Package: nrba (via r-universe)

October 29, 2024

Title Methods for Conducting Nonresponse Bias Analysis (NRBA)

Version 0.3.1

Description Facilitates nonresponse bias analysis (NRBA) for survey data. Such data may arise from a complex sampling design with features such as stratification, clustering, or unequal probabilities of selection. Multiple types of analyses may be conducted: comparisons of response rates across subgroups; comparisons of estimates before and after weighting adjustments; comparisons of sample-based estimates to external population totals; tests of systematic differences in covariate means between respondents and full samples; tests of independence between response status and covariates; and modeling of outcomes and response status as a function of covariates. Extensive documentation and references are provided for each type of analysis. Krenzke, Van de Kerckhove, and Mohadjer (2005) <http://www.asasrms.org/Proceedings/y2005/files/JSM2005-000572.pdf> and Lohr and Riddles (2016) <https://www150.statcan.gc.ca/n1/en/pub/12-001-x/2016002/article/14677-eng.pdf?st=q7PyNsGR> provide an overview of the methods implemented in this package.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports broom, dplyr, magrittr, rlang, srvyr, stats, survey (>= 4.1-1), svrep, tidyr

Suggests knitr, rmarkdown, stringr, testthat (>= 3.0.0), tibble

Config/testthat/edition 3

Depends R (>= 4.1.0)

VignetteBuilder knitr

NeedsCompilation no

Author Ben Schneider [aut, cre]
(<https://orcid.org/0000-0002-0406-8470>), Jim Green [aut],
Shelley Brock [aut] (Author of original SAS macro, WesNRBA),
Tom Krenzke [aut] (Author of original SAS macro, WesNRBA),
Michael Jones [aut] (Author of original SAS macro, WesNRBA),
Wendy Van de Kerckhove [aut] (Author of original SAS macro,
WesNRBA), David Ferraro [aut] (Author of original SAS macro,
WesNRBA), Laura Alvarez-Rojas [aut] (Author of original SAS
macro, WesNRBA), Katie Hubbell [aut] (Author of original SAS
macro, WesNRBA), Westat [cph]

Maintainer Ben Schneider <BenjaminSchneider@westat.com>

Date/Publication 2023-11-21 05:10:02 UTC

Repository https://westat-schneider.r-universe.dev

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

RemoteRef HEAD

RemoteSha 60c8caa611d7fc9c3269446c5cecdb0c3b585742

Contents

`assess_range_of_bias` *Assess the range of possible bias based on specified assumptions about how nonrespondents differ from respondents*

Description

This range-of-bias analysis assesses the range of possible nonresponse bias under varying assumptions about how nonrespondents differ from respondents. The range of potential bias is calculated for both unadjusted estimates (i.e., from using base weights) and nonresponse-adjusted estimates (i.e., based on nonresponse-adjusted weights).

Usage

```
assess_range_of_bias(
  survey_design,
  y_var,
  comparison_cell,
  status,
  status_codes,
  assumed_multiple = c(0.5, 0.75, 0.9, 1.1, 1.25, 1.5),
  assumed_percentile = NULL
)
```

Arguments

`survey_design` A survey design object created with the 'survey' package

`y_var` Name of a variable whose mean or proportion is to be estimated

`comparison_cell` (Optional) The name of a variable in the data dividing the sample into cells. If supplied, then the analysis is based on assumptions about differences between respondents and nonrespondents within the same cell. Typically, the variable used is a nonresponse adjustment cell or post-stratification variable.

`status` A character string giving the name of the variable representing response/eligibility status. The status variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).

`status_codes` A named vector, with four entries named 'ER', 'EN', 'IE', and 'UE'. `status_codes` indicates how the values of the status variable are to be interpreted.

`assumed_multiple` One or more numeric values. Within each nonresponse adjustment cell, the mean for nonrespondents is assumed to be a specified multiple of the mean for respondents. If `y_var` is a categorical variable, then the assumed nonrespondent mean (i.e., the proportion) in each cell is capped at 1.

`assumed_percentile` One or more numeric values, ranging from 0 to 1. Within each nonresponse adjustment cell, the mean of a continuous variable among nonrespondents is assumed to equal a specified percentile of the variable among respondents. The `assumed_percentile` parameter should be used only when the `y_var` variable is numeric. Quantiles are estimated with weights, using the function `svyquantile(..., qrule = "hf2")`.

Value

A data frame summarizing the range of bias under each assumption. For a numeric outcome variable, there is one row per value of `assumed_multiple` or `assumed_percentile`. For a categorical outcome variable, there is one row per combination of category and `assumed_multiple` or `assumed_percentile`.

The column `bias_of_unadj_estimate` is the nonresponse bias of the estimate from respondents produced using the unadjusted weights. The column `bias_of_adj_estimate` is the nonresponse bias of the estimate from respondents produced using nonresponse-adjusted weights, based on a weighting-class adjustment with `comparison_cell` as the weighting class variable. If no `comparison_cell` is specified, the two bias estimates will be the same.

References

See Petraglia et al. (2016) for an example of a range-of-bias analysis using these methods.

- Petraglia, E., Van de Kerckhove, W., and Krenzke, T. (2016). *Review of the Potential for Nonresponse Bias in FoodAPS 2012*. Prepared for the Economic Research Service, U.S. Department of Agriculture. Washington, D.C.

Examples

```
# Load example data

suppressPackageStartupMessages(library(survey))
data(api)

base_weights_design <- svydesign(
  data    = apiclus1,
  id      = ~dnum,
  weights = ~pw,
  fpc     = ~fpc
) |> as.svrepdesign(type = "JK1")

base_weights_design$variables$response_status <- sample(
  x = c("Respondent", "Nonrespondent"),
  prob = c(0.75, 0.25),
  size = nrow(base_weights_design),
  replace = TRUE
)

# Assess range of bias for mean of `api00`
# based on assuming nonrespondent means
# are equal to the 25th percentile or 75th percentile
# among respondents, within nonresponse adjustment cells

  assess_range_of_bias(
    survey_design = base_weights_design,
    y_var = "api00",
    comparison_cell = "stype",
    status = "response_status",
    status_codes = c("ER" = "Respondent",
```

```
                    "EN" = "Nonrespondent",
                    "IE" = "Ineligible",
                    "UE" = "Unknown"),
    assumed_percentile = c(0.25, 0.75)
  )

# Assess range of bias for proportions of `sch.wide`
# based on assuming nonrespondent proportions
# are equal to some multiple of respondent proportions,
# within nonresponse adjustment cells

  assess_range_of_bias(
    survey_design = base_weights_design,
    y_var = "sch.wide",
    comparison_cell = "stype",
    status = "response_status",
    status_codes = c("ER" = "Respondent",
                    "EN" = "Nonrespondent",
                    "IE" = "Ineligible",
                    "UE" = "Unknown"),
    assumed_multiple = c(0.25, 0.75)
  )
```

---

calculate_response_rates
Calculate Response Rates

Description

Calculates response rates using one of the response rate formulas defined by AAPOR (American Association of Public Opinion Research).

Usage

```
calculate_response_rates(
  data,
  status,
  status_codes = c("ER", "EN", "IE", "UE"),
  weights,
  rr_formula = "RR3",
  elig_method = "CASRO-subgroup",
  e = NULL
)
```

Arguments

data	A data frame containing the selected sample, one row per case.

status
: A character string giving the name of the variable representing response/eligibility status. The status variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).

status_codes
: A named vector, with four entries named 'ER', 'EN', 'IE', and 'UE'. status_codes indicates how the values of the status variable are to be interpreted.

weights
: (Optional) A character string giving the name of a variable representing weights in the data to use for calculating weighted response rates

rr_formula
: A character vector including any of the following: 'RR1', 'RR3', and 'RR5'. These are the names of formulas defined by AAPOR. See the *Formulas* section below for formulas.

elig_method
: If rr_formula='RR3', this specifies how to estimate an eligibility rate for cases with unknown eligibility. Must be one of the following:

  'CASRO-overall'
  Estimates an eligibility rate using the overall sample. If response rates are calculated for subgroups, the single overall sample estimate will be used as the estimated eligibility rate for subgroups as well.

