Package: rwetools (via r-universe)

May 29, 2026

Title Estimating Propensity Scores (PS), PS-Based Weights, and Effects

Version 0.1.2

Description Toolbox that provides a streamlined, end-to-end workflow for propensity score analysis in generating real-world evidence from real-world data. The package covers the full analytic pipeline - from estimating propensity scores via logistic regression, to calculating weights or creating a matched cohort, to generating publication-ready Table 1s with standardized mean differences and weighted balance diagnostics. It also estimates incidence rates, hazard ratios, risk ratios, and risk differences with support for stratified and direct-standardized analyses. All core functions produce formatted 'Excel' reports with embedded 'README' documentation, making results immediately shareable with collaborators and stakeholders. Methods are based on Rosenbaum and Rubin (1983) <doi:10.1093/biomet/70.1.41>, Austin (2011) <doi:10.1080/00273171.2011.568786>, and Desai et al. (2017) <doi:10.1097/EDE.0000000000000595>.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

Imports stats, utils, parallel, survival, survey

Suggests openxlsx, ggplot2, MatchIt, testthat (>= 3.0.0)

URL https://github.com/hanseul0618/rwetools

Config/testthat/edition 3

NeedsCompilation no

Author Hanseul Cho [aut, cre], Georg Hahn [aut], Janinne Ortega-Montiel [aut], Julie Paik [aut], Elisabetta Patorno [aut]

Maintainer Hanseul Cho <hanseul0618@gmail.com>

Config/pak/sysreqs make

Repository https://hanseul0618.r-universe.dev

Date/Publication 2026-05-28 17:07:25 UTC

RemoteUrl https://github.com/cran/rwetools

RemoteRef HEAD

RemoteSha 7c6a8eafd99415d77504464a1afb3a94e747921a

Contents

`build_table1` *Build a Table 1 comparing baseline characteristics between groups*

Description

Generates a descriptive Table 1 with counts, percentages, means, SDs, crude differences, standardized mean differences (SMD), and missingness. Supports inverse probability weighting via a user-supplied weight column and optionally saves the output to an Excel file.

Usage

```
build_table1(
  in_df,
  out_xlsxpath = NULL,
  exposure_var,
  exp_value = 1,
  ref_value = 0,
  use_weights = FALSE,
  weight_var = "psweight",
  cont_vars = NULL,
  cat_vars = NULL,
  binary_vars = NULL,
  drop_vars = NULL,
  drop_varpattern = NULL,
  Var_colname = "Variable",
  Vartype_colname = "Type",
  Total_colname = "Total",
  Exp_colname = "Exp",
```

```
  Ref_colname = "Ref",
  CrudeDiff_colname = "Crude_diff",
  StdDiff_colname = "Std_diff",
  MissingTotal_colname = "Missing_Total_N_Pct",
  MissingExp_colname = "Missing_Exp_N_Pct",
  MissingRef_colname = "Missing_Ref_N_Pct",
  ExistingTotal_colname = "Existing_Total_N_denom",
  ExistingExp_colname = NULL,
  ExistingRef_colname = NULL,
  mean_decimal = 2,
  sd_decimal = 2,
  n_decimal = 0,
  pct_decimal = 1,
  use_absolute_values_for_diff = FALSE,
  add_n_of_patients_row = TRUE,
  verbose = TRUE
)
```

Arguments

in_df	Data frame containing the analytic cohort.
out_xlsxpath	Character string. File path for Excel output, or NULL to skip (default NULL).
exposure_var	Character string. Name of the binary exposure column.
exp_value	Value in exposure_var representing the exposed group (default 1).
ref_value	Value in exposure_var representing the reference group (default 0).
use_weights	Logical. Apply weights? (default FALSE).
weight_var	Character string. Name of the weight column (default "psweight").
cont_vars	Character vector of continuous variable names.
cat_vars	Character vector or named list of categorical variable names. If a named list, each element should be a character vector of factor levels.
binary_vars	Character vector of binary variable names.
drop_vars	Character vector of variable names to exclude.
drop_varpattern	Character vector of regex patterns; matching variables are excluded.
Var_colname, Vartype_colname, Total_colname, Exp_colname, Ref_colname	Column-name overrides for the output table.
CrudeDiff_colname, StdDiff_colname	Column-name overrides for difference columns.
MissingTotal_colname, MissingExp_colname, MissingRef_colname	Column-name overrides for missingness columns.
ExistingTotal_colname, ExistingExp_colname, ExistingRef_colname	Column-name overrides for non-missing-count columns. Set to NULL to omit (default for Exp/Ref).
mean_decimal, sd_decimal	Integer. Decimal places for mean and SD.

n_decimal	Integer. Decimal places for counts (default 0).
pct_decimal	Integer. Decimal places for percentages (default 1).
use_absolute_values_for_diff	Logical. Show absolute differences? (default FALSE).
add_n_of_patients_row	Logical. Prepend an "n_of_patients" row? (default TRUE).
verbose	Logical. Print progress messages (default TRUE).

