Package 'smartcor'

August 9, 2026

Title Automatic Correlation Method Selection Based on Variable Types

Version 1.0.1

Description Detects variable types (continuous, count, binary, ordinal, categorical) and selects the appropriate correlation method for each pair. Supports Pearson, Spearman, Kendall's tau, point-biserial, rank-biserial, phi, tetrachoric, polychoric, polyserial, Cramer's V, Tschuprow's T, Theil's U, Yule's Q, and Goodman-Kruskal's gamma, each with a confidence interval and p-value. Explains the selection rationale in the output, follows tidy data principles, and works in both interactive and scripted workflows. The methodology is described in Harshvardhan and Ranjan (2026) <doi:10.48550/arXiv.2607.22285>.

License GPL (>= 3)

URL https://harshvardhaniimi.github.io/smartcor/, https://github.com/harshvardhaniimi/smartcor

BugReports https://github.com/harshvardhaniimi/smartcor/issues

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.6.0)

Imports cli (>= 3.0.0), generics, ggplot2 (>= 3.4.0), graphics, grDevices, mvtnorm, polycor, rlang (>= 1.0.0), stats, tibble, utils, withr

Suggests knitr, MASS, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

NeedsCompilation no

Author M. Harshvardhan [aut, cre, cph], Pritam Ranjan [aut, cph]

Maintainer M. Harshvardhan <harshvardhan@aus.edu>

Repository CRAN

Date/Publication 2026-08-09 07:30:02 UTC

Contents

`available_methods` *Which Methods Apply to a Pair of Variable Types*

Description

Lists the methods `smart_cor()` can compute for two variable types. Use it before forcing a `method`.

Usage

```
available_methods(type_x, type_y)
```

Arguments

`type_x`, `type_y` Character strings: `"continuous"`, `"count"`, `"binary"`, `"ordinal"`, or `"categorical"`.

Value

A tibble with columns `method` (code name) and `label` (human-readable description).

Examples

```
path = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(path)
available_methods(detect_type(gss$degree), detect_type(gss$happy))
```

compare_methods *Run Every Applicable Method on One Pair*

Description

Computes the correlation between x and y with every method that applies to their type combination and lines the results up side by side. Use it to check how much the method choice changes the estimate.

Usage

```
compare_methods(
  x,
  y,
  x_type = NULL,
  y_type = NULL,
  assume_latent_normal = "auto",
  ordinal_threshold = 10,
  ignore_na = TRUE,
  conf_level = 0.95,
  bootstrap = "auto",
  n_boot = 500,
  n_perm = 500,
  seed = NULL,
  verbose = TRUE
)
```

Arguments

x, y	Vectors of the same length. May be numeric, factor, character, or logical.
x_type, y_type	Optional character strings to override automatic type detection. One of "continuous", "count", "binary", "ordinal", or "categorical".
assume_latent_normal	"auto" (default) or logical. Decides which method gets the recommended flag, exactly as in smart_cor().
ordinal_threshold	Integer passed to detect_type(). Default: 10.
ignore_na	Logical. If TRUE (default), rows with missing values are removed. If FALSE, the function errors when NA values are present.
conf_level	Numeric between 0 and 1. Confidence level for every interval in the table. Default: 0.95.
bootstrap	Character or logical: "auto" (default) bootstraps only the methods without an analytic CI, TRUE bootstraps all of them, FALSE none.
n_boot	Integer. Bootstrap replications per method. Default: 500.
n_perm	Integer. Permutation shuffles for the Theil's U p-value. Default: 500.

seed	Optional integer seed for the bootstrap and permutation draws; the global RNG state is saved and restored.
verbose	Logical. If TRUE (default), prints informative messages.

Details

The function computes every method listed by available_methods() for the detected type pair. The recommended column marks the method that smart_cor() would select under the assume_latent_normal setting, so the recommended row always matches what smart_cor() and smart_cormat() return.

When one variable is continuous or count and the other is categorical, the continuous or count variable is binned into quantiles before Cramer's V is computed, exactly as smart_cor() does.

