

Package ‘gtsummary’

November 11, 2019

Title Presentation-Ready Data Summary and Analytic Result
Tables

Version 1.2.3

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. The package is enhanced when the 'gt' package is installed. Use this code to install: `'remotes::install_github("rstudio/gt")'`.

License MIT + file LICENSE

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

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

Depends R (>= 3.5)

Imports broom (>= 0.5.1),
broom.mixed (>= 0.2.3),
crayon (>= 1.3.4),
dplyr (>= 0.7.8),
glue (>= 1.3.0),
knitr (>= 1.21),
lifecycle (>= 0.1.0),
magrittr (>= 1.5),
purrr (>= 0.3.0),
rlang (>= 0.3.1),
stringr (>= 1.4.0),
survival,
tibble (>= 2.0.1),
tidyr (>= 1.0.0),
tidyselect (>= 0.2.5),
usethis (>= 1.5.1)

Suggests car (>= 3.0.2),
covr (>= 3.2.1),
geepack (>= 1.2.1),

ggplot2 (>= 3.1.0),
 Hmisc (>= 4.2.0),
 lme4 (>= 1.1.18.1),
 rmarkdown (>= 1.11),
 spelling (>= 2.0),
 testthat (>= 2.1.0),
 forcats (>= 0.4.0)

Enhances gt (>= 0.1.0)

VignetteBuilder knitr

RdMacros lifecycle

Additional_repositories <http://ddsjoberg.github.io/drat>

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 6.1.1

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	7
add_nevent.tbl_uvregression	8
add_overall	9
add_p	10
add_p_	12
add_q	13
add_q.tbl_summary	14
add_q.tbl_uvregression	15
add_stat_label	16
as_gt	17
as_kable	18
as_tibble_methods	19
bold_italicize_labels_levels	20
bold_p	21
bold_p.tbl_regression	22
bold_p.tbl_stack	23
bold_p.tbl_summary	24
bold_p.tbl_uvregression	25
gtsummary_logo	26
inline_text	26
inline_text.tbl_regression	27
inline_text.tbl_summary	28
inline_text.tbl_survival	29
inline_text.tbl_uvregression	31
modify_header	32

print_gtsummary	33
select_helpers	35
sort_p.tbl_regression	35
sort_p.tbl_summary	36
sort_p.tbl_uvregression	37
style_percent	38
style_pvalue	39
style_ratio	39
style_sigfig	40
tbl_merge	41
tbl_regression	42
tbl_stack	44
tbl_summary	46
tbl_summary_	49
tbl_survival	50
tbl_survival.survfit	50
tbl_uvregression	52
trial	55

Index	56
--------------	-----------

add_global_p	<i>Adds the global p-value for a categorical variables</i>
--------------	--

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 an [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

```
## S3 method for class 'tbl_regression'
add_global_p(x, terms = NULL, keep = FALSE,
  ...)
```

Arguments

x	Object with class <code>tbl_regression</code> from the tbl_regression function
terms	Character vector of terms for which to add global p-values. Default is <code>NULL</code> which will add global p-values for all categorical variables
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 <code>FALSE</code>
...	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 an [issue](#) to request support.

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_regression` tools: [add_nevent.tbl_regression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Examples

```
tbl_lm_global_ex1 <-
  lm(marker ~ age + grade, trial) %>%
  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.tbl_uvregression](#), [bold_italicize_labels.tbl_uvregression](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header.tbl_uvregression](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_uv_global_ex2 <-
  trial %>%
  dplyr::select(response, trt, age, grade) %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>%
  add_global_p()
```

add_n	<i>Add column with N</i>
-------	--------------------------

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

<code>x</code>	Object with class <code>tbl_summary</code> from the tbl_summary function
<code>statistic</code>	String indicating the statistic to report. Default is the number of non-missing observation for each variable, <code>statistic = "{n}"</code> . Other statistics available to report include: <ul style="list-style-type: none"> • <code>"{N}"</code> total number of observations, • <code>"{n}"</code> number of non-missing observations, • <code>"{n_miss}"</code> number of missing observations, • <code>"{p}"</code> percent non-missing data, • <code>"{p_miss}"</code> percent missing data The argument uses glue::glue syntax and multiple statistics may be reported, e.g. <code>statistic = "{n} / {N} ({p}%)"</code>
<code>col_label</code>	String indicating the column label. Default is <code>"**N**"</code>
<code>footnote</code>	Logical argument indicating whether to print a footnote clarifying the statistics presented. Default is <code>FALSE</code>
<code>last</code>	Logical indicator to include N column last in table. Default is <code>FALSE</code> , which will display N column first.
<code>missing</code>	DEPRECATED. Logical argument indicating whether to print N (<code>missing = FALSE</code>), or N missing (<code>missing = TRUE</code>). Default is <code>FALSE</code>

Value

A `tbl_summary` object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_summary` tools: [add_overall](#), [add_p](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_le](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

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

add_nevent

*Add number of events to a regression table***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_regeression](#) 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](#)

add_nevent.tbl_regression

*Add number of events to a regression table***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](#)).

Usage

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

Arguments

x	tbl_regression object
...	Not used

Value

A tbl_regression object

Reporting Event N

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.

Example Output**Author(s)**

Daniel D. Sjoberg

See Also

Other `tbl_regression` tools: [add_global_p.tbl_regression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Examples

```
tbl_reg_nevent_ex <-
  glm(response ~ trt, trial, family = binomial) %>%
  tbl_regression() %>%
  add_nevent()
```

```
add_nevent.tbl_uvregression
```

Add number of events to a regression table

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

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


Arguments

x	tbl_uvregerssion 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.tbl_uvregression](#), [bold_italicize_label.tbl_uvregression](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header.tbl_uvregression](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_uv_nevent_ex <-
  trial %>%
  dplyr::select(response, trt, age, grade) %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial)
  ) %>%
  add_nevent()
```

add_overall	<i>Add column with overall summary statistics</i>
-------------	---

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](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

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

add_p	<i>Adds p-values to summary tables</i>
-------	--

Description

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

Usage

```
add_p(x, test = NULL, pvalue_fun = NULL, group = NULL,
      include = NULL, 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,
 - "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,
- "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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
group	Column name of an ID or grouping variable. The column can be used 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	Character vector of variable names to include from output.
exclude	Character vector of variable names to exclude from output.

