Package 'tab'

September 20, 2016

Type Package

Title Functions for Creating Summary Tables for Statistical Reports

Version 3.1.2

Date 2016-09-16

Author Dane R. Van Domelen

Maintainer Dane R. Van Domelen <vandomed@gmail.com>

Description

Contains functions for generating tables for statistical reports written in Microsoft Word or LaTeX. There are functions for I-by-J frequency tables, comparison of means or medians across levels of a categorical variable, and summarizing fitted generalized linear models, generalized estimating equations, and Cox proportional hazards regression. Functions are available to handle data simple random samples or survey data. The package is intended to make it easier for researchers to translate results from statistical analyses in R to their reports or manuscripts.

License GPL-2

Depends R (>= 2.10)

Imports survey, survival, gee, grDevices, graphics, stats, xtable

Repository CRAN

Repository/R-Forge/Project tab

Repository/R-Forge/Revision 79

Repository/R-Forge/DateTimeStamp 2016-09-17 01:20:49

Date/Publication 2016-09-20 18:46:46

NeedsCompilation no

R topics documented:

---

tab-package *Functions for Creating Summary Tables for Statistical Reports*

Description

Contains functions for generating tables for statistical reports written in Microsoft Word or LaTeX. There are functions for I-by-J frequency tables, comparison of means or medians across levels of a categorical variable, and summarizing fitted generalized linear models, generalized estimating equations, and Cox proportional hazards regression. Functions are available to handle data simple random samples or survey data. The package is intended to make it easier for researchers to translate results from statistical analyses in R to their reports or manuscripts.

Details

Package:	tab
Type:	Package
Version:	3.1.2
Date:	2016-09-16
License:	GPL-2

The following functions are included:

`tabfreq`, `tabmeans`, `tabmedians`, `tabmulti`, `tabglm`, `tabcox`, `tabgee`, `tabfreq.svy`, `tabmeans.svy`, `tabmedians.svy`, `tabmulti.svy`, `tabglm.svy`, `formatp`

Author(s)

Dane R. Van Domelen

Maintainer: Dane R. Van Domelen <vandomed@gmail.com>

References

1. Therneau T (2013). A Package for Survival Analysis in S. R package version 2.37-4, https://cran.r-project.org/package=survival.

2. Terry M. Therneau and Patricia M. Grambsch (2000). Modeling Survival Data: Extending the Cox Model. Springer, New York. ISBN 0-387-98784-3.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

NA

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Compare race distribution by group, with group as column variable
freqtable <- tabfreq(x = d$Group, y = d$Race)

# Compare mean BMI in control group vs. treatment group
meanstable <- tabmeans(x = d$Group, y = d$BMI)

# Generate plot comparing mean BMI in control group vs. treatment group
meansfig <- tabmeans(x = d$Group, y = d$BMI, fig = TRUE)

# Compare median BMI in control group vs. treatment group
medianstable <- tabmedians(x = d$Group, y = d$BMI)

# Create a typical Table 1 for statistical report or manuscript
table1 <- tabmulti(dataset = d, xvarname = "Group",
                   yvarnames = c("Age", "Sex", "Race", "BMI"))

# Create vector of race labels for use in regression tables
races <- c("White", "Black", "Mexican American", "other")

# Test whether age, sex, race, and treatment group are associated with BMI
glmfit1 <- glm(BMI ~ Age + Sex + Race + Group, data = d)
lintable <- tabglm(glmfit = glmfit1,
                   xlabels = c("Intercept", "Age", "Male", "Race", races, "Treatment"))

# Test whether age, sex, race, and treatment group are associated with 1-year mortality
glmfit2 <- glm(death_1yr ~ Age + Sex + Race + Group, data = d, family = binomial)
logtable <- tabglm(glmfit = glmfit2, ci.beta = FALSE,
                   xlabels = c("Intercept", "Age", "Male", "Race", races, "Treatment"))

# Test whether age, sex, race, and treatment group are associated with survival
coxtable <- tabcox(x = d[,c("Age", "Sex", "Race", "Group")], time = d$time,
                   delta = d$delta,
                   xlabels = c("Age", "Male", "Race", races, "Treatment"))

# Click on freqtable, meanstable, table1, lintable, logtable, or coxtable in
```

```
# the Workspace tab of RStudio to see the tables that could be copied and pasted
# into a Word document. With newer versions of RStudio, it works better to
# set the print.html input to TRUE, and then copy the table from the .html file
# that prints to your current working directory. Alternatively, setting the latex
# input to TRUE produces tables that can be inserted into LaTeX using the xtable
# package.
```

---

d *Example Dataset for tab Package*

Description

This dataset is used to illustrate the various functions in the R package tab.

Usage

```
data(d)
```

Format

A data frame with 300 observations on the following 15 variables.

`ID` a numeric vector

`Group` a factor

`Age` a numeric vector

`Sex` a factor

`Race` a factor

`BMI` a numeric vector

`time` a numeric vector

`delta` a numeric vector

`death_1yr` a numeric vector

`bp.1` a numeric vector

`bp.2` a numeric vector

`bp.3` a numeric vector

`highbp.1` a numeric vector

`highbp.2` a numeric vector

`highbp.3` a numeric vector

Details

NA

Source

This dataset was generated in R.

References

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

Examples

```
NA
```

`formatp` *Format P-values for Functions in the tab Package*

Description

This function formats p-values for tables generated by the functions in the package tab. It handles rounding and presentation of p-values.

Usage

