Package 'SurveyNCD'

September 28, 2026

Title Survey-Weighted Analysis of Self-Reported Health Indicators

Version 0.1.0

Description Analyses population health survey data from the World Health Organization (WHO) Stepwise Approach to Non-Communicable Disease (NCD) Risk Factor Surveillance (STEPS), Demographic and Health Surveys (DHS), Multiple Indicator Cluster Surveys (MICS), and similar complex sample surveys, where chronic conditions are self-reported rather than coded using the International Classification of Diseases (ICD) and estimates must account for stratification, clustering, and sampling weights. Includes a self-reported multimorbidity index based on the Functional Comorbidity Index (FCI) described by Groll et al. (2005) <doi:10.1016/j.jclinepi.2004.10.018>, design-weighted population prevalence estimation via the 'survey' package, a survey-weighted concentration index for health inequality analysis, a DHS anthropometric z-score categoriser, a choropleth mapping helper, and exploratory survey-weighted gradient boosting (via 'xgboost') with SHapley Additive exPlanations (SHAP) based explainability. The gradient boosting component applies case weights but does not yet propagate cluster and strata design effects into variance estimates; it should be treated as exploratory rather than as design-based inference.

License MIT + file LICENSE

Encoding UTF-8

VignetteBuilder knitr

Imports dplyr, ggplot2, magrittr, rlang, stats, survey, tidyr

Suggests knitr, rmarkdown, sf, srvyr, testthat (>= 3.0.0), xgboost

Config/testthat/edition 3

URL https://github.com/StatAid-Research-Lab/SurveyNCD

BugReports https://github.com/StatAid-Research-Lab/SurveyNCD/issues

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Sujon Mia [aut, cre],
Md. Atiqul Islam [ctb] (Professor & Academic Mentor, Department of Statistics)

Maintainer Sujon Mia <sujonsgc@gmail.com>

Repository CRAN

Date/Publication 2026-09-28 08:30:24 UTC

Contents

`fci_items` *The 18 conditions of the Functional Comorbidity Index*

Description

Returns the standard condition list for the Functional Comorbidity Index.

Usage

```
fci_items()
```

Value

A character vector of the 18 FCI condition descriptions.

References

Groll, D. L., To, T., Bombardier, C., & Wright, J. G. (2005). The development of a comorbidity index with physical function as the outcome. *Journal of Clinical Epidemiology*, 58(6), 595-602. doi:10.1016/j.jclinepi.2004.10.018

Examples

```
fci_items()
length(fci_items())  # 18
```

`mm_prevalence` *Population-level multimorbidity estimates from complex survey data*

Description

Wraps the `survey` package to compute design-weighted multimorbidity prevalence and mean condition count, properly accounting for the stratification, clustering, and sampling weights of a complex survey design (WHO STEPS, DHS, MICS, etc.). This is the step that turns an individual-level score into a defensible population estimate – run `multimorbidity_index()` first to create `mm_category` and `mm_n_conditions`.

Usage

```
mm_prevalence(data, ids = NULL, strata = NULL, weights, by = NULL, nest = TRUE)
```

Arguments

`data`	A data frame already processed by `multimorbidity_index()` (must contain `mm_category` and `mm_n_conditions`), plus the survey design columns named below.
`ids`	Name of the cluster/PSU column, or `NULL` if the design has no clustering (e.g. simple random sample).
`strata`	Name of the stratification column, or `NULL` if the design is unstratified.
`weights`	Name of the sampling weight column.
`by`	Optional single column name to compute subgroup estimates by (e.g. `"sex"`, `"region"`). `NULL` (default) returns one overall row. Multi-variable grouping isn't supported yet – a natural v2 addition.
`nest`	Passed to `survey::svydesign()`. `TRUE` (default) assumes cluster IDs repeat across strata (true for most DHS/STEPS-style designs, where e.g. cluster "1" exists in every region).

Value

A data frame with one row (or one row per `by` group), containing `n` (unweighted sample size), `prevalence` / `prevalence_se` / `prevalence_lower` / `prevalence_upper` (design- weighted proportion "Multimorbid" with a logit-CI), and `mean_conditions` / `mean_conditions_se`.