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.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_le](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

```
add_p_ex1 <-
  trial %>%
  dplyr::select(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
  result$test <- "McNemar\\'s test"
  result
}

add_p_ex2 <-
  trial[c("response", "trt")] %>%
  tbl_summary(by = trt) %>%
  add_p(test = vars(response) ~ "my_mcnemar")
```

add_p_	<i>Standard evaluation version of add_p()</i>
--------	---

Description

The 'group =' argument can be passed as a string, rather than with non-standard evaluation as in [add_p](#). Review the help file for [add_p](#) fully documented options and arguments.

Usage

```
add_p(x, test = NULL, pvalue_fun = NULL, group = NULL,
      include = NULL, exclude = NULL)
```

Arguments

x	Object with class <code>tbl_summary</code> from the tbl_summary function
test	List of formulas specifying statistical tests to perform, e.g. <code>list(all_continuous() ~ "t.test", all_categorical() ~ "fisher.test")</code> . Options include <ul style="list-style-type: none"> • "t.test" for a t-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, • "fisher.test" for a Fisher's exact test, • "lme4" for a random intercept logistic regression model to account for clustered data, <code>lme4::glmer(by ~ variable + (1 group), family = binomial)</code>. 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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
group	Column name of an ID or grouping variable. The column can be used 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	Character vector of variable names to include from output.
exclude	Character vector of variable names to exclude from output.

add_q

Add a column of q values to account for multiple comparisons

Description

Add a column of q values to account for multiple comparisons

Usage

```
add_q(x, ...)
```

Arguments

x	tbl_summary or tbl_uvregression object
...	Additional arguments passed to other methods.

Author(s)

Esther Drill, Daniel D. Sjoberg

See Also

[add_q.tbl_summary](#), [add_q.tbl_uvregression](#), [tbl_summary](#), [tbl_uvregression](#)

add_q.tbl_summary	<i>Add a column of q-values to account for multiple comparisons</i>
-------------------	---

Description

Adjustments to are p-values are performed with [stats::p.adjust](#).

Usage

```
## S3 method for class 'tbl_summary'
add_q(x, method = "fdr",
      pvalue_fun = x$pvalue_fun, ...)
```

Arguments

x	tbl_summary 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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
...	Additional arguments passed to or from other methods

Value

A tbl_summary object

Example Output

Author(s)

Esther Drill, Daniel D. Sjoberg

See Also

Other tbl_summary tools: [add_n](#), [add_overall](#), [add_p](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

```
tbl_sum_q_ex <-
  trial %>%
  dplyr::select(trt, age, grade, response) %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  add_q()
```

```
add_q.tbl_uvregression
```

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

Description

Adjustments to are p-values are performed with [stats::p.adjust](#).

Usage

```
## S3 method for class 'tbl_uvregression'
add_q(x, method = "fdr",
      pvalue_fun = x$inputs$pvalue_fun, ...)
```

Arguments

x	tbl_uvregression 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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
...	Additional arguments passed to or from other methods

Value

A tbl_uvregression object

Example Output

Author(s)

Esther Drill, Daniel D. Sjoberg

See Also

Other tbl_uvregression tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_uvr_q_ex <-
  trial %>%
  dplyr::select(age, marker, grade, response) %>%
  tbl_uvregression(
    method = lm,
```

```

    y = age
  ) %>%
  add_global_p() %>%
  add_q()

```

add_stat_label	<i>Add statistic labels column</i>
----------------	------------------------------------

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](#), [add_q.tbl_summary](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

```

tbl_stat_ex <-
  trial %>%
  dplyr::select(trt, age, grade, response) %>%
  tbl_summary() %>%
  add_stat_label()

```


as_gt

*Convert gtsummary object to a gt_tbl object***Description**

Function converts gtsummary objects to a gt_tbl objects. 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

```
as_gt(x, include = NULL, exclude = NULL, omit = NULL)
```

Arguments

x	Object created by a function from the gtsummary package (e.g. tbl_summary or tbl_regression)
include	Character vector naming gt commands to include in printing. Default is NULL, which utilizes all commands in x\$gt_calls.
exclude	Character vector naming gt commands to exclude in printing. Default is NULL.
omit	DEPRECATED. Argument is synonymous with exclude vector of named gt commands to omit. Default is NULL

Value

A gt_tbl object

Example Output**Author(s)**

Daniel D. Sjoberg

See Also

[tbl_summary](#) [tbl_regression](#) [tbl_uvregression](#) [tbl_survival](#)

Examples

```
as_gt_ex <-
  trial[c("trt", "age", "response", "grade")] %>%
  tbl_summary(by = trt) %>%
  as_gt()
```

as_kable	<i>Convert to knitr_kable object</i>
----------	--------------------------------------

Description

Function converts gtsummary objects to a knitr_kable objects. 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](#).

Usage

```
as_kable(x, include = NULL, exclude = NULL, ...)
```

Arguments

x	Object created by a function from the gtsummary package (e.g. tbl_summary or tbl_regression)
include	Character vector naming kable commands to include in printing. Default is NULL, which utilizes all commands in x\$kable_calls.
exclude	Character vector naming kable commands to exclude in printing. Default is NULL.
...	Additional arguments passed to knitr::kable

Value

A knitr_kable object

Author(s)

Daniel D. Sjoberg

See Also

[tbl_summary](#) [tbl_regression](#) [tbl_uvregression](#) [tbl_survival](#)

Examples

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

as_tibble_methods *Convert gtsummary object to tibble*

Description

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

Usage

```
## S3 method for class 'tbl_summary'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)