  'CASRO-subgroup'
  Estimates eligibility rates separately for each subgroup.

  'specified'
  With this option, a numeric value is supplied by the user to the parameter e.

  For elig_method='CASRO-overall' or elig_method='CASRO-subgroup', the eligibility rate is estimated as $(ER)/(ER + NR + IE)$.

e
: (Required if elig_method='specified'). A numeric value between 0 and 1 specifying the estimated eligibility rate for cases with unknown eligibility. A character string giving the name of a numeric variable may also be supplied; in that case, the eligibility rate must be constant for all cases in a subgroup.

Value

Output consists of a data frame giving weighted and unweighted response rates. The following columns may be included, depending on the arguments supplied:

- RR1_Unweighted
- RR1_Weighted
- RR3_Unweighted
- RR3_Weighted
- RR5_Unweighted
- RR5_Weighted
- n: Total sample size
- Nhat: Sum of weights for the total sample

- `n_ER`: Number of eligible respondents
- `Nhat_ER`: Sum of weights for eligible respondents
- `n_EN`: Number of eligible nonrespondents
- `Nhat_EN`: Sum of weights for eligible nonrespondents
- `n_IE`: Number of ineligible cases
- `Nhat_IE`: Sum of weights for ineligible cases
- `n_UE`: Number of cases whose eligibility is unknown
- `Nhat_UE`: Sum of weights for cases whose eligibility is unknown
- `e_unwtd`: If *RR3* is calculated, the eligibility rate estimate *e* used for the unweighted response rate.
- `e_wtd`: If *RR3* is calculated, the eligibility rate estimate *e* used for the weighted response rate.

If the data frame is grouped (i.e. by using `df %>% group_by(Region)`), then the output contains one row per subgroup.

Formulas

Denote the sample totals as follows:

- **ER**: Total number of eligible respondents
- **EN**: Total number of eligible non-respondents
- **IE**: Total number of ineligible cases
- **UE**: Total number of cases whose eligibility is unknown

For weighted response rates, these totals are calculated using weights.

The response rate formulas are then as follows:

$$RR1 = ER/(ER + EN + UE)$$

RR1 essentially assumes that all cases with unknown eligibility are in fact eligible.

$$RR3 = ER/(ER + EN + (e * UE))$$

RR3 uses an estimate, *e*, of the eligibility rate among cases with unknown eligibility.

$$RR5 = ER/(ER + EN)$$

RR5 essentially assumes that all cases with unknown eligibility are in fact ineligible.

For *RR3*, an estimate, `e`, of the eligibility rate among cases with unknown eligibility must be used. AAPOR strongly recommends that the basis for the estimate should be explicitly stated and detailed.

The CASRO methods, which might be appropriate for the design, use the formula $e = 1 - (IE/(ER + EN + IE))$.

- For `elig_method='CASRO-overall'`, an estimate is calculated for the overall sample and this single estimate is used when calculating response rates for subgroups.
- For `elig_method='CASRO-subgroup'`, estimates are calculated separately for each subgroup.

Please consult AAPOR's current *Standard Definitions* for in-depth explanations.

References

The American Association for Public Opinion Research. 2016. *Standard Definitions: Final Dispositions of Case Codes and Outcome Rates for Surveys. 9th edition.* AAPOR.

Examples

```
# Load example data
data(involvement_survey_srs, package = "nrba")

involvement_survey_srs[["RESPONSE_STATUS"]] <- sample(1:4, size = 5000, replace = TRUE)

# Calculate overall response rates

involvement_survey_srs %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
    weights = "BASE_WEIGHT",
    rr_formula = "RR3",
    elig_method = "CASRO-overall"
  )

# Calculate response rates by subgroup

library(dplyr)

involvement_survey_srs %>%
  group_by(STUDENT_RACE, STUDENT_SEX) %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
    weights = "BASE_WEIGHT",
    rr_formula = "RR3",
    elig_method = "CASRO-overall"
  )

# Compare alternative approaches for handling of cases with unknown eligiblity

involvement_survey_srs %>%
  group_by(STUDENT_RACE) %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
    rr_formula = "RR3",
    elig_method = "CASRO-overall"
  )

involvement_survey_srs %>%
  group_by(STUDENT_RACE) %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
```

```
    rr_formula = "RR3",
    elig_method = "CASRO-subgroup"
  )

involvement_survey_srs %>%
  group_by(STUDENT_RACE) %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
    rr_formula = "RR3",
    elig_method = "specified",
    e = 0.5
  )

involvement_survey_srs %>%
  transform(e_by_email = ifelse(PARENT_HAS_EMAIL == "Has Email", 0.75, 0.25)) %>%
  group_by(PARENT_HAS_EMAIL) %>%
  calculate_response_rates(
    status = "RESPONSE_STATUS",
    status_codes = c("ER" = 1, "EN" = 2, "IE" = 3, "UE" = 4),
    rr_formula = "RR3",
    elig_method = "specified",
    e = "e_by_email"
  )
```

---

chisq_test_ind_response *Test the independence of survey response and auxiliary variables*

Description

Tests whether response status among eligible sample cases is independent of categorical auxiliary variables, using a Chi-Square test with Rao-Scott's second-order adjustment. If the data include cases known to be ineligible or who have unknown eligibility status, the data are subsetted to only include respondents and nonrespondents known to be eligible.

Usage

```
chisq_test_ind_response(
  survey_design,
  status,
  status_codes = c("ER", "EN", "UE", "IE"),
  aux_vars
)
```

Arguments

`survey_design` A survey design object created with the `survey` package.

`status`	A character string giving the name of the variable representing response/eligibility status. The `status` variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).
`status_codes`	A named vector, with four entries named 'ER', 'EN', 'IE', and 'UE'. `status_codes` indicates how the values of the `status` variable are to be interpreted.
`aux_vars`	A list of names of auxiliary variables.

Details

Please see svychisq for details of how the Rao-Scott second-order adjusted test is conducted.

Value

A data frame containing the results of the Chi-Square test(s) of independence between response status and each auxiliary variable. If multiple auxiliary variables are specified, the output data contains one row per auxiliary variable.

The columns of the output dataset include:

- `auxiliary_variable`: The name of the auxiliary variable tested
- `statistic`: The value of the test statistic
- `ndf`: Numerator degrees of freedom for the reference distribution
- `ddf`: Denominator degrees of freedom for the reference distribution
- `p_value`: The p-value of the test of independence
- `test_method`: Text giving the name of the statistical test
- `variance_method`: Text describing the method of variance estimation

References

- Rao, JNK, Scott, AJ (1984) "On Chi-squared Tests For Multiway Contigency Tables with Proportions Estimated From Survey Data" Annals of Statistics 12:46-60.

Examples

```
# Create a survey design object ----
library(survey)
data(involvement_survey_srs, package = "nrba")

involvement_survey <- svydesign(
```

```
    weights = ~BASE_WEIGHT,
    id = ~UNIQUE_ID,
    data = involvement_survey_srs
  )


  # Test whether response status varies by race or by sex ----

  test_results <- chisq_test_ind_response(
    survey_design = involvement_survey,
    status = "RESPONSE_STATUS",
    status_codes = c(
      "ER" = "Respondent",
      "EN" = "Nonrespondent",
      "UE" = "Unknown",
      "IE" = "Ineligible"
    ),
    aux_vars = c("STUDENT_RACE", "STUDENT_SEX")
  )

  print(test_results)
```

---

chisq_test_vs_external_estimate

Test of differences in survey percentages relative to external estimates

---

Description

Compare estimated percentages from the present survey to external estimates from a benchmark source. A Chi-Square test with Rao-Scott's second-order adjustment is used to evaluate whether the survey's estimates differ from the external estimates.

Usage

```
chisq_test_vs_external_estimate(survey_design, y_var, ext_ests, na.rm = TRUE)
```

Arguments

survey_design	A survey design object created with the survey package.
y_var	Name of dependent categorical variable.
ext_ests	A numeric vector containing the external estimate of the percentages for each category. The vector must have names, each name corresponding to a given category.
na.rm	Whether to drop cases with missing values

Details

Please see svygofchisq for details of how the Rao-Scott second-order adjusted test is conducted. The test statistic, `statistic` is obtained by calculating the Pearson Chi-squared statistic for the estimated table of population totals. The reference distribution is a Satterthwaite approximation. The p-value is obtained by comparing `statistic/scale` to a Chi-squared distribution with `df` degrees of freedom.

