Package 'dentomedical'

May 8, 2026

Type Package

Title Publication-Ready Descriptive, Bivariate, Regression, Correlation and Diagnostic Accuracy Tools for Medical and Dental Data

Version 0.2.0

Description The 'dentomedical' package provides a comprehensive suite of tools for medical and dental research. It includes automated descriptive statistics, bivariate analysis with intelligent test selection, logistic regression, and diagnostic accuracy assessment. All functions generate publication-ready tables using 'flextable', ensuring reproducibility and clarity suitable for manuscripts, reports, and clinical research workflows.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, stats, flextable, tibble, rlang, FSA, purrr, broom, tidyr

Depends R (>= 4.0.0)

URL https://github.com/umarhussain-git/dentomedical1

BugReports https://github.com/umarhussain-git/dentomedical/issues

NeedsCompilation no

Suggests testthat, knitr, rmarkdown

Author Umar Hussain [aut, cre], Nikolaos Pandis [aut]

Maintainer Umar Hussain <drumarhussain@gmail.com>

Repository CRAN

Date/Publication 2026-01-21 05:50:01 UTC

Contents

category 2
diag_accuracy 3
impute_missing 4
linreg 4
logreg 5
medical_data 6
recode_data 6
sum_cor 7
sum_crosstab 8
sum_norm 9
sum_posthoc 10
sum_stat 11
sum_stat_p 12
sum_stat_p_strata 13

Index **15**

`category` *Categorize a Numeric Variable into Custom Ranges*

Description

`category` creates a new categorical variable by splitting a numeric column into specified ranges. You can provide custom labels for each range, or it will default to the range values themselves.

Usage

```
category(data, var, level, new_var = NULL, labels = NULL)
```

Arguments

`data`	A data.frame or tibble containing the numeric variable.
`var`	The numeric variable to categorize (unquoted).
`level`	A character vector of numeric ranges, e.g., c("18-25", "26-35").
`new_var`	Optional. Name of the new categorical variable. Defaults to the variable name with "_group" appended.
`labels`	Optional. Custom labels for each category. Length must match `level`.

Value

A data.frame or tibble with a new categorical variable.

Examples

```
df <- data.frame(age = c(20, 28, 40, 55, 34, 10, 24, 55))

# Categorize without custom labels
category(df, var = age, level = c("10-25", "26-35", "36-50"))

# Categorize with custom labels
category(df, var = age, level = c("10-25", "26-35", "36-50"),
         labels = c("young", "adult", "old"))
```

`diag_accuracy` *Diagnostic Accuracy Metrics with Optional 2x2 Table*

Description

Calculates diagnostic accuracy measures (Sensitivity, Specificity, PPV, NPV, Accuracy, LR+, LR-, DOR) from a binary test and gold standard. Provides 95% confidence intervals using Wilson method for proportions and log method for ratios. Optionally, prints a descriptive 2x2 table.

Usage

```
diag_accuracy(data, test_col, gold_col, descriptive = FALSE)
```

Arguments

`data`	A data frame containing the test results and gold standard.
`test_col`	Character. Name of the column in `data` with test results ("positive"/"negative").
`gold_col`	Character. Name of the column in `data` with gold standard results ("positive"/"negative").
`descriptive`	Logical. If TRUE, prints a descriptive 2x2 table with counts (TN, TP, FP, FN). Default is FALSE.

Value

A `flextable` object summarizing diagnostic metrics with 95% CI. If `descriptive = TRUE`, also prints a 2x2 table of counts.

Examples

```
diagnostic_data <- data.frame(
  test = c("positive","negative","positive","
  negative","positive","negative","positive","negative"),
  goldstandard = c("positive","positive","negative",
  "negative","positive","negative","positive","negative")
)
diag_accuracy(diagnostic_data, test_col = "test",
gold_col = "goldstandard",
descriptive = FALSE)
```

impute_missing *Impute Missing Values in a Data Frame*

Description

This function imputes missing values in a data frame. For categorical variables (factor or character), missing values are replaced with the mode. For numeric variables, missing values can be imputed using the mean, median, or regression-based imputation. If no method is specified for numeric columns, missing values are left as NA.

Usage

```
impute_missing(data, method = NULL)
```

Arguments

data	A data frame containing numeric and/or categorical variables with missing values.
method	A character string specifying the imputation method for numeric columns. Options are "mean", "median", "regression", or NULL (default: NULL).

Value

A data frame with missing values imputed according to the specified method.

