(response ~ trt + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE) %>%
  sort_p()

style_percent

Style percentages to be displayed in tables or text

Description
Style percentages to be displayed in tables or text

Usage
style_percent(x, symbol = FALSE)

Arguments

  x      numeric vector of percentages

  symbol Logical indicator to include percent symbol in output. Default is FALSE.

Value
A character vector of styled percentages

Author(s)
Daniel D. Sjoberg

See Also
See Table Gallery vignette for example
Other style tools: style_pvalue(), style_ratio(), style_sigfig()

Examples

percent_vals <- c(-1, 0, 0.0001, 0.005, 0.01, 0.10, 0.45356, 0.99, 1.45)
style_percent(percent_vals)
style_percent(percent_vals, symbol = TRUE)
**style_pvalue**

*Style p-values to be displayed in tables or text*

**Description**

Style p-values to be displayed in tables or text

**Usage**

```r
style_pvalue(x, digits = 1, prepend_p = FALSE)
```

**Arguments**

- `x`: Numeric vector of p-values.
- `digits`: Number of digits large p-values are rounded. Must be 1 or 2. Default is 1.
- `prepend_p`: Logical. Should 'p=' be prepended to formatted p-value. Default is FALSE

**Value**

A character vector of styled p-values

**Author(s)**

Daniel D. Sjoberg

**See Also**

See tbl_summary vignette for examples

Other style tools: `style_percent()`, `style_ratio()`, `style_sigfig()`

**Examples**

```r
pvals <- c(
  1.5, 1, 0.999, 0.5, 0.25, 0.2, 0.197, 0.12, 0.10, 0.099, 0.06,
  0.03, 0.002, 0.001, 0.00099, 0.0002, 0.00002, -1
)
style_pvalue(pvals)
style_pvalue(pvals, digits = 2, prepend_p = TRUE)
```
Implement significant figure-like rounding for ratios

Description

When reporting ratios, such as relative risk or an odds ratio, we’ll often want the rounding to be similar on each side of the number 1. For example, if we report an odds ratio of 0.95 with a confidence interval of 0.70 to 1.24, we would want to round to two decimal places for all values. In other words, 2 significant figures for numbers less than 1 and 3 significant figures 1 and larger. `style_ratio()` performs significant figure-like rounding in this manner.

Usage

`style_ratio(x, digits = 2)`

Arguments

- `x` Numeric vector
- `digits` Integer specifying the number of significant digits to display for numbers below 1. Numbers larger than 1 will be be `digits + 1`. Default is `digits = 2`.

Value

A character vector of styled ratios

Author(s)

Daniel D. Sjoberg

See Also

Other style tools: `style_percent()`, `style_pvalue()`, `style_sigfig()`

Examples

```r
c( 0.123, 0.9, 1.1234, 12.345, 101.234, -0.123,  
   -0.9, -1.1234, -12.345, -101.234  
) %>%
style_ratio()
```
Implement significant figure-like rounding

Description

Converts a numeric argument into a string that has been rounded to a significant figure-like number. Scientific notation output is avoided, however, and additional significant figures may be displayed for large numbers. For example, if the number of significant digits requested is 2, 123 will be displayed (rather than 120 or 1.2x10^2).

Usage

style_sigfig(x, digits = 2)

Arguments

x Numeric vector
digits Integer specifying the minimum number of significant digits to display

Details

If 2 sig figs are input, the number is rounded to 2 decimal places when \( \text{abs}(x) < 1 \), 1 decimal place when \( \text{abs}(x) \geq 1 \) & \( \text{abs}(x) < 10 \), and to the nearest integer when \( \text{abs}(x) \geq 10 \).

