Package 'DescrTab2'

March 15, 2021

Type Package

Title Publication Quality Descriptive Statistics Tables

Version 2.0.7

Date 2021-03-11

Description

Provides functions to create descriptive statistics tables for continuous and categorical variables. By default, summary statistics such as mean, standard deviation, quantiles, minimum and maximum for continuous
variables and relative and absolute frequencies for categorical variables are calculated. 'DescrTab2' features a sophisticated algorithm to
choose appropriate test statistics for your data and provides p-
values. On top of this, confidence intervals for group
differences of appropriated summary measures are automatically produces for two-
group comparison.
Tables generated by 'DescrTab2' can be integrated in a variety of document formats, includ-
ing .html, .tex and .docx documents.
'DescrTab2' also allows printing tables to console and saving table objects for later use.

Depends R (>= 4.0.0)

Suggests testthat, covr, knitr, rmarkdown, tidyverse, here, shiny,
Exact

Imports stats, utils, nlme, exact2x2, DescTools, dplyr, rlang, tibble,
stringr, forcats, magrittr, tidyselect, scales, cli,
kableExtra, flextable, officer

VignetteBuilder knitr

License GPL (>= 3)

Copyright This package contains codes copyrighted by third parites.
See file COPYRIGHTS for details.

LazyLoad true

RoxygenNote 7.1.1

Encoding UTF-8

URL https://imbi-heidelberg.github.io/DescrTab2/

R **topics documented:**

---

descr   *Calculate descriptive statistics*

Description

Generate a list of descriptive statistics. By default, the function calculates summary statistics such as mean, standard deviation, quantiles, minimum and maximum for continuous variables and relative and absolute frequencies for categorical variables. Also calculates p-values for an appropriately chosen statistical test. For two-group comparisons, confidence intervals for appropriate summary measures of group differences are calculated aswell. In particular, Wilson score intervals [1] from prop.test are used for categorical variables with 2 levels, confidence intervals from t.test are used for continuous variables and confidence intervals for the Hodge-Lehman estimator [2] from wilcox.test are used for ordinal variables.

Usage

```
descr(
  dat,
  group = NULL,
  group_labels = list(),
  var_labels = list(),
  var_options = list(),
  summary_stats_cont = list(N = DescrTab2:::.N, Nmiss = DescrTab2:::.Nmiss, mean =
    DescrTab2:::.mean, sd = DescrTab2:::.sd, median = DescrTab2:::.median, Q1 =
    DescrTab2:::.Q1, Q3 = DescrTab2:::.Q3, min = DescrTab2:::.min, max =
    DescrTab2:::.max),
  summary_stats_cat = list(),
  format_summary_stats = list(N = function(x) format(x, digits = 2, scientific = 3),
    Nmiss = function(x) format(x, digits = 2, scientific = 3), mean = function(x)
    format(x, digits = 2, scientific = 3), sd = function(x) format(x, digits = 2,
   scientific = 3), median = function(x) format(x, digits = 2, scientific = 3), Q1 =
   function(x) format(x, digits = 2, scientific = 3), Q3 = function(x) format(x, digits
   = 2, scientific = 3), min = function(x) format(x, digits = 2, scientific = 3), max =
   function(x) format(x, digits = 2,    scientific = 3), CI = function(x) format(x,
    digits = 2, scientific = 3)),
  format_p = scales::pvalue_format(),
  format_options = list(print_p = TRUE, print_CI = TRUE, combine_mean_sd = FALSE,
   combine_median_Q1_Q3 = FALSE, omit_Nmiss_if_0 = TRUE, omit_missings_in_group = TRUE,
    percent_accuracy = NULL, percent_suffix = "%", row_percent = FALSE,
    Nmiss_row_percent = FALSE, absolute_relative_frequency_mode = c("both",
   "only_absolute", "only_relative"), omit_missings_in_categorical_var = FALSE,
    categorical_missing_percent_mode = c("no_missing_percent",
    "missing_as_regular_category", "missing_as_separate_category")),
  test_options = list(paired = FALSE, nonparametric = FALSE, exact = FALSE, indices =
   c(), include_group_missings_in_test = FALSE, include_categorical_missings_in_test =
    FALSE, test_override = NULL, additional_test_args = list()),
  reshape_rows = list(`Q1 - Q3` = list(args = c("Q1", "Q3"), fun = function(Q1, Q3)
    paste0(Q1, " -- ", Q3)), `min - max` = list(args = c("min", "max"), fun =
    function(min, max) paste0(min, " -- ", max))),
  ...
)
```