## S3 method for class 'tbl_regression'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)

## S3 method for class 'tbl_uvregression'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)

## S3 method for class 'tbl_merge'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)

## S3 method for class 'tbl_stack'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)

## S3 method for class 'tbl_survival'
as_tibble(x, include = NULL, exclude = NULL,
  col_labels = TRUE, ...)
```

Arguments

<code>x</code>	Object created by a function from the gtsummary package (e.g. tbl_summary or tbl_regression)
<code>include</code>	Character vector naming kable commands to include in printing. Default is NULL, which utilizes all commands in <code>x\$kable_calls</code> .
<code>exclude</code>	Character vector naming kable commands to exclude in printing. Default is NULL.
<code>col_labels</code>	Logical argument adding column labels to output tibble. Default is TRUE.
<code>...</code>	Not used

Value

a [tibble](#)

Author(s)

Daniel D. Sjoberg

See Also

[tbl_summary](#) [tbl_regression](#) [tbl_uvregression](#) [tbl_survival](#)

Examples

```
tbl <-  
  trial %>%  
  tbl_summary(by = trt)  
  
as_tibble(tbl)  
  
# without column labels  
as_tibble(tbl, col_names = FALSE)
```

bold_italicize_labels_levels

Bold or Italicize labels or levels in gtsummary tables

Description

Bold or Italicize labels or levels in gtsummary tables

Usage

```
bold_labels(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

See Also

Other `tbl_summary` tools: [add_n](#), [add_overall](#), [add_p](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Other `tbl_regression` tools: [add_global_p.tbl_regression](#), [add_nevent.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Other `tbl_uvregression` tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_bold_ital_ex <-
  trial %>%
  dplyr::select(trt, age, grade) %>%
  tbl_summary() %>%
  bold_labels() %>%
  bold_levels() %>%
  italicize_labels() %>%
  italicize_levels()
```

<code>bold_p</code>	<i>Bold significant p-values or q-values</i>
---------------------	--

Description

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

Usage

```
bold_p(x, ...)
```

Arguments

- `x` Object created using `gtsummary` functions
- `...` Additional arguments passed to other methods.

Author(s)

Daniel D. Sjoberg, Esther Drill

See Also

[bold_p.tbl_summary](#), [bold_p.tbl_regression](#), [bold_p.tbl_uvregression](#)

`bold_p.tbl_regression` *Bold significant p-values or q-values*

Description

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

Usage

```
## S3 method for class 'tbl_regression'
bold_p(x, t = 0.05, ...)
```

Arguments

<code>x</code>	Object created using <code>tbl_regression</code> function
<code>t</code>	Threshold below which values will be bold. Default is 0.05.
<code>...</code>	Not used

Value

A `tbl_regression` object

Example Output

Author(s)

Daniel D. Sjoberg, Esther Drill

See Also

Other `tbl_regression` tools: `add_global_p.tbl_regression`, `add_nevent.tbl_regression`, `bold_italicize_label.tbl_regression`, `bold_p.tbl_stack`, `inline_text.tbl_regression`, `modify_header.tbl_regression`, `sort_p.tbl_regression`, `tbl_merge.tbl_regression`, `tbl_stack.tbl_regression`

Examples

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

<code>bold_p.tbl_stack</code>	<i>Bold significant p-values or q-values</i>
-------------------------------	--

Description

Bold values below a chosen threshold (e.g. <0.05) in [tbl_stack](#) tables.

Usage

```
## S3 method for class 'tbl_stack'
bold_p(x, ...)
```

Arguments

<code>x</code>	Object created using tbl_stack function
<code>...</code>	arguments passed to <code>bold_p.*()</code> method that matches the first object in the <code>tbl_stack</code>

Value

A `tbl_stack` object

Example Output

Author(s)

Daniel D. Sjoberg

See Also

Other `tbl_uvregression` tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Other `tbl_regression` tools: [add_global_p.tbl_regression](#), [add_nevent.tbl_regression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_regression](#), [inline_text.tbl_regression](#), [modify_header](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Examples

```
t1 <- tbl_regression(lm(age ~ response, trial))
t2 <- tbl_regression(lm(age ~ grade, trial))

bold_p_stack_ex <-
  tbl_stack(list(t1, t2)) %>%
  bold_p(t = 0.10)
```

bold_p.tbl_summary	<i>Bold significant p-values or q-values</i>
--------------------	--

Description

Bold values below a chosen threshold (e.g. <0.05) in [tbl_summary](#) tables.

Usage

```
## S3 method for class 'tbl_summary'
bold_p(x, t = 0.05, q = FALSE, ...)
```

Arguments

x	Object created using <code>tbl_summary</code> function
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.
...	Not used

Value

A `tbl_summary` object

Example Output

Author(s)

Daniel D. Sjoberg, Esther Drill

See Also

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

Examples

```
tbl_sum_bold_p_ex <-
  trial %>%
  dplyr::select(age, grade, response, trt) %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  bold_p()
```

bold_p.tbl_uvregression

Bold significant p-values or q-values

Description

Bold values below a chosen threshold (e.g. <0.05) in [tbl_uvregression](#) tables.

Usage

```
## S3 method for class 'tbl_uvregression'
bold_p(x, t = 0.05, q = FALSE, ...)
```

Arguments

x	Object created using tbl_uvregression function
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.
...	Not used

Value

A `tbl_uvregression` object

Example Output

Author(s)

Daniel D. Sjoberg, Esther Drill

See Also

Other `tbl_uvregression` tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_uvglm_bold_p_ex <-
  trial %>%
  dplyr::select(age, marker, response, grade) %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial),
    exponentiate = TRUE
  ) %>%
  bold_p(t = 0.25)
```

gtsummary_logo	<i>The gtsummary logo, using ASCII or Unicode characters</i>
----------------	--

Description

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

Usage

```
gtsummary_logo(unicode = l10n_info()$`UTF-8`)
```

Arguments

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

Examples

```
gtsummary_logo()
```

inline_text	<i>Report statistics from gtsummary tables inline</i>
-------------	---

Description

Report statistics from gtsummary tables inline

Usage

```
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_survival](#)

inline_text.tbl_regression

Report statistics from regression summary tables inline

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 [tbl_regression vignette](#)

Usage

