

Package ‘ThesiStats’

October 5, 2026

Version 1.1.0

Title Statistical Tools for Quantitative Theses

Description Provides helpers for the analyses that quantitative theses in the social and behavioral sciences repeat: renaming and scoring items, recoding Likert responses, cleaning sociodemographic variables written in Spanish (age, academic term, degree and university), descriptive statistics, univariate and multivariate normality checks, omega reliability from ordinal confirmatory factor models, correlation matrices in table format, and two-group or several-group comparisons with effect sizes.

License GPL (>= 3)

URL <https://github.com/jventural/ThesiStats>

BugReports <https://github.com/jventural/ThesiStats/issues>

Encoding UTF-8

Depends R (>= 4.1.0)

Imports broom, dplyr, ggplot2 (>= 3.4.0), grid, gridExtra, gtable, lavaan, psych, readr, rlang, semTools, stats, stringdist, stringi, stringr, tibble, tidyr, utils

Suggests readxl, WRS2

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author José Ventura-León [aut, cre] (ORCID:
<<https://orcid.org/0000-0003-2996-4244>>)

Maintainer José Ventura-León <jventuraleon@gmail.com>

Repository CRAN

Date/Publication 2026-10-05 15:30:07 UTC

Contents

calcular_correlaciones	3
calcular_porcentajes	4
calculate_descriptives	4
calcula_omega_all	5
Calcule_Comparative	6
carreras_peruanas	7
clean_edad	7
convert_age_to_years	8
convert_age_to_years_months	9
convert_to_expresions	10
Correct_category	10
detect_expression_Likert	11
diag_aba_na	12
edad_stat	12
epsilon_cuadrado_kruskal	13
extract_items	14
Fiabilidad	14
generate_and_apply	15
generate_code	16
grafico_boxplots	16
mardia_test	17
Multivariate_plot	18
normality_test_SW	18
normalize_carreras	19
normalize_ciclo	20
normalize_tiempo_relacion	21
normalize_universidades	21
process_likert_blocks	22
replace_alternative_response	23
rename_columns	23
rename_items2	24
rename_items3	25
rename_items_only	26
universidades_peruanas	27
u_mann_whitney_superioridad	27
validation_categoria	28

Index

29

calcular_correlaciones*Correlation Matrix of a Range of Columns*

Description

Computes the Spearman or Pearson correlation matrix of the columns that go from `columna_inicial` to `columna_final`, optionally with Winsorized Pearson correlations and significance marks, and returns it in the lower triangular format used in thesis tables.

Usage

```
calcular_correlaciones(  
  data,  
  columna_inicial,  
  columna_final,  
  method = c("spearman", "pearson"),  
  winsorize = FALSE,  
  show_pval = FALSE  
)
```

Arguments

<code>data</code>	A data frame.
<code>columna_inicial, columna_final</code>	Names of the first and last columns of the range.
<code>method</code>	"spearman" (default) or "pearson".
<code>winsorize</code>	Logical. With <code>method = "pearson"</code> , compute 20% Winsorized correlations with <code>WRS2::winall()</code> (requires the <code>WRS2</code> package).
<code>show_pval</code>	Logical. Append significance marks to each correlation: ns $p > .05$, * $p \leq .05$, ** $p \leq .01$, *** $p \leq .001$, **** $p \leq .0001$.

Value

A list with `correlation` (data frame with the correlations in the lower triangle, "-" on the diagonal and NA above it) and `p_values` (the p values in the same format).

Examples

```
set.seed(1)  
x <- rnorm(80)  
df <- data.frame(ansiedad = x, depresion = 0.5 * x + rnorm(80),  
                 estres = 0.3 * x + rnorm(80))  
calcular_correlaciones(df, "ansiedad", "estres", method = "pearson",  
                       show_pval = TRUE)
```

`calcular_porcentajes` *Frequencies and Percentages of Categorical Variables*

Description

For each variable in `columnas`, counts the observations of each category and its percentage of the total.

Usage

```
calcular_porcentajes(data, columnas)
```

Arguments

`data` A data frame.
`columnas` Character vector with the names of the categorical variables.

Value

A named list with one tibble per variable, holding the categories, their counts (n) and percentages (Porcentaje).

Examples

```
df <- data.frame(sexo = c("F", "M", "F", "F", "M"),
                 ciclo = c(1, 2, 2, 3, 3))
calcular_porcentajes(df, c("sexo", "ciclo"))
```

`calculate_descriptives`
Descriptive Statistics of a Range of Columns

Description

Computes, with `psych::describe()`, the mean, standard deviation, minimum, maximum, skewness and kurtosis of the columns from `start_col` to `end_col`, plus the mean as a percentage of the maximum.