Arguments

dat	Data frame or tibble. The data set to be analyzed. Can contain continuous or factor (also ordered) variables.
group	name (as character) of the group variable in dat.
group_labels	named list of labels for the levels of the group variable in dat.
var_labels	named list of variable labels.
var_options	A named list of lists. For each variable, you can have special options that apply only to that variable. These options are specified in this argument. See the details and examples for more explanation.

summary_stats_cont

    named list of summary statistic functions to be used for numeric variables.

summary_stats_cat

    named list of summary statistic function to be used for categorical variables.

format_summary_stats

    named list of formatting functions for summary statistics.

format_p   formatting function for p-values.

format_options named list of formatting options.

test_options   A named list of test options.

reshape_rows   A named list of lists. Describes how to combine different summary statistics into the same row.

...   further argument to be passed along

Value

Returns a A `DescrList` object, which is a named list of descriptive statistics which can be passed along to the print function to create pretty summary tables.

Labels

`group_labels` and `var_labels` need to be named lists of character elements. The names of the list elements have to match the variable names in your dataset. The values of the list elements are the labels that will be assigned to these variables when printing.

Custom summary statistics

`summary_stats_cont` and `summary_stats_cat` are both named lists of functions. The names of the list elements are what will be displayed in the leftmost column of the descriptive table. These functions should take a vector and return a value.

Each summary statistic has to have an associated formatting function in the `format_summary_stats` list. The functions in `format_summary_stats` take a numeric value and convert it to a character string, e.g. 0.2531235 -> "0.2".

The `format_p` function converts p-values to character strings, e.g. 0.05 -> "0.05" or 0.000001 -> "<0.001".

Formatting options

Further formatting options can be specified in the `format_options` list. It contains the following members:

- `print_p` (logical) controls whether to print the p-value column.

- `print_CI` (logical) controls whether to print the confidence intervals for group-differences.

- `combine_mean_sd` (logical) controls whether to combine the mean and sd row into one mean ± sd row. This is a shortcut argument for the specification of an appropriate entry in the `reshape_rows` argument.

- `combine_median_Q1_Q3` (logical) controls whether to combine the median, Q1 and Q3 row into one median (Q1, Q3) row. This is a shortcut argument for the specification of an appropriate entry in the `reshape_rows` argument.

- `omit_Nmiss_if_0` (logical) controls whether to omit the Nmiss row in continuous variables there are no missings in the variable.

- `omit_missings_in_group` (logical) controls whether to omit all observations where the group variable is missing.

- `percent_accuracy` (numeric) A number to round to. Use (e.g.) 0.01 to show 2 decimal places of precision. If NULL, the default, uses a heuristic that should ensure breaks have the minimum number of digits needed to show the difference between adjacent values. See documentation of scales::label_percent

- `percent_suffix` (character) the symbol to be used where "%" is appropriate, sensible choices are usually "%" (default) or "" (i.e., empty string)

- `row_percent` (logical) controls wheter percentages of regular categorical variables should be calculated column-wise (default) or row-wise

- `Nmiss_row_percent` (logical) controls whether percentages of the "Nmiss"-statistic (number of missing values) should be calculated column-wise (default) or row-wise

- `absolute_relative_frequency_mode` (character) controls how to display frequencies. It may be set to one of the following options:

  - `"both"` will display absolute and relative frequencies.
  - `"only_absolute"` will only display absolute frequencies.
  - `"only_relative"` will only display relative frequencies.

- `omit_missings_in_categorical_var` (logical) controls whether to omit missing values in categorical variables completely.