Value

Invisibly returns the Table 1 data frame.

Side Effects

When out_xlsxpath is not NULL, creates the output directory (if needed) and writes an Excel workbook.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df <- read.csv(csv_path)

# Unweighted Table 1
tbl <- build_table1(
  in_df        = df,
  exposure_var = "exposure",
  cont_vars    = c("cont1", "cont2", "cont3"),
  binary_vars  = c("binary1", "binary2"),
  cat_vars     = c("cat1", "cat2"),
  verbose      = FALSE
)
head(tbl)


# Weighted Table 1 with Excel output (requires openxlsx)
if (requireNamespace("openxlsx", quietly = TRUE)) {
  df_ps <- estimate_ps(
    in_df        = df,
    exposure_var = "exposure",
    class_vars   = c("cat1", "cat2", "cat3", "cat4"),
    cont_vars    = c("cont1", "cont2", "cont3"),
    verbose      = FALSE
  )
  df_wt <- create_ps_weights(
    in_df         = df_ps,
    exposure_var  = "exposure",
    ps_var        = "ps",
    weight_method = "mw",
    weight_var    = "mw_wt",
    verbose       = FALSE
  )
```

```
  out_xlsx <- tempfile(fileext = ".xlsx")
  tbl_wt <- build_table1(
    in_df        = df_wt,
    out_xlsxpath = out_xlsx,
    exposure_var = "exposure",
    use_weights  = TRUE,
    weight_var   = "mw_wt",
    cont_vars    = c("cont1", "cont2", "cont3"),
    binary_vars  = c("binary1", "binary2"),
    cat_vars     = c("cat1", "cat2"),
    verbose      = FALSE
  )
}
```

`create_ps_fs_weights` *Calculate PS Fine Stratification Weights, Trim Non-overlapping Regions, and Generate Diagnostics*

Description

Combined function that (1) creates PS fine strata, (2) calculates stratification weights, (3) trims non-overlapping PS regions, and optionally (4) builds a crude vs weighted balance table (Table 1) and (5) generates diagnostic plots.

Usage

```
create_ps_fs_weights(
  in_df,
  out_csvpath = NULL,
  out_xlsxpath_report = NULL,
  out_dir_plots = NULL,
  trim_nonoverlap_region = TRUE,
  exposure_var = "exp",
  exp_value = 1,
  ref_value = 0,
  ps_var = NULL,
  weight_var = "ps_fs_wt",
  number_of_strata = 50,
  stratification_method = c("exposure", "cohort"),
  estimand = c("ATT", "ATE"),
  make_unwt_wt_table1 = FALSE,
  table1_cont_vars = NULL,
  table1_binary_vars = NULL,
  table1_cat_vars = NULL,
  std_diff_threshold = 0.1,
  readme_text = NULL,
```

```
  verbose = TRUE
)
```

Arguments

in_df	Data frame with PS already calculated (the "crude" / untrimmed data)
out_csvpath	Character. Path for output CSV of trimmed+weighted data (optional)
out_xlsxpath_report	Character. Path for Excel diagnostic report (optional)
out_dir_plots	Character. Directory to save diagnostic plots (NULL = skip plots)
trim_nonoverlap_region	Logical. Trim non-overlapping PS regions (default TRUE)
exposure_var	Character. Name of binary exposure variable (default "exp")
exp_value	Value representing exposed group (default 1)
ref_value	Value representing reference group (default 0)
ps_var	Character. Name of PS variable in the data (required)
weight_var	Character. Name for the stratification weight variable (default "ps_fs_wt")
number_of_strata	Integer. Number of strata to create (default 50)
stratification_method	Character. "exposure" or "cohort" (default "exposure")
estimand	Character. "ATT" or "ATE" (default "ATT")
make_unwt_wt_table1	Logical. Build crude vs weighted balance table (default FALSE)
table1_cont_vars	Character vector. Continuous vars for Table 1 (auto-detected if NULL)
table1_binary_vars	Character vector. Binary vars for Table 1 (auto-detected if NULL)
table1_cat_vars	Character vector. Categorical vars for Table 1 (auto-detected if NULL)
std_diff_threshold	Numeric. Threshold for acceptable standardised difference (default 0.1)
readme_text	Character. Optional message for README sheet in Excel report
verbose	Logical. Print progress messages (default TRUE).

Value

Invisibly returns the trimmed+weighted data frame.

Side Effects

- Writes a CSV file when `out_csvpath` is provided.
- Creates directories, writes an Excel diagnostic report, and saves PNG plot files when the corresponding path arguments are supplied.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df_ps <- estimate_ps(
  in_df        = read.csv(csv_path),
  exposure_var = "exposure",
  class_vars   = c("cat1", "cat2", "cat3", "cat4"),
  cont_vars    = c("cont1", "cont2", "cont3"),
  verbose      = FALSE
)

result <- create_ps_fs_weights(
  in_df                 = df_ps,
  exposure_var          = "exposure",
  ps_var                = "ps",
  weight_var            = "fs_wt",
  number_of_strata      = 10,
  stratification_method = "exposure",
  estimand              = "ATT",
  verbose               = FALSE
)
summary(result$fs_wt)
```