Value

A data frame containing the results of the Chi-Square test(s) of whether survey-based estimates systematically differ from external estimates.

The columns of the output dataset include:

- `statistic`: The value of the test statistic
- `df`: Degrees of freedom for the reference Chi-Squared distribution
- `scale`: Estimated scale parameter.
- `p_value`: The p-value of the test of independence
- `test_method`: Text giving the name of the statistical test
- `variance_method`: Text describing the method of variance estimation

References

- Rao, JNK, Scott, AJ (1984) "On Chi-squared Tests For Multiway Contigency Tables with Proportions Estimated From Survey Data" Annals of Statistics 12:46-60.

Examples

```
library(survey)

# Create a survey design ----
data("involvement_survey_pop", package = "nrba")
data("involvement_survey_str2s", package = "nrba")

involvement_survey_sample <- svydesign(
  data = involvement_survey_str2s,
  weights = ~BASE_WEIGHT,
  strata = ~SCHOOL_DISTRICT,
  ids = ~ SCHOOL_ID + UNIQUE_ID,
  fpc = ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL
)

# Subset to only include survey respondents ----
```

```
involvement_survey_respondents <- subset(
  involvement_survey_sample,
  RESPONSE_STATUS == "Respondent"
)

# Test whether percentages of categorical variable differ from benchmark ----

parent_email_benchmark <- c(
  "Has Email" = 0.85,
  "No Email" = 0.15
)

chisq_test_vs_external_estimate(
  survey_design = involvement_survey_respondents,
  y_var = "PARENT_HAS_EMAIL",
  ext_ests = parent_email_benchmark
)
```

---

get_cumulative_estimates

Calculate cumulative estimates of a mean/proportion

Description

Calculates estimates of a mean/proportion which are cumulative with respect to a predictor variable, such as week of data collection or number of contact attempts. This can be useful for examining whether estimates are affected by decisions such as whether to extend the data collection period or make additional contact attempts.

Usage

```
get_cumulative_estimates(
  survey_design,
  y_var,
  y_var_type = NULL,
  predictor_variable
)
```

Arguments

survey_design	A survey design object created with the survey package.
y_var	Name of a variable whose mean or proportion is to be estimated.
y_var_type	Either NULL, "categorical" or "numeric". For "categorical", proportions are estimated. For "numeric", means are estimated. For NULL (the default), then proportions are estimated if y_var is a factor or character variable. Otherwise, means are estimated. The data will be subset to remove any missing values in this variable.

predictor_variable
: Name of a variable for which cumulative estimates of y_var will be calculated. This variable should either be numeric or have categories which when sorted by their label are arranged in ascending order. The data will be subset to remove any missing values of the predictor variable.

Value

A dataframe of cumulative estimates. The first column–whose name matches predictor_variable– gives describes the values of predictor_variable for which a given estimate was computed. The other columns of the result include the following:

- outcome: The name of the variable for which estimates are computed
- outcome_category: For a categorical variable, the category of that variable
- estimate: The estimated mean or proportion.
- std_error: The estimated standard error
- respondent_sample_size: The number of cases used to produce the estimate (excluding missing values)

References

See Maitland et al. (2017) for an example of a level-of-effort analysis based on this method.

- Maitland, A. et al. (2017). *A Nonresponse Bias Analysis of the Health Information National Trends Survey (HINTS).* Journal of Health Communication 22, 545-553. doi:10.1080/10810730.2017.1324539

Examples

```
# Create an example survey design
# with a variable representing number of contact attempts
library(survey)
data(involvement_survey_srs, package = "nrba")


survey_design <- svydesign(
  weights = ~BASE_WEIGHT,
  id = ~UNIQUE_ID,
  fpc = ~N_STUDENTS,
  data = involvement_survey_srs
)

# Cumulative estimates from respondents for average student age ----
get_cumulative_estimates(
  survey_design = survey_design |>
    subset(RESPONSE_STATUS == "Respondent"),
  y_var = "STUDENT_AGE",
  y_var_type = "numeric",
  predictor_variable = "CONTACT_ATTEMPTS"
)
```

```
# Cumulative estimates from respondents for proportions of categorical variable ----
get_cumulative_estimates(
  survey_design = survey_design |>
    subset(RESPONSE_STATUS == "Respondent"),
  y_var = "WHETHER_PARENT_AGREES",
  y_var_type = "categorical",
  predictor_variable = "CONTACT_ATTEMPTS"
)
```

`involvement_survey_pop` *Parent involvement survey: population data*

Description

An example dataset describing a population of 20,000 students with disabilities in 20 school districts. This population is the basis for selecting a sample of students for a parent involvement survey.

Usage

```
involvement_survey_pop
```

Format

A data frame with 20,000 rows and 9 variables

Fields

UNIQUE_ID A unique identifier for students

SCHOOL_DISTRICT A unique identifier for school districts

SCHOOL_ID A unique identifier for schools, nested within districts

STUDENT_GRADE Student's grade level: 'PK', 'K', 1-12

STUDENT_AGE Student's age, measured in years

STUDENT_DISABILITY_CODE Code for student's disability category (e.g. 'VI' for 'Visual Impairments')

STUDENT_DISABILITY_CATEGORY Student's disability category (e.g. 'Visual Impairments')

STUDENT_SEX 'Female' or 'Male'

STUDENT_RACE Seven-level code with descriptive label (e.g. 'AS7 (Asian)')

Examples

```
involvement_survey_pop
```

involvement_survey_srs
Parent involvement survey: simple random sample

Description

An example dataset describing a simple random sample of 5,000 parents of students with disabilities, from a population of 20,000. The parent involvement survey measures a single key outcome: whether "parents perceive that schools facilitate parent involvement as a means of improving services and results for children with disabilities."

The variable `BASE_WEIGHT` provides the base sampling weight. The variable `N_STUDENTS_IN_SCHOOL` can be used to provide a finite population correction for variance estimation.

Usage

```
involvement_survey_srs
```

Format

A data frame with 5,000 rows and 17 variables

Fields

UNIQUE_ID A unique identifier for students

RESPONSE_STATUS Survey response/eligibility status: 'Respondent', 'Nonrespondent', 'Ineligble', 'Unknown'

WHETHER_PARENT_AGREES Parent agreement ('AGREE' or 'DISAGREE') for whether they perceive that schools facilitate parent involvement

SCHOOL_DISTRICT A unique identifier for school districts

SCHOOL_ID A unique identifier for schools, nested within districts

STUDENT_GRADE Student's grade level: 'PK', 'K', 1-12

STUDENT_AGE Student's age, measured in years

STUDENT_DISABILITY_CODE Code for student's disability category (e.g. 'VI' for 'Visual Impairments')

STUDENT_DISABILITY_CATEGORY Student's disability category (e.g. 'Visual Impairments')

STUDENT_SEX 'Female' or 'Male'

STUDENT_RACE Seven-level code with descriptive label (e.g. 'AS7 (Asian)')

PARENT_HAS_EMAIL Whether parent has an e-mail address ('Has Email' vs 'No Email')

PARENT_HAS_EMAIL_BENCHMARK Population benchmark for category of `PARENT_HAS_EMAIL`

PARENT_HAS_EMAIL_BENCHMARK Population benchmark for category of `STUDENT_RACE`

BASE_WEIGHT Sampling weight to use for weighted estimates

N_STUDENTS Total number of students in the population

CONTACT_ATTEMPTS The number of contact attempts made for each case (ranges between 1 and 6)

Examples

```
involvement_survey_srs
```

`involvement_survey_str2s`

Parent involvement survey: stratified, two-stage sample

Description

An example dataset describing a stratified, multistage sample of 1,000 parents of students with disabilities, from a population of 20,000. The parent involvement survey measures a single key outcome: whether "parents perceive that schools facilitate parent involvement as a means of improving services and results for children with disabilities."