- `categorical_missing_percent_mode` (character) controls how to display percentages in categorical variables with a (Missing) category. It may be set to one of the following options:

  - `"no_missing_percent"` omits a percentage in the missing category entirely.
  - `"missing_as_regular_category"` treats (Missing) as a regular category for %-calculation This means that if You have three categories: "A" with 10 counts, "B" with 10 counts and "(Missing)" with 10 counts, they will become "A": 10 (33%), "B": 10 (33%), "(Missing)": 10 (33% purposes.)
  - `"missing_as_separat_category"` calculates (Missing) percentages with respect to all observations (i.e. #(Missing) / N), but calculates all other catetgory percentages with respect to the non-missing observations (e.g. #A / N_nonmissing). This means that if You have three categories: "A" with 10 counts, "B" with 10 counts and "(Missing)" with 10 counts, they will become "A": 10 (50%), "B": 10 (50%), "(Missing)": 10 (33%)

Test options

`test_options` is a named list with test options. It's members `paired`, `nonparametric`, and `exact` (logicals) control which test in the corresponding situation. For details, check out the vignette: https://imbi-heidelberg.github.io/DescrTab2/articles/test_choice_tree_pdf.pdf. The `test_options = list(test_override="<some test name>")` option can be specified to force usage of a specific test. This will produce errors if the data does not allow calculation of that specific test, so be wary. Use `print_test_names()` to see a list of all available test names. If `paired = TRUE` is specified, you need to supply an index variable `indices` that specifies which datapoints in your dataset are paired. `indices` may either be a length one character vector that

describes the name of the index variable in your dataset, or a vector containing the respective indices. See https://imbi-heidelberg.github.io/DescrTab2/articles/usage_guide.html#Paired-observations-1 for a bit more explanation. The optional list `additional_test_args` can be used to pass arguments along to test functions, e.g. `additional_test_args=list(correct=TRUE)` will request continuity correction if available.

Customization for single variables

The `var_options` list can be used to conduct customizations that should only apply to a single variable and leave the rest of the table unchanged.
`var_options` is a list of named lists. This means that each member of `var_options` is itself a list again. The names of the list elements of `var_options` determine the variables to which the options will apply. Let's say you have an age variable in your dataset. To change 'descr' options only for age, you will need to pass a list of the form `var_options = list(age = list(<Your options here>))`.
You can replace `<Your options here>` with the following options:

- `label` a character string containing the label for the variable
- `summary_stats` a list of summary statistics. See section "Custom summary statistics"
- `format_summary_stats` a list of formatting functions for summary statistics. See section "Custom summary statistics"
- `format_p` a function to format p-values. See section "Custom summary statistics"
- `format_options` a list of formatting options. See section "Formatting options"
- `test_options` a list of test options. See section "Test options"
- `test_override` manually specify the name of the test you want to apply. You can see a list of choices by typing `print_test_names()`. Possible choices are:
  - `"Cochrans Q test"`
  - `"McNemars test"`
  - `"Chi-squared goodness-of-fit test"`
  - `"Pearsons chi-squared test"`
  - `"Exact McNemars test"`
  - `"Boschloos test"`
  - `"Wilcoxon one-sample signed-rank test"`
  - `"Mann-Whitney U test"`
  - `"Kruskal-Wallis one-way ANOVA"`
  - `"Students paired t-test"`
  - `"Mixed model ANOVA"`
  - `"Students one-sample t-test"`
  - `"Welchs two-sample t-test"`
  - `"F-test (ANOVA)"`

Combining rows

The `reshape_rows` argument offers a framework for combining multiple rows of the output table into a single one. `reshape_rows` is a named list of lists. The names of it's member-lists determine

the name that will be displayed as the name of the combined summary stats in the table (e.g. "mean ± sd "). The member lists need to contain two elements: `args`, contains the names of the summary statistics to be combined as characters, and `fun` which contains a function to combine these summary stats. The argument names of this function need to match the character strings specified in `args`. Check out the default options for an exemplary definition.