Value

A character vector of styled numbers

Author(s)

Daniel D. Sjoberg

See Also

Other style tools: style_percent(), style_pvalue(), style_ratio()

Examples

c(0.123, 0.9, 1.1234, 12.345, -0.123, -0.9, -1.1234, -12.345, NA, -0.001) %>%
  style_sigfig()
**Description**

**Experimental** The function creates a cross table of two categorical variables.

**Usage**

```r
tbl_cross(
  data,
  row = NULL,
  col = NULL,
  label = NULL,
  statistic = NULL,
  percent = c("none", "column", "row", "cell"),
  missing = c("ifany", "always", "no"),
  missing_text = "Unknown",
  margin_text = "Total"
)
```

**Arguments**

- **data** A data frame
- **row** A column name in data to be used for columns of cross table.
- **col** A column name in data to be used for rows of cross table.
- **label** List of formulas specifying variables labels, e.g. `list(age ~ "Age, yrs", stage ~ "Path T Stage")`. If a variable's label is not specified here, the label attribute `attr(data$age, "label")` is used. If attribute label is `NULL`, the variable name will be used.
- **statistic** A string with the statistic name in curly brackets to be replaced with the numeric statistic (see `glue::glue`). The default is `{n}`). If percent argument is "column", "row", or "cell", default is `{n}` (p%).
- **percent** Indicates the type of percentage to return. Must be one of "none", "column", "row", or "cell". Default is "cell" when `{N}` or `{p}` is used in statistic.
- **missing** Indicates whether to include counts of NA values in the table. Allowed values are "no" (never display NA values), "ifany" (only display if any NA values), and "always" (includes NA count row for all variables). Default is "ifany".
- **missing_text** String to display for count of missing observations. Default is "Unknown".
- **margin_text** Text to display for margin totals. Default is "Total"

**Value**

A `tbl_cross` object

**Example Output**
tbl_merge

Merge two or more gtsummary objects

Description

Merges two or more tbl_regression, tbl_uvregression, tbl_stack, or tbl_summary objects and adds appropriate spanning headers.

Usage

```r
tbl_merge(tbls, tab_spanner = NULL)
```

Arguments

- `tbls` List of gtsummary objects to merge
- `tab_spanner` Character vector specifying the spanning headers. Must be the same length as `tbls`. The strings are interpreted with `gt::md`. Must be same length as `tbls` argument

Value

A `tbl_merge` object

Example Output

Author(s)

Daniel D. Sjoberg
See Also

tbl_stack

Other tbl_regression tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels()`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_header()`, `tbl_regression()`, `tbl_stack()`

Other tbl_uvregression tools: `add_global_p.tbl_uvregression()`, `add_nevent.tbl_uvregression()`, `add_q()`, `bold_italicize_labels_levels()`, `inline_text.tbl_uvregression()`, `modify_header()`, `tbl_stack()`, `tbl_uvregression()`

Other tbl_summary tools: `add_n()`, `add_overall()`, `add_p.tbl_summary()`, `add_q()`, `add_stat_label()`, `bold_italicize_labels_levels()`, `inline_text.tbl_summary()`, `inline_text.tbl_survfit()`, `modify_header()`, `tbl_stack()`, `tbl_summary()`

Examples

# Side-by-side Regression Models
library(survival)
t1 <-
glm(response ~ trt + grade + age, trial, family = binomial) %>%
tbl_regression(exponentiate = TRUE)
t2 <-
coxph(Surv(ttdeath, death) ~ trt + grade + age, trial) %>%
tbl_regression(exponentiate = TRUE)
tbl_merge_ex1 <-
tbl_merge(
tbls = list(t1, t2),
tab_spanner = c("**Tumor Response**", "**Time to Death**")
)

# Descriptive statistics alongside univariate regression, with no spanning header
t3 <-
trial[, c("age", "grade", "response")]
%>%
tbl_summary(missing = "no") %>%
add_n()
t4 <-
tbl_uvregression(
trial[, c("ttdeath", "death", "age", "grade", "response")],
method = coxph,
y = Surv(ttdeath, death),
exponentiate = TRUE,
hide_n = TRUE
)

tbl_merge_ex2 <-
tbl_merge(tbls = list(t3, t4)) %>%
as_gt(include = -tab_spanner) %>%
gt::cols_label(stat_0_1 = gt::md("**Summary Statistics**"))
Description

This function takes a regression model object and returns a formatted table that is publication-ready. The function is highly customizable allowing the user to obtain a bespoke summary table of the regression model results. Review the tbl_regression vignette for detailed examples.