The sample was selected by sampling 5 schools from each of 20 districts, and then sampling parents of 10 children in each sampled school. The variable `BASE_WEIGHT` provides the base sampling weight. The variable `SCHOOL_DISTRICT` was used for stratification, and the variables `SCHOOL_ID` and `UNIQUE_ID` uniquely identify the first and second stage sampling units (schools and parents). The variables `N_SCHOOLS_IN_DISTRICT` and `N_STUDENTS_IN_SCHOOL` can be used to provide finite population corrections.

Usage

```
involvement_survey_str2s
```

Format

A data frame with 5,000 rows and 18 variables

Fields

UNIQUE_ID A unique identifier for students

RESPONSE_STATUS Survey response/eligibility status: 'Respondent', 'Nonrespondent', 'Ineligble', 'Unknown'

WHETHER_PARENT_AGREES Parent agreement ('AGREE' or 'DISAGREE') for whether they perceive that schools facilitate parent involvement

SCHOOL_DISTRICT A unique identifier for school districts

SCHOOL_ID A unique identifier for schools, nested within districts

STUDENT_GRADE Student's grade level: 'PK', 'K', 1-12

STUDENT_AGE Student's age, measured in years

STUDENT_DISABILITY_CODE Code for student's disability category (e.g. 'VI' for 'Visual Impairments')

STUDENT_DISABILITY_CATEGORY Student's disability category (e.g. 'Visual Impairments')

STUDENT_SEX 'Female' or 'Male'

STUDENT_RACE Seven-level code with descriptive label (e.g. 'AS7 (Asian)')

PARENT_HAS_EMAIL Whether parent has an e-mail address ('Has Email' vs 'No Email')

PARENT_HAS_EMAIL_BENCHMARK Population benchmark for category of `PARENT_HAS_EMAIL`

STUDENT_RACE_BENCHMARK Population benchmark for category of `STUDENT_RACE`

N_SCHOOLS_IN_DISTRICT Total number of schools in each district

N_STUDENTS_IN_SCHOOL Total number of students in each school

BASE_WEIGHT Sampling weight to use for weighted estimates

CONTACT_ATTEMPTS The number of contact attempts made for each case (ranges between 1 and 6)

Examples

```
# Load the data
involvement_survey_str2s

# Prepare the data for analysis with the 'survey' package

  library(survey)

  involvement_survey <- svydesign(
    data = involvement_survey_str2s,
    weights = ~ BASE_WEIGHT,
    strata =  ~ SCHOOL_DISTRICT,
    ids =     ~ SCHOOL_ID             + UNIQUE_ID,
    fpc =     ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL
  )
```

`predict_outcome_via_glm`

Fit a regression model to predict survey outcomes

Description

A regression model is fit to the sample data to predict outcomes measured by a survey. This model can be used to identify auxiliary variables that are predictive of survey outcomes and hence are potentially useful for nonresponse bias analysis or weighting adjustments.

Only data from survey respondents will be used to fit the model, since survey outcomes are only measured among respondents.

The function returns a summary of the model, including overall tests for each variable of whether that variable improves the model's ability to predict response status in the population of interest (not just in the random sample at hand).

Usage

```
predict_outcome_via_glm(
  survey_design,
  outcome_variable,
  outcome_type = "continuous",
  outcome_to_predict = NULL,
  numeric_predictors = NULL,
  categorical_predictors = NULL,
  model_selection = "main-effects",
 selection_controls = list(alpha_enter = 0.5, alpha_remain = 0.5, max_iterations = 100L)
)
```

Arguments

`survey_design` A survey design object created with the `survey` package.

`outcome_variable`
: Name of an outcome variable to use as the dependent variable in the model The value of this variable is expected to be `NA` (i.e. missing) for all cases other than eligible respondents.

`outcome_type` Either `"binary"` or `"continuous"`. For `"binary"`, a logistic regression model is used. For `"continuous"`, a generalized linear model is fit using using an identity link function.

`outcome_to_predict`
: Only required if `outcome_type="binary"`. Specify which category of `outcome_variable` is to be predicted.

`numeric_predictors`
: A list of names of numeric auxiliary variables to use for predicting response status.

`categorical_predictors`
: A list of names of categorical auxiliary variables to use for predicting response status.

`model_selection`
: A character string specifying how to select a model. The default and recommended method is 'main-effects', which simply includes main effects for each of the predictor variables.
The method `'stepwise'` can be used to perform stepwise selection of variables for the model. However, stepwise selection invalidates p-values, standard errors, and confidence intervals, which are generally calculated under the assumption that model specification is predetermined.

`selection_controls`
: Only required if `model-selection` isn't set to `"main-effects"`. Otherwise, a list of parameters for model selection to pass on to `stepwise_model_selection`, with elements `alpha_enter`, `alpha_remain`, and `max_iterations`.

Details

See Lumley and Scott (2017) for details of how regression models are fit to survey data. For overall tests of variables, a Rao-Scott Likelihood Ratio Test is conducted (see section 4 of Lumley and Scott

(2017) for statistical details) using the function `regTermTest(method = "LRT", lrt.approximation = "saddlepoint")` from the 'survey' package.

If the user specifies `model_selection = "stepwise"`, a regression model is selected by adding and removing variables based on the p-value from a likelihood ratio rest. At each stage, a single variable is added to the model if the p-value of the likelihood ratio test from adding the variable is below `alpha_enter` and its p-value is less than that of all other variables not already in the model. Next, of the variables already in the model, the variable with the largest p-value is dropped if its p-value is greater than `alpha_remain`. This iterative process continues until a maximum number of iterations is reached or until either all variables have been added to the model or there are no unadded variables for which the likelihood ratio test has a p-value below `alpha_enter`.

Value

A data frame summarizing the fitted regression model.

Each row in the data frame represents a coefficient in the model. The column `variable` describes the underlying variable for the coefficient. For categorical variables, the column `variable_category` indicates the particular category of that variable for which a coefficient is estimated.

The columns `estimated_coefficient`, `se_coefficient`, `conf_intrvl_lower`, `conf_intrvl_upper`, and `p_value_coefficient` are summary statistics for the estimated coefficient. Note that `p_value_coefficient` is based on the Wald t-test for the coefficient.

The column `variable_level_p_value` gives the p-value of the Rao-Scott Likelihood Ratio Test for including the variable in the model. This likelihood ratio test has its test statistic given by the column `LRT_chisq_statistic`, and the reference distribution for this test is a linear combination of `p` F-distributions with numerator degrees of freedom given by `LRT_df_numerator` and denominator degrees of freedom given by `LRT_df_denominator`, where `p` is the number of coefficients in the model corresponding to the variable being tested.

References

- Lumley, T., & Scott A. (2017). Fitting Regression Models to Survey Data. Statistical Science 32 (2) 265 - 278. https://doi.org/10.1214/16-STS605

Examples

```
library(survey)

# Create a survey design ----
data(involvement_survey_str2s, package = "nrba")

survey_design <- svydesign(
  weights = ~BASE_WEIGHT,
  strata = ~SCHOOL_DISTRICT,
  id = ~ SCHOOL_ID + UNIQUE_ID,
  fpc = ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL,
  data = involvement_survey_str2s
```

```
)

predict_outcome_via_glm(
  survey_design = survey_design,
  outcome_variable = "WHETHER_PARENT_AGREES",
  outcome_type = "binary",
  outcome_to_predict = "AGREE",
  model_selection = "main-effects",
  numeric_predictors = c("STUDENT_AGE"),
  categorical_predictors = c("STUDENT_DISABILITY_CATEGORY", "PARENT_HAS_EMAIL")
)
```

`predict_response_status_via_glm` *Fit a logistic regression model to predict response to the survey.*

Description

A logistic regression model is fit to the sample data to predict whether an individual responds to the survey (i.e. is an eligible respondent) rather than a nonrespondent. Ineligible cases and cases with unknown eligibility status are not included in this model.

The function returns a summary of the model, including overall tests for each variable of whether that variable improves the model's ability to predict response status in the population of interest (not just in the random sample at hand).

This model can be used to identify auxiliary variables associated with response status and compare multiple auxiliary variables in terms of their ability to predict response status.

