Package 'sumvar'

May 9, 2026

Title Summarise and Explore Continuous, Categorical and Date Variables

Version 0.2.0

Description Explore continuous, date and categorical variables with summary statistics, visualisations, and frequency tables. Brings the ease and simplicity of the sum and tab commands from 'Stata' to 'R', including support for two-way cross-tabulations, hypothesis tests, duplicate and missing data exploration, and automated HTML or PDF exploratory reports.

Encoding UTF-8

RoxygenNote 7.3.3

Imports cli, dplyr, ggplot2, kableExtra, knitr, magrittr, patchwork, purrr, rlang, rmarkdown, scales, stats, tibble, tidyr, utils

Suggests testthat (>= 3.0.0), tinytex

Config/testthat/edition 3

URL https://github.com/alstockdale/sumvar, https://alstockdale.github.io/sumvar/

BugReports https://github.com/alstockdale/sumvar/issues

License MIT + file LICENSE

VignetteBuilder knitr

NeedsCompilation no

Author Alexander Stockdale [aut, cre, cph]

Maintainer Alexander Stockdale <a.stockdale@liverpool.ac.uk>

Repository CRAN

Date/Publication 2026-03-19 07:10:02 UTC

Contents

`dist_date` *Summarize and visualize a date variable*

Description

Summarises the minimum, maximum, median, and interquartile range of a date variable, optionally stratified by a grouping variable. Produces a histogram and (optionally) a density plot.

Usage

```
dist_date(data, var, by = NULL)
```

Arguments

`data`	A data frame or tibble.
`var`	The date variable to summarise.
`by`	Optional grouping variable.

Value

A tibble with summary statistics for the date variable.

See Also

`dist_sum` for continuous variables.

Examples

```
# Example ungrouped
df <- tibble::tibble(
  dt = as.Date("2020-01-01") + sample(0:1000, 100, TRUE)
)
dist_date(df, dt)

# Example grouped
df2 <- tibble::tibble(
  dt = as.Date("2020-01-01") + sample(0:1000, 100, TRUE),
  grp = sample(1:2, 100, TRUE)
)
dist_date(df2, dt, grp)
```

`dist_sum` *Explore a continuous variable*

Description

Summarises a continuous variable, returning a tibble of descriptive statistics and a plot. When a grouping variable is supplied, results are stratified by group.

For two groups, a t-test and Wilcoxon rank-sum test are reported. For three or more groups, a one-way ANOVA and Kruskal-Wallis test are reported.

Usage

```
dist_sum(data, var, by = NULL)
```

Arguments

`data`	The data frame or tibble
`var`	The continuous variable to summarise
`by`	Optional grouping variable

Value

A tibble with one row per group (or one row when ungrouped) containing the following columns:

`n` Number of non-missing observations.

`n_miss` Number of missing (NA) values.

`median` Median value.

`p25`, `p75` 25th and 75th percentiles (interquartile range boundaries).

`mean` Arithmetic mean.

`sd` Standard deviation.

`ci_lower`, `ci_upper` Lower and upper bounds of the 95% confidence interval for the mean. Uses the t-distribution when n < 30, and the Z-distribution when n >= 30.

`min`, `max` Minimum and maximum observed values.

`n_outliers` Count of values more than 1.5 x IQR below Q1 or above Q3 (Tukey fence method).

`shapiro_p` P-value from the Shapiro-Wilk test of normality. Returns `NA` when n < 3 or n > 5000 (outside the valid range of the test).

`normal` Logical. `TRUE` if `shapiro_p` > `0.05`, indicating no significant departure from normality at the 5% level.

`p_ttest` Shown when two groups are compared. P-value from an independent samples t-test, testing whether the means of the two groups differ. Assumes approximately normal distributions or large samples. All p-values are reported on the first row only; remaining rows contain `NA`.

`p_wilcox` Shown when two groups are compared. P-value from the Wilcoxon rank-sum test (Mann-Whitney U test), a non-parametric alternative to the t-test. Preferred over `p_ttest` when data are skewed, ordinal, or contain outliers, as it compares ranks rather than means and makes no distributional assumptions.