Methods that fail (a latent-variable optimiser that does not converge, say) are kept in the output with estimate = NA.

Value

An object of class "smartcor_comparison": a list with elements:

results A tibble::tibble() with columns method, method_label, estimate, statistic, p.value, ci_lower, ci_upper, ci_method, ci_source, and recommended (logical flag for the default method).

x_name Name of the first variable.

y_name Name of the second variable.

x_type Detected (or specified) type of x.

y_type Detected (or specified) type of y.

n_complete Number of complete observations used.

conf_level Confidence level used for the intervals.

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)

# Two ordinal variables: Kendall, Spearman, gamma, and polychoric
compare_methods(gss$degree, gss$happy,
                assume_latent_normal = FALSE, bootstrap = FALSE)
```

---

detect_type *Detect the Statistical Type of a Variable*

Description

Classifies a variable as continuous, binary, ordinal, count, or categorical from its R class and the number of distinct non-missing values. The rest of the package leans on this to pick correlation methods, but it is exported because the classification is useful on its own.

Usage

```
detect_type(x, ordinal_threshold = 10, detect_count = TRUE, verbose = FALSE)
```

Arguments

x	A vector (numeric, factor, character, or logical).
ordinal_threshold	Integer. Numeric variables with this many or fewer unique non-missing values are classified as ordinal. Default: 10.
detect_count	Logical. If TRUE (default), numeric variables that are non-negative integer-valued and exceed ordinal_threshold distinct values are classified as "count". Set to FALSE to treat all such variables as "continuous".
verbose	Logical. If TRUE, prints the detection reasoning. Default: FALSE.

Details

The detection follows these rules, in order:

1. **Ordered factors** are always classified as "ordinal".
2. **Unordered factors and character vectors** are classified as "categorical", except when they have exactly 2 unique values, in which case they are "binary".
3. **Logical vectors** are classified as "binary".
4. **Numeric vectors** with exactly 2 unique values are "binary".
5. **Numeric vectors** with unique values at or below ordinal_threshold are "ordinal".
6. **Numeric vectors** that are non-negative integers with more than ordinal_threshold unique values are classified as "count". Disable by setting detect_count = FALSE.
7. All other numeric vectors are "continuous".

The ordinal_threshold parameter controls the heuristic for numeric variables: if a numeric variable has at most this many distinct values, it is assumed to represent ordered categories (e.g., a 1–5 Likert scale stored as integers). Set to 0 to disable this heuristic and treat all numeric variables as continuous (or count, if non-negative integer-valued).

Value

A single character string: "continuous", "binary", "ordinal", "count", or "categorical".

Examples

```
path = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(path)
vapply(gss, detect_type, character(1))
```

ggcor *ggplot2 Heatmaps for Smart Correlation Matrices*

Description

Two heatmaps for `smart_cormat()` output. `ggcor_heatmap()` shows the correlation values on a diverging scale centred at zero. `ggcor_method_heatmap()` colours each cell by the selected method.

Usage

```
ggcor_heatmap(
  x,
  palette = "blue_red",
  digits = 2,
  title = NULL,
  show_values = TRUE,
  text_size = 3.5,
  lower_only = FALSE,
  ...
)

ggcor_method_heatmap(
  x,
  title = NULL,
  show_values = TRUE,
  text_size = 3.5,
  digits = 2,
  ...
)
```

Arguments

`x`	A `smartcormat` object (from `smart_cormat()`), or a named list with a `correlations` element (e.g., from `smart_cor_df()`).
`palette`	Character string selecting the colour palette. For `ggcor_heatmap`: one of `"blue_red"` (default), `"purple_green"`, or a vector of 3 colours (low, mid, high).
`digits`	Integer. Number of decimal places in cell labels. Default: `2`.
`title`	Optional plot title. If `NULL`, a default is used.
`show_values`	Logical. Show correlation values in cells? Default: `TRUE`.
`text_size`	Numeric. Base text size for cell labels. Default: `3.5`.
`lower_only`	Logical. Show only the lower triangle? Default: `FALSE`.
`...`	Currently ignored.

Details

Both functions use the **ggplot2** package.

Value

A ggplot object that can be customised further.

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)
mat = smart_cormat(
  gss[, c("age", "coninc", "degree", "sex")],
  assume_latent_normal = FALSE,
  verbose = FALSE
)

# correlation heatmap
ggcor_heatmap(mat)

# lower triangle only
ggcor_heatmap(mat, lower_only = TRUE)

# method-selection heatmap
ggcor_method_heatmap(mat)

# customise further with ggplot2
ggcor_heatmap(mat) + ggplot2::labs(caption = "Frozen GSS 2024 extract")
```