Usage

```
calculate_descriptives(data, start_col, end_col)
```

Arguments

`data` A data frame.
`start_col, end_col` Names of the first and last columns of the range.

Value

A data frame with the columns Variables, Media, DE, Min., Max., g1 (skewness), g2 (kurtosis) and %, rounded to two decimals.

Examples

```
set.seed(1)
df <- data.frame(ansiedad = rnorm(50, 20, 4), depresion = rnorm(50, 15, 3))
calculate_descriptives(df, "ansiedad", "depresion")
```

calcula_omega_all	<i>Omega Reliability of Several Factors</i>
-------------------	---

Description

Applies [Fiabilidad\(\)](#) to every factor of a list of item names and returns the omega coefficient of each one.

Usage

```
calcula_omega_all(extracted, data)
```

Arguments

extracted	A named list: each element holds the item names of one factor, as returned by extract_items() .
data	A data frame with the item responses.

Value

A data frame with the columns Variables (factor name) and Omega.

Examples

```
set.seed(123)
n <- 300
eta <- matrix(rnorm(n * 2), n, 2) %%% chol(0.7 * diag(2) + 0.3)
items <- as.data.frame(sapply(1:8, function(j) as.numeric(cut(
  0.7 * eta[, ceiling(j / 4)] + rnorm(n, 0, 0.7),
  c(-Inf, -1.5, -0.5, 0.5, 1.5, Inf)))))
names(items) <- c(paste0("ANS", 1:4), paste0("DEP", 1:4))

factores <- extract_items(c("Ansiedad: ANS1, ANS2, ANS3, ANS4",
  "Depresion: DEP1, DEP2, DEP3, DEP4"))
calcula_omega_all(factores, items)
```

Calcule_Comparative *Compare Two Groups with Welch's t Test and Cohen's d*

Description

For each variable in cols, compares the two groups defined by group_var with Welch's t test and reports the means and standard deviations of each group, the t statistic, its degrees of freedom, the p value and Cohen's d with a verbal interpretation.

Usage

```
Calcule_Comparative(data, cols, group_var, Robust = FALSE)
```

Arguments

data	A data frame.
cols	Character vector with the names of the variables to compare.
group_var	Name of the grouping variable. It must have exactly two groups.
Robust	Logical. If FALSE (default), Cohen's d uses the pooled standard deviation. If TRUE, the robust effect size of Algina, Keselman and Penfield (WRS2::akp.effect()) is reported instead; this requires the WRS2 package.

Details

The two groups are taken in the order of levels(factor(data[[group_var]])), the same order used by `stats::t.test()`, so the means, standard deviations, the difference and the sign of d always refer to the same group. The interpretation uses `ldl > .80 "Grande", > .50 "Mediano", > .30 "Pequeno"` and "Trivial" otherwise.

Value

A data frame with one row per variable and the columns Variables_interes, <group 1>(SD1), <group 2>(SD2), t, gl, p, d_cohen and Interpretacion.

Examples

```
set.seed(1)
df <- data.frame(
  grupo = rep(c("Mujer", "Varon"), each = 50),
  ansiedad = c(rnorm(50, 20, 4), rnorm(50, 18, 4)),
  depresion = c(rnorm(50, 15, 3), rnorm(50, 15, 3))
)
Calcule_Comparative(df, cols = c("ansiedad", "depresion"), group_var = "grupo")
```

carreras_peruanas	<i>Catalog of Peruvian University Degrees</i>
-------------------	---

Description

Reference list of undergraduate degrees and their faculties, used by `normalize_carreras()` to standardize free-text degree names.

Usage

```
carreras_peruanas
```

Format

A tibble with 93 rows and 2 columns:

Carrera Standard name of the degree.

Facultad Faculty the degree belongs to.

Source

Compiled by the package author from the degree offer of Peruvian universities.

clean_edad	<i>Clean Age Strings into Fractional Years</i>
------------	--

Description

Standardizes a character column with ages written in Spanish (for example "19 años", "1 año 6 meses", "03 meses" or "2.5"): extracts years and months, converts months to fractions of a year and returns a numeric column. Rows without any digit are removed and reported with a message.