Usage

tbl_regression(
  x,
  label = NULL,
  exponentiate = FALSE,
  include = everything(),
  show_single_row = NULL,
  conf.level = NULL,
  intercept = FALSE,
  estimate_fun = NULL,
  pvalue_fun = NULL,
  tidy_fun = NULL,
  show_yesno = NULL,
  exclude = NULL
)

Arguments

x Regression model object
label List of formulas specifying variables labels, e.g. list(age ~ "Age, yrs", stage ~ "Path T Stage")
exponentiate Logical indicating whether to exponentiate the coefficient estimates. Default is FALSE.
include Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or tidyselect select helper functions. Default is everything().
show_single_row By default categorical variables are printed on multiple rows. If a variable is dichotomous (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name(s) here—quoted and unquoted variable name accepted.
conf.level Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
intercept Logical argument indicating whether to include the intercept in the output. Default is FALSE.
estimate_fun Function to round and format coefficient estimates. Default is style_sigfig when the coefficients are not transformed, and style_ratio when the coefficients have been exponentiated.
pvalue_fun Function to round and format p-values. Default is style_pvalue. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. pvalue_fun = function(x) style_pvalue(x, digits = 2) or equivalently, purrr::partial(style_pvalue, digits = 2)).
tidy_fun Option to specify a particular tidier function if the model is not a vetted model or you need to implement a custom method. Default is NULL.
show_yesno  DEPRECATED
exclude     DEPRECATED

Value
A `tbl_regression` object

Setting Defaults
If you prefer to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, `.Rprofile`. The default confidence level can also be set.

- `options(gtsummary.pvalue_fun = new_function)`
- `options(gtsummary.tbl_regression.estimate_fun = new_function)`
- `options(gtsummary.conf.level = 0.90)`

Note
The N reported in the output is the number of observations in the data frame `model.frame(x)`. Depending on the model input, this N may represent different quantities. In most cases, it is the number of people or units in your model. Here are some common exceptions.

1. Survival regression models including time dependent covariates.
2. Random- or mixed-effects regression models with clustered data.
3. GEE regression models with clustered data.

This list is not exhaustive, and care should be taken for each number reported.

Example Output

Author(s)
Daniel D. Sjoberg

See Also
See `tbl_regression` vignette for detailed examples

Other `tbl_regression` tools: `add_global_p.tbl_regression()`, `add_nevent.tbl_regression()`, `add_q()`, `bold_italicize_labels_levels()`, `combine_terms()`, `inline_text.tbl_regression()`, `modify_header()`, `tbl_merge()`, `tbl_stack()`

Examples
```r
library(survival)
tbl_regression_ex1 <-
  coxph(Surv(ttdeath, death) ~ age + marker, trial) %>%
  tbl_regression(exponentiate = TRUE)

tbl_regression_ex2 <-
  glm(response ~ age + grade, trial, family = binomial(link = "logit")) %>%
  tbl_regression(exponentiate = TRUE)
```
library(lme4)
tbl_regression_ex3 <-
glmer(am ~ hp + (1 | gear), mtcars, family = binomial) %>%
tbl_regression(exponentiate = TRUE)

# for convenience, you can also pass named lists to any arguments
# that accept formulas (e.g label, etc.)
glm(response ~ age + grade, trial, family = binomial(link = "logit")) %>%
tbl_regression(exponentiate = TRUE, label = list(age = "Patient Age"))

tbl_stack

Stacks two or more gtsummary objects

Description
Assists in patching together more complex tables. tbl_stack() appends two or more tbl_regression, tbl_summary, or tbl_merge objects. gt attributes from the first regression object are utilized for output table.