---

plot.smartcormat *Plot a Correlation Heatmap*

Description

Draws an annotated heatmap of a smart correlation matrix with base R. Each cell is coloured by the correlation and labelled with its value.

Usage

```
## S3 method for class 'smartcormat'
plot(
  x,
  palette = "blue_red",
  digits = 2,
  title = NULL,
  show_values = TRUE,
  text_size = 1,
  ...
```

```
)

plot_cor_heatmap(
  x,
  palette = "blue_red",
  digits = 2,
  title = NULL,
  show_values = TRUE,
  text_size = 1,
  ...
)
```

Arguments

x	A `smartcormat` object (from `smart_cormat()`), or a named list with a `correlations` element (e.g., from `smart_cor_df()`).
palette	Character string selecting the colour palette. One of: • `"blue_red"` (default): diverging blue-white-red • `"purple_green"`: diverging purple-white-green • `"viridis"`: sequential yellow-green-blue • `"heat"`: sequential white-yellow-red • `"cool"`: sequential white-cyan-blue
digits	Integer. Number of decimal places shown in cell labels. Default: `2`.
title	Optional character string for the plot title. If `NULL` (default), a generic title is used.
show_values	Logical. Whether to display numeric values in each cell. Default: `TRUE`.
text_size	Numeric. Scaling factor for cell text size. Default: `1`.
...	Additional arguments (currently ignored).

Details

The function works with any object that contains a `correlations` element (matrix or data frame). This includes objects returned by `smart_cormat()` and `smart_cor_df()`.

Custom colour palettes can be supplied by passing a character vector of colours to `palette` instead of a named preset. The vector should contain at least 3 colours and will be interpolated to 100 steps.

Value

Invisibly returns the correlation matrix (numeric).

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)

# Plot a smart_cormat result
mat = smart_cormat(
```

```
  gss[, c("age", "coninc", "degree", "sex")],
  assume_latent_normal = FALSE,
  verbose = FALSE
)
plot(mat)

# different palettes
plot(mat, palette = "purple_green")
plot(mat, palette = "viridis")

# Plot a smart_cor_df result
res = smart_cor_df(gss[, c("age", "coninc", "degree", "sex")],
                   assume_latent_normal = FALSE)
plot_cor_heatmap(res)

# custom colour vector
plot_cor_heatmap(res, palette = c("darkblue", "white", "darkred"))
```

`print.smartcor` *Print a smartcor Object*

Description

Shows the estimate, the method that was chosen and why, the inference payload, and (when the "auto" mode ran) the latent-normality test.

Usage

```
## S3 method for class 'smartcor'
print(x, ...)
```

Arguments

`x`	A `smartcor` object.
`...`	Ignored.

Value

Invisibly returns `x`.

print.smartcormat *Print a smartcormat Object*

Description

Prints the correlation matrix, the two-letter method codes for each cell, and a legend for the codes that appear.

Usage

```
## S3 method for class 'smartcormat'
print(x, digits = 2, show_methods = TRUE, ...)
```

Arguments

x	A smartcormat object.
digits	Integer. Number of significant figures. Default: 2.
show_methods	Logical. Whether to print the method matrix. Default: TRUE.
...	Ignored.

Value

Invisibly returns x.

smart_cor *Choose and Compute a Correlation*

Description

Detects the types of x and y, selects a suitable correlation method, and returns the estimate with its confidence interval and p-value.

Usage