Examples

```
set.seed(1)
n <- 40
df <- data.frame(
  psu    = rep(1:8, each = 5),
  region = rep(c("A", "B"), each = 20),
  wt     = round(runif(n, 0.8, 1.4), 2),
  htn    = rbinom(n, 1, 0.3),
```

```
  dm     = rbinom(n, 1, 0.2)
)
scored <- multimorbidity_index(df, conditions = c("htn", "dm"))
mm_prevalence(scored, ids = "psu", strata = "region", weights = "wt")
mm_prevalence(scored, ids = "psu", strata = "region", weights = "wt",
              by = "region")
```

`multimorbidity_index` *Calculate a self-reported multimorbidity index*

Description

Computes an individual-level multimorbidity score from a set of self-reported binary condition indicators (e.g. "Has a doctor ever told you that you have hypertension?"). This is designed for population health survey data (WHO STEPS, DHS, SAGE, etc.), where multimorbidity is captured through self-report rather than ICD-coded diagnoses – a data shape that existing comorbidity packages (which all assume ICD claims data) don't handle.

Usage

```
multimorbidity_index(
  data,
  conditions,
  weights = NULL,
  na_action = c("ignore", "na")
)
```

Arguments

`data`	A data frame, one row per individual.
`conditions`	A character vector of column names in `data`. Each column must already be coded 0/1/NA (1 = condition present). Use `recode_binary()` first if your raw data uses Yes/No, 1/2, etc.
`weights`	Optional named numeric vector giving a weight for each condition (names must match `conditions`). If `NULL` (the default), every condition counts equally – i.e. a simple unweighted sum, which is how the published Functional Comorbidity Index (Groll et al. 2005) is scored. See `fci_items()`.
`na_action`	How to handle a row with at least one missing condition. `"ignore"` (default) sums whatever conditions are non-missing for that person. `"na"` returns NA for that person's index entirely, which is the safer choice once you move to population-level (survey-weighted) estimates, since silently treating missing as absent can bias prevalence downward.

Value

`data` with three columns appended: `mm_n_conditions` (raw count of conditions present), `mm_index` (the possibly weighted score), and `mm_category` (factor: "None", "Single condition", "Multimorbid").

References

Groll, D. L., To, T., Bombardier, C., & Wright, J. G. (2005). The development of a comorbidity index with physical function as the outcome. *Journal of Clinical Epidemiology*, 58(6), 595-602. doi:10.1016/j.jclinepi.2004.10.018

Examples

```
df <- data.frame(
  id           = 1:5,
  hypertension = c(1, 0, 1, 1, 0),
  diabetes     = c(0, 0, 1, 1, 0),
  arthritis    = c(1, 0, 0, 1, NA)
)
multimorbidity_index(df, conditions = c("hypertension", "diabetes", "arthritis"))
```

---

plot_shap_summary *Plot SHapley Additive exPlanations (SHAP) Summary*

Description

Generates a SHAP summary plot (similar to the Python 'shap' package) where features are ranked on the y-axis by their overall importance (mean absolute SHAP value), SHAP values are shown on the x-axis, and each point (representing a respondent) is colored by its relative value for that feature (blue for low, red for high).

Usage

```
plot_shap_summary(
  shap_matrix,
  data,
  low_color = "#1e88e5",
  high_color = "#ff0052",
  title = "SHAP Summary Plot",
  subtitle = "Feature impact on model predictions (ranked by mean absolute SHAP)"
)
```

Arguments

`shap_matrix`	The matrix output from `survey_shap()`.
`data`	The matrix or data frame of training features (e.g. the `X` matrix returned by `survey_xgboost()`) or the original data frame. If the row count does not match the SHAP matrix, rows with missing values will be automatically filtered out.
`low_color`	A character string for the color representing low feature values. Default is `"#1e88e5"` (standard SHAP blue).
`high_color`	A character string for the color representing high feature values. Default is `"#ff0052"` (standard SHAP red).

title	A character string for the plot title. Default is "SHAP Summary Plot".
subtitle	A character string for the plot subtitle. Default is "Feature impact on model predictions (ranked by mean absolute SHAP)".

Value

A ggplot2 object.

Examples

```
set.seed(1)
n <- 30
df <- data.frame(
  outcome = rnorm(n, 100, 10),
  age     = round(runif(n, 18, 80)),
  bmi     = round(rnorm(n, 24, 4), 1),
  wt      = round(runif(n, 0.5, 2), 2)
)
design <- survey::svydesign(ids = ~1, weights = ~wt, data = df)
model  <- survey_xgboost(design, outcome ~ age + bmi, nrounds = 5)
shap   <- survey_shap(model)
plot_shap_summary(shap, model$X)
```

---

recode_binary *Recode a messy survey column into clean 0/1*

Description

Converts alternative survey response values into clean standard numeric 0 or 1 scores. When no is not specified, every value that is neither in yes nor NA is treated as 0. When no *is* specified, values matching neither yes nor no are left as NA and a warning is issued.