Usage
tbl_stack(tbls)

Arguments
tbls List of gtsummary objects

Value
A tbl_stack object

Example Output

Author(s)
Daniel D. Sjoberg

See Also
tbl_merge
Other tbl_summary tools: add_n(), add_overall(), add_p.tbl_summary(), add_q(), add_stat_label(), bold_italicize_labels_levels, inline_text.tbl_summary(), inline_text.tbl_survfit(), modify_header(), tbl_merge(), tbl_summary()

Other tbl_regression tools: add_global_p.tbl_regression(), add_nevent.tbl_regression(), add_q(), bold_italicize_labels_levels, combine_terms(), inline_text.tbl_regression(), modify_header(), tbl_merge(), tbl_regression()

Other tbl_uvregression tools: add_global_p.tbl_uvregression(), add_nevent.tbl_uvregression(), add_q(), bold_italicize_labels_levels, inline_text.tbl_uvregression(), modify_header(), tbl_merge(), tbl_uvregression()
### Examples

#### # Example 1 - stacking two tbl_regression objects

```r
t1 <-
  glm(response ~ trt, trial, family = binomial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (unadjusted)"
  )
)

t2 <-
  glm(response ~ trt + grade + stage + marker, trial, family = binomial) %>%
  tbl_regression(
    include = "trt",
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (adjusted)"
  )
)

tbl_stack_ex1 < - tbl_stack(list(t1, t2))
```

#### # Example 2 - stacking two tbl_merge objects

```r
library(survival)
t3 <-
  coxph(Surv(ttdeath, death) ~ trt, trial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (unadjusted)"
  )
)

t4 <-
  coxph(Surv(ttdeath, death) ~ trt + grade + stage + marker, trial) %>%
  tbl_regression(
    include = "trt",
    exponentiate = TRUE,
    label = list(trt ~ "Treatment (adjusted)"
  )
)

# first merging, then stacking
row1 < - tbl_merge(list(t1, t3), tab_spanner = c("Tumor Response", "Death"))
row2 < - tbl_merge(list(t2, t4))
tbl_stack_ex2 <-
  tbl_stack(list(row1, row2))
```

### tbl_summary

**Create a table of summary statistics**

**Description**

The **tbl_summary** function calculates descriptive statistics for continuous, categorical, and dichotomous variables. Review the **tbl_summary vignette** for detailed examples.

**Usage**

```r
tbl_summary()
```
data,
by = NULL,
label = NULL,
statistic = NULL,
digits = NULL,
type = NULL,
value = NULL,
missing = c("ifany", "always", "no"),
missing_text = "Unknown",
sort = NULL,
percent = c("column", "row", "cell"),
group = NULL
)

Arguments
data A data frame
by A column name (quoted or unquoted) in data. Summary statistics will be calculated separately for each level of the by variable (e.g. by = trt). If NULL, summary statistics are calculated using all observations.
label List of formulas specifying variables labels, e.g. list(age ~ "Age, yrs", stage ~ "Path T Stage"). If a variable's label is not specified here, the label attribute (attr(data$age,"label")) is used. If attribute label is NULL, the variable name will be used.
statistic List of formulas specifying types of summary statistics to display for each variable. The default is list(all_continuous() ~ "{median} ({p25},{p75})",all_categorical() ~ "{n} ({p}%)"). See below for details.
digits List of formulas specifying the number of decimal places to round continuous summary statistics. If not specified, tbl_summary guesses an appropriate number of decimals to round statistics. When multiple statistics are displayed for a single variable, supply a vector rather than an integer. For example, if the statistic being calculated is "{mean} ({sd})" and you want the mean rounded to 1 decimal place, and the SD to 2 use digits = list(age ~ c(1,2)).
type List of formulas specifying variable types. Accepted values are c("continuous", "categorical", "dichotomous") e.g. type = list(starts_with(age) ~ "continuous", female ~ "dichotomous"). If type not specified for a variable, the function will default to an appropriate summary type. See below for details.
value List of formulas specifying the value to display for dichotomous variables. See below for details.
missing Indicates whether to include counts of NA values in the table. Allowed values are "no" (never display NA values), "ifany" (only display if any NA values), and "always" (includes NA count row for all variables). Default is "ifany".
missing_text String to display for count of missing observations. Default is "Unknown".
sort List of formulas specifying the type of sorting to perform for categorical data. Options are frequency where results are sorted in descending order of frequency and alphanumeric, e.g. sort = list(everything() ~ "frequency")
percent Indicates the type of percentage to return. Must be one of "column", "row", or "cell". Default is "column".
group DEPRECATED. Migrated to add_p
Value

A tbl_summary object

select helpers

Select helpers from the tidyselect\ package and gtsummary\ package are available to modify default behavior for groups of variables. For example, by default continuous variables are reported with the median and IQR. To change all continuous variables to mean and standard deviation use statistic = list(all_continuous() ~ "\{mean\} (\{sd\})").