Usage

```
predict_response_status_via_glm(
  survey_design,
  status,
  status_codes = c("ER", "EN", "IE", "UE"),
  numeric_predictors = NULL,
  categorical_predictors = NULL,
  model_selection = "main-effects",
  selection_controls = list(alpha_enter = 0.5, alpha_remain = 0.5, max_iterations = 100L)
)
```

Arguments

`survey_design` A survey design object created with the `survey` package.

status
: A character string giving the name of the variable representing response/eligibility status. The status variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).

status_codes
: A named vector, with two entries named 'ER' and 'EN' indicating which values of the status variable represent eligible respondents (ER) and eligible nonrespondents (EN).

numeric_predictors
: A list of names of numeric auxiliary variables to use for predicting response status.

categorical_predictors
: A list of names of categorical auxiliary variables to use for predicting response status.

model_selection
: A character string specifying how to select a model. The default and recommended method is 'main-effects', which simply includes main effects for each of the predictor variables.
The method 'stepwise' can be used to perform stepwise selection of variables for the model. However, stepwise selection invalidates p-values, standard errors, and confidence intervals, which are generally calculated under the assumption that model specification is predetermined.

selection_controls
: Only required if model-selection isn't set to "main-effects". Otherwise, a list of parameters for model selection to pass on to stepwise_model_selection, with elements alpha_enter, alpha_remain, and max_iterations.

Details

See Lumley and Scott (2017) for details of how regression models are fit to survey data. For overall tests of variables, a Rao-Scott Likelihood Ratio Test is conducted (see section 4 of Lumley and Scott (2017) for statistical details) using the function regTermTest(method = "LRT", lrt.approximation = "saddlepoint") from the 'survey' package.

If the user specifies model_selection = "stepwise", a regression model is selected by adding and removing variables based on the p-value from a likelihood ratio rest. At each stage, a single variable is added to the model if the p-value of the likelihood ratio test from adding the variable is below alpha_enter and its p-value is less than that of all other variables not already in the model. Next, of the variables already in the model, the variable with the largest p-value is dropped if its p-value is greater than alpha_remain. This iterative process continues until a maximum number of iterations is reached or until either all variables have been added to the model or there are no unadded variables for which the likelihood ratio test has a p-value below alpha_enter.

Value

A data frame summarizing the fitted logistic regression model.

Each row in the data frame represents a coefficient in the model. The column variable describes the underlying variable for the coefficient. For categorical variables, the column variable_category indicates the particular category of that variable for which a coefficient is estimated.

The columns estimated_coefficient, se_coefficient, conf_intrvl_lower, conf_intrvl_upper, and p_value_coefficient are summary statistics for the estimated coefficient. Note that p_value_coefficient is based on the Wald t-test for the coefficient.

The column variable_level_p_value gives the p-value of the Rao-Scott Likelihood Ratio Test for including the variable in the model. This likelihood ratio test has its test statistic given by the column LRT_chisq_statistic, and the reference distribution for this test is a linear combination of p F-distributions with numerator degrees of freedom given by LRT_df_numerator and denominator degrees of freedom given by LRT_df_denominator, where p is the number of coefficients in the model corresponding to the variable being tested.

References

- Lumley, T., & Scott A. (2017). Fitting Regression Models to Survey Data. Statistical Science 32 (2) 265 - 278. https://doi.org/10.1214/16-STS605

Examples

```
library(survey)

# Create a survey design ----
data(involvement_survey_str2s, package = "nrba")

survey_design <- survey_design <- svydesign(
  data = involvement_survey_str2s,
  weights = ~BASE_WEIGHT,
  strata = ~SCHOOL_DISTRICT,
  ids = ~ SCHOOL_ID + UNIQUE_ID,
  fpc = ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL
)

predict_response_status_via_glm(
  survey_design = survey_design,
  status = "RESPONSE_STATUS",
  status_codes = c(
    "ER" = "Respondent",
    "EN" = "Nonrespondent",
    "IE" = "Ineligible",
    "UE" = "Unknown"
  ),
  model_selection = "main-effects",
  numeric_predictors = c("STUDENT_AGE"),
  categorical_predictors = c("PARENT_HAS_EMAIL", "STUDENT_GRADE")
)
```

`rake_to_benchmarks` *Re-weight data to match population benchmarks, using raking or post-stratification*

Description

Adjusts weights in the data to ensure that estimated population totals for grouping variables match known population benchmarks. If there is only one grouping variable, simple post-stratification is used. If there are multiple grouping variables, raking (also known as iterative post-stratification) is used.

Usage

```
rake_to_benchmarks(
  survey_design,
  group_vars,
  group_benchmark_vars,
  max_iterations = 100,
  epsilon = 5e-06
)
```

Arguments

- `survey_design` A survey design object created with the `survey` package.
- `group_vars` Names of grouping variables in the data dividing the sample into groups for which benchmark data are available. These variables cannot have any missing values
- `group_benchmark_vars` Names of group benchmark variables in the data corresponding to `group_vars`. For each category of a grouping variable, the group benchmark variable gives the population benchmark (i.e. population size) for that category.
- `max_iterations` If there are multiple grouping variables, then raking is used rather than post-stratification. The parameter `max_iterations` controls the maximum number of iterations to use in raking.
- `epsilon` If raking is used, convergence for a given margin is declared if the maximum change in a re-weighted total is less than `epsilon` times the total sum of the original weights in the design.

Details

Raking adjusts the weight assigned to each sample member so that, after reweighting, the weighted sample percentages for population subgroups match their known population percentages. In a sense, raking causes the sample to more closely resemble the population in terms of variables for which population sizes are known.

Raking can be useful to reduce nonresponse bias caused by having groups which are overrepresented in the responding sample relative to their population size. If the population subgroups systematically differ in terms of outcome variables of interest, then raking can also be helpful in terms of reduce sampling variances. However, when population subgroups do not differ in terms of outcome variables of interest, then raking may increase sampling variances.

There are two basic requirements for raking.

- Basic Requirement 1 - Values of the grouping variable(s) must be known for all respondents.
- Basic Requirement 2 - The population size of each group must be known (or precisely estimated).

When there is effectively only one grouping variable (though this variable can be defined as a combination of other variables), raking amounts to simple post-stratification. For example, simple post-stratification would be used if the grouping variable is "Age x Sex x Race", and the population size of each combination of age, sex, and race is known. The method of "iterative poststratification" (also known as "iterative proportional fitting") is used when there are multiple grouping variables, and population sizes are known for each grouping variable but not for combinations of grouping variables. For example, iterative proportional fitting would be necessary if population sizes are known for age groups and for gender categories but not for combinations of age groups and gender categories.

Value

A survey design object with raked or post-stratified weights

Examples

```
# Load the survey data

data(involvement_survey_srs, package = "nrba")

# Calculate population benchmarks
population_benchmarks <- list(
  "PARENT_HAS_EMAIL" = data.frame(
    PARENT_HAS_EMAIL = c("Has Email", "No Email"),
    PARENT_HAS_EMAIL_POP_BENCHMARK = c(17036, 2964)
  ),
  "STUDENT_RACE" = data.frame(
    STUDENT_RACE = c(
      "AM7 (American Indian or Alaska Native)", "AS7 (Asian)",
      "BL7 (Black or African American)",
      "HI7 (Hispanic or Latino Ethnicity)", "MU7 (Two or More Races)",
      "PI7 (Native Hawaiian or Other Pacific Islander)",
      "WH7 (White)"
    ),
    STUDENT_RACE_POP_BENCHMARK = c(206, 258, 3227, 1097, 595, 153, 14464)
  )
)
```

```
# Add the population benchmarks as variables in the data
involvement_survey_srs <- merge(
  x = involvement_survey_srs,
  y = population_benchmarks$PARENT_HAS_EMAIL,
  by = "PARENT_HAS_EMAIL"
)
involvement_survey_srs <- merge(
  x = involvement_survey_srs,
  y = population_benchmarks$STUDENT_RACE,
  by = "STUDENT_RACE"
)

# Create a survey design object
library(survey)

survey_design <- svydesign(
  weights = ~BASE_WEIGHT,
  id = ~UNIQUE_ID,
  fpc = ~N_STUDENTS,
  data = involvement_survey_srs
)

# Subset data to only include respondents
survey_respondents <- subset(
  survey_design,
  RESPONSE_STATUS == "Respondent"
)

# Rake to the benchmarks
raked_survey_design <- rake_to_benchmarks(
  survey_design = survey_respondents,
  group_vars = c("PARENT_HAS_EMAIL", "STUDENT_RACE"),
  group_benchmark_vars = c(
    "PARENT_HAS_EMAIL_POP_BENCHMARK",
    "STUDENT_RACE_POP_BENCHMARK"
  ),
)

# Inspect estimates from respondents, before and after raking

svymean(
  x = ~PARENT_HAS_EMAIL,
  design = survey_respondents
)
svymean(
  x = ~PARENT_HAS_EMAIL,
  design = raked_survey_design
)

svymean(
  x = ~WHETHER_PARENT_AGREES,
  design = survey_respondents
)
```

```
svymean(
  x = ~WHETHER_PARENT_AGREES,
  design = raked_survey_design
)
```