Usage

```
clean_edad(df, col_name = "Edad", round_decimals = 2)
```

Arguments

df A data frame.

col_name Name of the age column. Defaults to "Edad".

round_decimals Number of decimals of the result. Defaults to 2.

Value

df without the rows that had no digits, and with `col_name` replaced by the age in years.

Examples

```
df <- data.frame(Edad = c("19 años", "1 año 6 meses", "03 meses", "2.5",  
                          "sin dato"))  
clean_edad(df)
```

convert_age_to_years	<i>Convert Age Strings to Numeric Years</i>
----------------------	---

Description

Parses a character column with ages written in Spanish (for example "2 años", "un año" or "3,5") into numeric years. Unlike [convert_age_to_years_months\(\)](#), months are not added.

Usage

```
convert_age_to_years(  
  df,  
  col_name = "Edad",  
  round_decimals = 2,  
  drop_missing = FALSE  
)
```

Arguments

df	A data frame.
col_name	Name of the age column. Defaults to "Edad".
round_decimals	Number of decimals of the result. Defaults to 2.
drop_missing	Logical. If TRUE, rows whose age could not be converted are removed; if FALSE (default) they are kept as NA and listed in a message.

Value

A list with `cleaned_df` (the data frame with `col_name` converted) and `verify_df` (a tibble pairing each original string with its value).

Examples

```
df <- data.frame(Edad = c("20 años", "un año", "3,5", "sin dato"))  
res <- convert_age_to_years(df)  
res$verify_df
```

`convert_age_to_years_months`*Convert Age Strings with Years and Months to Numeric Years*

Description

Parses a character column with ages written in Spanish (for example "2 años", "6 meses", "1 año 3 meses" or "3,5") into numeric years, expressing months as fractions of a year.

Usage

```
convert_age_to_years_months(  
  df,  
  col_name = "Edad",  
  round_decimals = 2,  
  drop_missing = FALSE  
)
```

Arguments

<code>df</code>	A data frame.
<code>col_name</code>	Name of the age column. Defaults to "Edad".
<code>round_decimals</code>	Number of decimals of the result. Defaults to 2.
<code>drop_missing</code>	Logical. If TRUE, rows whose converted age is zero (entries without any usable number) are removed. Defaults to FALSE.

Value

A list with `cleaned_df` (the data frame with `col_name` converted) and `verify_df` (a tibble pairing each original string with its value).

Examples

```
df <- data.frame(Edad = c("2 años", "6 meses", "1 año 3 meses", "3,5"))  
res <- convert_age_to_years_months(df)  
res$verify_df
```

`convert_to_expressions` *Format Likert Response Options as a Single String*

Description

Turns a data frame of Likert response options and their scores into one string of the form "0. Option; 1. Option; ...", ordered by score. This is the format read by `replace_alternative_response()`.

Usage

```
convert_to_expressions(df)
```

Arguments

`df` A data frame with the columns `Alternativas` (response options) and `score` (their numeric values), such as the output of `detect_expression_Likert()`.

Value

A character string.

Examples

```
likert <- data.frame(
  Alternativas = c("Nunca", "A veces", "Siempre"),
  score = c(0, 1, 2)
)
convert_to_expressions(likert)
```

`Correct_category` *Recode Specified Values in Selected Columns*

Description

Applies recoding rules, written as two-sided formulas "old" ~ "new", to one or more columns of a data frame, replacing exact matches of each old value with the new one.

Usage

```
Correct_category(df, cols, ...)
```

Arguments

`df` A data frame or tibble.
`cols` Columns to recode, in dplyr selection syntax (for example `Q1:Q5` or `starts_with("Q")`).
`...` One or more two-sided formulas of the form "old_value" ~ "new_value".

Value

df with every occurrence of each old value replaced by the new value in the selected columns. Other columns are unchanged.

Examples

```
df <- data.frame(
  Q1 = c("Yes", "No", "yes", "No"),
  Q2 = c("Maybe", "maybe", "No", "Yes")
)
Correct_category(df, Q1:Q2, "yes" ~ "Yes", "maybe" ~ "Maybe")
```

detect_expression_Likert

Detect Likert Response Options and Assign Scores

Description

Looks in a data frame for the response options listed in likert_levels (ignoring case, accents and extra spaces), keeps those that appear, orders them as in likert_levels and assigns each one a score starting at 0 or 1.

