Package ‘tidysummary’

July 15, 2025

Title An Elegant Approach to Summarizing Clinical Data

Version 0.1.0

Description Streamlines the analysis of clinical data by automatically selecting appropriate statistical descriptions and inference methods based on variable types. For method details see Motulsky H J (2016) <`https://www.graphpad.com/guides/prism/10/statistics/index.htm`> and d'Agostino R B (1971) <doi:10.1093/biomet/58.2.341>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports car, cli, dplyr, fBasics, glue, qqplotr, rlang, stats, stringr, tibble, tidyplots, tidyr

Suggests knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 4.1.0)

NeedsCompilation no

Author Xiang Li [aut, cre]

Maintainer Xiang Li <`htqqdd@126.com`>

Repository CRAN

Date/Publication 2025-07-15 07:00:02 UTC

Contents

`add_p` *Add statistical test results to summary data*

Description

Calculates and appends p-values with optional statistical details to a summary table based on variable types and group comparisons. Handles both continuous and categorical variables with appropriate statistical tests.

Usage

```
add_p(
  summary,
  digit = 3,
  asterisk = FALSE,
  add_method = FALSE,
  add_statistic_name = FALSE,
  add_statistic_value = FALSE
)
```

Arguments

`summary`	A data frame that has been processed by `add_summary()`.
`digit`	A numeric determine decimal. Accepts: • 3:convert to 3 decimal, default • 4:convert to 4 decimal
`asterisk`	Logical indicating whether to show asterisk significance markers.
`add_method`	Control parameter for display of statistical methods. Accepts: • `'code'`: Show method as codes according to order of appearance • `TRUE`/`'true'`: Show method text • `FALSE`/`'false'`: Not show method text
`add_statistic_name`	Logical indicating whether to include test statistic names.
`add_statistic_value`	Logical indicating whether to include test statistic values.

Value

A data frame merged with statistical test results, containing: - Variable names - Summary - Formatted p-values - Optional method names/codes - Optional statistic names/values

Examples

```
# `summary` is a data frame processed by `add_var()` and `add_summary()`:
data <- add_var(iris, var = c("Sepal.Length", "Species"), group = "Species")
summary <- add_summary(data)

# Add statistical test results
result <- add_p(summary)
```

`add_summary` *Add summary statistics to a add_var object*

Description

This function generates summary statistics for variables from a data frame that has been processed by `add_var()`, with options to format outputs.

Usage

```
add_summary(
  data,
  add_overall = TRUE,
  continuous_format = NULL,
  norm_continuous_format = "{mean} ± {SD}",
  unnorm_continuous_format = "{median} ({Q1}, {Q3})",
  categorical_format = "{n} ({pct})",
  binary_show = "last",
  digit = 2
)
```

Arguments

`data`	A data frame that has been processed by `add_var()`.
`add_overall`	Logical indicating whether to include an "Overall" summary column. `TRUE`, by default.
`continuous_format`	Format string to override both normal/abnormal continuous formats. Accepted placeholders are `{mean}`, `{SD}`, `{median}`, `{Q1}`, `{Q3}`.
`norm_continuous_format`	Format string for normally distributed continuous variables. Default is `"{mean} ± {SD}"`. Accepted placeholders same as `continuous_format`.
`unnorm_continuous_format`	Format string for non-normal continuous variables. Default is `"{median} ({Q1}, {Q3})"`. Accepted placeholders same as `continuous_format`.
`categorical_format`	Format string for categorical variables. Default is `"{n} ({pct})"`. Accepted placeholders are `{n}` and `{pct}`.

`binary_show`	Display option for binary variables: • `"first"`: show only first level • `"last"`: show only last level, default • `"all"`: show all levels
`digit`	digit A numeric determine decimal.

Value

A data frame containing summary statistics with the following columns:

- `variable`: Variable name
- `Overall (n=X)`: Summary statistics for all data, if `add_overall=TRUE`
- Group-specific columns named `[group] (n=X)` with summary statistics

Examples

```
# `data` is a data frame processed by `add_var()`:
data <- add_var(iris, var = c("Sepal.Length", "Species"), group = "Species")
# Add summary statistics
result <- add_summary(data, add_overall = TRUE)
result <- add_summary(data, continuous_format = "{mean}, ({SD})")
```

`add_var` *Prepare variables for add_summary*

Description

This function processes a dataset for statistical analysis by categorizing variables into continuous and categorical types. It automatically handles normality checks, equality of variances checks, and expected frequency assumptions checks.

Usage

```
add_var(data, var = NULL, group = "group", norm = "auto", center = "median")
```

Arguments

`data`	A data frame containing the variables to analyze, with variables at columns and observations at rows.
`var`	A character vector of variable names to include. If `NULL`, by default, all columns except the `group` column will be used.
`group`	A character string specifying the grouping variable in `data`. If not specified, `'group'`, by default.
`norm`	Control parameter for normality tests. Accepts:

	• `'auto'`: Automatically decide based on p-values, but the same as `'ask'` when n > 1000, default • `'ask'`: Show p-values, plots QQ plots and prompts for decision • `TRUE`/`'true'`: Always assuming data are normally distributed • `FALSE`/`'false'`: Always assuming data are non-normally distributed
`center`	A character string specifying the `center` to use in Levene's test for equality of variances. Default is `'median'`, which is more robust than the mean.