---

create_ps_matched_cohort

Perform propensity score matching and optionally generate diagnostic reports.

Description

This function performs 1:k nearest neighbor PS matching using MatchIt and generates comprehensive diagnostic reports including assumption checks, balance tables, and plots.

Usage

```
create_ps_matched_cohort(
  in_df = NULL,
  in_csvpath = NULL,
  out_csvpath_matcheddata = NULL,
  out_csvpath_crudedata_w_matchindicator = NULL,
  out_xlsxpath_report = NULL,
  out_dir_plots = NULL,
  exposure_var = "exp",
  exp_value = 1,
  ref_value = 0,
  ps_var = "ps",
  ratio = 1,
  caliper = 0.2,
  replace = FALSE,
```

```
  make_crude_matched_table1 = FALSE,
  table1_cont_vars = NULL,
  table1_binary_vars = NULL,
  table1_cat_vars = NULL,
  std_diff_threshold = 0.1,
  readme_text = NULL,
  verbose = TRUE
)
```

Arguments

in_df	Data frame containing the input data with PS already calculated (optional if in_csvpath provided)
in_csvpath	Character string. Path to input CSV file with PS already calculated (optional if in_df provided)
out_csvpath_matcheddata	Character string. Path for matched cohort CSV file (optional)
out_csvpath_crudedata_w_matchindicator	Character string. Path for crude data with match indicator CSV file (optional)
out_xlsxpath_report	Character string. Path for output Excel diagnostic report (optional).
out_dir_plots	Character string. Directory to save plot files (optional).
exposure_var	Character string. Name of the binary exposure/treatment variable column (default: "exp")
exp_value	Value representing the exposed/treated group (default: 1)
ref_value	Value representing the reference/control group (default: 0)
ps_var	Character string. Name of the PS variable column in the data (default: "ps")
ratio	Integer. Matching ratio (1:k). Default is 1 for 1:1 matching.
caliper	Numeric. Caliper width in standard deviations of logit(PS). Default is 0.2.
replace	Logical. Whether to match with replacement. Default is FALSE.
make_crude_matched_table1	Logical. Whether to generate crude and matched Table 1 (default: FALSE).
table1_cont_vars	Character vector. Names of continuous variables for Table 1 (optional, auto-detected if NULL)
table1_binary_vars	Character vector. Names of binary variables for Table 1 (optional, auto-detected if NULL)
table1_cat_vars	Character vector. Names of categorical variables for Table 1 (optional, auto-detected if NULL)
std_diff_threshold	Numeric. Threshold for acceptable standardized difference (default: 0.1)
readme_text	Character string. Optional message to include in README sheet of Excel report
verbose	Logical. Print progress messages (default TRUE).

Value

A data frame containing the matched cohort only (crude data with match indicator can be saved to CSV via out_csvpath_crudedata_w_matchindicator).

Side Effects

- Writes matched-cohort and/or crude-data CSV files when the corresponding path arguments are provided.
- Creates directories, writes an Excel diagnostic report, and saves PNG plot files when the corresponding path arguments are supplied.

Examples

```
# Requires MatchIt
if (requireNamespace("MatchIt", quietly = TRUE)) {
  csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
  df_ps <- estimate_ps(
    in_df        = read.csv(csv_path),
    exposure_var = "exposure",
    class_vars   = c("cat1", "cat2", "cat3", "cat4"),
    cont_vars    = c("cont1", "cont2", "cont3"),
    verbose      = FALSE
  )

  matched <- create_ps_matched_cohort(
    in_df        = df_ps,
    exposure_var = "exposure",
    ps_var       = "ps",
    ratio        = 1,
    caliper      = 0.2,
    verbose      = FALSE
  )
  nrow(matched)
}
```

---

create_ps_weights *Calculate propensity score weights and optionally generate diagnostic reports.*

Description

This function calculates various types of propensity score weights (matching weights, overlap weights, IPTW, stabilized IPTW) and adds them to a dataset that already contains propensity scores. Optionally generates comprehensive diagnostic reports including assumption checks, balance tables, and plots.