Usage

```
detect_expression_Likert(
  df,
  start_zero = TRUE,
  likert_levels = c("Completamente en desacuerdo", "En desacuerdo", "Me es indiferente",
    "De acuerdo", "Completamente de acuerdo")
)
```

Arguments

df	A data frame whose columns hold Likert responses as text.
start_zero	Logical. Scores start at 0 (TRUE, default) or at 1.
likert_levels	Character vector with the response options in ascending order.

Value

A tibble with the columns Alternativas (the options found, as an ordered factor) and score. A warning lists the options of likert_levels that do not appear in df.

Examples

```
respuestas <- data.frame(
  P1 = c("De acuerdo", "En desacuerdo", "Me es indiferente"),
  P2 = c("Completamente de acuerdo", "De acuerdo", "En desacuerdo")
)
suppressWarnings(detect_expression_Likert(respuestas))
```

diag_aba_na

*Lower Triangular Format for a Correlation Matrix***Description**

Numbers the rows ("1. Name") and columns (1, 2, ...) of a square matrix, puts "-" on the diagonal and NA above it, the usual layout of a correlation table in a thesis.

Usage

```
diag_aba_na(matriz)
```

Arguments

matriz A square matrix with row names.

Value

A character matrix with the lower triangle of matriz.

Examples

```
m <- round(cor(mtcars[, 1:4]), 2)
diag_aba_na(m)
```

edad_stat

*Summary Statistics of an Age Column***Description**

Computes the mean, standard deviation, minimum and maximum of an age column.

Usage

```
edad_stat(obj, columna)
```

Arguments

obj A data frame, or the list returned by [convert_age_to_years\(\)](#) or [convert_age_to_years_months\(\)](#) (its cleaned_df is used).

columna The age column, as a bare name or a string.

Value

A one-row data frame with Media, DesviacionEstandar, Minimo and Maximo.

Examples

```
df <- data.frame(Edad = c(18, 20, 22, 25, 30))
edad_stat(df, Edad)
edad_stat(df, "Edad")
```

epsilon_cuadrado_kruskal

Kruskal-Wallis Test with Epsilon Squared

Description

Runs a Kruskal-Wallis test and adds its effect size, epsilon squared $\epsilon^2 = H/(n - 1)$, with a verbal interpretation.

Usage

```
epsilon_cuadrado_kruskal(data, formula)
```

Arguments

data	A data frame.
formula	A formula outcome ~ group.

Details

n counts the cases with non-missing outcome and group, the same cases used by the test. The interpretation is "Grande" for $\epsilon^2 \geq .50$, "Mediano" $\geq .30$, "Pequeño" $\geq .10$ and "No significativo" otherwise.

Value

A tibble with the test results (statistic, p.value, parameter, method), Epsilon and Interpretación.

Examples

```
df <- data.frame(
  grupo = rep(c("A", "B", "C"), each = 5),
  puntaje = c(3, 4, 5, 4, 3, 7, 8, 6, 7, 8, 10, 12, 11, 9, 10)
)
epsilon_cuadrado_kruskal(df, puntaje ~ grupo)
```

extract_items	<i>Extract the Items of Each Factor from Text Lines</i>
---------------	---

Description

Reads lines of the form "Factor: Item1, Item2, Item3" and returns a named list with the items of each factor.

Usage

```
extract_items(text, prefix = "")
```

Arguments

text	Character vector, one line per factor.
prefix	Optional text inserted in the item names between an uppercase letter and the lowercase letter that follows it. Defaults to "" (names unchanged).

Value

A named list: one character vector of item names per factor.

Examples

```
extract_items(c("Ansiedad: ANS1, ANS2, ANS3", "Depresion: DEP1, DEP2"))
```

Fiabilidad	<i>Omega Reliability of One Factor</i>
------------	--

Description

Fits a one-factor confirmatory model to the items in vars, treating them as ordinal (WLSMV estimator) and returns the composite reliability (omega) computed by [semTools::compRelSEM\(\)](#).

Usage

```
Fiabilidad(vars, data)
```

Arguments

vars	Character vector with the names of the items of the factor, for example one element of the list returned by extract_items() .
data	A data frame with the item responses.

Value

A named numeric value: the omega coefficient of the factor.

See Also

`calcula_omega_all()` for several factors at once.

Examples

```
set.seed(123)
n <- 300
eta <- rnorm(n)
items <- as.data.frame(sapply(1:4, function(j) as.numeric(cut(
  0.7 * eta + rnorm(n, 0, 0.7), c(-Inf, -1.5, -0.5, 0.5, 1.5, Inf)))))
names(items) <- paste0("ANS", 1:4)

Fiabilidad(vars = names(items), data = items)
```

generate_and_apply	<i>Add the Sum Score of Each Factor to a Data Frame</i>
--------------------	---

Description

Reads lines of the form "Factor: Item1, Item2, ..." and adds to df one column per factor with the row sum of its items.