```
smart_cor(
  x,
  y,
  x_type = NULL,
  y_type = NULL,
  method = NULL,
  assume_latent_normal = "auto",
  ordinal_threshold = 10,
  ignore_na = TRUE,
  conf_level = 0.95,
  bootstrap = "auto",
```

```
  n_boot = 500,
  n_perm = 500,
  seed = NULL,
  verbose = TRUE
)
```

Arguments

x, y	Vectors of the same length. May be numeric, factor, character, or logical.
x_type, y_type	Optional character strings ("continuous", "count", "binary", "ordinal", "categorical") to override automatic type detection.
method	Optional character string to force a specific method. One of "pearson", "spearman", "kendall", "phi", "tetrachoric", "yules_q", "polychoric", "polyserial", "cramers_v", "theils_u", "tschuprows_t", "gamma", "rank_biserial", or "point_biserial".
assume_latent_normal	"auto" (default), logical, or NULL. "auto" runs a likelihood-ratio chi-square test (via test_bivariate_normality()) and decides whether the data satisfies bivariate normality; a saturated 2 x 2 table defaults to the observed-scale method (phi). TRUE forces polychoric / polyserial / tetrachoric; FALSE forces Kendall's tau / Spearman / phi. NULL asks interactively (defaults to TRUE in non-interactive sessions).
ordinal_threshold	Integer passed to detect_type(). Default: 10.
ignore_na	Logical. If TRUE (default), rows with missing values in either x or y are silently removed (pairwise complete observations). If FALSE, the function errors when NA values are present.
conf_level	Numeric between 0 and 1. Confidence level for intervals. Default: 0.95.
bootstrap	Character or logical. Controls bootstrap inference: "auto" (default) bootstraps only when analytic CIs are unavailable; TRUE always replaces the analytic CI with a percentile-bootstrap CI; FALSE never bootstraps. P-values remain analytic (or permutation-based for Theil's U) under every setting.
n_boot	Integer. Number of bootstrap replications when bootstrapping. Default: 500.
n_perm	Integer. Number of shuffles for the permutation p-value of Theil's U. Default: 500.
seed	Optional integer seed for the bootstrap and permutation draws. The global .Random.seed is saved and restored around a seeded call, so the global RNG state is restored. Seeded results are reproducible across R sessions but do not match the Python package, whose numpy generator draws differently.
verbose	Logical. If TRUE (default), prints messages about the detection and selection process.

Details

The method selection follows Harshvardhan, M. and Ranjan, P. (2026), "smartcor: Intelligent Correlation Method Selection for Mixed Variable Types" (arXiv:2607.22285, doi:10.48550/arXiv.2607.22285); the package vignettes (browseVignettes("smartcor")) cover the same material in more depth:

Variable types	Default method	Alternative
Correlation methods (signed, from -1 to +1)		
continuous + continuous	Pearson	Spearman (suggested, especially if nonlinear)
continuous + binary	Pearson = point-biserial	–
binary + binary	Pearson = phi	Tetrachoric (preferred, if latent normality holds)
continuous + ordinal	Spearman/Kendall	Polyserial (preferred, if latent normality holds)
ordinal + ordinal	Kendall's tau	Polychoric (preferred, if latent normality holds)
binary + ordinal	Rank-biserial	Spearman
Association measures (unsigned, from 0 to 1)		
continuous + categorical	Cramer's V (binned)	–
binary + categorical	Cramer's V	–
ordinal + categorical	Cramer's V	–
categorical + categorical	Cramer's V	Theil's U, Tschuprow's T

Count variables are treated as numeric continuous variables, so each count combination follows the corresponding continuous row.

When `assume_latent_normal = "auto"` (the default), a likelihood-ratio chi-square test (`test_bivariate_normality()`) decides whether the data satisfies bivariate normality. A binary-binary pair is an exception: its 2 x 2 table is saturated (df = 0), the test has nothing to work with, and the data cannot support the latent-normality assumption, so smartcor reports phi. Tetrachoric remains available via `assume_latent_normal = TRUE` or `method = "tetrachoric"`.

When a latent-variable estimator fails numerically (singular Hessian, sparse table), the function falls back to Kendall's tau or Spearman and the returned `method`, `method_label`, and `rationale` name the method actually used, not the one that failed.

One caveat on the point-biserial interval: it reuses the Fisher-z variance 1/(n - 3), which assumes bivariate normality; a binary margin cannot satisfy that exactly, so treat the interval as approximate.

Theil's U is asymmetric: `method = "theils_u"` computes U(y|x), the proportion of the entropy in `y` explained by `x`. Swap the arguments for the other direction. Its p-value comes from a permutation test (`n_perm` shuffles of `y`), not from the bootstrap.

Value

An object of class `"smartcor"`: a named list with elements `estimate`, `method`, `method_label`, `x_type`, `y_type`, `n`, `n_complete`, `statistic`, `p.value`, `ci_lower`, `ci_upper`, `conf_level`, `ci_method`, `ci_source`, `p_method`, `null_hypothesis`, `p_interpretation`, `alternatives`, and `rationale`. Has custom `print.smartcor()` and `tidy.smartcor()` methods.

Examples

