Package ‘Rrepest’

November 24, 2023

Title  An Analyzer of International Large Scale Assessments in Education

Version 1.3.0

Description A fast way to analyze International Large-Scale Assessments (ILSAs) or any other dataset that includes replicated weights (Balanced Repeated Replication (BRR) weights, Jackknife replicate weights,...) and/or plausible values.

'Rrepest' contains functionalities that enable you to calculate basic statistics (means, correlations, etc.), frequencies, linear regression, or any other model already implemented in R that takes a data frame and weights as parameters. It also includes options to prepare the results for publication, following the table formatting standards of the Organization for Economic Cooperation and Development (OECD).

Depends R (>= 4.2.0)

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports data.table (>= 1.14.8), doParallel (>= 1.0.17), dplyr (>= 1.1.2), flextable (>= 0.7.2), foreach (>= 1.5.2), labelled (>= 2.9.1), magrittr (>= 2.0.3), officer (>= 0.6.2), parallel (>= 4.2.1), purrr (>= 0.3.4), stringr (>= 1.5.0), tibble (>= 3.1.8), tidyr (>= 1.2.0)

LazyData true

NeedsCompilation no

Author Rodolfo Ilizaliturri [aut, cre], Francesco Avvisati [aut], Francois Keslair [aut]

Maintainer Rodolfo Ilizaliturri <rodolfo.ilizaliturri@oecd.org>

Repository CRAN

Date/Publication 2023-11-24 22:30:07 UTC
R topics documented:

- `df_pisa18` ................................................................. 2
- `df_talis18` ................................................................. 2
- `est` .............................................................................. 3
- `grouped_sum_freqs` ...................................................... 3
- `grp` .............................................................................. 4
- `inv_test` ................................................................. 5
- `n_obs_x` ................................................................. 5
- `Rrepest` ................................................................. 6
- `weighted.corr` ............................................................. 7
- `weighted.corr.cov.n` .................................................. 8
- `weighted.cov` ............................................................. 9
- `weighted.iqr` .............................................................. 10
- `weighted.quant` .......................................................... 10
- `weighted.std` ............................................................. 11
- `weighted.var` ............................................................. 11

Index 13

---

**df_pisa18**

*PISA 2018 Student questionnaire Database*

**Description**

This dataset is a subset of the PISA 2018 database produced by the OECD for the countries of France, Italy, and Mexico.

**Usage**

```r
data(df_pisa18)
```

**Format**

A data frame with 1269 rows and 1120 variables

---

**df_talis18**

*TALIS 2018 Teachers Database*

**Description**

This dataset is a subset of the TALIS 2018 database produced by the OECD for the countries of France, Italy, and Mexico.

**Usage**

```r
data(df_talis18)
```
**est**

**Estimate list**

**Description**

Input the statistic wanted, target variable, and (optional) list of regressors

**Usage**

\[\text{est(statistic, target, regressor = NULL)}\]

**Arguments**

- **statistic** (string vector) accepts "mean","var","std", "quant", "iqr", "freq", "lm", "corr", "cov"
- **target** (string vector) variable from where to get estimation
- **regressor** (string vector) independent variable for regression (1+)

**Value**

list of components to estimate for repest

**Examples**

\[\text{est(c("mean","quant",.5,"corr"),c("pv1math","pv1read","Pv1SCIE"))}\]

---

**grouped_sum_freqs**

**Grouped Frequencies**

**Description**

Compute a DataFrame with frequency counts obtained from the sum of 'small.level' and 'big.level' after grouping, which can be used to calculate percentages.

**Usage**

\[\text{grouped_sum_freqs(data, small.level, big.level, w = NULL)}\]
Arguments

data (dataframe) Data to analyze
small.level (string vector) All variables to get grouped sum that will sum up to 100
big.level (string vector) Must be fully contained in variables from small.level
w (string) Numeric variable from which to get weights (optional)

Value

Dataframe with frequencies from the grouped sum of small.level and big.level used for getting percentages

Examples

grouped_sum_freqs(data = mtcars, small.level = c("cyl", "am"), big.level = c("cyl"))

Description

Obtain a list as argument for groups to be evaluated in data

Usage

grp(group.name, column, cases)

Arguments

group.name (string) Name of the group to be displayed
column (string) Column where the data is located
cases (string vector) List of values to be put into the group

Value

list of groups to redefine group_name = column, values_in_group

Examples

append(grp("OECD Average", "CNTRY", c("HUN", "MEX")), grp("Europe", "CNTRY", c("ITA", "FRA")))
Description

Invert test column from Rrepest test = TRUE by name on "b." and "se." in the column name and by sign (*-1) on "b."