```
formatp(p, decimals = c(2, 3), cuts = 0.01, lowerbound = 0.001,
        leading0 = TRUE, avoid1 = FALSE)
```

Arguments

`p`	Numeric value or vector of p-values.
`decimals`	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See cuts.
`cuts`	Cut-point(s) to control number of decimal places used for p-values. For example, by default cuts is 0.1 and decimals is c(2,3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
`lowerbound`	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
`leading0`	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
`avoid1`	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for decimals and cuts).

Details

NA

Value

Character value or vector.

Note

NA

Author(s)

Dane R. Van Domelen

References

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

NA

Examples

```
# Generate vector of numeric p-values
set.seed(123)
p <- c(runif(n = 5, min = 0, max = 1), 1, 0, 4e-7, 0.009)

# Round to nearest 2 decimals for p in (0.01, 1] and 3 decimals for p < 0.01
pvals <- formatp(p = p)

# Use 2 decimal places, a lower bound of 0.01, and omit the leading 0.
pvals <- formatp(p = p, decimals = 2, lowerbound = 0.01, leading0 = FALSE)
```

---

tabcox *Generate Summary Tables of Fitted Cox Proportional Hazards Regression Models for Statistical Reports*

Description

This function performs Cox proportional hazards regression using the R package survival [1, 2] and summarizes the results in a clean table for a statistical report.

Usage

```
tabcox(x, time, delta, latex = FALSE, xlabels = NULL, cluster = NULL, robust.se = TRUE,
       decimals = 2, p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001,
      p.leading0 = TRUE, p.avoid1 = FALSE, n = FALSE, events = FALSE, coef = "n",
       greek.beta = FALSE, binary.compress = TRUE, bold.colnames = TRUE,
      bold.varnames = FALSE, bold.varlevels = FALSE, predictor.colname = "Variable",
       suppress.beta = FALSE, print.html = FALSE, html.filename = "table1.html")
```

Arguments

x	For single predictor, vector of values; for multiple predictors, data frame or matrix with one column per predictor. Categorical variables should be of class "factor."
time	Numeric values for time to event or censoring.
delta	Indicator variable where 1 = event observed, 0 = censored.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlabels	Optional character vector to label the x variables and their levels. If unspecified, the function uses generic labels.
cluster	Optional vector indicating clusters of subjects.
robust.se	Only a valid option if clusters are specified. In that case, setting to TRUE requests robust sandwich method standard errors, while setting to FALSE requests normal standard errors.
decimals	Number of decimal places for the regression coefficients, standard errors, hazard ratios, and confidence intervals.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
n	If TRUE, the table returned will include a column for sample size.
events	If TRUE, the table returned will include a column for number of events observed (i.e. uncensored observations).
coef	If set to "x", function will standardize all variables in x that are continuous, providing standardized regression coefficients. Then, the interpretation of each hazard ratio is the hazard ratio associated with a one standard deviation increase in the predictor.
greek.beta	If TRUE, column headings refer to regression parameters as Greek letter beta rather than Beta. Only used when latex input is set to TRUE.
binary.compress	If TRUE, only one row of the table is dedicated to parameter estimates for each binary factor predictor. If FALSE, the table displays separate rows for the variable name and the two levels for each binary factor predictor, much like the presentation for factor variables with more than two levels.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable names in the first column of the table are printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of each factor variable are printed in bold font. Only applies if latex = TRUE and there is at least one factor variable included as a predictor.

`predictor.colname` Character string with desired column heading for the column of predictors.

`suppress.beta` If FALSE, the Beta (SE) column is not included. May often be preferred since the point and interval estimate for the hazard ratio contains the same information, but are easier to interpret.

`print.html` If TRUE, function prints a .html file to the current working directory.

`html.filename` Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

NA

Value

A character matrix with the results of the Cox PH regression. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

A function is currently being developed to create a summary table based on an object from coxph rather than the data vectors themselves (similarly to how tabglm works).

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Therneau T (2013). A Package for Survival Analysis in S. R package version 2.37-4, https://cran.r-project.org/package=survival.

2. Terry M. Therneau and Patricia M. Grambsch (2000). Modeling Survival Data: Extending the Cox Model. Springer, New York. ISBN 0-387-98784-3.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

coxph, tabfreq, tabmeans, tabmedians, tabmulti, tabglm, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Create labels for race levels
races <- c("White", "Black", "Mexican American", "Other")

# Test whether race is associated with survival
coxtable1 <- tabcox(x = d$Race, time = d$time, delta = d$delta,
                    xlabels = c("Race", races))

# Test whether age, sex, race, and treatment group are associated with survival
coxtable2 <- tabcox(x = d[,c("Age", "Sex", "Race", "Group")], time = d$time,
                    delta = d$delta,
                    xlabels = c("Age", "Male", "Race", races, "Treatment"))

# Click on coxtable1 or coxtable2 in the Workspace tab of RStudio to see the tables
# that could be copied and pasted into a report or manuscript. Alternatively, setting
# the latex input to TRUE produces tables that can be inserted into LaTeX using the
# xtable package.
```

---

tabfreq *Generate Frequency Tables for Statistical Reports*

Description

This function creates an I-by-J frequency table and summarizes the results in a clean table for a statistical report.