Usage

```
create_ps_weights(
  in_df = NULL,
  in_csvpath = NULL,
  out_csvpath = NULL,
  out_xlsxpath_report = NULL,
  out_dir_plots = NULL,
  exposure_var = "exp",
  exp_value = 1,
  ref_value = 0,
  ps_var = "ps",
  weight_method = c("mw", "ow", "iptw", "stabiptw"),
  weight_var = "psweight",
  estimand = c("ATT", "ATE", "ATO"),
  make_unwt_wt_table1 = FALSE,
  table1_cont_vars = NULL,
  table1_binary_vars = NULL,
  table1_cat_vars = NULL,
  std_diff_threshold = 0.1,
  readme_text = NULL,
  verbose = TRUE
)
```

Arguments

in_df	Data frame containing the input data with PS already calculated (optional if in_csvpath provided)
in_csvpath	Character string. Path to input CSV file with PS already calculated (optional if in_df provided)
out_csvpath	Character string. Path for output CSV file (optional)
out_xlsxpath_report	Character string. Path for output Excel diagnostic report (optional). If NULL, no diagnostic report is generated.
out_dir_plots	Character string. Directory to save plot files (optional). If NULL, no plots are saved.
exposure_var	Character string. Name of the binary exposure/treatment variable column (default: "exp")
exp_value	Value representing the exposed/treated group (default: 1)
ref_value	Value representing the reference/control group (default: 0)
ps_var	Character string. Name of the PS variable column in the data (default: "ps")
weight_method	Character string. Type of weight to calculate: "mw" (matching weights), "ow" (overlap weights), "iptw" (inverse probability weights), "stabiptw" (stabilized IPTW). Only ONE method can be specified.
weight_var	Character string. Name for the weight variable column (default: "psweight")

estimand — Character string. Target estimand: "ATT" (average treatment effect on treated), "ATE" (average treatment effect), or "ATO" (average treatment effect in overlap population). Only used for IPTW methods. For MW/OW, estimand is determined by the method itself.

make_unwt_wt_table1 — Logical. Whether to generate unweighted and weighted Table 1 (default: FALSE). Only used if out_xlsxpath_report is provided.

table1_cont_vars — Character vector. Names of continuous variables for Table 1 (optional, auto-detected if NULL)

table1_binary_vars — Character vector. Names of binary variables for Table 1 (optional, auto-detected if NULL)

table1_cat_vars — Character vector. Names of categorical variables for Table 1 (optional, auto-detected if NULL)

std_diff_threshold — Numeric. Threshold for acceptable standardized difference (default: 0.1)

readme_text — Character string. Optional message to include in README sheet of Excel report

verbose — Logical. Print progress messages (default TRUE).

Value

A data frame with the weight column added. Also saves to CSV if path specified.

Side Effects

- Writes a CSV file when `out_csvpath` is provided.
- Creates directories, writes an Excel diagnostic report, and saves PNG plot files when the corresponding path arguments are supplied.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df_ps <- estimate_ps(
  in_df        = read.csv(csv_path),
  exposure_var = "exposure",
  class_vars   = c("cat1", "cat2", "cat3", "cat4"),
  cont_vars    = c("cont1", "cont2", "cont3"),
  verbose      = FALSE
)

# Add matching weights
df_mw <- create_ps_weights(
  in_df         = df_ps,
  exposure_var  = "exposure",
  ps_var        = "ps",
  weight_method = "mw",
  weight_var    = "mw_wt",
```

```
  verbose       = FALSE
)
summary(df_mw$mw_wt)


# Add IPTW weights with Excel diagnostic report (requires openxlsx, ggplot2)
if (requireNamespace("openxlsx", quietly = TRUE) &&
    requireNamespace("ggplot2", quietly = TRUE)) {
  out_xlsx <- tempfile(fileext = ".xlsx")
  out_dir  <- tempdir()
  df_iptw  <- create_ps_weights(
    in_df               = df_ps,
    exposure_var        = "exposure",
    ps_var              = "ps",
    weight_method       = "iptw",
    weight_var          = "iptw_wt",
    estimand            = "ATE",
    out_xlsxpath_report = out_xlsx,
    make_unwt_wt_table1 = TRUE,
    out_dir_plots       = out_dir,
    verbose             = FALSE
  )
}
```

`estimate_hr_ir` *Estimate Incidence Rates and Hazard Ratios*

Description

This function estimates incidence rates and hazard ratios for time-to-event outcomes. It can perform unweighted analysis, weighted (propensity score adjusted) analysis, or both, using potentially different datasets for each analysis type. Incidence rate CIs are estimated using Poisson distribution.

Usage

```
estimate_hr_ir(
  in_df_unwt = NULL,
  in_df_wted = NULL,
  out_xlsxpath = NULL,
  exposure_var = "exp",
  exp_value = 1,
  ref_value = 0,
  outcome_var = NULL,
  weight_var = NULL,
  survival_time = NULL,
  time_unit = c("days", "years"),
  ir_per_pyears = 1000,
  confidence_level = 0.95,
```