```
path = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(path)

# Continuous and binary variables use point-biserial correlation.
smart_cor(gss$coninc, gss$sex, verbose = FALSE)

# Two ordinal variables use Kendall's tau when latent normality is not assumed.
smart_cor(gss$degree, gss$happy,
```

```
            assume_latent_normal = FALSE, verbose = FALSE)

# Force a method when the analysis calls for it.
smart_cor(gss$coninc, gss$age, method = "spearman", verbose = FALSE)
```

`smart_cormat` *Build a Mixed-Type Correlation Matrix*

Description

Runs `smart_cor()` on every pair of columns and records both the estimate and the method used for each pair.

Usage

```
smart_cormat(
  data,
  cols = NULL,
  types = NULL,
  assume_latent_normal = "auto",
  ordinal_threshold = 10,
  ignore_na = TRUE,
  conf_level = 0.95,
  bootstrap = "auto",
  n_boot = 500,
  n_perm = 500,
  seed = NULL,
  verbose = TRUE
)
```

Arguments

`data`	A data frame or tibble. All columns are used unless `cols` is specified.
`cols`	A character vector of column names to include. Default: all columns.
`types`	An optional named list mapping column names to types (`"continuous"`, `"count"`, `"binary"`, `"ordinal"`, `"categorical"`). Columns not listed are auto-detected.
`assume_latent_normal`	`"auto"` (default), logical, or `NULL`. Passed to each `smart_cor()` call. `"auto"` tests each pair individually via `test_bivariate_normality()`. `NULL` asks interactively once for all pairs.
`ordinal_threshold`	Integer passed to `detect_type()`. Default: `10`.
`ignore_na`	Logical. If `TRUE` (default), missing values are handled pairwise (each pair uses its own complete observations). If `FALSE`, any pair with missing values fails; the failure is reported as a warning (when `verbose = TRUE`) and the cell is left `NA`.

conf_level	Numeric between 0 and 1. Confidence level passed to each smart_cor() call. Default: 0.95.
bootstrap	Character or logical, passed to each smart_cor() call: "auto" (default), TRUE, or FALSE.
n_boot	Integer. Bootstrap replications per pair when bootstrapping. Default: 500.
n_perm	Integer. Permutation shuffles for Theil's U p-values. Default: 500.
seed	Optional integer seed passed to each smart_cor() call; the global RNG state is saved and restored around seeded draws.
verbose	Logical. If TRUE (default), prints progress information. Set to FALSE for silent operation (useful inside scripts).

Details

For each pair of columns, smart_cor() detects variable types and selects the appropriate method. The result is a symmetric matrix of correlation/association measure estimates annotated with the method used for each cell.

Since different methods are used for different pairs, the resulting matrix may not be positive semi-definite. This is expected and reflects the heterogeneous nature of the data.

Diagonal cells hold 1 by definition; the methods matrix records for each diagonal cell the method that a self-pair of that type would get, so every cell of the matrix is labelled the same way.

Value

An object of class "smartcormat": a list with elements:

correlations Numeric matrix of correlation estimates.

methods Character matrix of method codes used for each pair.

types Named character vector of detected/specified variable types.

details List of smart_cor() result objects (one per pair).

Examples