`p_anova` Shown when three or more groups are compared. P-value from a one-way analysis of variance (ANOVA) F-test, testing whether at least one group mean differs from the others. Assumes approximately normal distributions and equal variances across groups.

`p_kruskal` Shown when three or more groups are compared. P-value from the Kruskal-Wallis test, a non-parametric alternative to one-way ANOVA. Preferred over `p_anova` when data are skewed or the normality assumption is not met, as it compares rank distributions and makes no distributional assumptions.

Examples

```
example_data <- dplyr::tibble(id = 1:100, age = rnorm(100, mean = 30, sd = 10),
                              group = sample(c("a", "b", "c", "d"),
                              size = 100, replace = TRUE))
dist_sum(example_data, age, group)
example_data <- dplyr::tibble(id = 1:100, age = rnorm(100, mean = 30, sd = 10),
                             sex = sample(c("male", "female"),
                             size = 100, replace = TRUE))
dist_sum(example_data, age, sex)
summary <- dist_sum(example_data, age, sex) # Save summary statistics as a tibble.
```

---

dup *Explore duplicate and missing data*

Description

Provides an integer value for the number of duplicates found within a variable The function accepts an input from a dplyr pipe "%>%" and outputs the results as a tibble.

eg. example_data %>% dup(variable)

Usage

```
dup(data, var = NULL)
```

Arguments

`data`	The data frame or tibble
`var`	The variable to assess

Value

A tibble with the number and percentage of duplicate values found, and the number of missing values (NA), together with percentages.

Examples

```
example_data <- dplyr::tibble(id = 1:200, age = round(rnorm(200, mean = 30, sd = 50), digits=0))
example_data$age[sample(1:200, size = 15)] <- NA  # Replace 15 values with missing.
dup(example_data, age)
# It is also possible to pass a whole database to dup and it will explore all variables.
example_data <- dplyr::tibble(age = round(rnorm(200, mean = 30, sd = 50), digits=0),
                              sex = sample(c("Male", "Female"), 200, TRUE),
                    favourite_colour = sample(c("Red", "Blue", "Purple"), 200, TRUE))
example_data$age[sample(1:200, size = 15)] <- NA  # Replace 15 values with missing.
example_data$sex[sample(1:200, size = 32)] <- NA  # Replace 32 values with missing.
dup(example_data)
```

---

explorer *Explore all variables and generate an HTML or PDF summary report*

Description

Analyses a data frame or tibble, summarising all continuous, date and categorical variables, missing data and duplicate values, and produces an HTML or PDF report.

Usage

```
explorer(
  data,
  output_file = NULL,
  format = c("html", "pdf"),
  progress = TRUE,
  id_var = NULL
)
```

Arguments

`data`	A data frame or tibble to explore.
`output_file`	The name of the output file. Default uses `<df_name>_report.html` or `.pdf`
`format`	Output format, either `"html"` (default) or `"pdf"`.
`progress`	If `TRUE` (default), show a progress bar while building the report.
`id_var`	Character vector of column names to treat as IDs (not summarised).

Value

Outputs an html or PDF summary. Output in PDF typically takes longer.

For PDF output, a LaTeX distribution must be installed. TinyTeX is recommended. To install, run:

```
install.packages("tinytex")
tinytex::install_tinytex()
```

Examples

```
## Not run:
# Build example data from mtcars with some factors and a date column:
  cars_example <- mtcars %>%
    dplyr::mutate(
      across(c(vs, am, gear, carb, cyl), as.factor),
      date_var = as.Date("2025-06-01") +
        sample(-300:300, nrow(mtcars), replace = TRUE),
      id = dplyr::row_number()
    )
# To run explorer:
  explorer(mtcars)                  # with progress bar
  explorer(mtcars, progress = FALSE) # omit progress bar
  explorer(mtcars, format = "pdf")   # PDF output
  explorer(mtcars, format = "pdf", id_var = "id")   # Identify ID variable

## End(Not run)
```

`sumvar` *sumvar: Summarise Continuous and Categorical Variables in R*

Description

The sumvar package explores continuous and categorical variables. sumvar brings the ease and simplicity of the "sum" and "tab" functions from Stata to R.