```
  readme_text = NULL,
  stratify_by = NULL,
  verbose = TRUE
)
```

Arguments

in_df_unwt	Data frame for unweighted analysis. Set to NULL to skip unweighted analysis. This could be the full cohort, matched data, or any specific population
in_df_wted	Data frame for weighted analysis. Set to NULL to skip weighted analysis. This could be the same as unweighted data or a different population (e.g., trimmed)
out_xlsxpath	Character string. Path for output Excel file with all results
exposure_var	Character string. Name of binary exposure/treatment variable (default: "exp")
exp_value	Value representing the exposed/treated group (default: 1)
ref_value	Value representing the reference/control group (default: 0)
outcome_var	Character string. Name of the event indicator variable (0=censored, 1=event)
weight_var	Character string. Name of weight variable in the weighted dataset. Required when in_df_wted is not NULL.
survival_time	Character string. Name of survival/follow-up time variable (required)
time_unit	Character string. Unit of time for survival analysis: "days" or "years". Default: "days"
ir_per_pyears	Numeric. Multiplier for incidence rate per person-years. Allowed values: 1, 100, 1000, 10000, 100000. Default: 1000 (i.e., IR per 1,000 person-years). Column name is fixed as IR_per_Npy_Unwt / IR_per_Npy_Wted. The actual multiplier value is documented in the Analysis Summary sheet.
confidence_level	Numeric. Confidence level for confidence intervals (default: 0.95 for 95% CI)
readme_text	Character string. Optional message to include in README sheet at beginning of Excel file. Use this to document the analysis parameters, data sources, or any notes
stratify_by	Character string. Name of the stratification variable. When specified: - Cox model uses strata() in the formula (separate baseline hazards per stratum) - IR/IRD are computed via direct standardization using total cohort person-time distribution as weights Default: NULL (no stratification)
verbose	Logical. Print progress messages (default TRUE).

Details

The function allows flexible analysis configurations:

- Unweighted only: Set in_df_unwt = data, in_df_wted = NULL
- Weighted only: Set in_df_unwt = NULL, in_df_wted = data
- Both: Provide both datasets (can be the same or different populations)

For unweighted analysis:

- Uses standard Cox regression without weights
- Can be applied to matched data, trimmed data, or full cohort

For weighted analysis:

- Uses weighted Cox regression with specified weights
- Typically applied to PS-weighted data (IPTW, MW, OW, FS weights)

Stratification (stratify_by parameter): When stratify_by is specified:

- Cox model: Fits a stratified Cox proportional hazards model using strata() in the model formula. This allows separate baseline hazard functions for each stratum while estimating a common treatment effect (HR) across strata.
- IR/IRD: Uses direct standardization with total cohort person-time distribution as standard weights.
  - IR_std = sum(Rate_k * W_k) where W_k = PT_total_k / PT_total
  - Variance via delta method: Var = sum(W_k^2 * d_k / PT_k^2)
  - IRD_std = IR_std_exposed - IR_std_unexposed
  - CI for standardized IR uses log-transformation to ensure positive bounds

ir_per_pyears parameter: Controls the person-year multiplier for incidence rates and incidence rate differences. For example:

- ir_per_pyears = 1000: rates expressed per 1,000 person-years (default)
- ir_per_pyears = 100000: rates expressed per 100,000 person-years Output column names are fixed (e.g., IR_per_Npy_Unwt, IR_per_Npy_Wted). The multiplier value is recorded in the Analysis Summary sheet.

Value

Invisibly returns a list containing:

- incidence_rates: Data frame with event counts, person-years, and incidence rates
- hazard_ratios: Data frame with unweighted and/or weighted HR estimates
- unweighted_model: The unweighted Cox model object (if unweighted analysis performed)
- weighted_model: The weighted Cox model object (if weighted analysis performed)
- stratum_details_unwt: Data frame with stratum-level details (if stratified, unweighted)
- stratum_details_wted: Data frame with stratum-level details (if stratified, weighted) Also saves results to Excel file when out_xlsxpath is provided

Side Effects

Creates the output directory and writes an Excel workbook when `out_xlsxpath` is provided.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df <- read.csv(csv_path)

# Unweighted analysis only
result <- estimate_hr_ir(
  in_df_unwt    = df,
  in_df_wted    = NULL,
  exposure_var  = "exposure",
  outcome_var   = "outcome",
  survival_time = "follow_up_days",
  time_unit     = "days",
  ir_per_pyears = 1000,
  verbose       = FALSE
)
result$incidence_rates


# Weighted analysis with Excel output (requires openxlsx)
if (requireNamespace("openxlsx", quietly = TRUE)) {
  df_ps <- estimate_ps(
    in_df        = df,
    exposure_var = "exposure",
    class_vars   = c("cat1", "cat2", "cat3", "cat4"),
    cont_vars    = c("cont1", "cont2", "cont3"),
    verbose      = FALSE
  )
  df_wt <- create_ps_weights(
    in_df         = df_ps,
    exposure_var  = "exposure",
    ps_var        = "ps",
    weight_method = "mw",
    weight_var    = "mw_wt",
    verbose       = FALSE
  )
  out_xlsx <- tempfile(fileext = ".xlsx")
  result2 <- estimate_hr_ir(
    in_df_unwt    = df,
    in_df_wted    = df_wt,
    out_xlsxpath  = out_xlsx,
    exposure_var  = "exposure",
    outcome_var   = "outcome",
    survival_time = "follow_up_days",
    weight_var    = "mw_wt",
    ir_per_pyears = 1000,
    verbose       = FALSE
  )
}
```