Usage

```
tabfreq(x, y, latex = FALSE, xlevels = NULL, yname = NULL, ylevels = NULL,
        quantiles = NULL, quantile.vals = FALSE, cell = "n", parenth = NULL,
        text.label = NULL, parenth.sep = "-", test = "chi", decimals = NULL,
      p.include = TRUE, p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001,
      p.leading0 = TRUE, p.avoid1 = FALSE, overall.column = TRUE, n.column = FALSE,
      n.headings = TRUE, compress = FALSE, compress.val = NULL, bold.colnames = TRUE,
      bold.varnames = FALSE, bold.varlevels = FALSE, variable.colname = "Variable",
        print.html = FALSE, html.filename = "table1.html")
```

Arguments

x	Vector of values indicating group membership for columns of IxJ table.
y	Vector of values indicating group membership for rows of IxJ table.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [1]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlevels	Optional character vector to label the levels of x, used in the column headings. If unspecified, the function uses the values that x takes on.
yname	Optional label for the y (row) variable. If unspecified, variable name of y is used.
ylevels	Optional character vector to label the levels of y. If unspecified, the function uses the values that y takes on. Note that levels of y will be listed in the order that they appear when you run table(y, x).
quantiles	If specified, function compares distribution of the y variable across quantiles of the x variable. For example, if x contains continuous BMI values and y is race, setting quantiles to 3 would result in the distribution of race being compared across tertiles of BMI.
quantile.vals	If TRUE, labels for x show quantile number and corresponding range of the x variable. For example, Q1 [0.00, 0.25). If FALSE, labels for quantiles just show quantile number (e.g. Q1). Only used if xlevels is not specified.
cell	Controls what value is placed in each cell of the table. Possible choices are "n" for counts, "tot.percent" for table percentage, "col.percent" for column percentage, "row.percent" for row percentage, "tot.prop" for table proportion, "col.prop" for column proportion, "row.prop" for row proportion, "n/totn" for count/total counts, "n/coln" for count/column count, and "n/rown" for count/row count.
parenth	Controls what values (if any) are placed in parentheses after the values in each cell. By default, if cell is "n", "n/totn", "n/coln", or "n/rown" then the corresponding percentage is shown in parentheses; if cell is "tot.percent", "col.percent", "row.percent", "tot.prop", "col.prop", or "row.prop" then a 95% confidence interval for the requested percentage of proportion is shown in parentheses. Possible values are "none", "se" (for standard error of requested percentage or proportion based on cell), "ci" (for 95% confidence interval for requested percentage of proportion based on cell), "tot.percent", "col.percent", "row.percent", "tot.prop", "col.prop", and "row.prop".

`text.label` Optional text to put after the y variable name, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on cell and parenth settings. Set to "none" for no text labels.

`parenth.sep` Optional character specifying the separator between lower and upper bound of confidence interval (when requested). Usually either "-" or ", "" depending on user preference.

`test` Controls test for association between x and y. Use "chi" for Pearson's chi-squared test, which is valid only in large samples; "fisher" for Fisher's exact test, which is valid in small or large samples; "z" for z test without continuity correction; or "z.continuity" for z test with continuity correction. "z" and "z.continuity" can only be used if x and y are binary.

`decimals` Number of decimal places for values in table (no decimals are used for counts). If unspecified, function uses 1 decimal for percentages and 3 decimals for proportions.

`p.include` If FALSE, statistical test is not performed and p-value is not returned.

`p.decimals` Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts input.

`p.cuts` Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.

`p.lowerbound` Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.

`p.leading0` If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

`p.avoid1` If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).

`overall.column` If FALSE, column showing distribution of y in full sample is suppressed.

`n.column` If TRUE, the table will have a column for sample size.

`n.headings` If TRUE, the table will indicate the sample size overall and in each group in parentheses after the column headings.

`compress` If y has only two levels, setting compress to TRUE will produce a single row rather than two rows. For example, if y is sex with 0 for female, 1 for male, and cell = "n" and parenth = "col.pecent", setting compress = TRUE will return a table with n (percent) for males only. If FALSE, the table would show n (percent) for both males and females, which is somewhat redundant.

`compress.val` When x and y are both binary, and compress is TRUE, compress.val can be used to specify which level of the y variable should be shown. For example, if x is sex and y is obesity status with levels "Obese" and "Not Obese", setting compress to TRUE and compress.val to "Not Obese" would result in the table comparing the proportions of subjects that are not obese by sex.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of the y variable are printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name and levels.

`print.html` If TRUE, function prints a .html file to the current working directory.

`html.filename` Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

NA

Value

A character matrix with the requested frequency table. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [1].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

tabmeans, tabmedians, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Compare sex distribution by group, with group as column variable
freqtable1 <- tabfreq(x = d$Group, y = d$Sex)

# Same comparison, but compress table to show Female row only, show percent (SE) rather
# than n (percent), and suppress (n = ) from column headings
freqtable2 <- tabfreq(x = d$Group, y = d$Sex, compress = TRUE, compress.val = "Female",
                      cell = "col.percent", parenth = "se", n.headings = FALSE)

# Compare sex distribution by race, suppressing (n = ) from column headings and
# showing percent (95% CI) rather than n (percent)
freqtable3 <- tabfreq(x = d$Race, y = d$Sex, n.headings = FALSE, cell = "col.percent")

# Use rbind to create single table comparing sex and race in control vs. treatment group
freqtable4 <- rbind(tabfreq(x = d$Group, y = d$Sex), tabfreq(x = d$Group, y = d$Race))

# A (usually) faster way to make the above table is to call the the tabmulti function
freqtable5 <- tabmulti(dataset = d, xvarname = "Group", yvarnames = c("Sex", "Race"))

# freqtable4 and freqtable5 are equivalent
all(freqtable4 == freqtable5)

# Click on freqtable1, ... , freqtable5 in the Workspace tab of RStudio to see the tables
# that could be copied and pasted into a report. Alternatively, setting the latex input to
# TRUE produces tables that can be inserted into LaTeX using the xtable package.
```

---

tabfreq.svy *Generate Frequency Tables for Statistical Reports (Survey Data)*

Description

This function creates an I-by-J frequency table and summarizes the results in a clean table for a statistical report. Similar to tabfreq, but for survey data. Relies heavily on the 'survey' package [1,2].

Usage