```
path = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(path)

smart_cormat(
  gss[, c("age", "coninc", "degree", "sex")],
  assume_latent_normal = FALSE,
  verbose = FALSE
)

# Override a detected type when the study design requires it.
smart_cormat(
  gss[, c("age", "coninc", "degree")],
  types = list(age = "continuous", degree = "ordinal"),
  assume_latent_normal = FALSE,
  verbose = FALSE
)
```

smart_cor_df *Correlation Matrix as Plain Data Frames*

Description

A thin wrapper around `smart_cormat()` for pipeline work. It never prompts, prints nothing, and returns two plain data frames (the estimates and the method used for each pair) instead of an S3 object.

Usage

```
smart_cor_df(
  data,
  cols = NULL,
  types = NULL,
  assume_latent_normal = "auto",
  ordinal_threshold = 10,
  ignore_na = TRUE,
  conf_level = 0.95,
  bootstrap = "auto",
  n_boot = 500,
  n_perm = 500,
  seed = NULL
)
```

Arguments

`data`	A data frame or tibble.
`cols`	Optional character vector of column names to include. Default: all columns.
`types`	Optional named list mapping column names to types (`"continuous"`, `"count"`, `"binary"`, `"ordinal"`, `"categorical"`). Columns not listed are auto-detected.
`assume_latent_normal`	`"auto"` (default) or logical, passed to `smart_cormat()`. `NULL` is not useful here because this wrapper never prompts.
`ordinal_threshold`	Integer passed to `detect_type()`. Default: `10`.
`ignore_na`	Logical. If `TRUE` (default), missing values are handled pairwise. If `FALSE`, pairs with missing values come back `NA`.
`conf_level`	Numeric between 0 and 1. Confidence level passed through to each `smart_cor()` call. Default: `0.95`.
`bootstrap`	Character or logical, passed through: `"auto"` (default), `TRUE`, or `FALSE`.
`n_boot`	Integer. Bootstrap replications per pair. Default: `500`.
`n_perm`	Integer. Permutation shuffles for Theil's U p-values. Default: `500`.
`seed`	Optional integer seed for the bootstrap and permutation draws; the global RNG state is saved and restored.

Details

By default it uses `assume_latent_normal = "auto"`, so each pair is screened with `test_bivariate_normality()` and the method follows the test (a saturated 2 x 2 pair gets phi).

The two returned data frames share identical row and column names (the variable names from `data`). Diagonal entries in `correlations` are 1; diagonal entries in `methods` name the method a self-pair of that type would get, matching `smart_cormat()`.

Value

A named list with two elements:

`correlations` A data frame of correlation estimates (numeric).

`methods` A data frame of method codes (character) used for each pair.

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)

# Correlations and methods for four columns in the frozen GSS extract
result = smart_cor_df(
  gss[, c("age", "coninc", "degree", "sex")],
  assume_latent_normal = FALSE
)
result$correlations
result$methods

# Missing values are handled pairwise by default
example = gss[, c("age", "coninc", "sex")]
example$coninc[1:3] = NA
smart_cor_df(example, assume_latent_normal = FALSE)
```

---

test_bivariate_normality

Test Bivariate Normality for Two Ordinal Variables

Description

Likelihood-ratio chi-square test for bivariate normality, the assumption underlying polychoric correlation, following Joreskog (2005). The observed cell frequencies of the contingency table are compared with the expected frequencies under a bivariate normal threshold model.