Usage

```
generate_and_apply(df, text_lines, new_name = NULL)
```

Arguments

df	A data frame with the item columns.
text_lines	Character vector, one line per factor.
new_name	Deprecated and ignored. Earlier versions also saved the result in the global environment under this name; assign the returned data frame instead, for example <code>df2 <- generate_and_apply(df, lines)</code> .

Value

df with one new column per factor (the row sum of its items, ignoring missing values). The item columns are converted to numeric.

Examples

```
df <- data.frame(A1 = c(1, 2, 3), A2 = c(2, 2, 2), B1 = c(0, 1, 1))
generate_and_apply(df, c("Ansiedad: A1, A2", "Estres: B1"))
```

generate_code

Generate dplyr Code that Adds Factor Sum Scores

Description

Reads text with one line per factor ("Factor: Item1, Item2, ...") and returns, as a string, the dplyr code that adds the row sum of each factor to a data frame. The code can be printed with `cat()` and pasted into a script.

Usage

```
generate_code(text, name = "df_new_renombrado")
```

Arguments

text	A single string with one line per factor, separated by "\n".
name	Name of the data frame used in the generated code.

Value

A character string with R code.

Examples

```
codigo <- generate_code("Ansiedad: A1, A2, A3\nEstres: B1, B2", name = "datos")
cat(codigo)
```

grafico_boxplots

Boxplots of Several Variables

Description

Draws one boxplot per variable, each in its own panel with its own scale, to inspect the distribution and outliers of the scores.

Usage

```
grafico_boxplots(data, cols)
```

Arguments

data	A data frame.
cols	Character vector with the names of the variables.

Value

A ggplot object.

Examples

```
set.seed(1)
df <- data.frame(ansiedad = rnorm(60, 20, 4), depresion = rexp(60, 0.2))
grafico_boxplots(df, c("ansiedad", "depresion"))
```

mardia_test

Mardia's Test of Multivariate Normality

Description

Computes Mardia's multivariate skewness and kurtosis tests with `psych::mardia()` and reports whether each one is compatible with multivariate normality ($p \geq .05$).

Usage

```
mardia_test(data)
```

Arguments

data A data frame or matrix of numeric variables.

Value

A data frame with the columns Test, Statistic, p.value (formatted, " $p < .001$ " when $p \leq .001$) and Result ("YES" when normality is not rejected).

Examples

```
set.seed(1)
df <- as.data.frame(matrix(rnorm(200 * 3), ncol = 3))
mardia_test(df)
```

Multivariate_plot	<i>Q-Q Plot of Mahalanobis Distances with Mardia's Coefficients</i>
-------------------	---

Description

Plots the squared Mahalanobis distances of the observations against the quantiles of a chi-square distribution (a check of multivariate normality) and overlays a table with Mardia's skewness and kurtosis tests from `mardia_test()`.

Usage

```
Multivariate_plot(data, xmin = 30, xmax = 40, ymin = 2, ymax = 7)
```

Arguments

data	A data frame or matrix of numeric variables.
xmin, xmax, ymin, ymax	Position of the Mardia table inside the plot, in the units of the axes.

Value

A ggplot object.

Examples

```
set.seed(1)
df <- as.data.frame(matrix(rnorm(200 * 4), ncol = 4))
Multivariate_plot(df, xmin = 8, xmax = 14, ymin = 1, ymax = 5)
```

normality_test_SW	<i>Shapiro-Wilk Normality Test for Several Variables</i>
-------------------	--

Description

Applies the Shapiro-Wilk test to each variable and classifies it as "Normal" ($p \geq .05$) or "No-normal".

Usage

```
normality_test_SW(data, variables)
```

Arguments

data	A data frame.
variables	Variables to test, in dplyr selection syntax (for example <code>c(ansiedad, depression)</code> or <code>ansiedad:estres</code>).

Value

A tibble with the columns Variables, Shapiro-Wilk (the W statistic), p.value (formatted, "p < .001" when $p < .001$) and Normality.

Examples

```
set.seed(1)
df <- data.frame(ansiedad = rnorm(60), depresion = rexp(60))
normality_test_SW(df, c(ansiedad, depresion))
```

normalize_carreras	<i>Normalize Peruvian University Degree Names</i>
--------------------	---

Description

Matches free-text degree names (for example "psicologia", "Ing. de Sistemas") against the catalog [carreras_peruanas](#) with fuzzy string matching, and adds a column with the standardized name (and optionally the faculty).