Examples

```
library(dplyr)
# Impute numeric columns using regression and categorical with mode
impute_missing(starwars, method = "regression")

# Impute numeric columns using mean
impute_missing(starwars, method = "mean")

# Impute numeric columns using median
impute_missing(starwars, method = "median")
```

linreg *Linear Regression Table with Univariable and Multivariable Analysis*

Description

Fits univariable and multivariable linear regression models for a continuous outcome, summarizing beta coefficients, 95% confidence intervals, and p-values. Factor predictors include reference levels in the table. Returns a formatted flextable and optionally provides an automatic textual interpretation of results.

Usage

```
linreg(data, outcome, predictors, report = TRUE)
```

Arguments

data	A data frame containing the outcome and predictor variables.
outcome	Name of the continuous outcome variable (character).
predictors	Character vector of predictor variable names.
report	Logical; if TRUE, prints an automatic textual interpretation of multivariable results (default: TRUE).

Value

If report = FALSE, returns a flextable summarizing univariable and multivariable beta coefficients, 95% CI, and p-values. If report = TRUE, returns a list with table (the flextable) and interpretation (textual summary of multivariable results).

Examples

```
# Apply linear regression on iris dataset
linreg(
  data = iris,
  outcome = "Sepal.Length",
  predictors = c("Sepal.Width", "Petal.Length", "Species"),
  report = TRUE
)
```

---

logreg *Binary Logistic Regression Table with Univariable and Multivariable Analysis*

Description

Fits univariable and multivariable logistic regression models for a binary outcome, summarizing odds ratios (ORs), 95% confidence intervals, and p-values. Factor predictors include reference levels in the table. Returns a formatted flextable and optionally provides an automatic textual interpretation of results.

Arguments

data	A data frame containing the outcome and predictor variables.
outcome	Name of the binary outcome variable (character).
predictors	Character vector of predictor variable names.
report	Logical; if TRUE, prints an automatic textual interpretation of multivariable results (default: FALSE).

Value

A `flextable` summarizing univariable and multivariable logistic regression results, including ORs, 95% CI, and p-values.

Examples

```
logreg(data=medical_data(), outcome="case" ,
   predictors= c("age" ,  "parity" ,    "induced" ), report = TRUE)
```

`medical_data` *Load Infertility Dataset*

Description

Load Infertility Dataset

Usage

```
medical_data()
```

Value

A data.frame containing infertility cases with labeled predictors suitable for logistic regression

---

`recode_data` *Recode values in a data frame using a lookup table*

Description

This function replaces values in a data frame according to a named lookup vector. All columns are converted to character, and any value matching a name in `lookup` will be replaced by its corresponding value.

Usage

```
recode_data(data, lookup)
```

Arguments

`data`	A data frame whose values you want to recode.
`lookup`	A named character vector where names are original values and elements are the new values.

Value

A data frame with the same structure as `data`, with values recoded according to `lookup`.

Examples

```
df <- data.frame(
  gender = c("male", "F", "male", "female"),
  status = c("single", "Married", "oo", "M"),
  stringsAsFactors = FALSE
)

lookup <- c(
  "male" = "Male",
  "M" = "Married",
  "oo" = "Widow",
  "female" = "Female",
  "F" = "Female"
)

df_recode <- recode_data(df, lookup)
print(df_recode)
```

---

sum_cor *Summarize Correlations Between a Reference Variable and Others*

Description

Computes correlations between a reference variable and one or more comparison variables. For Pearson correlations, 95% confidence intervals are also calculated. Can optionally stratify by a grouping variable. Returns a formatted `flextable` and optionally prints a narrative summary describing weak, moderate, and strong correlations.

Usage

```
sum_cor(
  data,
  ref_var,
  compare_vars,
  by = NULL,
  method = "pearson",
  digits = 3,
  report = TRUE
)
```

Arguments

`data`	A data frame containing the variables of interest.
`ref_var`	The reference variable (numeric) for correlation calculations.
`compare_vars`	A character vector of variables to correlate with the reference variable.

by	Optional grouping variable. If provided, correlations are calculated within each group.
method	Correlation method. Options: "pearson", "spearman", or "kendall" (default: "pearson").
digits	Number of decimal places to round correlation coefficients and CIs (default: 3).
report	Logical. If TRUE, prints a narrative summary of correlations (default: TRUE).

Value

A `flextable` object showing correlations, 95% CI (Pearson only), p-values, and interpretation.

