Package 'cardx'

March 18, 2024

Title Extra Analysis Results Data Utilities

Version 0.1.0

Description Create extra Analysis Results Data (ARD) summary objects. The package supplements the simple ARD functions from the 'cards' package, exporting functions to put statistical results in the ARD format. These objects are used and re-used to construct summary tables, visualizations, and written reports.

License Apache License 2.0

URL https://github.com/insightsengineering/cardx

BugReports https://github.com/insightsengineering/cardx/issues

Depends R (>= 4.1)

Imports cards (>= 0.1.0), cli (>= 3.6.1), dplyr (>= 1.1.2), glue (>= 1.6.2), rlang (>= 1.1.1), tidyr (>= 1.3.0)

Suggests broom (>= 1.0.5), broom.helpers (>= 1.13.0), spelling, testthat (>= 3.2.0), withr

Config/Needs/website insightsengineering/nesttemplate

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Language en-US

RoxygenNote 7.3.1

NeedsCompilation no

Author Daniel Sjoberg [aut, cre], F. Hoffmann-La Roche AG [cph, fnd]

Maintainer Daniel Sjoberg <danield.sjoberg@gmail.com>

Repository CRAN

Date/Publication 2024-03-18 14:30:02 UTC

R topics documented:

`ard_chisqtest` *ARD Chi-squared Test*

Description

Analysis results data for Pearson's Chi-squared Test. Calculated with `chisq.test(x = data[[variable]], y = data[[by]], ...)`

Usage

```
ard_chisqtest(data, by, variable, ...)
```

Arguments

`data`	(`data.frame`) a data frame.
`by, variable`	(`tidy-select`) column names to compare
`...`	additional arguments passed to `fisher.test(...)`

Value

ARD data frame

Examples

```
cards::ADSL |>
  ard_chisqtest(by = "ARM", variable = "AGEGR1")
```

ard_fishertest *ARD Fisher's Exact Test*

Description

Analysis results data for Fisher's Exact Test. Calculated with `fisher.test(x = data[[variable]], y = data[[by]], ...)`

Usage

```
ard_fishertest(data, by, variable, ...)
```

Arguments

`data`	(`data.frame`) a data frame.
`by, variable`	(`tidy-select`) column names to compare
`...`	additional arguments passed to `fisher.test(...)`

Value

ARD data frame

Examples

```
cards::ADSL[1:30, ] |>
  ard_fishertest(by = "ARM", variable = "AGEGR1")
```

ard_kruskaltest *ARD Kruskal-Wallis Test*

Description

Analysis results data for Kruskal-Wallis Rank Sum Test.

Calculated with `kruskal.test(data[[variable]], data[[by]], ...)`

Usage

```
ard_kruskaltest(data, by, variable)
```

Arguments

`data`	(`data.frame`) a data frame.
`by`	(tidy-select) column name to compare by
`variable`	(tidy-select) column name to be compared

Value

ARD data frame

Examples

```
cards::ADSL |>
  ard_kruskaltest(by = "ARM", variable = "AGE")
```

`ard_mcnemartest` *ARD McNemar's Test*

Description

Analysis results data for McNemar's statistical test.

Usage

```
ard_mcnemartest(data, by, variable, ...)
```

Arguments

`data`	(`data.frame`) a data frame. See below for details.
`by`	(tidy-select) column name to compare by.
`variable`	(tidy-select) column name to be compared.
`...`	arguments passed to `stats::mcnemar.test(...)`

Details

For the `ard_mcnemartest()` function, the data is expected to be one row per subject. The data is passed as `stats::mcnemar.test(x = data[[variable]], y = data[[by]], ...)`. Please use `table(x = data[[variable]], y = data[[by]])` to check the contingency table.

Value

ARD data frame

Examples

```
cards::ADSL |>
  ard_mcnemartest(by = "SEX", variable = "EFFFL")
```

`ard_moodtest` *ARD Mood Test*

Description

Analysis results data for Mood two sample test of scale. Note this not to be confused with the Brown-Mood test of medians.

Usage

```
ard_moodtest(data, by, variable, ...)
```

Arguments

`data`	(`data.frame`) a data frame. See below for details.
`by`	(`tidy-select`) column name to compare by.
`variable`	(`tidy-select`) column name to be compared.
`...`	arguments passed to `mood.test(...)`

Details

For the `ard_moodtest()` function, the data is expected to be one row per subject. The data is passed as `mood.test(data[[variable]] ~ data[[by]], ...)`.