Usage

```
normalize_carreras(
  df,
  col_name = "Carrera",
  max_dist = 0.15,
  facultad = FALSE,
  manual_path = NULL,
  force_match = FALSE,
  fallback_dist = 0.4,
  remove_unmatched = FALSE
)
```

Arguments

df	A data frame.
col_name	Name of the column with the degree names. Defaults to "Carrera".
max_dist	Maximum string distance accepted as a match (0 to 1).
facultad	Logical. Also add a column with the faculty.
manual_path	Optional path to an Excel file with the columns raw and normalized, a manual dictionary applied before the fuzzy matching (requires the readxl package).
force_match	Logical. Retry unmatched values with the more permissive fallback_dist (not recommended).
fallback_dist	Distance used when force_match = TRUE.
remove_unmatched	Logical. Remove the rows whose degree could not be normalized.

Value

df with the column <col_name>_norm (and <col_name>_facultad when facultad = TRUE) placed after col_name. Messages report the values that could not be normalized.

Examples

```
df <- data.frame(Carrera = c("psicologia", "Derecho", "zzz"))
normalize_carreras(df)
```

normalize_ciclo

Normalize the Academic Term (Ciclo) to an Integer

Description

Converts free-text academic terms in Spanish ("3er ciclo", "V", "quinto", "10") into integers. Rows that cannot be converted are removed and listed in a message.

Usage

```
normalize_ciclo(df, col_name = "Ciclo")
```

Arguments

df A data frame.

col_name Name of the column with the term. Defaults to "Ciclo".

Value

df without the rows that could not be converted, and with col_name as an integer.

Examples

```
df <- data.frame(Ciclo = c("3er ciclo", "V", "quinto", "10", "no se"))
normalize_ciclo(df)
```

`normalize_tiempo_relacion`*Normalize the Length of a Relationship to Months*

Description

Converts free-text durations in Spanish ("1 año y 6 meses", "2 semanas", "un año y medio", "8") into months.

Usage

```
normalize_tiempo_relacion(df, col_name = "Tiempo_Relacion", remover = TRUE)
```

Arguments

<code>df</code>	A data frame.
<code>col_name</code>	Name of the column with the duration. Defaults to "Tiempo_Relacion".
<code>remover</code>	Logical. Remove the rows that could not be converted (TRUE, default) or keep them with NA.

Value

A data frame with the new column `<col_name>_norm` (months).

Examples

```
df <- data.frame(Tiempo_Relacion = c("1 año y 6 meses", "2 semanas", "8"))
normalize_tiempo_relacion(df)
```

`normalize_universidades`*Normalize Peruvian University Names to Their Acronyms*

Description

Matches free-text university names or acronyms against the catalog [universidades_peruanas](#) with Jaro-Winkler distance and adds a column with the standard acronym.

Usage

```
normalize_universidades(df, col_name)
```

Arguments

<code>df</code>	A data frame.
<code>col_name</code>	Name of the column with the university names.

Value

df with the column <col_name>_norm (the acronym, or NA when there is no match within a distance of .15) placed after col_name.

Examples

```
df <- data.frame(U = c("Universidad Nacional Mayor de San Marcos", "unmsm"))
normalize_universidades(df, "U")
```

process_likert_blocks *Detect and Score Several Blocks of Likert Items*

Description

For each block of items, detects the response options with `detect_expression_likert()`, builds the score mapping with `convert_to_expressions()` and replaces the text responses by their scores with `replace_alternative_response()`.

Usage

```
process_likert_blocks(df, specs)
```

Arguments

df	A data frame.
specs	A list of blocks. Each block is a list with prefix (item prefix, for example "ANS"), n_items (number of items, named prefix1 to prefixN), levels (response options in ascending order) and start_zero (logical, scores start at 0 or 1).

Value

df with the items of every block scored.

Examples

```
df <- data.frame(ANS1 = c("Nunca", "Siempre", "A veces"),
                 ANS2 = c("A veces", "Nunca", "Siempre"))
process_likert_blocks(df, list(
  list(prefix = "ANS", n_items = 2, start_zero = TRUE,
       levels = c("Nunca", "A veces", "Siempre"))
))
```

 replace_alternative_response

Replace Likert Text Responses with Their Scores

Description

Converts text responses into numbers in one or more blocks of item columns. Each block is described by its first and last column and by a string that maps each response option to its score ("0. Nunca; 1. A veces; ...", the format returned by `convert_to_expresions()`).