All columns with class logical are displayed as dichotomous variables showing the proportion of events that are TRUE on a single row. To show both rows (i.e. a row for TRUE and a row for FALSE) use type = list(all_logical() ~ "categorical").

The select helpers are available for use in any argument that accepts a list of formulas (e.g. statistic, type, digits, value, sort, etc.)

statistic argument

The statistic argument specifies the statistics presented in the table. The input is a list of formulas that specify the statistics to report. For example, statistic = list(age ~ "\{mean\} (\{sd\})") would report the mean and standard deviation for age; statistic = list(all_continuous() ~ "\{mean\} (\{sd\})") would report the mean and standard deviation for all continuous variables. A statistic name that appears between curly brackets will be replaced with the numeric statistic (see glue::glue).

For categorical variables the following statistics are available to display.

- \{n\} frequency
- \{N\} denominator, or cohort size
- \{p\} formatted percentage

For continuous variables the following statistics are available to display.

- \{median\} median
- \{mean\} mean
- \{sd\} standard deviation
- \{var\} variance
- \{min\} minimum
- \{max\} maximum
- \{p##\} any integer percentile, where ## is an integer from 0 to 100
- \{foo\} any function of the form foo(x) is accepted where x is a numeric vector

type argument

tbl_summary displays summary statistics for three types of data: continuous, categorical, and dichotomous. If the type is not specified, tbl_summary will do its best to guess the type. Dichotomous variables are categorical variables that are displayed on a single row in the output table, rather than one row per level of the variable. Variables coded as TRUE/FALSE, 0/1, or yes/no are assumed to be dichotomous, and the TRUE, 1, and yes rows are displayed. Otherwise, the value to display must be specified in the value argument, e.g. value = list(varname ~ "level to show")
Example Output

Author(s)

Daniel D. Sjoberg

See Also

See tbl_summary vignette for detailed examples

Other tbl_summary tools: add_n(), add_overall(), add_p.tbl_summary(), add_q(), add_stat_label(), bold_italicize_labels_levels(), inline_text.tbl_summary(), inline_text.tbl_survfit(), modify_header(), tbl_merge(), tbl_stack()

Examples

```r
# tbl_summary_ex1
trial[c("age", "grade", "response")]
  %>%
  tbl_summary()

# tbl_summary_ex2
trial[c("age", "grade", "response", "trt")]
  %>%
  tbl_summary(
    by = trt,
    label = list(age ~ "Patient Age"),
    statistic = list(all_continuous() ~ "{mean} \{sd\}",
    digits = list(age ~ c(0, 1))
  )

# for convenience, you can also pass named lists to any arguments
# that accept formulas (e.g label, digits, etc.)
trial[c("age", "trt")]
  %>%
  tbl_summary(
    by = trt,
    label = list(age = "Patient Age")
  )
```

### tbl_survfit

Creates table of survival probabilities

**Description**

Experimental Function takes a survfit object as an argument, and provides a formatted summary table of the results

**Usage**

```r
tbl_survfit(
  x,
  times = NULL,
  probs = NULL,
```
**tbl_survfit**

```r
statistic = "\{estimate\} \{conf.low\}, \{conf.high\}",
label = NULL,
label_header = NULL,
estimate_fun = NULL,
missing = "--",
conf.level = 0.95,
failure = FALSE
)
```

**Arguments**

- **x**
  - survfit object. Object may have no stratification (e.g. `survfit(Surv(ttdeath, death) ~ 1, trial)`), or a single stratifying variable (e.g. `survfit(Surv(ttdeath, death) ~ trt, trial)`)

- **times**
  - numeric vector of times for which to return survival probabilities.

