Package 'joinless'

May 8, 2026

Title Exploratory Analysis of Relationships Between Variables

Version 0.0.1

Maintainer Braylin Alexander Jiménez Reynoso <braylinjr1511@gmail.com>

Description Provides tools to explore and summarize relationship patterns between variables across one or multiple datasets. The package relies on efficient sampling strategies to estimate pairwise associations and supports quick exploratory data analysis for large or heterogeneous data sources.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0), tibble

Config/testthat/edition 3

NeedsCompilation no

Author Braylin Alexander Jiménez Reynoso [aut, cre]

Repository CRAN

Date/Publication 2025-12-11 19:30:02 UTC

Contents

`joinless` *Infer relationship types between variables in two datasets using sampling*

Description

This function compares selected variables from two data frames and infers their relational structure (e.g., one-to-one, many-to-one). It uses random sampling—either automatic or user-defined—to estimate match behavior, uniqueness patterns, and missingness characteristics. The goal is to help diagnose potential join keys or detect unrelated fields without performing full-table comparisons.

Usage

```
joinless(
  x,
  y,
  x_vars = NULL,
  y_vars = NULL,
  conf = 0.95,
  error = 0.05,
  n_x = NULL,
  n_y = NULL,
  max_vars = 20,
  ignore = character(0),
  missingness_tol = 0.1,
  type_coerce = TRUE,
  seed = NULL,
  verbose = FALSE,
  info = FALSE
)
```

Arguments

x, y	Data frames. Input datasets to be compared.
x_vars, y_vars	Character vectors specifying the column names to compare. If NULL, the function selects up to max_vars variables from each dataset.
conf	Numeric. Confidence level used to compute automatic sample sizes (default: 0.95).
error	Numeric. Margin of error used in sample size calculation (default: 0.05).
n_x, n_y	Optional fixed sample sizes for x and y. If not provided, sample sizes are computed automatically based on population size, conf, and error.
max_vars	Integer. Maximum number of variables to compare per dataset. Defaults to 20.
ignore	Character vector of relation types to exclude from the output. By default, no types are excluded.
missingness_tol	Numeric. Maximum tolerated proportion of missing/problematic values within a variable (default: 0.1). Problematic values include: NA, NaN, NULL, Inf, -Inf, empty strings, and whitespace-only strings.
type_coerce	Logical. If TRUE (default), attempts to coerce variables to a common type (typically character) when domains differ. If FALSE, type mismatches produce an "error_type" result.

seed	Optional integer. Random seed to make the sampling reproducible.
verbose	Logical. If TRUE, prints progress messages during execution.
info	Logical. If FALSE (default), returns only the inferred relationship type for each variable pair. If TRUE, returns additional diagnostics such as match rate, missingness rates, inferred types, and notes.

Details

Relationship inference is determined using:

- **Match rate**: proportion of keys in x found in y
- **Key uniqueness**: frequency distribution of non-missing values

Based on these, relationships are classified as:

- "one-one"
- "many-one"
- "one-many"
- "many-many"
- "unrelated" (very low or zero match rate)
- "null" (missingness above tolerance)
- "error_type" (incompatible types and coercion disabled)

Value

A data frame summarizing the inferred relationship between every variable pair. If info = FALSE, the output contains:

- x_var: variable name in x
- y_var: variable name in y
- relation_type: inferred relationship

If info = TRUE, additional columns include:

- n_used: sample size used
- match_rate: proportion of sampled values from x found in y
- null_rate_x, null_rate_y: missingness/problematic rates
- type_x, type_y: underlying storage types
- notes: diagnostic messages

Examples

```
df1 <- data.frame(
  id    = 1:5,
  value = 1:5
)

df2 <- data.frame(
  id    = 3:7,
  value = 3:7
)

joinless(df1, df2, x_vars = "id", y_vars = "id")
joinless(df1, df2, conf = 0.99, error = 0.02, info = TRUE)
joinless(df1, df2, ignore = "unrelated")
```

`joinless_multiple` *Infer relationship types between one dataset and multiple counterparts*

Description

This function is a convenience wrapper around `joinless()` that compares a single dataset `x` against multiple datasets supplied in a list `ys`. Internally it calls `joinless()` once per dataset and row-binds the results, adding an extra column that identifies the target dataset.

Usage

```
joinless_multiple(
  x,
  ys,
  x_vars = NULL,
  y_vars = NULL,
  dataset_names = NULL,
  ...
)
```

Arguments

`x`	A data frame. The reference dataset to compare from.
`ys`	A named or unnamed list of data frames. Each element is treated as a separate target dataset to compare `x` against.
`x_vars`	Optional character vector of column names in `x` to compare. Passed directly to `joinless()`. If `NULL`, the default behavior of `joinless()` is used (i.e., it selects up to `max_vars` variables).
`y_vars`	Optional character vector of column names to use in each target dataset. If not `NULL`, the function filters `y_vars` to those that exist in each dataset in `ys` before calling `joinless()`. If `NULL`, each dataset is allowed to use its own default variable selection in `joinless()` (up to `max_vars`).

`dataset_names` Optional character vector with labels for each dataset in `ys`. If `NULL` and `ys` is a named list, the list names are used (empty names are replaced by `"y1"`, `"y2"`, ...). If `ys` is unnamed, generic names `"y1"`, `"y2"`, ... are generated. The length of `dataset_names` must match the length of `ys` if supplied.

`...` Additional arguments passed on to `joinless()`, such as `conf`, `error`, `n_x`, `n_y`, `max_vars`, `ignore`, `missingness_tol`, `type_coerce`, `seed`, `verbose`, and `info`.

Details

For each dataset in `ys`, the function:

- optionally restricts the variables in `x` via `x_vars`,
- optionally restricts the variables in that dataset via `y_vars`,
- calls `joinless()` with the provided settings,
- tags the output with a dataset name.

When `y_vars` is not `NULL`, the function intersects `y_vars` with the column names of each dataset in `ys`. This means that:

- Variables listed in `y_vars` but missing in a given dataset are silently dropped for that dataset.
- If *none* of the variables in `y_vars` exist in a particular dataset, that dataset is skipped and a warning is emitted.

This behavior avoids producing `"error_type"` rows solely due to missing columns in some of the target datasets.

If all datasets are skipped (e.g., because none contain the requested `y_vars`), the function returns an empty data frame.

Value

A data frame that row-binds the result of `joinless()` for all target datasets that were processed. It contains all columns returned by `joinless()` plus an additional column:

- `dataset`: identifier of the target dataset (one per element of `ys`).

If all datasets are skipped, an empty data frame is returned.

Examples

```
df_base <- data.frame(id = 1:5, value = 1:5)
df_a <- data.frame(id = 3:7, value = 3:7)
df_b <- data.frame(id_alt = 1:5, value = 11:15)

# Compare the same key from df_base against multiple datasets
res <- joinless_multiple(
  x = df_base,
  ys = list(a = df_a, b = df_b),
  x_vars = "id",
  y_vars = c("id", "id_alt"),
```

```
  info = TRUE
)
```

Index