- To explore a continuous variable, use `dist_sum()`. You can stratify by a grouping variable: `df %>% dist_sum(var, group)`
- To explore dates, use `dist_date()`; usage is the same as `dist_sum()`.
- To summarise a single categorical variable use `tab1()`, e.g. `df %>% tab1(var)`. For a two-way table, use `tab()`, e.g. `df %>% tab(var1, var2)`. Both include options for frequentist hypothesis tests.
- Explore duplicates and missing values with with `dup()`.

All functions are tidyverse/dplyr-friendly and accept the `%>%` pipe, outputting results as a tibble. You can save outputs for further manipulation, e.g. `summary <- df %>% dist_sum(var)`.

Author(s)

Maintainer: Alexander Stockdale <a.stockdale@liverpool.ac.uk> [copyright holder]

See Also

Useful links:

- https://github.com/alstockdale/sumvar
- https://alstockdale.github.io/sumvar/
- Report bugs at https://github.com/alstockdale/sumvar/issues

tab *Create a cross-tabulation of two categorical variables*

Description

Creates a cross-tabulation of two categorical variables with row or column percentages, row and column totals, and optional hypothesis tests. Prints a formatted table to the console (similar to Stata's `tab` command) with the column variable name displayed as a spanning header above its levels.

Usage

```
tab(
  data,
  variable1,
  variable2,
  show = c("row", "col", "n"),
  test = c("both", "chi", "exact", "none"),
  totals = TRUE,
  dp = 1
)
```

Arguments

`data`	The data frame or tibble.
`variable1`	The row variable (first categorical variable).
`variable2`	The column variable (second categorical variable).
`show`	What to display in each cell: `"row"` (default) shows `n (row%)`, `"col"` shows `n (col%)`, `"n"` shows counts only.
`test`	Hypothesis test(s) to report: `"both"` (default) runs both chi-squared and Fisher's exact; `"chi"` for chi-squared only; `"exact"` for Fisher's exact only (falls back to simulated p-value for large tables); `"none"` to suppress tests.
`totals`	If `TRUE` (default), includes a Total column (row totals) and a Total row (column totals). Set to `FALSE` to suppress both.
`dp`	Number of decimal places for percentages. Default is `1`.

Value

A wide-format tibble (invisibly) with:

- First column: levels of `variable1` (plus `"Total"` if `totals = TRUE`).
- For each level of `variable2`: `{level}_n` (integer count) and, when `show != "n"`, `{level}_pct` (numeric percentage).
- `total_n`: row totals (when `totals = TRUE`).
- `p_chi`, `p_fisher`: p-values on the first row, `NA` elsewhere (when `test = "both"`; individual tests add `test`, `statistic`, `p_value` instead).

Examples

```
example_data <- dplyr::tibble(
  group1 = sample(c("a", "b", "c"), 100, replace = TRUE),
  group2 = sample(c("male", "female"), 100, replace = TRUE)
)
tab(example_data, group1, group2)
tab(example_data, group1, group2, show = "col")
tab(example_data, group1, group2, test = "none")
result <- tab(example_data, group1, group2)
```

---

tab1 *Summarise a categorial variable*

Description

Summarises frequencies and percentages for a categorical variable.

The function accepts an input from a dplyr pipe "%>%" and outputs the results as a tibble. eg. example_data %>% tab1(variable)

Usage

```
tab1(data, variable, ..., dp = 1)
```

Arguments

data	The data frame or tibble
variable	The categorical variable you would like to summarise
...	Not used. Passing additional variables raises an informative error suggesting tab() for two-way tables.
dp	The number of decimal places for percentages (default=1)

Value

A tibble with frequencies and percentages

Examples

```
example_data <- dplyr::tibble(id = 1:100, group = sample(c("a", "b", "c", "d"),
                                                         size = 100, replace = TRUE))
example_data$group[sample(1:100, size = 10)] <- NA  # Replace 10 with missing
tab1(example_data, group)
summary <- tab1(example_data, group) # Save summary statistics as a tibble.
```

Index