- **probs**
  - numeric vector of probabilities with values in (0,1) specifying the survival quantiles to return.

- **statistic**
  - string defining the statistics to present in the table. Default is "\{estimate\} \{conf.low\}, \{conf.high\}".

- **label**
  - string specifying variable or overall label. Default is stratifying variable name or "Overall" when no stratifying variable present.

- **label_header**
  - string specifying column labels above statistics. Default is "\{prob\} Percentile" for survival percentiles, and "Time \{time\}" for n-year survival estimates.

- **estimate_fun**
  - function to format the Kaplan-Meier estimates. Default is `style_percent` for survival probabilities and `style_sigfig` for survival times.

- **missing**
  - text to fill when estimate is not estimable. Default is "--".

- **conf.level**
  - Confidence level for confidence intervals. Default is 0.95.

- **failure**
  - Calculate failure probabilities rather than survival probabilities. Default is FALSE. Does not apply to survival quantile requests.

**Example Output**

**Author(s)**

Daniel D. Sjoberg

**Examples**

```r
library(survival)
fit1 <- survfit(Surv(ttdeath, death) ~ trt, trial)
fit2 <- survfit(Surv(ttdeath, death) ~ 1, trial)

tbl_survfit_ex1 <- tbl_survfit(
  fit1,
times = c(12, 24),
label = "Treatment",
label_header = "**\{time\} Month**"
)

tbl_survfit_ex2 <- tbl_survfit(
  fit2,
times = c(12, 24),
label = "Overall",
label_header = "\{prob\} Percentile"
)
```
tbl_uvregression

Display univariate regression model results in table

Description

This function estimates univariate regression models and returns them in a publication-ready table. It can create univariate regression models holding either a covariate or outcome constant.

For models holding outcome constant, the function takes as arguments a data frame, the type of regression model, and the outcome variable y. Each column in the data frame is regressed on the specified outcome. The tbl_uvregression function arguments are similar to the tbl_regression arguments. Review the tbl_uvregression vignette for detailed examples.

You may alternatively hold a single covariate constant. For this, pass a data frame, the type of regression model, and a single covariate in the x= argument. Each column of the data frame will serve as the outcome in a univariate regression model. Take care using the x argument that each of the columns in the data frame are appropriate for the same type of model, e.g. they are all continuous variables appropriate for lm, or dichotomous variables appropriate for logistic regression with glm.

Usage

```r
tbl_uvregression(
  data,
  method,
  y = NULL,
  x = NULL,
  method.args = NULL,
  formula = "{y} ~ {x}",
  exponentiate = FALSE,
  label = NULL,
  include = everything(),
  exclude = NULL,
  hide_n = FALSE,
  show_single_row = NULL,
  conf.level = NULL,
  estimate_fun = NULL,
  pvalue_fun = NULL,
  show_yesno = NULL,
  tidy_fun = NULL
)
```

Arguments

data
  Data frame to be used in univariate regression modeling. Data frame includes the outcome variable(s) and the independent variables.

method
  Regression method (e.g. lm, glm, survival::coxph, and more).

y
  Model outcome (e.g. y = recurrence or y = Surv(time, recur)). All other column in data will be regressed on y. Specify one and only one of y or x.
x  Model covariate (e.g. \( x = \text{trt} \)). All other columns in data will serve as the outcome in a regression model with \( x \) as a covariate. Output table is best when \( x \) is a continuous or dichotomous variable displayed on a single row. Specify one and only one of \( y \) or \( x \).

method.args List of additional arguments passed on to the regression function defined by method.

formula String of the model formula. Uses `glue::glue` syntax. Default is \( \{y\} \sim \{x\} \), where \( y \) is the dependent variable, and \( x \) represents a single covariate. For a random intercept model, the formula may be `formula = \{y\} \sim \{x\} + (1 \mid \text{gear})`.

exponentiate Logical indicating whether to exponentiate the coefficient estimates. Default is `FALSE`.

