Package ‘DirectStandardisation’

July 21, 2025

Type Package

Title Adjusted Means and Proportions by Direct Standardisation

Version 1.3

Date 2020-03-15

Author Christiana Kartsonaki

Maintainer Christiana Kartsonaki <christiana.kartsonaki@gmail.com>

Description Calculate adjusted means and proportions of a variable by groups defined by another variable by direct standardisation, standardised to the structure of the dataset.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2020-03-16 05:10:02 UTC

Contents

`DirectStandardisation-package`
Adjusted Means and Proportions by Direct Standardisation

Description

Adjusted Means and Proportions by Direct Standardisation

Details

Calculates adjusted means and proportions of a variable by groups defined by another variable, standardised to the structure of the adjustment variables in the whole dataset using direct standardisation.

Index of help topics:

```
DirectStandardisation-package
                        Adjusted Means and Proportions by Direct
                        Standardisation
adjmeans                Calculate adjusted means using direct
                        standardisation
adjprop                 Calculate adjusted proportions using direct
                        standardisation
```

Author(s)

Christiana Kartsonaki

Maintainer: Christiana Kartsonaki <christiana.kartsonaki@gmail.com>

---

adjmeans *Calculate adjusted means using direct standardisation*

Description

Calculates adjusted means of a variable by groups defined by another variable using direct standardisation to the structure of the dataset, as defined by one or more variables.

Usage

```
adjmeans(dataset, outcome_vars, categorical_vars, outcome_var_labels = NULL,
categorical_var_labels = NULL, adjustment_vars = c("age", "sex"),
adjustment_var_labels = NULL, format_table = FALSE,
transpose_table = FALSE, ndigits = 2, title = "")
```

Arguments

`dataset`	A dataframe containing all variables to be used.
`outcome_vars`	The names of the outcome variables in `dataset` (character vector). These are the variables of which adjusted means will be calculated.
`categorical_vars`	A character vector containing the names of categorical variables which define the groups by which adjusted means will be calculated. They must exist in `dataset`.

`outcome_var_labels`	Labels for the outcome variable to be printed in the table produced. This must be a list of length equal to the number of outcome variables, with each element a list of length two, each element of which is a character with a label. Note that if there is only one variable it should be `list(list("Label 1", "Label 2"))`. Defaults to `outcome_vars`.
`categorical_var_labels`	Labels for the categorical variables by which means will be calculated, to be printed in the table produced. This must be a list of length equal to the number of variables by which adjusted means will be calculated, with each element a list of length two, the first element of which is a character with a label for the variable and the second element a character vector with labels for the levels of the variable. For example for two variables, the first of which has 3 levels and the second 2, `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")), list("Variable 2", c("Group 1", "Group 2")))`. Note that if there is only one variable it should be `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")))`. If null, the levels of the variable are used.
`adjustment_vars`	A character vector containing the variable names of categorical variables to be adjusted for. The default is `age` and `sex`, which standardises means of the subgroups to the age and sex structure of the overall dataset.
`adjustment_var_labels`	A character vector with labels for the variables adjusted for, to be printed in the table produced. If null, the variable names are used.
`format_table`	Whether the output table should be formatted. Defaults to `FALSE`.
`transpose_table`	Whether the output table should be transposed. Defaults to `FALSE`.
`ndigits`	Number of digits to be printed (defaults to 2).
`title`	A title for the table (defaults to blank).

Details

The function produces a table of means of some outcome variable by one or more categorical variables using direct standardisation with target population a population with proportions within each group specified by some variables (default is age and sex) identical for all categories of the categorical variable and equal to the overall proportion in the data.

Value

A data frame of adjusted means with categorical variables defining the groupings as rows and outcome categories as columns.

Author(s)

Christiana Kartsonaki <christiana.kartsonaki@gmail.com>

See Also

`adjprop`

Examples

```
# Example 1

# generate a dataframe with sleep duration, sex and age group
data <- data.frame("sleep" = rnorm(50, mean = 8, sd = 1.5),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

adjmeans(dataset = data, outcome_vars = "sleep",
categorical_vars = "sex", outcome_var_label = "Sleep duration",
categorical_var_labels = list(list("Sex", c("Female", "Male"))),
adjustment_vars = "age_group", adjustment_var_labels = "age",
title = "Means of sleep duration by sex.")

# Example 2

# generate a dataframe with sleep duration, sex and age group
data <- data.frame("sleep" = rnorm(50, mean = 8, sd = 1.5),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

# no labels, more digits
adjmeans(dataset = data, outcome_vars = "sleep",
categorical_vars = "sex", adjustment_vars = "age_group",
ndigits = 4, title = "Means of sleep duration by sex.")
```

---

adjprop *Calculate adjusted proportions using direct standardisation*

Description

Calculates adjusted proportions of a variable by groups defined by another variable using direct standardisation to the structure of the dataset, as defined by one or more variables.