---

stepwise_model_selection *Select and fit a model using stepwise regression*

Description

A regression model is selected by iteratively adding and removing variables based on the p-value from a likelihood ratio rest. At each stage, a single variable is added to the model if the p-value of the likelihood ratio test from adding the variable is below `alpha_enter` and its p-value is less than that of all other variables not already in the model. Next, of the variables already in the model, the variable with the largest p-value is dropped if its p-value is greater than `alpha_remain`. This iterative process continues until a maximum number of iterations is reached or until either all variables have been added to the model or there are no variables not yet in the model whose likelihood ratio test has a p-value below `alpha_enter`.

Stepwise model selection generally invalidates inferential statistics such as p-values, standard errors, or confidence intervals and leads to overestimation of the size of coefficients for variables included in the selected model. This bias increases as the value of `alpha_enter` or `alpha_remain` decreases. The use of stepwise model selection should be limited only to reducing a large list of candidate variables for nonresponse adjustment.

Usage

```
stepwise_model_selection(
  survey_design,
  outcome_variable,
  predictor_variables,
  model_type = "binary-logistic",
  max_iterations = 100L,
  alpha_enter = 0.5,
  alpha_remain = 0.5
)
```

Arguments

`survey_design` A survey design object created with the `survey` package.

`outcome_variable` The name of an outcome variable to use as the dependent variable.

`predictor_variables` A list of names of variables to consider as predictors for the model.

model_type A character string describing the type of model to fit. 'binary-logistic' for a binary logistic regression, 'ordinal-logistic' for an ordinal logistic regression (cumulative proportional-odds), 'normal' for the typical model which assumes residuals follow a Normal distribution.

max_iterations Maximum number of iterations to try adding new variables to the model.

alpha_enter The maximum p-value allowed for a variable to be added to the model. Large values such as 0.5 or greater are recommended to reduce the bias of estimates from the selected model.

alpha_remain The maximum p-value allowed for a variable to remain in the model. Large values such as 0.5 or greater are recommended to reduce the bias of estimates from the selected model.

Details

See Lumley and Scott (2017) for details of how regression models are fit to survey data. For overall tests of variables, a Rao-Scott Likelihood Ratio Test is conducted (see section 4 of Lumley and Scott (2017) for statistical details) using the function `regTermTest(method = "LRT", lrt.approximation = "saddlepoint")` from the 'survey' package.

See Sauerbrei et al. (2020) for a discussion of statistical issues with using stepwise model selection.

Value

An object of class svyglm representing a regression model fit using the 'survey' package.

References

- Lumley, T., & Scott A. (2017). Fitting Regression Models to Survey Data. Statistical Science 32 (2) 265 - 278. https://doi.org/10.1214/16-STS605
- Sauerbrei, W., Perperoglou, A., Schmid, M. et al. (2020). State of the art in selection of variables and functional forms in multivariable analysis - outstanding issues. Diagnostic and Prognostic Research 4, 3. https://doi.org/10.1186/s41512-020-00074-3

Examples

```
library(survey)

# Load example data and prepare it for analysis
data(involvement_survey_str2s, package = 'nrba')

involvement_survey <- svydesign(
  data = involvement_survey_str2s,
  ids = ~ SCHOOL_ID + UNIQUE_ID,
  fpc = ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL,
  strata = ~ SCHOOL_DISTRICT,
  weights = ~ BASE_WEIGHT
)

involvement_survey <- involvement_survey |>
    transform(WHETHER_PARENT_AGREES = factor(WHETHER_PARENT_AGREES))
```

```
# Fit a regression model using stepwise selection
selected_model <- stepwise_model_selection(
  survey_design = involvement_survey,
  outcome_variable = "WHETHER_PARENT_AGREES",
  predictor_variables = c("STUDENT_RACE", "STUDENT_DISABILITY_CATEGORY"),
  model_type = "binary-logistic",
  max_iterations = 100,
  alpha_enter = 0.5,
  alpha_remain = 0.5
)
```

t_test_by_response_status

t-test of differences in means/percentages between responding sample and full sample, or between responding sample and eligible sample

Description

The function `t_test_resp_vs_full` tests whether means of auxiliary variables differ between respondents and the full selected sample, where the full sample consists of all cases regardless of response status or eligibility status.
The function `t_test_resp_vs_elig` tests whether means differ between the responding sample and the eligible sample, where the eligible sample consists of all cases known to be eligible, regardless of response status.

See Lohr and Riddles (2016) for the statistical theory of this test.

Usage

```
t_test_resp_vs_full(
  survey_design,
  y_vars,
  na.rm = TRUE,
  status,
  status_codes = c("ER", "EN", "IE", "UE"),
  null_difference = 0,
  alternative = "unequal",
  degrees_of_freedom = survey::degf(survey_design) - 1
)

t_test_resp_vs_elig(
  survey_design,
  y_vars,
  na.rm = TRUE,
  status,
  status_codes = c("ER", "EN", "IE", "UE"),
  null_difference = 0,
```

```
  alternative = "unequal",
  degrees_of_freedom = survey::degf(survey_design) - 1
)
```

Arguments

- `survey_design` — A survey design object created with the `survey` package.
- `y_vars` — Names of dependent variables for tests. For categorical variables, percentages of each category are tested.
- `na.rm` — Whether to drop cases with missing values for a given dependent variable.
- `status` — The name of the variable representing response/eligibility status.
  The `status` variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).
- `status_codes` — A named vector, with four entries named 'ER', 'EN', 'IE', and 'UE'.
  `status_codes` indicates how the values of the `status` variable are to be interpreted.
- `null_difference` — The difference between the two means under the null hypothesis. Default is `0`.
- `alternative` — Can be one of the following:
  - `'unequal'` (the default): two-sided test of whether difference in means is equal to `null_difference`
  - `'less'`: one-sided test of whether difference is less than `null_difference`
  - `'greater'`: one-sided test of whether difference is greater than `null_difference`
- `degrees_of_freedom` — The degrees of freedom to use for the test's reference distribution. Unless specified otherwise, the default is the design degrees of freedom minus one, where the design degrees of freedom are estimated using the `survey` package's `degf` method.

Value

A data frame describing the results of the t-tests, one row per dependent variable.

Statistical Details

The t-statistic used for the test has as its numerator the difference in means between the two samples, minus the `null_difference`. The denominator for the t-statistic is the estimated standard error of the difference in means. Because the two means are based on overlapping groups and thus have correlated sampling errors, special care is taken to estimate the covariance of the two estimates. For designs which use sets of replicate weights for variance estimation, the two means and their difference are estimated using each set of replicate weights; the estimated differences from the sets of replicate weights are then used to estimate sampling error with a formula appropriate to the replication method (JKn, BRR, etc.). For designs which use linearization methods for variance estimation, the covariance between the two means is estimated using the method of linearization based on influence functions implemented in the survey package. See Osier (2009) for an overview of the method of linearization based on influence functions. Eckman et al. (2023) showed in a simulation study that linearization and replication performed similarly in estimating the variance of a difference in means for overlapping samples.

Unless specified otherwise using the degrees_of_freedom parameter, the degrees of freedom for the test are set to the design degrees of freedom minus one. Design degrees of freedom are estimated using the survey package's degf method.