label List of formulas specifying variables labels, e.g. `list(age \sim "Age.yrs", stage \sim "Path T Stage")`

include Variables to include in output. Input may be a vector of quoted variable names, unquoted variable names, or tidyselect select helper functions. Default is `everything()`.

exclude DEPRECATED

hide_n Hide N column. Default is `FALSE`.

show_single_row By default categorical variables are printed on multiple rows. If a variable is dichotomous (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name(s) here–quoted and unquoted variable name accepted.

conf.level Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

estimate_fun Function to round and format coefficient estimates. Default is `style_sigfig` when the coefficients are not transformed, and `style_ratio` when the coefficients have been exponentiated.

pvalue_fun Function to round and format p-values. Default is `style_pvalue`. The function must have a numeric vector input (the numeric, exact p-value), and return a string that is the rounded/formatted p-value (e.g. `pvalue_fun = function(x) style_pvalue(x, digits = 2)` or equivalently, `purrr::partial(style_pvalue, digits = 2)`).

show_yesno DEPRECATED

tidy_fun Option to specify a particular tidier function if the model is not a vetted model or you need to implement a custom method. Default is `NULL`.

Value

A `tbl_uvregression` object

Example Output

Setting Defaults

If you prefer to consistently use a different function to format p-values or estimates, you can set options in the script or in the user- or project-level startup file, `.Rprofile`. The default confidence level can also be set.
• options(gtsummary.pvalue_fun = new_function)
• options(gtsummary.tbl_regression.estimate_fun = new_function)
• options(gtsummary.conf.level = 0.90)

Note

The N reported in the output is the number of observations in the data frame model.frame(x). Depending on the model input, this N may represent different quantities. In most cases, it is the number of people or units in your model. Here are some common exceptions.

1. Survival regression models including time dependent covariates.
2. Random- or mixed-effects regression models with clustered data.
3. GEE regression models with clustered data.

This list is not exhaustive, and care should be taken for each number reported.

Author(s)

Daniel D. Sjoberg

See Also

See tbl_regression vignette for detailed examples

Other tbl_uvregression tools: add_global_p.tbl_uvregression(), add_nevent.tbl_uvregression(), add_q(), bold_italicize_labels_levels, inline_text.tbl_uvregression(), modify_header(), tbl_merge(), tbl_stack()

Examples

tbl_uv_ex1 <-
  tbl_uvregression(
    trial[c("response", "age", "grade")],
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  )

# rounding pvalues to 2 decimal places
library(survival)
tbl_uv_ex2 <-
  tbl_uvregression(
    trial[c("ttdeath", "death", "age", "grade", "response")],
    method = coxph,
    y = Surv(ttdeath, death),
    exponentiate = TRUE,
    pvalue_fun = function(x) style_pvalue(x, digits = 2)
  )
## Results from a simulated study of two chemotherapy agents

### Description
A dataset containing the baseline characteristics of 200 patients who received Drug A or Drug B. Dataset also contains the outcome of tumor response to the treatment.

### Usage

```
trial
```

### Format
A data frame with 200 rows—one row per patient

- **trt**: Chemotherapy Treatment
- **age**: Age, yrs
- **marker**: Marker Level, ng/mL
- **stage**: T Stage
- **grade**: Grade
- **response**: Tumor Response
- **death**: Patient Died
- **ttdeath**: Months to Death/Censor
Index