`estimate_ps` *Calculate propensity scores and add them to the dataset*

Description

This function calculates propensity scores using logistic regression and adds them as a new column to the dataset. It can output both an R data frame and/or CSV file. Additionally, it can save odds ratio table from the PS model to Excel or data frame.

Usage

```
estimate_ps(
  in_df = NULL,
  in_csvpath = NULL,
  out_csvpath = NULL,
  out_xlsxpath_odds_ratio = NULL,
  exposure_var = "exposure",
  exp_value = 1,
  ref_value = 0,
  class_vars = NULL,
  cont_vars = NULL,
  ps_var = "ps",
  interactions = NULL,
  exclude_vars_w_extreme_distribution = FALSE,
  verbose = TRUE
)
```

Arguments

`in_df`	Data frame containing the input data (optional if in_csvpath provided)
`in_csvpath`	Character string. Path to input CSV file (optional if in_df provided)
`out_csvpath`	Character string. Path for output CSV file (optional, if NULL no CSV is saved)
`out_xlsxpath_odds_ratio`	Character string. Path for output Excel file with OR table (optional)
`exposure_var`	Character string. Name of the binary exposure/treatment variable column (default: "exposure")
`exp_value`	Value representing the exposed/treated group (default: 1)
`ref_value`	Value representing the reference/control group (default: 0)
`class_vars`	Character vector. Names of categorical/factor variables to include in PS model
`cont_vars`	Character vector. Names of continuous variables to include in PS model
`ps_var`	Character string. Name for the calculated PS variable column (default: "ps")
`interactions`	Character string. Interaction terms to include (e.g., "var1:var2 + var1:var3")

exclude_vars_w_extreme_distribution
: Logical. If TRUE, automatically excludes variables with extreme distribution (categorical: only 1 unique level in either group, or any level with count < 5 in either group; continuous: constant/single unique value (SD < 1e-6) in either group, or SMD > 1.5 between groups) instead of throwing error (default: FALSE). The categorical screen unions levels across the two exposure groups, so a level present in one group but absent in the other is counted as n = 0 and flagged by the cnt < 5 rule.

verbose
: Logical. Print progress messages (default TRUE).

Value

A data frame with the PS column added. Also saves to CSV if path specified.

Side Effects

- Writes a CSV file when `out_csvpath` is provided.
- Writes an Excel file with odds-ratio table when `out_xlsxpath_odds_ratio` is provided.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df <- read.csv(csv_path)

# Basic usage: estimate PS and add it as a new column
result <- estimate_ps(
  in_df        = df,
  exposure_var = "exposure",
  class_vars   = c("cat1", "cat2", "cat3", "cat4"),
  cont_vars    = c("cont1", "cont2", "cont3"),
  verbose      = FALSE
)
head(result$ps)


# With CSV output and Excel OR table (requires openxlsx)
if (requireNamespace("openxlsx", quietly = TRUE)) {
  out_csv  <- tempfile(fileext = ".csv")
  out_xlsx <- tempfile(fileext = ".xlsx")
  result2 <- estimate_ps(
    in_df                              = df,
    out_csvpath                        = out_csv,
    out_xlsxpath_odds_ratio            = out_xlsx,
    exposure_var                       = "exposure",
    class_vars                         = c("cat1", "cat2"),
    cont_vars                          = c("cont1", "cont2"),
    exclude_vars_w_extreme_distribution = TRUE,
    verbose                            = FALSE
  )
}
```

`estimate_rr_rd` *Estimate Risk Ratios and Risk Differences using Cumulative Incidence*

Description

This function estimates risk ratios (RR) and risk differences (RD) for time-to-event outcomes using cumulative incidence at a specified timepoint. Two estimators are supported:

Kaplan-Meier (KM) Standard 1 - S(t) cumulative incidence. Appropriate when there are no competing risks or competing events are treated as censored.

Aalen-Johansen (AJ) Cumulative incidence function accounting for competing risks. Produces unbiased risk estimates in the presence of competing events by treating them as a separate absorbing state rather than censoring. Requires a competing event indicator via `competing_event_var`. Uses multi-state `survfit` internally (`survival` >= 3.1).

It can perform unweighted analysis, weighted (propensity score adjusted) analysis, or both, using potentially different datasets for each analysis type. Confidence intervals can be estimated using bootstrapping, analytical (delta method / Greenwood), or both.

Usage