See Lohr and Riddles (2016) for the statistical details of this test. See Van de Kerckhove et al. (2009) and Amaya and Presser (2017) for examples of a nonresponse bias analysis which uses t-tests to compare responding samples to eligible samples.

References

- Amaya, A., Presser, S. (2017). *Nonresponse Bias for Univariate and Multivariate Estimates of Social Activities and Roles*. Public Opinion Quarterly, Volume 81, Issue 1, 1 March 2017, Pages 1–36, https://doi.org/10.1093/poq/nfw037
- Eckman, S., Unangst, J., Dever, J., Antoun, A. (2023). *The Precision of Estimates of Nonresponse Bias in Means*. Journal of Survey Statistics and Methodology, 11(4), 758-783. https://doi.org/10.1093/jssam/smac019
- Lohr, S., Riddles, M. (2016). *Tests for Evaluating Nonresponse Bias in Surveys*. Survey Methodology 42(2): 195-218. https://www150.statcan.gc.ca/n1/pub/12-001-x/2016002/article/14677-eng.pdf
- Osier, G. (2009). *Variance estimation for complex indicators of poverty and inequality using linearization techniques*. Survey Research Methods, 3(3), 167-195. https://doi.org/10.18148/srm/2009.v3i3.369
- Van de Kerckhove, W., Krenzke, T., and Mohadjer, L. (2009). *Adult Literacy and Lifeskills Survey (ALL) 2003: U.S. Nonresponse Bias Analysis (NCES 2009-063)*. National Center for Education Statistics, Institute of Education Sciences, U.S. Department of Education. Washington, DC.

Examples

```
library(survey)

# Create a survey design ----
data(involvement_survey_srs, package = 'nrba')

survey_design <- svydesign(weights = ~ BASE_WEIGHT,
                           id = ~ UNIQUE_ID,
                           fpc = ~ N_STUDENTS,
                           data = involvement_survey_srs)

# Compare respondents' mean to the full sample mean ----

t_test_resp_vs_full(survey_design = survey_design,
                    y_vars = c("STUDENT_AGE", "WHETHER_PARENT_AGREES"),
```

```
                    status = 'RESPONSE_STATUS',
                    status_codes = c('ER' = "Respondent",
                                     'EN' = "Nonrespondent",
                                     'IE' = "Ineligible",
                                     'UE' = "Unknown"))

# Compare respondents' mean to the mean of all eligible cases ----

t_test_resp_vs_full(survey_design = survey_design,
                    y_vars = c("STUDENT_AGE", "WHETHER_PARENT_AGREES"),
                    status = 'RESPONSE_STATUS',
                    status_codes = c('ER' = "Respondent",
                                     'EN' = "Nonrespondent",
                                     'IE' = "Ineligible",
                                     'UE' = "Unknown"))
# One-sided tests ----

  ## Null Hypothesis: Y_bar_resp - Y_bar_full <= 0.1
  ## Alt. Hypothesis: Y_bar_resp - Y_bar_full >  0.1

t_test_resp_vs_full(survey_design = survey_design,
                    y_vars = c("STUDENT_AGE", "WHETHER_PARENT_AGREES"),
                    status = 'RESPONSE_STATUS',
                    status_codes = c('ER' = "Respondent",
                                     'EN' = "Nonrespondent",
                                     'IE' = "Ineligible",
                                     'UE' = "Unknown"),
                    null_difference = 0.1, alternative = 'greater')

  ## Null Hypothesis: Y_bar_resp - Y_bar_full >= 0.1
  ## Alt. Hypothesis: Y_bar_resp - Y_bar_full <  0.1

t_test_resp_vs_full(survey_design = survey_design,
                    y_vars = c("STUDENT_AGE", "WHETHER_PARENT_AGREES"),
                    status = 'RESPONSE_STATUS',
                    status_codes = c('ER' = "Respondent",
                                     'EN' = "Nonrespondent",
                                     'IE' = "Ineligible",
                                     'UE' = "Unknown"),
                    null_difference = 0.1, alternative = 'less')
```

---

t_test_of_weight_adjustment

t-test of differences in estimated means/percentages from two different sets of replicate weights.

Description

Tests whether estimates of means/percentages differ systematically between two sets of replicate weights: an original set of weights, and the weights after adjustment (e.g. post-stratification or nonresponse adjustments) and possibly subsetting (e.g. subsetting to only include respondents).

Usage

```
t_test_of_weight_adjustment(
  orig_design,
  updated_design,
  y_vars,
  na.rm = TRUE,
  null_difference = 0,
  alternative = "unequal",
  degrees_of_freedom = NULL
)
```

Arguments

`orig_design` A replicate design object created with the `survey` package.

`updated_design` A potentially updated version of `orig_design`, for example where weights have been adjusted for nonresponse or updated using post-stratification. The type and number of sets of replicate weights must match that of `orig_design`. The number of rows may differ (e.g. if `orig_design` includes the full sample but `updated_design` only includes respondents).

`y_vars` Names of dependent variables for tests. For categorical variables, percentages of each category are tested.

`na.rm` Whether to drop cases with missing values for a given dependent variable.

`null_difference` The difference between the two means/percentages under the null hypothesis. Default is `0`.

`alternative` Can be one of the following:

- `'unequal'` (the default): two-sided test of whether difference in means is equal to `null_difference`
- `'less'`: one-sided test of whether difference is less than `null_difference`
- `'greater'`: one-sided test of whether difference is greater than `null_difference`

`degrees_of_freedom` The degrees of freedom to use for the test's reference distribution. Unless specified otherwise, the default is the design degrees of freedom minus one, where the design degrees of freedom are estimated using the `survey` package's `degf` method applied to the 'stacked' design formed by combining `orig_design` and `updated_design`.

Value

A data frame describing the results of the t-tests, one row per dependent variable.

Statistical Details

The t-statistic used for the test has as its numerator the difference in means/percentages between the two samples, minus the `null_difference`. The denominator for the t-statistic is the estimated standard error of the difference in means. Because the two means are based on overlapping groups and thus have correlated sampling errors, special care is taken to estimate the covariance of the two estimates. For designs which use sets of replicate weights for variance estimation, the two means and their difference are estimated using each set of replicate weights; the estimated differences from the sets of replicate weights are then used to estimate sampling error with a formula appropriate to the replication method (JKn, BRR, etc.).

This analysis is not implemented for designs which use linearization methods for variance estimation.
Unless specified otherwise using the `degrees_of_freedom` parameter, the degrees of freedom for the test are set to the design degrees of freedom minus one. Design degrees of freedom are estimated using the `survey` package's `degf` method.

See Van de Kerckhove et al. (2009) for an example of this type of nonresponse bias analysis (among others). See Lohr and Riddles (2016) for the statistical details of this test.

References

- Lohr, S., Riddles, M. (2016). *Tests for Evaluating Nonresponse Bias in Surveys*. Survey Methodology 42(2): 195-218. https://www150.statcan.gc.ca/n1/pub/12-001-x/2016002/article/14677-eng.pdf
- Van de Kerckhove, W., Krenzke, T., and Mohadjer, L. (2009). *Adult Literacy and Lifeskills Survey (ALL) 2003: U.S. Nonresponse Bias Analysis (NCES 2009-063)*. National Center for Education Statistics, Institute of Education Sciences, U.S. Department of Education. Washington, DC.