```
tabfreq.svy(x, y, svy, latex = FALSE, xlevels = NULL, yname = "Y variable",
          ylevels = NULL, test = "F", decimals = 1, p.decimals = c(2, 3), p.cuts = 0.01,
          p.lowerbound = 0.001, p.leading0 = TRUE, p.avoid1 = FALSE, n.column = FALSE,
            n.headings = TRUE, compress = FALSE, compress.val = NULL,
           bold.colnames = TRUE, bold.varnames = FALSE, bold.varlevels = FALSE,
            variable.colname = "Variable")
```

Arguments

svy	Survey design object created by a call to svydesign [1,2].
x	Character string specifying column variable name. Must match one of names(svy$variables).
y	Character string specifying row variable name. Must match one of names(svy$variables).
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlevels	Optional character vector to label the levels of x. If unspecified, the function uses the values that x takes on.
yname	Optional label for the y (row) variable.
ylevels	Optional character vector to label the levels of y. If unspecified, the function uses the values that y takes on. Note that levels of y will be listed in the order that they appear when you run table(y, x).
test	Controls test for association between x and y. Must be a possible value for the 'statistic' input of the svychisq function in the survey package [1,2]: 'F', 'Chisq', 'Wald', 'adjWald', 'lincom', or 'saddlepoint'.
decimals	Number of decimal places for percentages.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts input.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
n.column	If TRUE, the table will have a column for (unweighted) sample size.
n.headings	If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.
compress	If y has only two levels, setting compress to TRUE will produce a single row for n (percent) for the higher level. For example, if y is gender with 0 for female, 1 for male, setting compress = TRUE will return a table with n (percent) for males only.
compress.val	When x and y are both binary, and compress is TRUE, compress.val can be used to specify which level of the y variable should be shown. For example, if x is sex and y is obesity status with levels "Obese" and "Not Obese", setting compress to TRUE and compress.val to "Not Obese" would result in the table comparing the proportions of subjects that are not obese by sex.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of the y variable are printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name and levels.

Details

NA

Value

A character matrix with the requested frequency table. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

Currently this function is fairly basic. Future versions should allow for more flexibility. If you have any specific suggestions for additional options, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Lumley T (2012). survey: analysis of complex survey samples. R package version 3.28-2, https://cran.r-project.org/package=survey.

2. Lumley T (2014). Analysis of complex survey samples. Journal of Statistical Software 9(1): 1-19.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

svydesign, svychisq, tabfreq, tabmeans, tabmedians, tabmulti, tabglm, tabcox, tabgee, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
NA
```

`tabgee` *Generate Summary Tables of Fitted Generalized Estimating Equations for Statistical Reports*

Description

This function takes an object returned from the gee function in the package gee [1] and generates a clean summary table for a statistical report.

Usage

```
tabgee(geefit, latex = FALSE, xlabels = NULL, ci.beta = TRUE, decimals = 2,
      p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001, p.leading0 = TRUE,
       p.avoid1 = FALSE, basic.form = FALSE, intercept = TRUE, n.id = FALSE,
      n.total = FALSE, or = TRUE, robust = TRUE, data = NULL, greek.beta = FALSE,
       binary.compress = TRUE, bold.colnames = TRUE, bold.varnames = FALSE,
      bold.varlevels = FALSE, predictor.colname = "Variable", print.html = FALSE,
       html.filename = "table1.html")
```

Arguments

`geefit`	An object returned from gee function call.
`latex`	If TRUE, object returned is formatted for printing in LaTeX using xtable [2]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
`xlabels`	Optional character vector to label the x variables and their levels. If unspecified, the function uses the variable names and values themselves.
`ci.beta`	If TRUE, the table returned will include a column for Wald 95% confidence interval for the estimated coefficients.
`decimals`	Number of decimal places for numeric values in the table (except p-values).
`p.decimals`	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
`p.cuts`	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
`p.lowerbound`	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
`p.leading0`	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
basic.form	If TRUE, there is no attempt to neatly format factor variables and their levels, and the table returned is very similar to what you see when you run summary(glmfit).
intercept	If FALSE, the table returned will not include a row for the intercept.
n.id	If TRUE, the table returned will include a column for number of unique IDs (e.g. clusters).
n.total	If TRUE, the table returned will include a column for total number of observations used.
or	If TRUE, the table returned will include columns for odds ratios and Wald 95% confidence intervals for odds ratios. Only meaningful for logistic regression.
robust	If TRUE, robust standard errors are used (i.e. from sandwich estimator); if FALSE, naive standard errors are used.
data	Data frame or matrix containing variables passed to gee to create geefit. Only necessary when one or more of the predictors is a factor variable and basic.form is FALSE.
greek.beta	If TRUE, column headings refer to regression parameters as Greek letter beta rather than Beta. Only used when latex input is set to TRUE.
binary.compress	If TRUE, only one row of the table is dedicated to parameter estimates for each binary factor predictor. If FALSE, the table displays separate rows for the variable name and the two levels for each binary factor predictor, much like the presentation for factor variables with more than two levels.
bold.colnames	If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.
bold.varnames	If TRUE, variable names in the first column of the table are printed in bold font. Only applies if latex = TRUE.
bold.varlevels	If TRUE, levels of each factor variable are printed in bold font. Only applies if latex = TRUE and there is at least one factor variable included as a predictor.
predictor.colname	Character string with desired column heading for the column of predictors.
print.html	If TRUE, function prints a .html file to the current working directory.
html.filename	Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

The function should work well with categorical predictors (factors), provided they are not ordered. For ordered factors, just convert to unordered before creating the gee object to pass to tabgee. Note that you can define the levels of an unordered factor to control, which dictates which level is used as the reference group in regression models. For example, suppose a factor variable x takes values "low", "medium", and "high". If you write x = factor(x = x, levels = c("low", "medium", "high")), then you can run levels(x) to see that the levels are now arranged "low", "medium", "high". It is

still a regular factor, but now if you use x as a predictor in a call to gee, "low" will be the reference group when you call gee.