Usage

```
replace_alternative_response(df, columnas_valores_entradas)
```

Arguments

`df` A data frame.

`columnas_valores_entradas`

A list of blocks. Each block is a list with two elements: a character vector `c(first_column, last_column)` (for example `c("ANS1", "ANS9")`) and the mapping string.

Value

`df` with the responses of the listed columns replaced by their scores; responses not found in the mapping become NA.

Examples

```
df <- data.frame(P1 = c("Nunca", "Siempre"), P2 = c("A veces", "Nunca"))
replace_alternative_response(
  df, list(list(c("P1", "P2"), "0. Nunca; 1. A veces; 2. Siempre"))
)
```

 rename_columns

Rename Columns by Position

Description

Renames the columns at the positions given in `columns` with `new_names`.

Usage

```
rename_columns(df, new_names, columns)
```

Arguments

df	A data frame.
new_names	Character vector with the new names.
columns	Integer vector with the positions of the columns to rename, in the same order as new_names.

Value

df with the columns renamed.

Examples

```
df <- data.frame(a = 1:2, b = 3:4, c = 5:6)
rename_columns(df, c("Edad", "Sexo"), 1:2)
```

rename_items2	<i>Rename a Block of Item Columns with Two Prefixes</i>
---------------	---

Description

Renames the consecutive columns from inicio to final: the first n_items1 as prefix1 1, 2, ... and the next n_items2 as prefix2 1, 2, ... If the counts are not given, the block is split in halves.

Usage

```
rename_items2(
  df,
  prefix1 = "COPE",
  prefix2 = "E",
  inicio = NULL,
  final = NULL,
  n_items1 = NULL,
  n_items2 = NULL
)
```

Arguments

df	A data frame.
prefix1	Prefix of the new names.
prefix2	Prefix of the second group of items.
inicio, final	Names of the first and last columns of the block. Defaults to the first and last columns of df.
n_items1	Number of items; if given, it must equal the number of columns in the block.
n_items2	Number of items of the second group.

Value

df with the block renamed.

Examples

```
df <- data.frame(p1 = 1, p2 = 2, p3 = 3, p4 = 4, p5 = 5)
rename_items2(df, prefix1 = "ANS", prefix2 = "DEP", n_items1 = 3)
```

rename_items3	<i>Rename a Block of Item Columns with Three Prefixes</i>
---------------	---

Description

Like `rename_items2()` with three groups of items. If the counts are not given, the block is split in thirds.

Usage

```
rename_items3(
  df,
  prefix1 = "COPE",
  prefix2 = "E",
  prefix3 = "F",
  inicio = NULL,
  final = NULL,
  n_items1 = NULL,
  n_items2 = NULL,
  n_items3 = NULL
)
```

Arguments

df	A data frame.
prefix1	Prefix of the new names.
prefix2	Prefix of the second group of items.
prefix3	Prefix of the third group of items.
inicio, final	Names of the first and last columns of the block. Defaults to the first and last columns of df.
n_items1	Number of items; if given, it must equal the number of columns in the block.
n_items2	Number of items of the second group.
n_items3	Number of items of the third group.

Value

df with the block renamed.

Examples

```
df <- data.frame(p1 = 1, p2 = 2, p3 = 3, p4 = 4, p5 = 5, p6 = 6)
rename_items3(df, prefix1 = "A", prefix2 = "B", prefix3 = "C")
```

rename_items_only*Rename a Block of Item Columns with One Prefix*

Description

Renames the consecutive columns from inicio to final as prefix1 followed by 1, 2, 3, ...

Usage

```
rename_items_only(  
  df,  
  prefix1 = "COPE",  
  inicio = NULL,  
  final = NULL,  
  n_items1 = NULL  
)
```

Arguments

df	A data frame.
prefix1	Prefix of the new names.
inicio, final	Names of the first and last columns of the block. Defaults to the first and last columns of df.
n_items1	Number of items; if given, it must equal the number of columns in the block.

Value

df with the block renamed.

Examples

```
df <- data.frame(id = 1:2, p1 = 1:2, p2 = 3:4, p3 = 5:6)
rename_items_only(df, prefix1 = "ANS", inicio = "p1", final = "p3")
```

universidades_peruanas*Catalog of Peruvian Universities*

Description

Reference list of Peruvian universities and their acronyms, used by `normalize_universidades()`.