Examples

```
library(survey)

# Create a survey design ----

data(involvement_survey_srs, package = 'nrba')

survey_design <- svydesign(weights = ~ BASE_WEIGHT,
                           id = ~ UNIQUE_ID,
                           fpc = ~ N_STUDENTS,
                           data = involvement_survey_srs)

# Create replicate weights for the design ----
rep_svy_design <- as.svrepdesign(survey_design, type = "subbootstrap",
                                 replicates = 500)

# Subset to only respondents (always subset *after* creating replicate weights)

rep_svy_respondents <- subset(rep_svy_design,
                              RESPONSE_STATUS == "Respondent")
```

```
# Apply raking adjustment ----

raked_rep_svy_respondents <- rake_to_benchmarks(
  survey_design = rep_svy_respondents,
  group_vars = c("PARENT_HAS_EMAIL", "STUDENT_RACE"),
  group_benchmark_vars = c("PARENT_HAS_EMAIL_BENCHMARK",
                           "STUDENT_RACE_BENCHMARK"),
)

# Compare estimates from respondents in original vs. adjusted design ----

t_test_of_weight_adjustment(orig_design = rep_svy_respondents,
                            updated_design = raked_rep_svy_respondents,
                            y_vars = c('STUDENT_AGE', 'STUDENT_SEX'))

t_test_of_weight_adjustment(orig_design = rep_svy_respondents,
                            updated_design = raked_rep_svy_respondents,
                            y_vars = c('WHETHER_PARENT_AGREES'))

# Compare estimates to true population values ----

data('involvement_survey_pop', package = 'nrba')

mean(involvement_survey_pop$STUDENT_AGE)

prop.table(table(involvement_survey_pop$STUDENT_SEX))
```

`t_test_vs_external_estimate`
t-test of differences in means/percentages relative to external estimates

---

Description

Compare estimated means/percentages from the present survey to external estimates from a benchmark source. A t-test is used to evaluate whether the survey's estimates differ from the external estimates.

Usage

```
t_test_vs_external_estimate(
  survey_design,
  y_var,
  ext_ests,
  ext_std_errors = NULL,
  na.rm = TRUE,
  null_difference = 0,
  alternative = "unequal",
```

```
  degrees_of_freedom = survey::degf(survey_design) - 1
)
```

Arguments

survey_design
: A survey design object created with the survey package.

y_var
: Name of dependent variable. For categorical variables, percentages of each category are tested.

ext_ests
: A numeric vector containing the external estimate of the mean for the dependent variable. If variable is a categorical variable, a named vector of means must be provided.

ext_std_errors
: (Optional) The standard errors of the external estimates. This is useful if the external data are estimated with an appreciable level of uncertainty, for instance if the external data come from a survey with a small-to-moderate sample size. If supplied, the variance of the difference between the survey and external estimates is estimated by adding the variance of the external estimates to the estimated variance of the survey's estimates.

na.rm
: Whether to drop cases with missing values for y_var

null_difference
: The hypothesized difference between the estimate and the external mean. Default is 0.

alternative
: Can be one of the following:

- 'unequal': two-sided test of whether difference in means is equal to null_difference
- 'less': one-sided test of whether difference is less than null_difference
- 'greater': one-sided test of whether difference is greater than null_difference

degrees_of_freedom
: The degrees of freedom to use for the test's reference distribution. Unless specified otherwise, the default is the design degrees of freedom minus one, where the design degrees of freedom are estimated using the survey package's degf method.

Value

A data frame describing the results of the t-tests, one row per mean being compared.

References

See Brick and Bose (2001) for an example of this analysis method and a discussion of its limitations.

- Brick, M., and Bose, J. (2001). *Analysis of Potential Nonresponse Bias*. in Proceedings of the Section on Survey Research Methods. Alexandria, VA: American Statistical Association. http://www.asasrms.org/Proceedings/y2001/Proceed/00021.pdf

Examples

```
library(survey)

# Create a survey design ----
data("involvement_survey_str2s", package = 'nrba')

involvement_survey_sample <- svydesign(
  data = involvement_survey_str2s,
  weights = ~ BASE_WEIGHT,
  strata =  ~ SCHOOL_DISTRICT,
  ids =     ~ SCHOOL_ID             + UNIQUE_ID,
  fpc =     ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL
)

# Subset to only include survey respondents ----

involvement_survey_respondents <- subset(involvement_survey_sample,
                                         RESPONSE_STATUS == "Respondent")

# Test whether percentages of categorical variable differ from benchmark ----

parent_email_benchmark <- c(
  'Has Email' = 0.85,
  'No Email' = 0.15
)

t_test_vs_external_estimate(
  survey_design = involvement_survey_respondents,
  y_var = "PARENT_HAS_EMAIL",
  ext_ests = parent_email_benchmark
)

# Test whether the sample mean differs from the population benchmark ----

average_age_benchmark <- 11

t_test_vs_external_estimate(
  survey_design = involvement_survey_respondents,
  y_var = "STUDENT_AGE",
  ext_ests = average_age_benchmark,
  null_difference = 0
)
```

`wt_class_adjust` *Adjust weights in a replicate design for nonresponse or unknown eligibility status, using weighting classes*

Description

Updates weights in a survey design object to adjust for nonresponse and/or unknown eligibility using the method of weighting class adjustment. For unknown eligibility adjustments, the weight in each class is set to zero for cases with unknown eligibility, and the weight of all other cases in the class is increased so that the total weight is unchanged. For nonresponse adjustments, the weight in each class is set to zero for cases classified as eligible nonrespondents, and the weight of eligible respondent cases in the class is increased so that the total weight is unchanged.

This function currently only works for survey designs with replicate weights, since the linearization-based estimators included in the `survey` package (or Stata or SAS for that matter) are unable to fully reflect the impact of nonresponse adjustment. Adjustments are made to both the full-sample weights and all of the sets of replicate weights.

Usage

```
wt_class_adjust(
  survey_design,
  status,
  status_codes,
  wt_class = NULL,
  type = c("UE", "NR")
)
```

Arguments

`survey_design` A replicate survey design object created with the `survey` package.

`status` A character string giving the name of the variable representing response/eligibility status.
The `status` variable should have at most four categories, representing eligible respondents (ER), eligible nonrespondents (EN), known ineligible cases (IE), and cases whose eligibility is unknown (UE).

`status_codes` A named vector, with four entries named 'ER', 'EN', 'IE', and 'UE'.
`status_codes` indicates how the values of the `status` variable are to be interpreted.

`wt_class` (Optional) A character string giving the name of the variable which divides sample cases into weighting classes.
If `wt_class=NULL` (the default), adjustment is done using the entire sample.

`type` A character vector including one or more of the following options:

- `'UE'`: Adjust for unknown eligibility.
- `'NR'`: Adjust for nonresponse.
  To sequentially adjust for unknown eligibility and then nonresponse, set `type=c('UE', 'NR')`.

Details

See the vignette "Nonresponse Adjustments" from the svrep package for a step-by-step walkthrough of nonresponse weighting adjustments in R:
`vignette(topic = "nonresponse-adjustments", package = "svrep")`

Value

A replicate survey design object, with adjusted full-sample and replicate weights

References

See Chapter 2 of Heeringa, West, and Berglund (2017) or Chapter 13 of Valliant, Dever, and Kreuter (2018) for an overview of nonresponse adjustment methods based on redistributing weights.

- Heeringa, S., West, B., Berglund, P. (2017). Applied Survey Data Analysis, 2nd edition. Boca Raton, FL: CRC Press. "Applied Survey Data Analysis, 2nd edition." Boca Raton, FL: CRC Press.
- Valliant, R., Dever, J., Kreuter, F. (2018). "Practical Tools for Designing and Weighting Survey Samples, 2nd edition." New York: Springer.

See Also

`svrep::redistribute_weights()`, `vignette(topic = "nonresponse-adjustments", package = "svrep")`

Examples

```
library(survey)
# Load an example dataset
data("involvement_survey_str2s", package = "nrba")

# Create a survey design object

involvement_survey_sample <- svydesign(
  data = involvement_survey_str2s,
  weights = ~BASE_WEIGHT,
  strata = ~SCHOOL_DISTRICT,
  ids = ~ SCHOOL_ID + UNIQUE_ID,
  fpc = ~ N_SCHOOLS_IN_DISTRICT + N_STUDENTS_IN_SCHOOL
)

rep_design <- as.svrepdesign(involvement_survey_sample, type = "mrbbootstrap")

# Adjust weights for nonresponse within weighting classes
nr_adjusted_design <- wt_class_adjust(
  survey_design = rep_design,
  status = "RESPONSE_STATUS",
  status_codes = c(
    "ER" = "Respondent",
    "EN" = "Nonrespondent",
    "IE" = "Ineligible",
```

```
    "UE" = "Unknown"
  ),
  wt_class = "PARENT_HAS_EMAIL",
  type = "NR"
)
```

Index