*Topic** datasets

  trial, 55

add_global_p, 3
add_global_p.tbl_regression, 3, 4, 8, 15, 23, 24, 29, 34, 43, 45, 46
add_global_p.tbl_uvregression, 3, 5, 9, 15, 23, 33, 34, 43, 46, 54
add_n, 6, 10, 14–16, 23, 30, 31, 34, 43, 46, 50
add_nevent, 7
add_nevent.tbl_regression, 5, 7, 8, 15, 23, 24, 29, 34, 43, 45, 46
add_nevent.tbl_uvregression, 5, 7, 9, 15, 23, 33, 34, 43, 46, 54
add_overall, 7, 10, 14–16, 23, 30, 31, 34, 43, 46, 50
add_p, 11, 48
add_p.tbl_cross, 11, 11, 27, 42
add_p.tbl_summary, 7, 10, 11, 12, 15, 16, 23, 30, 31, 34, 43, 46, 50
add_q, 5, 7–10, 14, 14, 16, 23, 24, 29–31, 33, 34, 43, 45, 46, 50, 54
add_stat_label, 7, 10, 14, 15, 15, 23, 30, 31, 34, 43, 46, 50
all_categorical (select_helpers), 35
all_character (select_helpers), 35
all_dichotomous (select_helpers), 35
all_continuous (select_helpers), 35
all_factor (select_helpers), 35
all_numeric (select_helpers), 35
as_flextable, 16, 18–21
as_flextable(), 17
as_gt, 17, 19, 29–31, 33, 34, 43, 45, 46, 50, 54
bold_labels

(bold_italicize_labels_levels), 22
bold_labels(), 19
bold_levels

(bold_italicize_labels_levels), 22
bold_p, 23
car::Anova, 3–5
combine_terms, 5, 8, 15, 23, 24, 29, 34, 43, 45, 46
crayon::strip_style(), 25
flextable::add_header_row(), 17
flextable::align(), 17
flextable::autofit(), 17
flextable::bold(), 17
flextable::flextable(), 17
flextable::footnote(), 17
flextable::italic(), 17
flextable::padding(), 17
flextable::set_header_labels(), 17
flextable::width(), 17
ggepack::geeglm, 7–9
glm, 52
 glue::glue, 6, 27, 28, 30, 32–34, 49, 53
gtsummary_logo, 25
inline_text, 8, 9, 26
inline_text.tbl_cross, 12, 26, 42
inline_text.tbl_regression, 5, 8, 15, 23, 24, 26, 28, 34, 43, 45, 46
inline_text.tbl_summary, 7, 10, 14–16, 23, 26, 29, 31, 34, 43, 46, 50
inline_text.tbl_survfit, 7, 10, 14–16, 23, 26, 30, 34, 43, 46, 50
inline_text.tbl_survival, 34
inline_text.tbl_uvregression, 5, 9, 15, 23, 26, 32, 34, 43, 46, 54
italicize_labels

(bold_italicize_labels_levels), 22
italicize_levels
   (bold_italicize_labels_levels), 22
italicize_levels(), 19

knit_print.gtsummary(print_gtsummary), 35
knitr::kable, 19, 20

lm, 52
lme4::glmer, 7–9

modify_header, 5, 7–10, 14–16, 23, 24,
   29–31, 33, 33, 43, 45, 46, 50, 54

print.gtsummary (print_gtsummary), 35
print_gtsummary, 35

select_helpers, 35
sort_p, 36
stats::anova, 24
stats::anova(), 24
stats::glm, 7–9
stats::p.adjust, 14
stats::update, 24
style_percent, 37, 38–40, 51
style_pvalue, 11, 13, 14, 27, 30, 31, 37, 38,
   39, 40, 44, 53
style_ratio, 31, 37, 38, 39, 40, 44, 53
style_sigfig, 31, 37–39, 40, 44, 51, 53
survival::coxph, 7–9, 52

tbl_cross, 11, 12, 27, 41
tbl_merge, 5, 7–10, 14–16, 23, 24, 29–31,
   33–35, 42, 45, 46, 50, 54
tbl_regression, 4, 5, 7, 8, 15, 17–21, 23, 24,
   28, 29, 34, 35, 43, 43, 46, 52
tbl_stack, 5, 7–10, 14–16, 23, 24, 29–31,
   33–35, 43, 45, 46, 50, 54
tbl_summary, 6, 7, 10, 12, 14–21, 23, 30, 31,
   34, 35, 43, 46, 47
tbl_survfit, 31, 50
tbl_survival.survfit, 34
tbl_uvregression, 5, 7, 9, 15, 23, 32–35, 43,
   46, 52
tibble, 21
trial, 55

vetted model, 44, 53