```
## 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$inputs$estimate_fun, pvalue_fun = function(x)
  style_pvalue(x, prepend_p = TRUE), ...)
```

Arguments

<code>x</code>	Object created from tbl_regression
<code>variable</code>	Variable name of statistics to present
<code>level</code>	Level of the variable to display for categorical variables. Default is <code>NULL</code> , returning the top row in the table for the variable.
<code>pattern</code>	String indicating the statistics to return. Uses glue::glue formatting. Default is <code>"{estimate} ({conf.level }% CI {conf.low},{conf.high}; {p.value})"</code> . All columns from <code>x\$table_body</code> are available to print as well as the confidence level (<code>conf.level</code>). See below for details.
<code>estimate_fun</code>	function to style model coefficient estimates. Columns <code>'estimate'</code> , <code>'conf.low'</code> , and <code>'conf.high'</code> are formatted. Default is <code>x\$inputs\$estimate_fun</code>
<code>pvalue_fun</code>	function to style p-values and/or q-values. Default is <code>function(x) style_pvalue(x,prepend_p = TRUE)</code>
<code>...</code>	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](#), [bold_italicize_label.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [modify_header.tbl_regression](#), [sort_p.tbl_regression](#), [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 [tbl_summary vignette](#)

Usage

```
## S3 method for class 'tbl_summary'
inline_text(x, variable, level = NULL,
  column = ifelse(is.null(x$by), "stat_0", stop("Must specify column")),
  pvalue_fun = function(x) style_pvalue(x, prepend_p = TRUE), ...)
```

Arguments

<code>x</code>	Object created from tbl_summary
<code>variable</code>	Variable name of statistic to present
<code>level</code>	Level of the variable to display for categorical variables. Can also specify the 'Unknown' row. Default is <code>NULL</code>
<code>column</code>	Column name to return from <code>x\$table_body</code> . Can also pass the level of a by variable.
<code>pvalue_fun</code>	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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
<code>...</code>	Not used

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](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

```
t1 <- tbl_summary(trial)
t2 <- tbl_summary(trial, by = trt) %>% add_p()

inline_text(t1, variable = "age")
inline_text(t2, variable = "grade", level = "I", column = "Drug")
inline_text(t2, variable = "grade", column = "p.value")
```

```
inline_text.tbl_survival
```

Report statistics from survival summary tables inline

Description

for inline reporting in an R markdown document.

Usage

```
## S3 method for class 'tbl_survival'
inline_text(x, strata = NULL, time = NULL,
  prob = NULL, pattern = "{estimate} ({conf.level*100}% CI {ci})",
  estimate_fun = x$estimate_fun, ...)
```

Arguments

x	Object created from tbl_survival
strata	If tbl_survival estimates are stratified, level of the stratum to report. Default is NULL when tbl_survival have no specified strata.
time	Time for which to return survival probability
prob	Probability for which to return survival time. For median survival use prob = 0.50
pattern	String indicating the statistics to return. Uses glue::glue formatting. Default is '{estimate} ({conf.level*100}% {ci})'. All columns from x\$table_long are available to print as well as the confidence level (conf.level). See below for details.
estimate_fun	function to round/style estimate and lower/upper confidence interval estimates. Note, this does not style the 'ci' column, which is a string. Default is x\$estimate_fun
...	Not used

Value

A string reporting results from a gtsummary table

pattern argument

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

- `{label}` 'time' or 'prob' label
- `{estimate}` survival or survival time 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` (pre-formatted)
- `{time}/{prob}` time or survival quantile (numeric)
- `{n.risk}` number at risk at 'time' (within stratum if applicable)
- `{n.event}` number of observed events at 'time' (within stratum if applicable)
- `{n}` number of observations (within stratum if applicable)
- `{variable}` stratum variable (if applicable)
- `{level}` stratum level (if applicable)
- `{groupname}` label_level from original `tbl_survival()` call

Author(s)

Karissa Whiting

See Also

Other `tbl_survival` tools: [modify_header](#), [tbl_survival.survfit](#)

Examples

```
library(survival)
surv_table <-
  survfit(Surv(ttdeath, death) ~ trt, trial) %>%
  tbl_survival(times = c(12, 24))

inline_text(surv_table,
  strata = "Drug",
  time = 12
)
```

inline_text.tbl_uvregression

Report statistics from regression summary tables inline

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 [tbl_regression vignette](#)

Usage

```
## 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

<code>x</code>	Object created from tbl_uvregression
<code>variable</code>	Variable name of statistics to present
<code>level</code>	Level of the variable to display for categorical variables. Default is <code>NULL</code> , returning the top row in the table for the variable.
<code>pattern</code>	String indicating the statistics to return. Uses glue::glue formatting. Default is <code>"{estimate} ({conf.level }% CI {conf.low},{conf.high}; {p.value})"</code> . All columns from <code>x\$table_body</code> are available to print as well as the confidence level (<code>conf.level</code>). See below for details.
<code>estimate_fun</code>	function to style model coefficient estimates. Columns <code>'estimate'</code> , <code>'conf.low'</code> , and <code>'conf.high'</code> are formatted. Default is <code>x\$inputs\$estimate_fun</code>
<code>pvalue_fun</code>	function to style p-values and/or q-values. Default is <code>function(x) style_pvalue(x, prepend_p = TRUE)</code>
<code>...</code>	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

See Also

Other `tbl_uvregression` tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
inline_text_ex1 <-
  trial %>%
  dplyr::select(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	<i>Modify column headers in gtsummary tables</i>
---------------	--

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"))
```

Arguments

x	gtsummary object, e.g. <code>tbl_summary</code> or <code>tbl_regression</code>
stat_by	String specifying text to include above the summary statistics stratified by a variable. Only use with stratified <code>tbl_summary</code> objects. The following fields are available for use in the headers: <ul style="list-style-type: none">• {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. <code>stat_by = "**{level}**, N = {n} ({style_percent(p)\%})"</code> . The <code>by</code> argument from the parent <code>tbl_summary()</code> cannot be NULL.
...	Specifies column label of any other column in <code>.\$table_body</code> . Argument is the column name, and the value is the new column header (e.g. <code>p.value = "Model P-values"</code>). Use <code>print(x\$table_body)</code> to see columns available.
text_interpret	indicates whether text will be interpreted as markdown ("md") or HTML ("html"). The text is interpreted with the <code>gt</code> package's <code>md()</code> or <code>html()</code> functions. The default is "md", and is ignored when the print engine is not <code>gt</code> .

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](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Other tbl_regression tools: [add_global_p.tbl_regression](#), [add_nevent.tbl_regression](#), [bold_italicize_labels_levels.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Other tbl_uvregression tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Other tbl_survival tools: [inline_text.tbl_survival](#), [tbl_survival.survfit](#)

Examples

```
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 'tbl_summary'  
print(x, ...)  
  