Examples

```
# Example 1: Correlations across entire dataset
sum_cor(
  data = iris,
  ref_var = "Sepal.Length",
  compare_vars = c("Petal.Length", "Petal.Width", "Sepal.Width"),
  method = "pearson",
  digits = 2,
  report = TRUE
)

# Example 2: Correlations by Species
sum_cor(
  data = iris,
  ref_var = "Sepal.Length",
  by = "Species",
  compare_vars = c("Petal.Length", "Petal.Width", "Sepal.Width"),
  method = "pearson",
  digits = 2,
  report = TRUE
)
```

---

sum_crosstab *Crosstabulation with Counts and Percentages*

Description

Creates a cross-tabulation (contingency table) between two categorical variables and returns a publication-ready table as a `flextable`. The table displays cell counts with optional percentages (cell-wise, row-wise, or column-wise), and includes row totals (right) and column totals (bottom).

Usage

```
sum_crosstab(
  data,
  row_var,
```

```
  col_var,
  percent = c("none", "cell", "row", "col"),
  digits = 2
)
```

Arguments

data	A data frame containing the variables to be cross-tabulated.
row_var	A character string specifying the row variable (categorical).
col_var	A character string specifying the column variable (categorical).
percent	Type of percentage to display alongside counts. One of "none" (counts only), "cell" (percentage of total), "row" (row percentage), or "col" (column percentage).
digits	Integer specifying the number of decimal places for percentages.

Value

A flextable object containing the formatted cross-tabulation with counts, optional percentages, and marginal totals.

Examples

```
data(CO2)

sum_crosstab(
  data = CO2,
  row_var = "Treatment",
  col_var = "Type",
  percent = "cell",
  digits = 2
)
```

---

sum_norm *Normality Test Summary Table for Numeric Variables*

Description

This function performs the Shapiro-Wilk normality test on all numeric variables in a dataset and returns the results in a publication-ready flextable. Extremely small p-values are displayed as "p < 0.001". The function automatically detects numeric variables and ignores non-numeric columns.

Arguments

data	A data frame containing numeric and non-numeric variables. Only numeric variables are assessed for normality.
sample_size	Integer. Maximum number of observations to use for the Shapiro-Wilk test per variable (default = 5000).

Value

A `flextable` summarizing each numeric variable with Shapiro-Wilk W statistic, formatted p-value, and distribution classification ("Normal" or "Skewed").

Examples

```
sum_norm(iris)
```

`sum_posthoc` *Summarize Variables with Post-hoc Tests: Mutiple comparisons*

Description

Produces a summary table of numeric or categorical variables grouped by a factor, optionally performing global tests (ANOVA or Kruskal-Wallis) and post-hoc comparisons (Tukey or Dunn test). Numeric variables can be summarized using mean (SD) or median (IQR). Returns a `flextable` suitable for reporting.

Usage

```
sum_posthoc(
  data,
  by,
  variables = NULL,
  digits = 2,
  format_lt = "<0.001",
  na.rm = TRUE,
  statistic = "auto",
  test_type = "auto"
)
```

Arguments

`data`	A data frame containing the variables to summarize.
`by`	The grouping variable for comparisons (factor or character).
`variables`	A character vector of variable names to summarize. If NULL, all variables except by are used.
`digits`	Number of decimal places for numeric summaries (default: 2).
`format_lt`	Character string to display very small p-values, e.g., "<0.001" (default: "<0.001").
`na.rm`	Logical. If TRUE, removes missing values before computations (default: TRUE).
`statistic`	Summary statistic for numeric variables. Options: "auto" (default), "mean_sd", "med_iqr".
`test_type`	Global test type. Options: "auto" (default), "anova", "kruskal".

Value

A `flextable` containing the summary table, global p-values, and post-hoc results.

Examples

```
sum_posthoc(
  data = iris,
  by = "Species",
  variables = c("Sepal.Length","Sepal.Width","Petal.Length","Petal.Width"),
  digits = 2,
  statistic = "auto",
  test_type = "auto"
)
```

---

sum_stat *Summarize Continuous and Categorical Variables with Optional Grouping*

Description

`sum_stat` provides a summary of both continuous and categorical variables in a dataset. Continuous variables can be summarized using mean (SD) or median (IQR), optionally with 95% confidence intervals. Categorical variables are summarized as counts and percentages, optionally with confidence intervals. Summaries can also be generated by a grouping variable, and a narrative interpretation is optionally printed.