Usage

inv_test(data, name_index)

Arguments

data (dataframe) df to analyze
name_index (string/numeric) name or index for the estimate (b.) columns containing the data for the test in Rrepest

Value

Dataframe containing inverted test column names for "b." and "se." according to Rrepest structure and column multiplied by (-1) for "b."

n_obs_x

Number of observations valid for column x

Description

Number of observations valid for column x

Usage

n_obs_x(df, by, x, svy = NULL)

Arguments

df (dataframe) data to analyze
by (string vector) column by which we’ll break down results
x (string) variable from where to get means
svy (string) Possible projects to analyse.must be equal to ALL, IALS, IELS, PIAAC, PISA, PISA2015, PISAOOS, TALISSCH, TALISTCH

Value

Dataframe containing the number of observations valid for the target variable x
Examples

data(df_pisa18)
data(df_talis18)

n_obs_x(df = df_pisa18, by = "cnt", x = "wb173q03ha", svy = "PISA2015")
n_obs_x(df = df_talis18, by = "cntry", x = "tt3g01", svy = "TALISTCH")

Description

Estimates statistics using replicate weights (Balanced Repeated Replication (BRR) weights, Jackknife replicate weights,...), thus accounting for complex survey designs in the estimation of sampling variances. It is specially designed to be used with the data sets produced by the Organization for Economic Cooperation and Development (OECD), some of which include the Programme for International Student Assessment (PISA) and Teaching and Learning International Survey (TALIS) data sets, but works for all International Large Scale Assessments that use replicated weights. It also allows for analyses with multiply imputed variables (plausible values); where plausible values are included in a pvvarlist, the average estimator across plausible values is reported and the imputation error is added to the variance estimator.

Usage

Rrepest(
  data,
  svy,
  est,
  by = NULL,
  over = NULL,
  test = FALSE,
  user_na = FALSE,
  show_na = FALSE,
  flag = FALSE,
  fast = FALSE,
  tabl = FALSE,
  average = NULL,
  group = NULL,
  ...
)

Arguments

data (dataframe) df to analyze
svy (string) Possible projects to analyse.must be equal to ALL, IALS, IELS, PIAAC, PISA, PISA2015, PISAOOS, TALISSCH, TALISTCH .
weighted.corr

**est**
(est function) that takes arguments stimate, target variable, regressor (optional for linear regressions)

**by**
(string vector) column in which we’ll break down results

**over**
(vector string) columns over which to do analysis

**test**
(bool) TRUE: will calculate the difference between over variables

**user_na**
(bool) TRUE: show nature of user defined missing values for by.var

**show_na**
(bool) TRUE: include na in frequencies of x

**flag**
(bool) TRUE: Show NaN when there is not enough cases (or schools)

**fast**
(bool) TRUE: Only do 6 replicated weights

**tabl**
(bool) TRUE: Creates a flextable with all examples

**average**
(grp function) that takes arguments group.name, column, cases to create averages at the end of df

**group**
(grp function) that takes arguments group.name, column, cases to create groups at the end of df

... Optional filtering parameters: i.e.: isced = 2, n.pvs = 5, cm.weights = c("finw",paste0("repw",1:22))

var.factor = 1/(0.5^2) z.score = qnorm(1-0.05/2)

**Value**

Dataframe containing estimation "b." and standard error "se." of desired processes

**Examples**

```r
data(df_pisa18)

grepest(data = df_pisa18,
svy = "PISA2015",
est = est("mean","AGE"),
by = c("CNT"))
```

---

**weighted.corr** *Weighted Bivariate Correlation*

**Description**

Compute weighted pearson correlation coefficient of two numeric vectors

**Usage**

```r
weighted.corr(x, y, w, na.rm = TRUE)
```
**weighted.corr.cov.n**

**Arguments**
- **x** (numeric vector) variable from where to get correlation
- **y** (numeric vector) variable from where to get correlation
- **w** (numeric vector) vector of weights
- **na.rm** (bool) True: NAs be stripped before computation proceeds