## S3 method for class 'tbl_summary'  
knit_print(x, ...)  
  
## S3 method for class 'tbl_regression'  
print(x, ...)  
  
## S3 method for class 'tbl_regression'  
knit_print(x, ...)  
  
## S3 method for class 'tbl_uvregression'  
print(x, ...)  
  
## S3 method for class 'tbl_uvregression'  
knit_print(x, ...)  
  
## S3 method for class 'tbl_survival'  
print(x, ...)  
  
## S3 method for class 'tbl_survival'  
knit_print(x, ...)  
  
## S3 method for class 'tbl_merge'  
print(x, ...)  
  
## S3 method for class 'tbl_merge'  
knit_print(x, ...)  
  
## S3 method for class 'tbl_stack'  
print(x, ...)  
  
## S3 method for class 'tbl_stack'  
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](#)

select_helpers	<i>Select helper functions</i>
----------------	--------------------------------

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

```
all_numeric()

all_character()

all_integer()

all_double()

all_logical()

all_factor()

all_continuous()

all_categorical(dichotomous = TRUE)

all_dichotomous()
```

Arguments

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

Value

A character vector of column names selected

sort_p.tbl_regression	<i>Sort variables in table by ascending p-values</i>
-----------------------	--

Description

Sort variables in tables created by [tbl_regression](#) by ascending p-values

Usage

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

Arguments

- x An object created using tbl_regression function
- ... Not used

Value

A tbl_regression object

Example Output

Author(s)

Karissa Whiting

See Also

Other tbl_regression tools: [add_global_p.tbl_regression](#), [add_nevent.tbl_regression](#), [bold_italicize_label.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header.tbl_merge](#), [tbl_regression](#), [tbl_stack](#)

Examples

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

sort_p.tbl_summary	<i>Sort variables in table by ascending p-values</i>
--------------------	--

Description

Sort variables in tables created by [tbl_summary](#) by ascending p-values

Usage

```
## S3 method for class 'tbl_summary'  
sort_p(x, q = FALSE, ...)
```

Arguments

- x An object created using tbl_summary function
- q Logical argument. When TRUE will sort by the q-value column rather than the p-values
- ... Not used

Value

A tbl_summary object

Example Output**Author(s)**

Karissa Whiting

See Also

Other tbl_summary tools: [add_n](#), [add_overall](#), [add_p](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [tbl_merge](#), [tbl_stack](#), [tbl_summary](#)

Examples

```
tbl_sum_sort_p_ex <-
  trial %>%
  dplyr::select(age, grade, response, trt) %>%
  tbl_summary(by = trt) %>%
  add_p() %>%
  sort_p()
```

```
sort_p.tbl_uvregression
```

Sort variables in table by ascending p-values

Description

Sort variables in tables created by [tbl_uvregression](#) by ascending p-values

Usage

```
## S3 method for class 'tbl_uvregression'
sort_p(x, q = FALSE, ...)
```

Arguments

x	an object created using tbl_uvregression function
q	logical argument. When TRUE will sort by the q-value column rather than the p-values
...	Not used

Value

A [tbl_uvregression](#) object

Example Output

Author(s)

Karissa Whiting

See Also

Other `tbl_uvregression` tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [tbl_merge](#), [tbl_stack](#), [tbl_uvregression](#)

Examples

```
tbl_uvglm_sort_p_ex <-
  trial %>%
  dplyr::select(age, marker, response, grade) %>%
  tbl_uvregression(
    method = glm,
    y = response,
    method.args = list(family = binomial),
    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

<code>x</code>	numeric vector of percentages
<code>symbol</code>	Logical indicator to include percent symbol in output. Default is FALSE.

Value

A character vector of styled percentages

Author(s)

Daniel D. Sjoberg

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	<i>Style p-values to be displayed in tables or text</i>
--------------	---

Description

Style p-values to be displayed in tables or text

Usage

```
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

Examples

```
pvals <- c(
  1.5, 1, 0.999, 0.5, 0.25, 0.2, 0.197, 0.12, 0.10, 0.0999, 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)
```

style_ratio	<i>Implement significant figure-like rounding for ratios</i>
-------------	--

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

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

Value

A character vector of styled ratios

Author(s)

Daniel D. Sjoberg

See Also

[style_sigfig](#)

Examples

```
c(
  0.123, 0.9, 1.1234, 12.345, 101.234, -0.123,
  -0.9, -1.1234, -12.345, -101.234
) %>%
  style_ratio()
```

style_sigfig

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 1.2×10^2).

Usage

```
style_sigfig(x, digits = 2)
```

Arguments

<code>x</code>	Numeric vector
<code>digits</code>	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

Examples

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

tbl_merge	<i>Merge two or more gtsummary objects</i>
-----------	--

Description

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

Usage

```
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](#), [bold_italicize_label.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header.tbl_regression](#), [sort_p.tbl_regression](#), [tbl_regression](#), [tbl_stack](#)

Other tbl_uvregression tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels.tbl_uvregression](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header.tbl_uvregression](#), [sort_p.tbl_uvregression](#), [tbl_stack](#), [tbl_uvregression](#)

Other tbl_summary tools: [add_n.tbl_summary](#), [add_overall.tbl_summary](#), [add_p.tbl_summary](#), [add_q.tbl_summary](#), [add_stat_label.tbl_summary](#), [bold_italicize_labels_levels.tbl_summary](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header.tbl_summary](#), [sort_p.tbl_summary](#), [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 %>%
  dplyr::select(age, grade, response) %>%
  tbl_summary(missing = "no") %>%
  add_n()
t4 <-
  tbl_uvregression(
    trial %>% dplyr::select(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(exclude = "tab_spanner") %>%
  gt::cols_label(stat_0_1 = gt::md("**Summary Statistics**"))
```