Usage

```
sum_stat(
  data,
  by = NULL,
  statistic = "mean_sd",
  percent = "col",
  ci = FALSE,
  conf = 0.95,
  report = FALSE
)
```

Arguments

`data`	A data.frame or tibble containing the variables to summarize.
`by`	Optional. A single variable name (as string) to group the summary by.
`statistic`	Character string indicating how to summarize continuous variables. Options are `"mean_sd"` (default) or `"med_iqr"`.
`percent`	Character string specifying how percentages should be calculated for categorical variables: `"col"` (column-wise), `"row"` (row-wise), or `"none"` (no percentage). Default is `"col"`.

ci	Logical. If TRUE, 95% confidence intervals are included in the summary for continuous and categorical variables. Default is FALSE.
conf	Numeric. Confidence level for CI calculation (between 0 and 1). Default is 0.95.
report	Logical. If TRUE, prints a narrative summary of the variables. Default is TRUE.

Value

A flextable object displaying the summarized variables, optionally including confidence intervals and group comparisons.

Examples

```
# Basic summary of iris dataset
sum_stat(iris, ci = FALSE, report = TRUE)

# Summary of CO2 dataset by 'Treatment' with CI
sum_stat(CO2, by = "Treatment", ci = TRUE, report = TRUE, percent = "row")
```

---

sum_stat_p *Summarize Continuous and Categorical Variables with Grouping and P-Values*

Description

sum_stat_p generates a descriptive summary table for both continuous and categorical variables, stratified by a grouping variable. It automatically computes appropriate statistical tests (Chi-square, Fisher's exact, t-test, Wilcoxon, ANOVA, or Kruskal–Wallis) based on variable type, number of groups, and data distribution. Continuous variables can be summarized as mean (SD) or median (IQR), and categorical variables as counts and percentages.

Usage

```
sum_stat_p(data, by, statistic = "mean_sd", test_type = "auto", paired = FALSE)
```

Arguments

data	A data.frame or tibble containing the variables to summarize.
by	A string specifying the grouping variable for stratified summaries.
statistic	Character string indicating how to summarize continuous variables: "mean_sd" (default) or "med_iqr".
test_type	Character string specifying the statistical test for group comparisons: "auto" (default, chooses test based on data), "t.test", "wilcox", "anova", "kruskal", "chisq", or "fisher".
paired	Logical. If TRUE and only two groups exist, performs a paired t-test for continuous variables. Default is FALSE.

Details

The output is formatted as a `flextable` with footnotes indicating which summary statistics were used and which statistical tests were applied.

Value

A `flextable` object displaying a publication-ready summary table including:

- Counts and percentages for categorical variables
- Mean (SD) or median (IQR) for continuous variables
- P-values for group comparisons
- Footnotes describing which summary statistics and tests were used

Examples

```
# Summary of iris dataset by species
sum_stat_p(iris, by = "Species", statistic = "mean_sd", test_type = "auto")

# Summary of CO2 dataset by Type with paired t-test
sum_stat_p(CO2, by = "Type", statistic = "mean_sd", test_type = "t.test", paired = TRUE)

# Summary using median and IQR
sum_stat_p(iris, by = "Species", statistic = "med_iqr", test_type = "kruskal")
```

---

sum_stat_p_strata *Summarize Variables with Optional Stratification and Statistical Tests*

Description

Produces summary tables for numeric and categorical variables in a dataset, optionally stratified by a grouping variable. Numeric variables are summarized with mean (SD) or median (IQR), and categorical variables with counts and percentages. Appropriate statistical tests (t-test, Wilcoxon, ANOVA, Kruskal-Wallis, Chi-square, or Fisher's Exact) are performed depending on the variable type, number of groups, and user-specified options.

Usage

```
sum_stat_p_strata(
  data,
  by,
  strata = NULL,
  statistic = "mean_sd",
  test_type = "auto",
  paired = FALSE
)
```

Arguments

data	A data frame containing the variables to summarize.
by	The main grouping variable for comparison.
strata	Optional stratification variable. Summaries are produced within each stratum.
statistic	Statistic to report for numeric variables: "mean_sd" (default) or "med_iqr".
test_type	Statistical test to use. Options: "auto" (default, selects appropriate test), "t.test", "wilcox", "anova", "kruskal", "fisher", or "chisq".
paired	Logical. If TRUE, paired t-tests are used when applicable (default: FALSE).

Value

A flextable object containing the summary table with optional p-values, counts, percentages, and numeric summaries. Footnotes describe the statistics used and tests performed.

Examples

```
# Example : Summary of CO2 dataset by Type, stratified by Treatment
sum_stat_p_strata(data = CO2, by = "Type", strata = "Treatment")
```

Index

category, 2

diag_accuracy, 3

impute_missing, 4

linreg, 4
logreg, 5

medical_data, 6

recode_data, 6

sum_cor, 7
sum_crosstab, 8
sum_norm, 9
sum_posthoc, 10
sum_stat, 11
sum_stat_p, 12
sum_stat_p_strata, 13