Usage

```
test_bivariate_normality(x, y = NULL, alpha = 0.05, max_levels = 10L)
```

Arguments

x, y	Integer or factor vectors representing ordinal (or binary) variables. Alternatively, a pre-computed contingency table (matrix or table object) can be passed as x with y = NULL.
alpha	Significance level for the test. Default: 0.05.
max_levels	Integer. Cap on the number of categories per variable when building the contingency table from numeric vectors. A numeric variable with more than max_levels distinct values (e.g. a continuous or count variable passed in for a polyserial-style screen) is binned into max_levels quantile groups before the table is formed. This keeps the maximum-likelihood threshold model tractable: an unbinned continuous variable would induce one threshold per distinct value, making the ML fit prohibitively slow and the chi-square reference distribution meaningless. Genuine ordinal/binary inputs (at most max_levels distinct values) are never altered. Default: 10.

Details

The bivariate normal threshold model is fitted with the two-step estimator (Olsson 1979) via polycor::polychor(): the thresholds come from the marginal proportions and the correlation is then estimated by one-dimensional maximum likelihood given those thresholds. The likelihood-ratio statistic $G^2 = 2\sum_{ij} O_{ij} \log(O_{ij}/E_{ij})$ compares the observed counts O_{ij} with the expected counts E_{ij} under the fitted model (cells with zero observed count are skipped). The degrees of freedom equal $m_1 \times m_2 - m_1 - m_2$, where m_1 and m_2 are the numbers of categories of the two variables. For a 2×2 table the model is saturated (df = 0) and no test is possible. The Python smartcor package computes the same statistic, so the two implementations agree.

Value

An object of class "smartcor_normtest", a list with elements:

statistic The likelihood-ratio chi-square statistic.

df Degrees of freedom.

p.value The p-value from pchisq(statistic, df).

rho The two-step estimate of the polychoric correlation.

row_thresholds Estimated thresholds for the first variable.

col_thresholds Estimated thresholds for the second variable.

n Total number of observations.

m1 Number of categories for the first variable.

m2 Number of categories for the second variable.

alpha The significance level used.

conclusion One of "reject", "fail_to_reject", or "saturated".

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)

# Check the latent-normality assumption for two ordinal variables
test_bivariate_normality(gss$degree, gss$happy)
```

`tidy.smartcor` *Tidy a smartcor Object*

Description

Lays a `smart_cor()` result out as a one-row tibble in the manner of the **broom** package, so results drop straight into a dplyr pipeline.

Usage

```
## S3 method for class 'smartcor'
tidy(x, ...)
```

Arguments

`x`	A `smartcor` object.
`...`	Ignored.

Value

A one-row `tibble::tibble()` with columns covering identifiers (`var_x`, `var_y`, `x_type`, `y_type`), the point estimate, the chosen method (`method`, `method_label`, `rationale`), inference details (`statistic`, `p.value`, `p_method`, `null_hypothesis`, `p_interpretation`, `ci_lower`, `ci_upper`, `conf_level`, `ci_method`, `ci_source`), and sample sizes (`n`, `n_complete`).

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)
res = smart_cor(gss$coninc, gss$age, verbose = FALSE)
tidy(res)
```

tidy.smartcormat *Tidy a smartcormat Object*

Description

Turns a `smart_cormat()` result into a long tibble with one row per variable pair (lower triangle only, no self-correlations).

Usage

```
## S3 method for class 'smartcormat'
tidy(x, ...)
```

Arguments

`x`	A `smartcormat` object.
`...`	Ignored.

Value

A `tibble::tibble()` with the same columns as `tidy.smartcor()`, one row per unique pair.

Examples

```
csv = system.file("extdata", "gss_2024_casestudy.csv", package = "smartcor")
gss = read.csv(csv)
mat = smart_cormat(
  gss[, c("age", "coninc", "sex")],
  assume_latent_normal = FALSE,
  verbose = FALSE
)
tidy(mat)
```

tidy.smartcor_comparison *Tidy a smartcor Comparison*

Description

Lays a `compare_methods()` result out as a tibble with one row per candidate method, in the manner of the **broom** package.

Usage

```
## S3 method for class 'smartcor_comparison'
tidy(x, ...)
```

Arguments

x	A `smartcor_comparison` object.
...	Ignored.

Value

A `tibble::tibble()` with one row per method.

Index