**Value**
Pearson correlation coefficient

**Examples**
```r
data(df_talis18)

weighted.corr(x = df_talis18$T3STAKE, y = df_talis18$T3TEAM, w = df_talis18$TCHWGT)
```

**Description**
Multivariate Correlation and Covariance

**Usage**
```r
weighted.corr.cov.n(
  data,
  x,
  w = rep(1, length(data[x[1]])),
  corr = TRUE,
  na.rm = TRUE
)
```

**Arguments**
- **data** (dataframe) data to analyze
- **x** (vector string) variables names from where to get correlation/covariance
- **w** (string) weight name
- **corr** (bool) True: get correlation. False: get covariance
- **na.rm** (bool) True: NAs be stripped before computation proceeds

**Value**
Dataframe containing 2 Choose length(x) columns with each bivariate correlation/covariance
**weighted.cov**

**Examples**

```r
data(df_talis18)
weighted.corr.cov.n(df_talis18,c("T3STAKE","T3TEAM","T3STUD"),"TCHWGT")
```

---

**weighted.cov**

**Weighted Bivariate Covariance**

**Description**

Compute weighted covariance coefficient of two numeric vectors

**Usage**

```r
weighted.cov(x, y, w, na.rm = TRUE)
```

**Arguments**

- `x` (numeric vector) variable from where to get covariance
- `y` (numeric vector) variable from where to get covariance
- `w` (numeric vector) vector of weights
- `na.rm` (bool) True: NAs be stripped before computation proceeds

**Value**

Pearson correlation coefficient

**Examples**

```r
data(df_talis18)
weighted.cov(x = df_talis18$T3STAKE, y = df_talis18$T3TEAM, w = df_talis18$TCHWGT)
```
### weighted.iqr  
**Weighted Interquantile Range**

**Description**

Compute interquantile range

**Usage**

```r
weighted.iqr(x, w = rep(1, length(x)), rang = c(0.25, 0.75))
```

**Arguments**

- `x` (numeric vector) variable from where to get quantiles
- `w` (numeric vector) vector of weights
- `rang` (numeric vector) two numbers indicating the range of the quantiles

**Value**

Interquantile range

**Examples**

```r
weighted.iqr(x = mtcars$mpg, w = mtcars$wt, rang = c(0.5, 0.9))
```

### weighted.quant  
**Weighted Quantile**

**Description**

Computation of weighted quantiles

**Usage**

```r
weighted.quant(x, w = rep(1, length(x)), q = 0.5)
```

**Arguments**

- `x` (numeric vector) variable from where to get quantiles
- `w` (numeric vector) vector of weights
- `q` (numeric vector) From 0 to 1 (exclusive) for the quantile desired

**Value**

Weighted quantile of a numeric vector
**Examples**

```r
weighted.quant(x = mtcars$mpg, w = mtcars$wt, q = seq(.1,.9,.1))
```

---

**weighted.std**  
Weighted Standard Deviation

**Description**

Calculate the standard deviation of a numeric vector.

**Usage**

```r
weighted.std(x, w, na.rm = TRUE)
```

**Arguments**

- `x` (numeric vector) variable to analyze
- `w` (numeric vector) vector of weights
- `na.rm` (bool) if TRUE remove missing values.

**Value**

Scalar with Variance or Standard Deviation.

**Examples**

```r
data(df_talis18)
weighted.std(df_talis18$TT3G02, df_talis18$TRWGT1)
```

---

**weighted.var**  
Weighted variance

**Description**

Calculate the weighted variance numeric vector.

**Usage**

```r
weighted.var(x, w, na.rm = TRUE)
```
Arguments

- \textbf{x} \quad \text{(numeric vector) variable to analyze}
- \textbf{w} \quad \text{(numeric vector) vector of weights}
- \textbf{na.rm} \quad \text{(bool) if TRUE remove missing values.}

Value

Scalar with Variance or Standard Deviation

Examples

```r
data(df_talis18)

weighted.var(df_talis18$TT3G02, df_talis18$TRWGT1)
```
Index

* datasets
  df_pisa18, 2
  df_talis18, 2

df_pisa18, 2
df_talis18, 2

est, 3

grouped_sum_freqs, 3
grp, 4

inv_test, 5

n_obs_x, 5

Rrepest, 6

weighted.corr, 7
weighted.corr.cov.n, 8
weighted.cov, 9
weighted.iqr, 10
weighted.quant, 10
weighted.std, 11
weighted.var, 11