tbl_regression

Display regression model results in table

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 = NULL,
  exclude = NULL, show_single_row = NULL, conf.level = NULL,
  intercept = FALSE, estimate_fun = NULL, pvalue_fun = NULL,
  show_yesno = NULL)
```

Arguments

<code>x</code>	Regression model object
<code>label</code>	List of formulas specifying variables labels, e.g. <code>list("age" ~ "Age, yrs", "ptstage" ~ "Path T Stage")</code>
<code>exponentiate</code>	Logical indicating whether to exponentiate the coefficient estimates. Default is FALSE.
<code>include</code>	Character vector of variable names to include from output.
<code>exclude</code>	Character vector of variable names to exclude from output.
<code>show_single_row</code>	By default categorical variables are printed on multiple rows. If a variable is binary (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name here, e.g. <code>show_single_row = c("var1", "var2")</code>
<code>conf.level</code>	Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
<code>intercept</code>	Logical argument indicating whether to include the intercept in the output. Default is FALSE
<code>estimate_fun</code>	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.
<code>pvalue_fun</code>	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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
<code>show_ynsno</code>	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](#), [bold_italicize_label.tbl_regression](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header.tbl_regression](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_stack](#)

Examples

```
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	<i>Stacks two or more gtsummary objects</i>
-----------	---

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](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_summary](#)

Other tbl_regression tools: [add_global_p.tbl_regression](#), [add_nevent.tbl_regression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_regression](#), [bold_p.tbl_stack](#), [inline_text.tbl_regression](#), [modify_header](#), [sort_p.tbl_regression](#), [tbl_merge](#), [tbl_regression](#)

Other tbl_uvregression tools: [add_global_p.tbl_uvregression](#), [add_nevent.tbl_uvregression](#), [add_q.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_uvregression](#)

Examples

```
# Example 1 - stacking two tbl_regression objects
t1 <-
  glm(response ~ trt, trial, family = binomial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(vars(trt) ~ "Treatment (unadjusted)")
  )

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

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

# Example 2 - stacking two tbl_merge objects
library(survival)
t3 <-
  coxph(Surv(ttdeath, death) ~ trt, trial) %>%
  tbl_regression(
    exponentiate = TRUE,
    label = list(vars(trt) ~ "Treatment (unadjusted)")
  )

t4 <-
```

```

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

```

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

<code>data</code>	A data frame
<code>by</code>	A column name in data. Summary statistics will be calculated separately for each level of the <code>by</code> variable (e.g. <code>by = trt</code>). If <code>NULL</code> , summary statistics are calculated using all observations.
<code>label</code>	List of formulas specifying variables labels, e.g. <code>list(vars(age) ~ "Age, yrs", vars(ptstage) ~ "Path T Stage")</code> . If a variable's label is not specified here, the function will take the label attribute (<code>attr(data\$age, "label")</code>). If attribute label is <code>NULL</code> , the variable name will be used.
<code>statistic</code>	List of formulas specifying types of summary statistics to display for each variable. The default is <code>list(all_continuous() ~ "{median} ({p25},{p75})", all_categorical() ~ "{n} ({p}%)")</code> . See below for details.
<code>digits</code>	List of formulas specifying the number of decimal places to round continuous summary statistics. If not specified, <code>tbl_summary</code> 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 <code>"{mean} ({sd})"</code> and you want the mean rounded to 1 decimal place, and the SD to 2 use <code>digits = list("age" ~ c(1, 2))</code> .
<code>type</code>	List of formulas specifying variable types. Accepted values are <code>c("continuous", "categorical", "dichotomous")</code> . e.g. <code>type = list(starts_with(age) ~ "continuous", "female" ~ "dichotomous")</code> . 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. <code>sort = list(everything() ~ "frequency")</code>
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 will be 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](#), [add_q.tbl_summary](#), [add_stat_label](#), [bold_italicize_labels_levels](#), [bold_p.tbl_summary](#), [inline_text.tbl_summary](#), [modify_header](#), [sort_p.tbl_summary](#), [tbl_merge](#), [tbl_stack](#)

Examples

```
tbl_summary_ex1 <-
  trial %>%
  dplyr::select(age, grade, response) %>%
  tbl_summary()

tbl_summary_ex2 <-
  trial %>%
  dplyr::select(age, grade, response, trt) %>%
  tbl_summary(
    by = trt,
    label = list(vars(age) ~ "Patient Age"),
    statistic = list(all_continuous() ~ "{mean} ({sd})"),
    digits = list(vars(age) ~ c(0, 1))
  )

# for convenience, you can also pass named lists to any arguments
# that accept formulas (e.g label, digits, etc.)
trial %>%
  dplyr::select(age, grade, response, trt) %>%
  tbl_summary(
    by = trt,
    label = list(age = "Patient Age"),
    statistic = list(all_continuous() ~ "{mean} ({sd})"),
    digits = list(vars(age) ~ c(0, 1))
  )
```


tbl_summary_

Standard evaluation version of tbl_summary()

Description

The 'by =' argument can be passed as a string, rather than with non-standard evaluation as in [tbl_summary](#). Review the help file for [tbl_summary](#) fully documented options and arguments.

Usage

```
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 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(vars(age) ~ "Age, yrs", vars(ptstage) ~ "Path T Stage"). If a variable's label is not specified here, the function will take the label attribute (attr(data\$age, "label")). 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. <code>sort = list(everything() ~ "frequency")</code>
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

tbl_survival	<i>Creates table of univariate summary statistics for time-to-event endpoints</i>
--------------	---

Description

Creates table of univariate summary statistics for time-to-event endpoints

Usage

```
tbl_survival(x, ...)
```

Arguments

x	A survfit object
...	Additional arguments passed to other methods

See Also

[tbl_survival.survfit](#)

tbl_survival.survfit	<i>Creates table of survival probabilities</i>
----------------------	--

Description

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

Usage