Usage

```
recode_binary(x, yes, no = NULL)
```

Arguments

x	A vector coding a single condition.
yes	The raw value(s) that should become 1.
no	Optional raw value(s) that should become 0. If NULL (the default), every non-yes, non-NA value becomes 0.

Value

A numeric vector of 0, 1, or NA.

Examples

```
# STEPS-style coding: 1 = yes, 2 = no
recode_binary(c(1, 2, 1, NA), yes = 1, no = 2)

# When `no` is omitted, everything that isn't `yes` becomes 0
recode_binary(c(1, 2, 9, NA), yes = 1)

# With `no` specified, unrecognised codes (9) become NA with a warning
recode_binary(c(1, 2, 9, NA), yes = 1, no = 2)
```

`survey_concentration_index`

Calculate Survey-Weighted Concentration Index

Description

Computes the concentration index (the Wagstaff/O'Donnell "convenient covariance" formula) for a health outcome across a socioeconomic ranking variable, using proper survey sampling weights, and computes design-consistent standard errors, confidence intervals, and p-values.

Usage

```
survey_concentration_index(design, outcome, wealth, conf.level = 0.95)
```

Arguments

`design`	A survey design object created by `survey::svydesign()` or `srvyr::as_survey_design()` – anything inheriting from `"survey.design"` (srvyr's `tbl_svy` objects do).
`outcome`	Unquoted name of the health indicator (e.g. stunting_clean).
`wealth`	Unquoted name of the wealth/ranking variable (e.g. wealth_index).
`conf.level`	Confidence level for the confidence interval. Default is 0.95.

Details

$$CI = \frac{2}{\mu} \times Cov_w(y_i, R_i)$$

where y_i is the outcome for individual i, μ is the weighted mean outcome, and R_i is each individual's weighted fractional rank in the wealth distribution.

The standard error, confidence intervals, and p-value are calculated using the "convenient regression" method (Kakwani, Wagstaff, and van Doorslaer 1997) run via `survey::svyglm()`:

$$2\sigma_R^2(y_i/\mu) = \alpha + \beta R_i + \epsilon_i$$

where σ_R^2 is the weighted variance of the fractional rank. The coefficient β is mathematically identical to the concentration index, and its standard error from the regression model is a design-consistent standard error that fully accounts for stratification and clustering.

Value

A tibble with `Concentration_Index`, `Standard_Error`, `Lower_CI`, `Upper_CI`, `p_value`, `Outcome_Mean`, and `n`.

References

Kakwani, N., Wagstaff, A., & van Doorslaer, E. (1997). Socioeconomic inequalities in health: Measurement, computation, and statistical inference. *Journal of Econometrics*, 77(1), 87-103. doi:10.1016/S03044076(96)018076

Examples

```
set.seed(42)
n <- 50
df <- data.frame(
  wealth  = rnorm(n, 50, 15),
  outcome = pmax(0, 0.3 * rnorm(n, 50, 15) + rnorm(n, 0, 5)),
  wt      = sample(1:5, n, replace = TRUE)
)
design <- survey::svydesign(ids = ~1, weights = ~wt, data = df)
survey_concentration_index(design, outcome = outcome, wealth = wealth)
```

`survey_map_indicator` *Map Survey Indicators Universally*

Description

This function merges calculated survey indicators with any provided spatial shapefile to generate a publication-ready thematic map.

Usage

```
survey_map_indicator(
  survey_data,
  shapefile,
  join_by,
  fill_var,
  palette = c("magma", "viridis", "plasma", "inferno", "cividis"),
  legend_title = NULL,
  border_color = "white",
  border_width = 0.2
)
```

Arguments

survey_data	A data frame containing the aggregated survey indicators.
shapefile	An sf spatial object containing regional boundaries.
join_by	A character string of the column name present in both datasets to merge on.
fill_var	A character string of the variable in survey_data to map (e.g., "Concentration_Index").
palette	A character string specifying the viridis color palette option to use: "magma" (default), "viridis", "plasma", "inferno", or "cividis".
legend_title	A character string for the legend title. If NULL (default), the fill_var name is used.
border_color	A character string for the region border line color. Default is "white".
border_width	A numeric value for the border line width. Default is 0.2.