Usage

```
universidades_peruanas
```

Format

A tibble with 99 rows and 2 columns: the full name of the university (Nombre) and its acronym (second column).

Source

Compiled by the package author from public lists of Peruvian universities.

u_mann_whitney_superioridad*Mann-Whitney U Test with the Probability of Superiority*

Description

Runs a Mann-Whitney U test (Wilcoxon rank-sum) between two groups and adds the probability of superiority, $PS = U/(n_1 n_2)$: the probability that a random case of the first group scores higher than a random case of the second.

Usage

```
u_mann_whitney_superioridad(data, formula, alternative = "two.sided")
```

Arguments

<code>data</code>	A data frame.
<code>formula</code>	A formula outcome ~ group; the group must have exactly two levels.
<code>alternative</code>	Alternative hypothesis passed to <code>stats::wilcox.test()</code> .

Details

PS = .50 means no effect. The size is judged on the distance from .50 in either direction, using $\max(PS, 1 - PS)$: at least .71 is "Grande", at least .64 "Mediano", at least .56 "Pequeño" and below that "No efecto".

Value

A tibble with the test results (statistic is U for the first group, p.value, method, alternative), PSest and Interpretación.

Examples

```
set.seed(1)
df <- data.frame(grupo = rep(c("A", "B"), each = 30),
                 puntaje = c(rnorm(30, 10), rnorm(30, 11)))
u_mann_whitney_superioridad(df, puntaje ~ grupo)
```

validation_categoria	<i>Unique Values of Several Columns</i>
----------------------	---

Description

Lists, sorted, the distinct values that appear in the selected columns, useful to check the categories before recoding them with [Correct_category\(\)](#).

Usage

```
validation_categoria(df, cols)
```

Arguments

df	A data frame.
cols	Columns in dplyr selection syntax.

Value

A sorted vector with the distinct values.

Examples

```
df <- data.frame(Q1 = c("Si", "No", "si"), Q2 = c("No", "Tal vez", "Si"))
validation_categoria(df, Q1:Q2)
```

Index

- * **datasets**
 - carreras_peruanas, [7](#)
 - universidades_peruanas, [27](#)
- calcula_omega_all, [5](#)
- calcula_omega_all(), [15](#)
- calcular_correlaciones, [3](#)
- calcular_porcentajes, [4](#)
- calculate_descriptives, [4](#)
- Calcule_Comparative, [6](#)
- carreras_peruanas, [7](#), [19](#)
- cat(), [16](#)
- clean_edad, [7](#)
- convert_age_to_years, [8](#)
- convert_age_to_years(), [12](#)
- convert_age_to_years_months, [9](#)
- convert_age_to_years_months(), [8](#), [12](#)
- convert_to_expresions, [10](#)
- convert_to_expresions(), [22](#), [23](#)
- Correct_category, [10](#)
- Correct_category(), [28](#)
- detect_expression_Likert, [11](#)
- detect_expression_Likert(), [10](#), [22](#)
- diag_abana, [12](#)
- edad_stat, [12](#)
- epsilon_cuadrado_kruskal, [13](#)
- extract_items, [14](#)
- extract_items(), [5](#), [14](#)
- Fiabilidad, [14](#)
- Fiabilidad(), [5](#)
- generate_and_apply, [15](#)
- generate_code, [16](#)
- grafico_boxplots, [16](#)
- mardia_test, [17](#)
- mardia_test(), [18](#)
- Multivariate_plot, [18](#)
- normality_test_SW, [18](#)
- normalize_carreras, [19](#)
- normalize_carreras(), [7](#)
- normalize_ciclo, [20](#)
- normalize_tiempo_relacion, [21](#)
- normalize_universidades, [21](#)
- normalize_universidades(), [27](#)
- process_likert_blocks, [22](#)
- psych::describe(), [4](#)
- psych::mardia(), [17](#)
- replace_alternative_response, [23](#)
- replace_alternative_response(), [10](#), [22](#)
- rename_columns, [23](#)
- rename_items2, [24](#)
- rename_items2(), [25](#)
- rename_items3, [25](#)
- rename_items_only, [26](#)
- semTools::compRelSEM(), [14](#)
- stats::t.test(), [6](#)
- stats::wilcox.test(), [27](#)
- u_mann_whitney_superioridad, [27](#)
- universidades_peruanas, [21](#), [27](#)
- validation_categoria, [28](#)