References

[1] Wilson, E. B. (1927). "Probable inference, the law of succession, and statistical inference". Journal of the American Statistical Association. 22 (158): 209-212. doi:10.1080/01621459.1927.10502953. JSTOR 2276774

[2] Hodges, J. L.; Lehmann, E. L. (1963). "Estimation of location based on ranks". Annals of Mathematical Statistics. 34 (2): 598-611. doi:10.1214/aoms/1177704172. JSTOR 2238406. MR 0152070. Zbl 0203.21105. PE euclid.aoms/1177704172

Examples

```
descr(iris)
DescrList <- descr(iris)
DescrList$variables$results$Sepal.Length$Total$mean
print(DescrList)
descr(iris, "Species")
```

DescrTab2   *DescrTab2*

Description

Publication quality descriptive statistics tables with R

Details

Provides functions to create descriptive statistics tables for continuous and categorical variables. By default, summary statistics such as mean, standard deviation, quantiles, minimum and maximum for continuous variables and relative and absolute frequencies for categorical variables are calculated. 'DescrTab2' features a sophisticated algorithm to choose appropriate test statistics for your data and provides p-values. On top of this, confidence intervals for group differences of appropriated summary measures are automatically produces for two-group comparison. Tables generated by 'DescrTab2' can be integrated in a variety of document formats, including .html, .tex and .docx documents. 'DescrTab2' also allows printing tables to console and saving table objects for later use.

Check out our documentation online: https://imbi-heidelberg.github.io/DescrTab2/ or browse the help files in the Rstudio viewer. You can access the vignettes by typing: `browseVignettes("DescrTab2")`

---

farrington.manning   *Farrington-Manning test for rate difference*

Description

The Farrington-Manning test for rate differences can be used to compare the rate difference of successes between two groups to a preset value. It uses an explicit formula for the standard deviation of the test statistic under the null hypothesis [1].

Usage

```
farrington.manning(
  group1,
  group2,
  delta = 0,
  alternative = "greater",
  alpha = 0.025
)
```

Arguments

group1	a logical vector of data from group 1, where TRUE indicates a success
group2	a logical vector of data from group 2, where TRUE indicates a success
delta	the rate difference under the null hypothesis
alternative	character string indicating the alternative to use, either of "two.sided", "less", "greater"
alpha	the significance level (acceptable error of the first kind), a two-sided confidence intnerval is returned with confidence level 1 - 2*alpha, such that the lower bound is a valid one sided confidence interval at the confidence level 1 - alpha.

Details

The Farrington-Maning test for rate differences test the null hypothesis of

$$H_0 : p_1 - p_2 = \delta$$

for the "two.sided" alternative (or $\geq$ for the "greater" respectively $\leq$ for the "less" alternative). This formulation allows to specify non-inferiority and superiority test in a consistent manner:

non-inferiority for delta < 0 and alternative == "greater" the null hypothesis reads $H_0 : p_1 - p_2 \geq \delta$ and consequently rejection allows concluding that $p_1 \geq p_2 + \delta$ i.e. that the rate of success in group one is at least the success rate in group two plus delta - as delta is negagtive this is equivalent to the success rate of group 1 being at worst |delta| smaller than that of group 2.

superiority for delta >= 0 and alternative == "greater" the null hypothesis reads $H_0 : p_1 - p_2 \geq \delta$ and consequently rejection allows concluding that $p_1 \geq p_2 + \delta$ i.e. that the rate of success in group one is at least delta greater than the success rate in group two.

The confidence interval is always computed as two-sided, but with $1-2\alpha$ confidence level in case of a one-sided hypthesis. This means that the lower or upper vound are valid one-sided confidence bounds at level α in this case. The confidence interval is constructed by inverting the two-sided test directly.

Value

A list of class "htest" containing the following components:

statistic:	the value of the Z-statistic
parameter:	delta, rate difference (group 1 - group 2) under the null hypothesis
p.value:	the p-value for the Farrington-Manning test
null.value:	rate difference (group 1 - group 2) under the null
alternative:	a character string indicating the alternative hypothesis
method:	a character string indicating the exact method employed
data.name:	a character string giving the names of the data used
estimate:	the estimated rate difference (maximum likelihood)
conf.int:	a confidence interval for the rate difference
sample.size:	the total sample size used for the test

Author(s)

Kevin Kunzmann

References

[1] Farrington, Conor P., and Godfrey Manning. "Test statistics and sample size formulae for comparative binomial trials with null hypothesis of non-zero risk difference or non-unity relative risk." Statistics in medicine 9.12 (1990): 1447-1454.