Interaction terms are compatible with tabgee, but the table will be formatted a little differently if interaction terms are present. Basically including an interaction is equivalent to setting basic.form to TRUE. All variable names and levels will be exactly as they appear when you run summary(geefit), where geefit is the object returned from a call to gee.

Value

A character matrix that summarizes the fitted GEE. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [2].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

While tabgee should work with any object generated by a call to gee, not all possibilities have been tested. Therefore in general I recommend always doing a quick check that the table created by tabgee matches the information in the gee object itself.

Author(s)

Dane R. Van domelen

References

1. Carey VJ (2012). gee: Generalized estimation equation solver. R package version 4.13-18. https://cran.r-project.org/package=gee.

2. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

gee tabfreq, tabmeans, tabmedians, tabmulti, tabglm, tabcox, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and convert to long format
data(d)
d2 <- reshape(data = d,
```

```
              varying = c("bp.1", "bp.2", "bp.3", "highbp.1", "highbp.2", "highbp.3"),
              timevar = "bp.visit", direction = "long")
d2 <- d2[order(d2$id), ]

# Load required package gee
library("gee")

# Create labels for race levels
races <- c("White", "Black", "Mexican American", "Other")

# Test whether predictors are associated with blood pressure at 1, 2, and 3 months
geefit1 <- gee(bp ~ Age + Sex + Race + BMI + Group, id = id, data = d2,
               corstr = "unstructured")

# Create summary table using tabgee
geetable1 <- tabgee(geefit = geefit1, data = d2, n.id = TRUE, n.total = TRUE,
                    xlabels = c("Intercept", "Age", "Male", "Race", races, "BMI",
                                "Treatment"))

# Test whether predictors are associated with high blood pressure at 1, 2, and 3 months
geefit2 <- gee(highbp ~ Age + Sex + Race + BMI + Group, id = id, data = d2,
               family = binomial, corstr = "unstructured")

# Create summary table using tabgee
geetable2 <- tabgee(geefit = geefit2, data = d2, ci.beta = FALSE,
                    xlabels = c("Intercept", "Age", "Male", "Race", races, "BMI",
                                "Treatment"))

# Click on geetable1 or geetable2 in the Workspace tab of RStudio to see the tables that
# could be copied and pasted into a report or manuscript. Alternatively, setting the
# latex input to TRUE produces tables that can be inserted into LaTeX using the xtable
# package.
```

`tabglm` *Generate Summary Tables of Fitted Generalized Linear Models for Statistical Reports*

Description

This function takes an object returned from the glm function and generates a clean summary table for a statistical report.

Usage