Value

ARD data frame

Examples

```
cards::ADSL |>
  ard_moodtest(by = "SEX", variable = "AGE")
```

`ard_proportion_ci` *ARD Proportion Confidence Intervals*

Description

[Experimental]
Calculate confidence intervals for proportions.

Usage

```
ard_proportion_ci(
  data,
  variables,
  by = dplyr::group_vars(data),
  conf.level = 0.95,
  strata,
  weights = NULL,
  max.iterations = 10,
 method = c("waldcc", "wald", "clopper-pearson", "wilson", "wilsoncc", "strat_wilson",
    "strat_wilsoncc", "agresti-coull", "jeffreys")
)
```

Arguments

`data`	(`data.frame`) a data frame
`variables`	(`tidy-select`) columns to include in summaries. Columns must be class `<logical>` or `<numeric>` values coded as `c(0, 1)`.
`by`	(`tidy-select`) columns to stratify calculations by
`conf.level`	(`numeric`) a scalar in `(0, 1)` indicating the confidence level. Default is `0.95`
`strata, weights, max.iterations`	arguments passed to `proportion_ci_strat_wilson()`, when `method='strat_wilson'`
`method`	(`string`) string indicating the type of confidence interval to calculate. Must be one of 'waldcc', 'wald', 'clopper-pearson', 'wilson', 'wilsoncc', 'strat_wilson', 'strat_wilsoncc', 'agresti-coull', 'jeffreys'. See `?proportion_ci` for details.

Value

an ARD data frame

Examples

```
ard_proportion_ci(mtcars, variables = c(vs, am), method = "wilson")
```

ard_regression *Regression ARD*

Description

Function takes a regression model object and converts it to a ARD structure using the broom.helpers package.

Usage

```
ard_regression(x, ...)

## Default S3 method:
ard_regression(x, tidy_fun = broom.helpers::tidy_with_broom_or_parameters, ...)
```

Arguments

x	regression model object
...	Arguments passed to broom.helpers::tidy_plus_plus()
tidy_fun	(function) a tidier. Default is broom.helpers::tidy_with_broom_or_parameters

Value

data frame

Examples

```
lm(AGE ~ ARM, data = cards::ADSL) |>
  ard_regression(add_estimate_to_reference_rows = TRUE)
```

ard_ttest *ARD t-test*

Description

Analysis results data for paired and non-paired t-tests.

Usage

```
ard_ttest(data, by, variable, ...)

ard_paired_ttest(data, by, variable, id, ...)
```

Arguments

data	(data.frame) a data frame. See below for details.
by	(tidy-select) column name to compare by
variable	(tidy-select) column name to be compared
...	arguments passed to t.test(...)
id	(tidy-select) column name of the subject or participant ID

Details

For the ard_ttest() function, the data is expected to be one row per subject. The data is passed as t.test(data[[variable]] ~ data[[by]], paired = FALSE, ...).

For the ard_paired_ttest() function, the data is expected to be one row per subject per by level. Before the t-test is calculated, the data are reshaped to a wide format to be one row per subject. The data are then passed as t.test(x = data_wide[[<by level 1>]], y = data_wide[[<by level 2>]], paired = TRUE, .

Value

ARD data frame

Examples

```
cards::ADSL |>
  dplyr::filter(ARM %in% c("Placebo", "Xanomeline High Dose")) |>
  ard_ttest(by = ARM, variable = AGE)

# constructing a paired data set,
# where patients receive both treatments
cards::ADSL[c("ARM", "AGE")] |>
  dplyr::filter(ARM %in% c("Placebo", "Xanomeline High Dose")) |>
  dplyr::mutate(.by = ARM, USUBJID = dplyr::row_number()) |>
  dplyr::arrange(USUBJID, ARM) |>
  ard_paired_ttest(by = ARM, variable = AGE, id = USUBJID)
```

---

ard_wilcoxtest *ARD Wilcoxon Rank-Sum Test*

Description

Analysis results data for paired and non-paired Wilcoxon Rank-Sum tests.

Usage

```
ard_wilcoxtest(data, by, variable, ...)

ard_paired_wilcoxtest(data, by, variable, id, ...)
```

Arguments

data	(data.frame) a data frame. See below for details.
by	(tidy-select) column name to compare by.
variable	(tidy-select) column name to be compared.
...	arguments passed to wilcox.test(...)
id	(tidy-select) column name of the subject or participant ID.

Details

For the ard_wilcoxtest() function, the data is expected to be one row per subject. The data is passed as wilcox.test(data[[variable]] ~ data[[by]], paired = FALSE, ...).