Usage

```
adjprop(dataset, outcome_vars, categorical_vars, outcome_var_labels = NULL,
categorical_var_labels = NULL, adjustment_vars = c("age", "sex"),
adjustment_var_labels = NULL, format_table = FALSE,
transpose_table = FALSE, percentage = FALSE, ndigits = 2, title = "")
```

Arguments

`dataset`	A data frame containing all variables to be used.
`outcome_vars`	The names of the outcome variables in `dataset` (character vector). These are the variables of which adjusted proportions will be calculated.

`categorical_vars`	A character vector containing the names of categorical variables which define the groups by which adjusted proportions will be calculated. They must exist in `dataset`.
`outcome_var_labels`	Labels for the outcome variable to be printed in the table produced. This must be a list of length equal to the number of outcome variables, with each element a list of length two, the first element of which is a character with a label for the variable and the second element a character vector with labels for the levels of the variable. For example for two variables, the first of which has 3 levels and the second 2, `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")), list("Variable 2", c("Group 1", "Group 2")))`. Note that if there is only one variable it should be `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")))`. If null, the names and levels of the variables are used.
`categorical_var_labels`	Labels for the categorical variables by which proportions will be calculated, to be printed in the table produced. This must be a list of length equal to the number of variables by which adjusted proportions will be calculated, with each element a list of length two, the first element of which is a character with a label for the variable and the second element a character vector with labels for the levels of the variable. For example for two variables, the first of which has 3 levels and the second 2, `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")), list("Variable 2", c("Group 1", "Group 2")))`. Note that if there is only one variable it should be `list(list("Variable 1", c("Group 1", "Group 2", "Group 3")))`. If null, the levels of the variable are used.
`adjustment_vars`	A character vector containing the names of categorical variables to be adjusted for. The default is `age` and `sex`, which standardises proportions of the subgroups to the age and sex structure of the overall dataset.
`adjustment_var_labels`	A character vector with labels for the variables adjusted for, to be printed in the table produced. If null, the variable names are used.
`format_table`	Whether the output table should be formatted. Defaults to `FALSE`.
`transpose_table`	Whether the output table should be transposed. Defaults to `FALSE`.
`ndigits`	Number of digits to be printed (defaults to 2).
`percentage`	If `TRUE`, percentages instead of proportions are calculated. The default is `FALSE`.
`title`	A title for the table (defaults to blank).

Details

The function produces a table of proportions of some outcome variable by one or more categorical variables using direct standardisation with target population a population with proportions within each group specified by some variables (default is age and sex) identical for all categories of the categorical variable and equal to the overall proportion in the data.

Value

A data frame of adjusted proportions with categorical variables defining the groupings as rows and outcome categories as columns.

Author(s)

Christiana Kartsonaki <christiana.kartsonaki@gmail.com>

See Also

adjmeans

Examples

```
# Example 1

# generate a dataframe with sleep deprivation (binary), sex and age group
data <- data.frame("sleep_deprivation" = rbinom(50, size = 1, prob = 0.5),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

adjprop(dataset = data, outcome_vars = "sleep_deprivation",
categorical_vars = "sex",
outcome_var_labels = list(list("Sleep deprivation", levels(as.factor(data$sleep_deprivation)))),
categorical_var_labels = list(list("Sex", c("Female", "Male"))),
adjustment_vars = "age_group", adjustment_var_labels = "age",
title = "Proportions of sleep deprivation by sex.")

# Example 2

# generate a dataframe with sleep duration group (3 categories), sex and age group
data <- data.frame("sleep_group" = rbinom(500, size = 2, prob = 0.3),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

adjprop(dataset = data, outcome_vars = "sleep_group",
categorical_vars = "sex", outcome_var_labels = list(list("Sleep duration group",
c("Group 1", "Group 2", "Group 3"))),
categorical_var_labels = list(list("Sex", c("Female", "Male"))),
adjustment_vars = "age_group", adjustment_var_labels = "age",
title = "Proportions of sleep duration groups by sex.")

# Example 3

# generate a dataframe with sleep duration group (3 categories), sex and age group
data <- data.frame("sleep_group" = rbinom(500, size = 2, prob = 0.3),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

# no labels, more digits
adjprop(dataset = data, outcome_vars = "sleep_group",
categorical_vars = "sex", adjustment_vars = "age_group",
adjustment_var_labels = "age", ndigits = 4,
title = "Proportions of sleep duration groups by sex.")

# Example 4

# generate a dataframe with sleep duration group (4 categories), sex and age group
data <- data.frame("sleep_group" = rbinom(500, size = 3, prob = 0.25),
"sex" = c(rep("m", 25), rep("f", 25)),
"age_group" = rep(c("20-29", "30-39", "40-49", "50-59", "60-69"), 5))

# no labels, proportions
adjprop(dataset = data, outcome_vars = "sleep_group",
categorical_vars = "sex", adjustment_vars = "age_group",
adjustment_var_labels = "age", title = "Proportions of sleep duration groups by sex.")

# no labels, percentages
adjprop(dataset = data, outcome_vars = "sleep_group",
categorical_vars = "sex", adjustment_vars = "age_group",
adjustment_var_labels = "age", percentage = TRUE,
title = "Proportions of sleep duration groups by sex.")
```

Index