```
## S3 method for class 'survfit'
tbl_survival(x, times = NULL, probs = NULL,
  label = ifelse(is.null(probs), "{time}", "{prob*100}%"),
  level_label = "{level}", N = {n}", header_label = NULL,
  header_estimate = NULL, failure = FALSE, missing = "-",
  estimate_fun = NULL, ...)
```

Arguments

x	A survfit object with a no stratification (e.g. <code>survfit(Surv(ttdeath,death) ~ 1,trial)</code>), or a single stratifying variable (e.g. <code>survfit(Surv(ttdeath,death) ~ trt,trial)</code>)
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
label	String defining the label shown for the time or prob column. Default is "{time}" or "{prob*100}%". The input uses <code>glue::glue</code> notation to convert the string into a label. A common label may be "{time} Months", which would resolve to "6 Months" or "12 Months" depending on specified times.
level_label	Used when survival results are stratified. It is a string defining the label shown. The input uses <code>glue::glue</code> notation to convert the string into a label. The default is "{level},N={n}". Other information available to call are '{n}', '{level}', '{n.event.tot}', '{n.event.strata}', and '{strata}'. See below for details.
header_label	String to be displayed as column header. Default is '**Time**' when time is specified, and '**Quantile**' when probs is specified.
header_estimate	String to be displayed as column header of the Kaplan-Meier estimate. Default is '**Probability**' when time is specified, and '**Time**' when probs is specified.
failure	Calculate failure probabilities rather than survival probabilities. Default is FALSE. Does NOT apply to survival quantile requests
missing	String indicating what to replace missing confidence limits with in output. Default is missing = "-"
estimate_fun	Function used to format the estimate and confidence limits. The default is <code>style_percent(x, symbol = TRUE)</code> for survival probabilities, and <code>style_sigfig(x, digits = 3)</code> for time estimates.
...	Not used

Value

A `tbl_survival` object

level_label argument

The `level_label` is used to modify the stratum labels. The default is `level_label = "{level},N={n}"`. The quantities in the curly brackets evaluate to stratum-specific values. For example, in the trial data set, there is a column called `trt` with levels 'Drug' and 'Placebo'. In this example, `{level}` would evaluate to either 'Drug' or 'Placebo' depending on the stratum. Other quantities available to print are:

- `{level}` level of the stratification variable
- `{level_label}` label of level for the stratification variable
- `{n}` number of observations, or number within stratum
- `{n.event.tot}` total number of events (total across stratum, if applicable)
- `{n.event.strata}` total number of events within stratum, if applicable
- `{strata}` raw stratum specification from `survfit` object

Example Output

Author(s)

Daniel D. Sjöberg

See Also

Other `tbl_survival` tools: [inline_text.tbl_survival](#), [modify_header](#)

Examples

```
library(survival)
fit1 <- survfit(Surv(ttdeath, death) ~ trt, trial)
tbl_strata_ex1 <-
  tbl_survival(fit1,
    times = c(12, 24),
    label = "{time} Months"
  )

fit2 <- survfit(Surv(ttdeath, death) ~ 1, trial)
tbl_nostrata_ex2 <-
  tbl_survival(fit2,
    probs = c(0.1, 0.2),
    header_estimate = "**Months**"
  )
```

tbl_uvregression	<i>Display univariate regression model results in table</i>
------------------	---

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 binary variables appropriate for logistic regression with [glm](#).

Usage

```
tbl_uvregression(data, method, y = NULL, x = NULL,
  method.args = NULL, formula = "{y} ~ {x}", exponentiate = FALSE,
  label = NULL, include = NULL, exclude = NULL, hide_n = FALSE,
  show_single_row = NULL, conf.level = NULL, estimate_fun = NULL,
  pvalue_fun = NULL, show_yesno = 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. <code>y = recurrence</code> or <code>y = Surv(time, recur)</code>). All other column in data will be regressed on y. Specify one and only one of y or x
x	Model covariate (e.g. <code>x = trt</code>). 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 binary 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 " <code>{y} ~ {x}</code> ", where <code>{y}</code> is the dependent variable, and <code>{x}</code> represents a single covariate. For a random intercept model, the formula may be <code>formula = "{y} ~ {x} + (1 gear)"</code> .
exponentiate	Logical indicating whether to exponentiate the coefficient estimates. Default is FALSE.
label	List of formulas specifying variables labels, e.g. <code>list("age" ~ "Age, yrs", "ptstage" ~ "Path T Stage")</code>
include	Character vector of variable names to include from output.
exclude	Character vector of variable names to exclude from output.
hide_n	Hide N column. Default is FALSE
show_single_row	By default categorical variables are printed on multiple rows. If a variable is binary (e.g. Yes/No) and you wish to print the regression coefficient on a single row, include the variable name here, e.g. <code>show_single_row = c("var1", "var2")</code>
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. <code>pvalue_fun = function(x) style_pvalue(x, digits = 2)</code> or equivalently, <code>purrr::partial(style_pvalue, digits = 2)</code>).
show_ynsno	deprecated

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.tbl_uvregression](#), [bold_italicize_labels_levels](#), [bold_p.tbl_stack](#), [bold_p.tbl_uvregression](#), [inline_text.tbl_uvregression](#), [modify_header](#), [sort_p.tbl_uvregression](#), [tbl_merge](#), [tbl_stack](#)

Examples

```
tbl_uv_ex1 <-
tbl_uvregression(
  trial %>% dplyr::select(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 %>% dplyr::select(ttdeath, death, age, grade, response),
  method = coxph,
  y = Surv(ttdeath, death),
  label = list(vars(grade) ~ "Grade"),
  exponentiate = TRUE,
  pvalue_fun = function(x) style_pvalue(x, digits = 2)
)
```

```
# for convenience, you can also pass named lists to any arguments
# that accept formulas (e.g label, etc.)
library(survival)
trial %>%
  dplyr::select(ttdeath, death, age, grade, response) %>%
  tbl_uvregression(
    method = coxph,
    y = Surv(ttdeath, death),
    label = list(grade = "Grade"),
    exponentiate = TRUE)
```

trial

Results from a simulated trial of Placebo vs Drug

Description

A dataset containing the baseline characteristics of 200 patients randomized to Placebo or Drug. Dataset also contains the trial outcome: tumor response to the treatment.