For the ard_paired_wilcoxtest() function, the data is expected to be one row per subject per by level. Before the test is calculated, the data are reshaped to a wide format to be one row per subject. The data are then passed as wilcox.test(x = data_wide[[<by level 1>]], y = data_wide[[<by level 2>]], paired =

Value

ARD data frame

Examples

```
cards::ADSL |>
  dplyr::filter(ARM %in% c("Placebo", "Xanomeline High Dose")) |>
  ard_wilcoxtest(by = "ARM", variable = "AGE")

# constructing a paired data set,
# where patients receive both treatments
cards::ADSL[c("ARM", "AGE")] |>
  dplyr::filter(ARM %in% c("Placebo", "Xanomeline High Dose")) |>
  dplyr::mutate(.by = ARM, USUBJID = dplyr::row_number()) |>
  dplyr::arrange(USUBJID, ARM) |>
  ard_paired_wilcoxtest(by = ARM, variable = AGE, id = USUBJID)
```

proportion_ci *Functions for Calculating Proportion Confidence Intervals*

Description

Functions to calculate different proportion confidence intervals for use in `ard_proportion()`.

Usage

```
proportion_ci_wald(x, conf.level = 0.95, correct = FALSE)

proportion_ci_wilson(x, conf.level = 0.95, correct = FALSE)

proportion_ci_clopper_pearson(x, conf.level = 0.95)

proportion_ci_agresti_coull(x, conf.level = 0.95)

proportion_ci_jeffreys(x, conf.level = 0.95)

proportion_ci_strat_wilson(
  x,
  strata,
  weights = NULL,
  conf.level = 0.95,
  max.iterations = 10L,
  correct = FALSE
)
```

Arguments

`x`	vector of a binary values, i.e. a logical vector, or numeric with values `c(0, 1)`
`conf.level`	(`numeric`) a scalar in `(0, 1)` indicating the confidence level. Default is `0.95`
`correct`	(`flag`) include the continuity correction. For further information, see for example `stats::prop.test()`.
`strata`	(`factor`) variable with one level per stratum and same length as `x`.
`weights`	(`numeric` or `NULL`) weights for each level of the strata. If `NULL`, they are estimated using the iterative algorithm that minimizes the weighted squared length of the confidence interval.
`max.iterations`	(`count`) maximum number of iterations for the iterative procedure used to find estimates of optimal weights.

Value

Confidence interval of a proportion.

Functions

- `proportion_ci_wald()`: Calculates the Wald interval by following the usual textbook definition for a single proportion confidence interval using the normal approximation.
- `proportion_ci_wilson()`: Calculates the Wilson interval by calling `stats::prop.test()`. Also referred to as Wilson score interval.
- `proportion_ci_clopper_pearson()`: Calculates the Clopper-Pearson interval by calling `stats::binom.test()`. Also referred to as the `exact` method.
- `proportion_ci_agresti_coull()`: Calculates the `Agresti-Coull` interval (created by `Alan Agresti` and `Brent Coull`) by (for 95% CI) adding two successes and two failures to the data and then using the Wald formula to construct a CI.
- `proportion_ci_jeffreys()`: Calculates the Jeffreys interval, an equal-tailed interval based on the non-informative Jeffreys prior for a binomial proportion.
- `proportion_ci_strat_wilson()`: Calculates the stratified Wilson confidence interval for unequal proportions as described in Xin YA, Su XG. Stratified Wilson and Newcombe confidence intervals for multiple binomial proportions. *Statistics in Biopharmaceutical Research*. 2010;2(3).

Examples

```
x <- c(
  TRUE, TRUE, TRUE, TRUE, TRUE,
  FALSE, FALSE, FALSE, FALSE, FALSE
)

proportion_ci_wald(x, conf.level = 0.9)
proportion_ci_wilson(x, correct = TRUE)
proportion_ci_clopper_pearson(x)
proportion_ci_agresti_coull(x)
proportion_ci_jeffreys(x)
# Stratified Wilson confidence interval with unequal probabilities

set.seed(1)
rsp <- sample(c(TRUE, FALSE), 100, TRUE)
strata_data <- data.frame(
  "f1" = sample(c("a", "b"), 100, TRUE),
  "f2" = sample(c("x", "y", "z"), 100, TRUE),
  stringsAsFactors = TRUE
)
strata <- interaction(strata_data)
n_strata <- ncol(table(rsp, strata)) # Number of strata

proportion_ci_strat_wilson(
  x = rsp, strata = strata,
  conf.level = 0.90
)
```

```
# Not automatic setting of weights
proportion_ci_strat_wilson(
  x = rsp, strata = strata,
  weights = rep(1 / n_strata, n_strata),
  conf.level = 0.90
)
```

Index