```
tabglm(glmfit, latex = FALSE, xlabels = NULL, ci.beta = TRUE, inference = "wald",
       decimals = 2, p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001,
      p.leading0 = TRUE, p.avoid1 = FALSE, basic.form = FALSE, intercept = TRUE,
       n = FALSE, events = FALSE, greek.beta = FALSE, binary.compress = TRUE,
       bold.colnames = TRUE, bold.varnames = FALSE, bold.varlevels = FALSE,
       predictor.colname = "Variable", print.html = FALSE,
       html.filename = "table1.html")
```

Arguments

glmfit	An object returned from glm function call.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [1]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlabels	Optional character vector to label the x variables and their levels. If unspecified, the function uses the variable names and values themselves.
ci.beta	If TRUE, the table returned will include a column for 95% confidence interval for the regression coefficients.
inference	If "wald", CI's and p-values are based on t or z statistics, depending on the GLM family (i.e. Gaussian, Poisson, binomial, etc.); if "wald.z", CI's and p-values are based on z statistics; if "profile", CI's are based on profile likelihood (confint function), and p-values are based on t or z statistics, depending on the GLM family; if "profile.z", CI's are based on profile likelihood, and p-values are based on z statistics.
decimals	Number of decimal places for numeric values in the table (except p-values).
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
basic.form	If TRUE, there is no attempt to neatly format factor variables and their levels, and the table returned is very similar to what you see when you run summary(glmfit).
intercept	If FALSE, the table returned will not include a row for the intercept.
n	If TRUE, the table returned will include a column for sample size.
events	If TRUE, the table returned will include a column for number of events observed. Only meaningful when the outcome variable is binary.
greek.beta	If TRUE, column headings refer to regression parameters as Greek letter beta rather than Beta. Only used when latex input is set to TRUE.
binary.compress	If TRUE, only one row of the table is dedicated to parameter estimates for each binary factor predictor. If FALSE, the table displays separate rows for the variable name and the two levels for each binary factor predictor, much like the presentation for factor variables with more than two levels.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable names in the first column of the table are printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of each factor variable are printed in bold font. Only applies if latex = TRUE and there is at least one factor variable included as a predictor.

`predictor.colname` Character string with desired column heading for the column of predictors.

`print.html` If TRUE, function prints a .html file to the current working directory.

`html.filename` Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

The function should work well with categorical predictors (factors), provided they are not ordered. For ordered factors, just convert to unordered before creating the glm object to pass to tabglm. Note that you can define the levels of an unordered factor to control, which dictates which level is used as the reference group in regression models. For example, suppose a factor variable x takes values "low", "medium", and "high". If you write x = factor(x = x, levels = c("low", "medium", "high")), then you can run levels(x) to see that the levels are now arranged "low", "medium", "high". It is still a regular factor, but now if you use x as a predictor in a call to glm, "low" will be the reference group.

Interaction terms are compatible with tabglm, but the table will be formatted a little differently if interaction terms are present. Basically including an interaction is equivalent to setting basic.form to TRUE. All variable names and levels will be exactly as they appear when you run summary(glmfit), where glmfit is the object returned from a call to glm.

Value

A character matrix that summarizes the fitted generalized linear model. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [1].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

This function replaces the previous functions tablin and tablog, which were for linear and logistic regression, respectively. The only capability those functions had that tabglm does not currently have is the ability to calculate standardized regression coefficients. This feature may be added to tabglm in the future.

While tabglm should work with any object generated by a call to glm, not all possibilities have been tested. Therefore in general I recommend always doing a quick check that the table created by tabglm matches the information in the glm object itself.

Author(s)

Dane R. Van Domelen

References

1. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

glm tabfreq, tabmeans, tabmedians, tabmulti, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Create labels for race levels
races <- c("White", "Black", "Mexican American", "Other")

# Test whether age, sex, race, and treatment group are associated with BMI
glmfit1 <- glm(BMI ~ Age + Sex + Race + Group, data = d)
lintable <- tabglm(glmfit = glmfit1,
                   xlabels = c("Intercept", "Age", "Male", "Race", races, "Treatment"))

# Test whether age, sex, race, and treatment group are associated with 1-year mortality
glmfit2 <- glm(death_1yr ~ Age + Sex + Race + Group, data = d, family = binomial)
logtable <- tabglm(glmfit = glmfit2, ci.beta = FALSE,
                   xlabels = c("Intercept", "Age", "Male", "Race", races, "Treatment"))

# Click on lintable or logtable in the Workspace tab of RStudio to see the tables that
# could be copied and pasted into a report or manuscript. Alternatively, setting the
# latex input to TRUE produces tables that can be inserted into LaTeX using the xtable
# package.
```

tabglm.svy *Generate Summary Tables of Fitted Generalized Linear Models for Statistical Reports (Survey Data)*

Description

This function takes an object returned from the svyglm function and generates a clean summary table for a statistical report. Similar to tabglm, but for survey data. Relies heavily on the 'survey' package [1,2].

Usage

```
tabglm.svy(svyglmfit, latex = FALSE, xlabels = NULL, ci.beta = TRUE, inference = "wald.t",
         decimals = 2, p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001,
         p.leading0 = TRUE, p.avoid1 = FALSE, basic.form = FALSE, intercept = TRUE,
          n = FALSE, events = FALSE, greek.beta = FALSE, binary.compress = TRUE,
           bold.colnames = TRUE, bold.varnames = FALSE, bold.varlevels = FALSE,
           predictor.colname = "Variable")
```

Arguments

svyglmfit	An object returned from svyglm function call [1,2].
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlabels	Optional character vector to label the x variables and their levels. If unspecified, the function uses the variable names and values themselves.
ci.beta	If TRUE, the table returned will include a column for Wald 95% confidence interval for the regression coefficients.
inference	If "wald.t", confidence intervals and p-values are based on t distributions with degrees of freedom given by svyglmfit$df.residual; if "wald.z", confidence intervals and p-values are based on z distributions.
decimals	Number of decimal places for numeric values in the table (except p-values).
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

`p.avoid1` If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).

`basic.form` If TRUE, there is no attempt to neatly format factor variables and their levels, and the table returned is very similar to what you see when you run summary(glmfit).

`intercept` If FALSE, the table returned will not include a row for the intercept.

`n` If TRUE, the table returned will include a column for sample size.

`events` If TRUE, the table returned will include a column for number of events observed. Only meaningful when the outcome variable is binary.

`greek.beta` If TRUE, column headings refer to regression parameters as Greek letter beta rather than Beta. Only used when latex input is set to TRUE.

`binary.compress`
If TRUE, only one row of the table is dedicated to parameter estimates for each binary factor predictor. If FALSE, the table displays separate rows for the variable name and the two levels for each binary factor predictor, much like the presentation for factor variables with more than two levels.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable names in the first column of the table are printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of each factor variable are printed in bold font. Only applies if latex = TRUE and there is at least one factor variable included as a predictor.

`predictor.colname`
Character string with desired column heading for the column of predictors.

Details

The function should work well with categorical predictors (factors), provided they are not ordered. For ordered factors, just convert to unordered before creating the svyglm object to pass to tabglm.svy. Note that you can define the levels of an unordered factor to control, which dictates which level is used as the reference group in regression models. For example, suppose a factor variable x takes values "low", "medium", and "high". If you write x = factor(x = x, levels = c("low", "medium", "high")), then you can run levels(x) to see that the levels are now arranged "low", "medium", "high". It is still a regular factor, but now if you use x as a predictor in a call to svyglm, "low" will be the reference group.

Interaction terms are compatible with tabglm.svy, but the table will be formatted a little differently if interaction terms are present. Basically including an interaction is equivalent to setting basic.form to TRUE. All variable names and levels will be exactly as they appear when you run summary(svyglmfit), where svyglmfit is the object returned from a call to svyglm.