Examples

```
x <- c(rep(TRUE, 20), rep(FALSE, 15))
y <- c(rep(TRUE, 30), rep(FALSE, 25))

farrington.manning(x, y, -.3)
```

---

format_freqs	*Make pretty frequencies*

Description

Make pretty frequencies

Usage

```
format_freqs(
  numerator,
  denominator = 1,
 absolute_relative_frequency_mode = c("both", "only_absolute", "only_relative"),
  percent_accuracy = NULL,
  percent_suffix = "%"
)
```

Arguments

numerator   numeric

denominator   numeric

absolute_relative_frequency_mode

                 "both","only_absolute" or "only_relative"

percent_accuracy

                 NULL or numeric

percent_suffix   "%" or "" (or something else)

Value

Character element of formatted frequencies

---

ignore_unused_args   *do.call but without an error for unused arguments*

Description

do.call but without an error for unused arguments

Usage

```
ignore_unused_args(what, args)
```

Arguments

what   either a function or a non-empty character string naming the function to be
                  called.

args   a list of arguments to the function call. The names attribute of args gives the
                  argument names.

Value

The result of the (evaluated) function call.

Examples

```
# works:
DescrTab2:::ignore_unused_args(chisq.test,
 list(x=factor(c(1,0,1,1,1,0)), y=factor(c(0,1,1,0,1,0)), abc=3 ) )

# would produce error:
# do.call(chisq.test, list(x=factor(c(1,0,1,1,1,0)), y=factor(c(0,1,1,0,1,0)), abc=3 ) )
```

---

print.DescrList   *S3 override for print function for DescrList objects.*

Description

This function takes a DescrList object and converts it to either a DescrPrintCharacter or DescrPrint-Numeric object, depending on the print_format option. This object is then printed in an appropriate format.

Usage

```
## S3 method for class 'DescrList'
print(x, print_format = options("print_format")[[1]], silent = FALSE, ...)
```

Arguments

x	A DescrList object returned from [descr]().
print_format	Possible values: "console" (default), "tex", "html", "word", "numeric"
silent	I TRUE, suppresses output to stdout.
...	further arguments to be passed along to print method

Details

There is no way to convert between DescrPrintCharacter and DescrPrintNumeric objects. The first type is for what you would usually want, the second type is mostly for debugging purposes. A DescrPrintCharacter object can be printed as html, tex code, as a flextable object or simply to the console.

Value

A DescrPrint object which can be printed in various formats.

You can use the print_format option to control the output type. If you use 'DescrTab2' inside an .Rmd document, you can set the clobal option option(print_format="tex") or option(print_format="html") or option(print_format="word") depending on your document type. This way, all your tables will be printed in the right format by default inside this document.

Examples

```
print(descr(iris), print_format="console")
print(descr(iris), print_format="tex")
print(descr(iris), print_format="html")
print(descr(iris), print_format="word")
print(descr(iris), print_format="numeric")
options(print_format="tex")
descr(iris)
options(print_format="console")
descr(iris)
DescrPrint <- print(descr(iris))
DescrPrint$variables$results$Sepal.Length$Total$mean
print(DescrPrint)
```

---

print_test_names   *Prints all possible tests names*

Description

Prints all possible tests names

Usage

```
print_test_names()
```

Value

Returns the names of all possible test names you can specify.

Examples

```
print_test_names()
```

---

test_cat   *Calculate a statistical test for a categorical variable.*

Description

Calculate a statistical test for a categorical variable.

Arguments

`var`	A variable (a vector).
`group`	A variable containing the grouping information.
`test_options`	Named list containing test options.
`test`	Name of a statistical test.
`var_name`	Name of variable to be tested (only used in warning messages).

Value

A list of test test results.

Examples

```
cont_var <- c(1,2,3)
test_cont(cont_var)
```

---

write_in_tmpfile_for_cran

Function that returns true in CRAN submission

Description

Function that returns true in CRAN submission

Usage

```
write_in_tmpfile_for_cran()
```

Value

TRUE for CRAN submission, FALSE otherwise

Index