```
estimate_rr_rd(
  in_df_unwt = NULL,
  in_df_wted = NULL,
  out_xlsxpath = NULL,
  exposure_var = "exp",
  exp_value = 1,
  ref_value = 0,
  outcome_var = NULL,
  weight_var = NULL,
  survival_time = NULL,
  time_unit = c("days", "months", "years"),
  rr_rd_at_timepoint = 365,
  rr_rd_per_individuals = 1000,
  confidence_level = 0.95,
  conf_int_method = c("bootstrap", "analytical", "both"),
  risk_estimator = c("KM", "AJ"),
  competing_event_var = NULL,
  bootstrap_count = 500,
  stratify_by = NULL,
  n_cores = NULL,
  seed = NULL,
  readme_text = NULL,
  verbose = TRUE
)
```

Arguments

in_df_unwt	Data frame for unweighted analysis. Set to NULL to skip unweighted analysis. This could be the full cohort, matched data, or any specific population.
in_df_wted	Data frame for weighted analysis. Set to NULL to skip weighted analysis. This could be the same as unweighted data or a different population (e.g., trimmed).
out_xlsxpath	Character string. Path for output Excel file with all results.
exposure_var	Character string. Name of binary exposure/treatment variable (default: "exp").
exp_value	Value representing the exposed/treated group (default: 1).
ref_value	Value representing the reference/control group (default: 0).
outcome_var	Character string. Name of the event indicator variable (0=censored, 1=event).
weight_var	Character string. Name of weight variable in the weighted dataset. Required when in_df_wted is not NULL.
survival_time	Character string. Name of survival/follow-up time variable (required).
time_unit	Character string. Unit of time for survival analysis: "days", "months", or "years". Default: "days".
rr_rd_at_timepoint	Numeric. Timepoint at which to calculate cumulative incidence. Units depend on time_unit parameter. Default: 365.
rr_rd_per_individuals	Numeric. Denominator for expressing risk and risk difference. Default: 1000.
confidence_level	Numeric. Confidence level for confidence intervals (default: 0.95).
conf_int_method	Character string. Method for confidence interval estimation: **"bootstrap"** Percentile bootstrap CIs only. **"analytical"** Analytical CIs only (delta method for RR, Greenwood-based for RD). bootstrap_count is ignored. **"both"** Compute both bootstrap and analytical CIs. Default: "bootstrap".
risk_estimator	Character string. Cumulative incidence estimator: **"KM"** Kaplan-Meier (1 - survival). Aliases: "Kaplan-Meier", "kaplan-meier". **"AJ"** Aalen-Johansen cumulative incidence function for competing risks. Requires competing_event_var. Aliases: "Aalen-Johansen", "aalen-johansen". Default: "KM".
competing_event_var	Character string. Name of the competing event indicator variable (1 = competing event occurred, e.g., death; 0 = no competing event). Required when risk_estimator = "AJ". Ignored when risk_estimator = "KM". The multistate status is constructed internally: outcome_var == 1 -> event of interest (1), outcome_var == 0 & competing_event_var == 1 -> competing event (2), outcome_var == 0 & competing_event_var == 0 -> censored (0).

`bootstrap_count`
: Integer. Number of bootstrap iterations for CI estimation (default: 500). Ignored when `conf_int_method = "analytical"`.

`stratify_by`
: Character string. Name of stratification variable for direct standardization (default: NULL). When specified, the function: (1) calculates stratum weights (w_k = N_k / N_total) from the total population, (2) computes risk within each stratum using the selected estimator, (3) standardizes: Risk_std = Sum(Risk_k * w_k), (4) derives RR and RD from standardized risks. Bootstrap CIs use stratified resampling (within each stratum). Analytical SEs use Greenwood formula: Var(Risk_std) = Sum(w_k^2 * Var(R_k)).

`n_cores`
: Integer. Number of CPU cores for parallel bootstrap. Default uses all available cores minus 1. Set to 1 to disable parallelization. Ignored when `conf_int_method = "analytical"`.

`seed`
: Integer or NULL. Optional seed for the bootstrap random number generator. When NULL (default), the caller's current RNG state is used and not modified. When an integer is supplied, it is passed to `clusterSetRNGStream` (parallel) or `set.seed` (sequential) to make the bootstrap CIs reproducible. In the sequential case the caller's `.Random.seed` is saved and restored on exit. Ignored when `conf_int_method = "analytical"`.