Value

A character matrix that summarizes the fitted generalized linear model. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

While tabglm.svy should work with any object generated by a call to svyglm, not all possibilities have been tested. Therefore in general I recommend always doing a quick check that the table created by tabglm.svy matches the information in the svyglm object itself.

Author(s)

Dane R. Van domelen

References

1. Lumley T (2012). survey: analysis of complex survey samples. R package version 3.28-2, https://cran.r-project.org/package=survey.

2. Lumley T (2014). Analysis of complex survey samples. Journal of Statistical Software 9(1): 1-19.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

svydesign svyglm tabfreq, tabmeans, tabmedians, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy

Examples

```
NA
```

---

tabmeans *Generate Summary Tables of Mean Comparisons for Statistical Reports*

Description

This function compares the mean of a continuous variable across levels of a categorical variable and summarizes the results in a clean table (or figure) for a statistical report.

Usage

```
tabmeans(x, y, latex = FALSE, variance = "unequal", xname = NULL, xlevels = NULL,
         yname = NULL, quantiles = NULL, quantile.vals = FALSE, parenth = "sd",
        text.label = NULL, parenth.sep = "-", decimals = NULL, p.include = TRUE,
       p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001, p.leading0 = TRUE,
       p.avoid1 = FALSE, overall.column = TRUE, n.column = FALSE, n.headings = TRUE,
       bold.colnames = TRUE, bold.varnames = FALSE, variable.colname = "Variable",
       fig = FALSE, fig.errorbars = "z.ci", fig.title = NULL, print.html = FALSE,
         html.filename = "table1.html")
```

Arguments

x	Vector of values for the categorical x variable.
y	Vector of values for the continuous y variable.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [1]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
variance	Controls whether equal variance t-test or unequal variance t-test is used when x has two levels. Possible values are "equal" for equal variance, "unequal" for unequal variance, or "ftest" for F test to determine which version of the t-test to use. Note that unequal variance t-test is less restrictive than equal variance t-test, and the F test is only valid when y is normally distributed in both x groups.
xname	Label for the categorical variable. Only used if fig is TRUE.
xlevels	Optional character vector to label the levels of x, used in the column headings. If unspecified, the function uses the values that x takes on.
yname	Optional label for the continuous y variable. If unspecified, variable name of y is used.
quantiles	If specified, function compares means of the y variable across quantiles of the x variable. For example, if x contains continuous BMI values and y contains continuous HDL cholesterol levels, setting quantiles to 3 would result in mean HDL being compared across tertiles of BMI.
quantile.vals	If TRUE, labels for x show quantile number and corresponding range of the x variable. For example, Q1 [0.00, 0.25). If FALSE, labels for quantiles just show quantile number (e.g. Q1). Only used if xlevels is not specified.
parenth	Controls what values (if any) are placed in parentheses after the means in each cell. Possible values are "none", "sd" for standard deviation, "se" for standard error, "t.ci" for 95% confidence interval for population mean based on t distribution, and "z.ci" for 95% confidence interval for population mean based on z distribution.
text.label	Optional text to put after the y variable name, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on parenth, e.g. M (SD) if parenth is "sd". Set to "none" for no text labels.
parenth.sep	Optional character specifying the separator between lower and upper bound of confidence interval (when requested). Usually either "-" or ", " depending on user preference.
decimals	Number of decimal places for means and standard deviations/standard errors/confidence intervals. If unspecified, function uses 0 decimal places if the largest mean (in magnitude) is in [1,000, Inf), 1 decimal place if [10, 1,000), 2 decimal places if [0.1, 10), 3 decimal places if [0.01, 0.1), 4 decimal places if [0.001, 0.01), 5 decimal places if [0.0001, 0.001), and 6 decimal places if [0, 0.0001).
p.include	If FALSE, t-test is not performed and p-value is not returned.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.

p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
overall.column	If FALSE, column showing mean of y in full sample is suppressed.
n.column	If TRUE, the table will have a column for (unweighted) sample size.
n.headings	If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.
bold.colnames	If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.
bold.varnames	If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.
variable.colname	Character string with desired heading for first column of table, which shows the y variable name.
fig	If TRUE, a figure is returned rather than a table. The figure shows mean (95 percent confidence interval) for each level of x.
fig.errorbars	Controls error bars around mean when fig is TRUE. Possible values are "sd" for +/- 1 standard deviation, "se" for +/- 1 standard error, "t.ci" for 95% confidence interval based on t distribution, "z.ci" for 95% confidence interval based on z distribution, and "none" for no error bars.
fig.title	Title of figure. If unspecified, title is set to "Mean yname by xname".
print.html	If TRUE, function prints a .html file to the current working directory.
html.filename	Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

If x has two levels, a t-test is used to test for a difference in means. If x has more than two levels, a one-way analysis of variance is used to test for a difference in means across the groups.

Both x and y can have missing values. The function drops observations with missing x or y.