Value

A modified data frame with an attribute `'add_var'` containing a list of categorized variables and their properties:

- `var`: List of categorized variables:
  - `valid`: All valid variable names after checks
  - `continuous`: Sublist of continuous variables (further divided by normality/equal variance)
  - `categorical`: Sublist of categorical variables (further divided by ordered/expected frequency)
- `group`: Grouping variable name
- `overall_n`: Total number of observations
- `group_n`: Observation counts per group
- `group_nlevels`: Number of groups
- `group_levels`: Group level names
- `norm`: Normality check method used

Examples

```
data <- add_var(iris, var = c("Sepal.Length", "Species"), group = "Species")
```

`equal_test` *Test for Equality of Variances*

Description

Performs Levene's test to assess equality of variances between groups.

Usage

```
equal_test(data, var, group, center = "median")
```

Arguments

`data`	A data frame containing the variables to be tested.
`var`	A character string specifying the numeric variable in `data` to test.
`group`	A character string specifying the grouping variable in `data`.
`center`	A character string specifying the `center` to use in Levene's test. Default is `'median'`, which is more robust than the mean.

Value

Logical value:

- `TRUE`: Variances are equal, p-value more than 0.05
- `FALSE`: Variances are unequal or an error occurred during testing

Methodology for Equality of Variances

Levene's test is the default method adopted in SPSS, the original Levene's test select center = mean, but here select center = median for a more robust test

Examples

```
equal_test(iris, "Sepal.Length", "Species")
```

`format_p` *Format p-values with significance markers*

Description

Formats p-values as strings with specified precision and optional significance asterisks.

Usage

```
format_p(p, digit = 3, asterisk = FALSE)
```

Arguments

`p`	A numeric p-value between 0 and 1.
`digit`	A numeric determine decimal. Accepts: • 3:convert to 3 decimal, default • 4:convert to 4 decimal
`asterisk`	Logical indicating whether to return significance asterisks.

Value

Character of formatted p-value or asterisks.

Examples

```
format_p(0.00009, 4)
format_p(0.03, 3)
format_p(0.02, asterisk = TRUE)
```

`normal_test` *Perform normality test on a variable*

Description

Conducts normality tests for a specified variable, optionally by group. Supports automatic testing and interactive visualization.

Usage

```
normal_test(data = NULL, var = NULL, group = NULL, norm = "auto")
```

Arguments

`data`	A data frame containing the variables to be tested.
`var`	A character string specifying the numeric variable in `data` to test.
`group`	A character string specifying the grouping variable in `data`. If `NULL`, treated as one group.
`norm`	Control parameter for test behavior. Accepts: • `'auto'`: Automatically decide based on p-values, but the same as `'ask'` when n > 1000, default • `'ask'`: Show p-values, plots QQ plots and prompts for decision • `TRUE`/`'true'`: Always returns `TRUE` • `FALSE`/`'false'`: Always returns `FALSE`

Value

A logical value:

- `TRUE`: data are normally distributed
- `FALSE`: data are not normally distributed

Methodology for p-values

Automatically selects test based on sample size per group:

- n < 3: Too small, assuming non-normal
- (3, 50] Shapiro-Wilk test
- (50, 1000]: D'Agostino Chi2 test, instead of Kolmogorov-Smirnov test
- n > 1000: Show p-values, plots QQ plots and prompts for decision

Examples

```
normal_test(iris, "Sepal.Length", "Species", norm = "auto")
normal_test(iris, "Sepal.Length", "Species", norm = TRUE)
```

`small_test` *Check Sample Size Adequacy for Chi-Squared Test*

Description

This function determines if a contingency table meets the expected frequency assumptions for a valid chi-squared test. It categorizes the data into "not_small", "small", or "very_small" based on sample size and expected frequencies.

Usage

```
small_test(data, var, group)
```

Arguments

`data`	A data frame containing the variables to be tested.
`var`	A character string specifying the factor variable in `data` to test.
`group`	A character string specifying the grouping variable in `data`.

Value

A character string with one of three values:

- `"not_small"`: Sample size more than or euqal to 40 and all expected frequencies more than or euqal to 5
- `"small"`: Sample size more than or euqal to 40, all expected frequencies more than or euqal to 1 and at least one <5, only for 2*2 contingency tables
- `"very_small"`: Other conditions, including sample size <40 or any expected frequency <1

Examples

```
df <- data.frame(
  category = factor(c("A", "B", "A", "B")),
  group    = factor(c("X", "X", "Y", "Y"))
)
small_test(data = df, var = "category", group = "group")
```

Index