`readme_text`
: Character string. Optional message to include in README sheet of Excel output.

`verbose`
: Logical. Print progress messages (default TRUE).

Details

When `stratify_by` is specified, the function performs direct standardization: stratum-specific risks are combined using the total population distribution as the standard. Bootstrap resampling is stratified to preserve stratum structure.

The function proceeds as follows:

1. Fits cumulative incidence curves by exposure group using the selected estimator
2. Extracts cumulative incidence (risk) at the specified timepoint
3. Computes Risk Ratio (RR) = Risk_exposed / Risk_reference
4. Computes Risk Difference (RD) = Risk_exposed - Risk_reference
5. Estimates confidence intervals using the selected method

Kaplan-Meier (KM) estimator: Risk = 1 - S(t), where S(t) is the KM survival estimate. Competing events, if present, are treated as censored, which can overestimate cumulative incidence.

Aalen-Johansen (AJ) estimator: Cumulative incidence function from a multi-state model via `survfit` with `Surv(time, factor(status))`. Competing events are modelled as a separate absorbing state, producing unbiased risk estimates. Variance is based on the counting-process variance estimator (Aalen, 1978).

Analytical CI method:

- RR: delta method on log scale. Var(log(RR)) = Var(R_exp)/R_exp^2 + Var(R_ref)/R_ref^2. CI = exp(log(RR) +/- z * SE(log(RR))).

- RD: Var(RD) = (Var(R_exp) + Var(R_ref)) * per_n^2. CI = RD +/- z * SE(RD).
- Variance from survfit (Greenwood for KM, counting-process for AJ).

Bootstrap CI method: Percentile bootstrap with parallel computation via parLapply using L'Ecuyer-CMRG random number streams. Pass an integer to seed for reproducible bootstrap CIs; by default (seed = NULL) the caller's current RNG state is used and not modified.

When stratify_by is specified (Direct Standardization):

1. Stratum weights w_k = N_k / N_total from the total population
2. Within each stratum k, risk R_{ik} is computed for each exposure group i
3. Standardized risk: Risk_std_i = Sum_k(R_{ik} * w_k)
4. RR = Risk_std_1 / Risk_std_0; RD = Risk_std_1 - Risk_std_0
5. Analytical SE: Var(Risk_std_i) = Sum_k(w_k^2 * Var(R_{ik}))
6. Bootstrap CIs use stratified resampling with fixed original population weights

Value

Invisibly returns a list containing:

estimates Data frame with RR, RD, confidence intervals, and Risk_Estimator ("KM" or "AJ").

cumulative_incidence Data frame with cumulative incidence by group. Columns: Risk (probability), Risk_LCI / Risk_UCI (log-transformed CI), Risk_SE (SE on 0-1 scale), Risk_Var (variance), RiskperN (scaled risk), RiskperN_LCI / RiskperN_UCI (scaled CI), and Risk_Estimator.

stratum_details (if stratify_by specified) Data frame with stratum-level results and Risk_Estimator.

unweighted_bootstrap (if applicable) Bootstrap results matrix.

weighted_bootstrap (if applicable) Bootstrap results matrix.

Also saves results to Excel file when out_xlsxpath is provided.

Side Effects

Creates the output directory and writes an Excel workbook when out_xlsxpath is provided.

Examples

```
csv_path <- system.file("extdata", "sample_data.csv", package = "rwetools")
df <- read.csv(csv_path)

# KM with analytical CIs (fast, no bootstrap)
result <- estimate_rr_rd(
  in_df_unwt            = df,
  in_df_wted            = NULL,
  exposure_var          = "exposure",
  outcome_var           = "outcome",
  survival_time         = "follow_up_days",
  time_unit             = "days",
  rr_rd_at_timepoint    = 365,
  rr_rd_per_individuals = 1000,
```

```
  conf_int_method       = "analytical",
  verbose               = FALSE
)
result$estimates


# KM with bootstrap CIs (slow) and competing risk (AJ estimator)
result2 <- estimate_rr_rd(
  in_df_unwt            = df,
  in_df_wted            = NULL,
  exposure_var          = "exposure",
  outcome_var           = "outcome",
  survival_time         = "follow_up_days",
  time_unit             = "days",
  rr_rd_at_timepoint    = 365,
  rr_rd_per_individuals = 1000,
  risk_estimator        = "AJ",
  competing_event_var   = "competing_event",
  conf_int_method       = "bootstrap",
  bootstrap_count       = 100,
  n_cores               = 1,
  verbose               = FALSE
)
```

Index