Usage

```
trial
```

Format

A data frame with 200 rows—one row per patient

trt Treatment Randomization

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, 21–23, 28, 33, 36, 41, 44, 45
- add_global_p.tbl_uvregression, 3, 5, 9, 15, 21, 23, 25, 32, 33, 38, 41, 45, 54
- add_n, 6, 10, 11, 14, 16, 21, 24, 29, 33, 37, 41, 45, 48
- add_nevent, 7
- add_nevent.tbl_regression, 4, 7, 7, 21–23, 28, 33, 36, 41, 44, 45
- add_nevent.tbl_uvregression, 5, 7, 8, 15, 21, 23, 25, 32, 33, 38, 41, 45, 54
- add_overall, 6, 9, 11, 14, 16, 21, 24, 29, 33, 37, 41, 45, 48
- add_p, 6, 10, 10, 12, 14, 16, 21, 24, 29, 33, 37, 41, 45, 47, 48, 50
- add_p_, 12
- add_q, 13
- add_q.tbl_summary, 6, 10, 11, 13, 14, 16, 21, 24, 29, 33, 37, 41, 45, 48
- add_q.tbl_uvregression, 5, 9, 13, 15, 21, 23, 25, 32, 33, 38, 41, 45, 54
- add_stat_label, 6, 10, 11, 14, 16, 21, 24, 29, 33, 37, 41, 45, 48
- all_categorical(select_helpers), 35
- all_character(select_helpers), 35
- all_continuous(select_helpers), 35
- all_dichotomous(select_helpers), 35
- all_double(select_helpers), 35
- all_factor(select_helpers), 35
- all_integer(select_helpers), 35
- all_logical(select_helpers), 35
- all_numeric(select_helpers), 35
- as_gt, 17
- as_kable, 18
- as_tibble.tbl_merge
 - (as_tibble_methods), 19
- as_tibble.tbl_regression
 - (as_tibble_methods), 19
- as_tibble.tbl_stack
 - (as_tibble_methods), 19
- as_tibble.tbl_summary
 - (as_tibble_methods), 19
- as_tibble.tbl_survival
 - (as_tibble_methods), 19
- as_tibble.tbl_uvregression
 - (as_tibble_methods), 19
- as_tibble_methods, 19
- bold_italicize_labels_levels, 4–6, 8–11, 14–16, 20, 22–25, 28, 29, 32, 33, 36–38, 41, 44, 45, 48, 54
- bold_labels
 - (bold_italicize_labels_levels), 20
- bold_levels
 - (bold_italicize_labels_levels), 20
- bold_p, 21
- bold_p.tbl_regression, 4, 8, 21, 22, 23, 28, 33, 36, 41, 44, 45
- bold_p.tbl_stack, 4, 5, 8, 9, 15, 21, 22, 23, 25, 28, 32, 33, 36, 38, 41, 44, 45, 54
- bold_p.tbl_summary, 6, 10, 11, 14, 16, 21, 24, 29, 33, 37, 41, 45, 48
- bold_p.tbl_uvregression, 5, 9, 15, 21, 23, 25, 32, 33, 38, 41, 45, 54
- car::Anova, 3–5
- crayon::strip_style(), 26
- geepack::geeglm, 7, 8
- glm, 52, 53
- glue::glue, 6, 27, 29, 31, 32, 47, 51, 53
- gtsummary_logo, 26
- inline_text, 8, 9, 26
- inline_text.tbl_regression, 4, 8, 21–23, 26, 27, 33, 36, 41, 44, 45
- inline_text.tbl_summary, 6, 10, 11, 14, 16, 21, 24, 26, 28, 33, 37, 41, 45, 48
- inline_text.tbl_survival, 26, 29, 33, 52
- inline_text.tbl_uvregression, 5, 9, 15, 21, 23, 25, 26, 31, 33, 38, 41, 45, 54

italicize_labels
 (*bold_italicize_labels_levels*),
 20
italicize_levels
 (*bold_italicize_labels_levels*),
 20
knit_print.tbl_merge (*print_gtsummary*),
 33
knit_print.tbl_regression
 (*print_gtsummary*), 33
knit_print.tbl_stack (*print_gtsummary*),
 33
knit_print.tbl_summary
 (*print_gtsummary*), 33
knit_print.tbl_survival
 (*print_gtsummary*), 33
knit_print.tbl_uvregression
 (*print_gtsummary*), 33
knitr::kable, 18

lm, 52, 53
lme4::glmer, 7, 8

modify_header, 4–6, 8–11, 14–16, 21–25,
 28–30, 32, 32, 36–38, 41, 44, 45, 48,
 52, 54

print.tbl_merge (*print_gtsummary*), 33
print.tbl_regression (*print_gtsummary*),
 33
print.tbl_stack (*print_gtsummary*), 33
print.tbl_summary (*print_gtsummary*), 33
print.tbl_survival (*print_gtsummary*), 33
print.tbl_uvregression
 (*print_gtsummary*), 33
print_gtsummary, 33

select_helpers, 35
sort_p.tbl_regression, 4, 8, 21–23, 28, 33,
 35, 41, 44, 45
sort_p.tbl_summary, 6, 10, 11, 14, 16, 21,
 24, 29, 33, 36, 41, 45, 48
sort_p.tbl_uvregression, 5, 9, 15, 21, 23,
 25, 32, 33, 37, 41, 45, 54
stats::glm, 7, 8
stats::p.adjust, 14, 15
style_percent, 38
style_pvalue, 11, 13–15, 28, 39, 43, 53
style_ratio, 39, 43, 53
style_sigfig, 40, 40, 43, 53
survival::coxph, 7, 8, 53

tbl_merge, 4–6, 8–11, 14–16, 21–25, 28, 29,
 32, 33, 36–38, 41, 44, 45, 48, 54
tbl_regression, 4, 7, 8, 17–23, 27, 28,
 33–36, 41, 42, 45, 52
tbl_stack, 4–6, 8–11, 14–16, 21–25, 28, 29,
 32, 33, 36–38, 41, 44, 44, 48, 54
tbl_summary, 6, 10–14, 16–21, 24, 28, 29, 33,
 34, 36, 37, 41, 45, 46, 49
tbl_summary_, 49
tbl_survival, 17, 18, 20, 29, 50
tbl_survival.survfit, 30, 33, 50, 50
tbl_uvregression, 5, 7–9, 13, 15, 17, 18, 20,
 21, 23, 25, 31–34, 37, 38, 41, 45, 52
tibble, 19
trial, 55