Value

A ggplot2 spatial map object.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
  regional_map <- sf::st_read(
    system.file("shape/nc.shp", package = "sf"), quiet = TRUE
  )
  metrics <- data.frame(
    NAME = c("Ashe", "Alleghany", "Surry"),
    prevalence = c(0.12, 0.08, 0.15)
  )
  survey_map_indicator(metrics, regional_map, "NAME", "prevalence")
}
```

---

survey_shap *Extract SHAP Values from a Survey-Weighted XGBoost Model*

Description

This function cracks open a trained survey-weighted XGBoost model and calculates the SHapley Additive exPlanations (SHAP values) for every respondent.

Usage

```
survey_shap(sxgb_model)
```

Arguments

sxgb_model	The list output from the survey_xgboost() function.

Value

A matrix of SHAP values detailing the marginal contribution of each feature to the final prediction for every observation.

Examples

```
set.seed(1)
n <- 30
df <- data.frame(
  outcome = rnorm(n, 100, 10),
  age     = round(runif(n, 18, 80)),
  bmi     = round(rnorm(n, 24, 4), 1),
  wt      = round(runif(n, 0.5, 2), 2)
)
design <- survey::svydesign(ids = ~1, weights = ~wt, data = df)
model  <- survey_xgboost(design, outcome ~ age + bmi, nrounds = 5)
shap   <- survey_shap(model)
head(shap)
```

---

survey_xgboost *Train an XGBoost Model with Complex Survey Weights*

Description

This function trains an Extreme Gradient Boosting (XGBoost) model while strictly enforcing sampling weights from a complex survey design object.

Usage

```
survey_xgboost(design, formula, params = list(), nrounds = 100)
```

Arguments

design	A survey.design object containing the data and weights.
formula	A formula specifying the response and predictor variables.
params	A list of XGBoost parameters (e.g., objective, eta, max_depth).
nrounds	The number of boosting iterations.

Value

A list containing the trained xgb.Booster model, the xgb.DMatrix, the feature names, the observation count, and the cleaned training feature matrix X.

Examples

```
set.seed(1)
n <- 30
df <- data.frame(
  outcome = rnorm(n, 100, 10),
  age     = round(runif(n, 18, 80)),
  bmi     = round(rnorm(n, 24, 4), 1),
  wt      = round(runif(n, 0.5, 2), 2)
)
design <- survey::svydesign(ids = ~1, weights = ~wt, data = df)
model  <- survey_xgboost(design, outcome ~ age + bmi, nrounds = 5)
model$features
```

---

who_anthro_score *Calculate WHO Anthropometric Categories from DHS/MICS Data*

Description

Cleans raw DHS/MICS z-score variables (e.g., hw70, hw71, hw72), adjusts the decimal scaling if needed, handles DHS-specific missing flags, applies WHO biologically implausible flagging rules, and categorizes them into WHO severity tiers.

Usage

```
who_anthro_score(
  x,
  indicator = c("stunting", "wasting", "underweight"),
  scaled_by_100 = TRUE,
  remove_implausible = TRUE
)
```

Arguments

x	A numeric vector of raw DHS/MICS z-scores.
indicator	The type of indicator for labeling and flagging: "stunting" (height-for-age, HAZ), "wasting" (weight-for-height, WHZ), or "underweight" (weight-for-age, WAZ).
scaled_by_100	Logical. If TRUE (default), divides input values by 100 (standard for DHS raw recode files like hw70). If FALSE, assumes values are already on the standard z-score scale (like MICS or cleaned DHS data).
remove_implausible	Logical. If TRUE (default), applies WHO child growth standard flagging rules to set biologically implausible values to NA. WHO flags are: • Stunting (HAZ): < -6.0 or > 6.0 • Wasting (WHZ): < -5.0 or > 5.0 • Underweight (WAZ): < -6.0 or > 5.0

Value

A factor vector of WHO categories (Severe, Moderate, Normal) with the indicator label appended.

References

World Health Organization (2006). *WHO Child Growth Standards: Length/height-for-age, weight-for-age, weight-for-length, weight-for-height and body mass index-for-age: Methods and development*. Geneva: WHO.

Examples

```
# DHS-style raw z-scores (scaled by 100)
raw <- c(-310, -254, 50, 9999)
who_anthro_score(raw, indicator = "stunting")

# Already on standard z-score scale
who_anthro_score(c(-3.1, -2.5, 0.5), indicator = "wasting",
                 scaled_by_100 = FALSE)
```

Index