Value

A character matrix with the requested table comparing mean y across levels of x. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [1].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

tabfreq, tabmedians, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Compare mean BMI in control group vs. treatment group - table and figure
meanstable1 <- tabmeans(x = d$Group, y = d$BMI)
meansfig1 <- tabmeans(x = d$Group, y = d$BMI, fig = TRUE)

# Compare mean BMI by race - table and figure
meanstable2 <- tabmeans(x = d$Race, y = d$BMI)
meansfig2 <- tabmeans(x = d$Race, y = d$BMI, fig = TRUE)

# Compare mean baseline systolic BP across tertiles of BMI - table and figure
meanstable3 <- tabmeans(x = d$BMI, y = d$bp.1, yname = "Systolic BP", quantiles = 3)
meansfig3 <- tabmeans(x = d$BMI, y = d$bp.1, quantiles = 3, fig = TRUE,
                      yname = "Systolic BP", xname = "BMI Tertile")

# Create single table comparing mean BMI and mean age in control vs. treatment group
meanstable4 <- rbind(tabmeans(x = d$Group, y = d$BMI), tabmeans(x = d$Group, y = d$Age))

# An easier way to make the above table is to call the tabmulti function
meanstable5 <- tabmulti(dataset = d, xvarname = "Group", yvarnames = c("BMI", "Age"))
```

```
# meanstable4 and meanstable5 are equivalent
all(meanstable4 == meanstable5)

# Click on meanstable 1, ... , meanstable5 in the Workspace tab of RStudio to see the
# tables that could be copied and pasted into a report. Alternatively, setting the latex
# input to TRUE produces tables that can be inserted into LaTeX using the xtable package.
```

`tabmeans.svy` *Generate Summary Tables of Mean Comparisons for Statistical Reports (Survey Data)*

Description

This function compares the mean of a continuous variable across levels of a categorical variable and summarizes the results in a clean table for a statistical report. Similar to tabmeans, but for survey data. Relies heavily on the 'survey' package [1,2].

Usage

```
tabmeans.svy(x, y, svy, latex = FALSE, xlevels = NULL, yname = "Y variable",
             test = "Wald", decimals = 1, p.decimals = c(2, 3), p.cuts = 0.01,
          p.lowerbound = 0.001, p.leading0 = TRUE, p.avoid1 = FALSE, n.column = FALSE,
             n.headings = TRUE, bold.colnames = TRUE, bold.varnames = FALSE,
             variable.colname = "Variable")
```

Arguments

`svy`	Survey design object created by a call to svydesign [1,2].
`x`	Character string specifying categorical x variable name. Must match one of names(svy$variables).
`y`	Character string specifying continuous y variable name. Must match one of names(svy$variables).
`latex`	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
`xlevels`	Optional character vector to label the levels of x. If unspecified, the function uses the values that x takes on.
`yname`	Optional label for the continuous variable.
`test`	Either "Wald" for Wald test or "LRT" for likelihood ratio test to test for equivalent mean y across levels of x.
`decimals`	Number of decimal places for means and standard deviations or standard errors.
`p.decimals`	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.

`p.cuts` Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.

`p.lowerbound` Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.

`p.leading0` If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

`p.avoid1` If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).

`n.column` If TRUE, the table will have a column for (unweighted) sample size.

`n.headings` If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name.

Details

NA

Value

A character matrix with the requested table comparing mean y across levels of x. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

Currently this function is fairly basic. Future versions should allow for more flexibility. If you have any specific suggestions for additional options, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Lumley T (2012). survey: analysis of complex survey samples. R package version 3.28-2, https://cran.r-project.org/package=survey.

2. Lumley T (2014). Analysis of complex survey samples. Journal of Statistical Software 9(1): 1-19.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

svydesign, svyglm, tabfreq, tabmeans, tabmedians, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
NA
```

---

tabmedians *Generate Summary Tables of Median Comparisons for Statistical Reports*

Description

This function compares the median of a continuous variable across levels of a categorical variable and summarizes the results in a clean table for a statistical report.

Usage

```
tabmedians(x, y, latex = FALSE, xlevels = NULL, yname = NULL, quantiles = NULL,
         quantile.vals = FALSE, parenth = "iqr", text.label = NULL, parenth.sep = "-",
          decimals = NULL, p.include = TRUE, p.decimals = c(2, 3), p.cuts = 0.01,
            p.lowerbound = 0.001, p.leading0 = TRUE, p.avoid1 = FALSE,
            overall.column = TRUE, n.column = FALSE, n.headings = TRUE,
         bold.colnames = TRUE, bold.varnames = FALSE, variable.colname = "Variable",
            print.html = FALSE, html.filename = "table1.html")
```

Arguments

x	Vector of values for the categorical variable.
y	Vector of values for the continuous variable.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [1]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlevels	Optional character vector to label the levels of x, used in the column headings. If unspecified, the function uses the values that x takes on.

yname	Optional label for the continuous y variable. If unspecified, variable name of y is used.
quantiles	If specified, function compares medians of the y variable across quantiles of the x variable. For example, if x contains continuous BMI values and y contains continuous HDL cholesterol levels, setting quantiles to 3 would result in median HDL being compared across tertiles of BMI.
quantile.vals	If TRUE, labels for x show quantile number and corresponding range of the x variable. For example, Q1 [0.00, 0.25). If FALSE, labels for quantiles just show quantile number (e.g. Q1). Only used if xlevels is not specified.
parenth	Controls what values (if any) are placed in parentheses after the medians in each cell. Possible values are "none", "iqr" for difference between first and third quartiles, "range" for difference between minimum and maximum, "minmax" for minimum and maximum, "q1q3" for first and third quartiles, or "ci.90", "ci.95", or "ci.99" for confidence intervals for the medians (based on binomial probabilities if one or more groups have n less than 10, otherwise based on normal approximation to binomial).
text.label	Optional text to put after the y variable name, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on parenth, e.g. Median (IQR) if parenth is "iqr". Set to "none" for no text labels.
parenth.sep	Optional character specifying the separator for the two numbers in parentheses when parenth is set to "minmax" or "q1q3". The default is a dash, so values in the table are formatted as Median (Lower-Upper). If you set parenth.sep to ", " the values in the table would instead be formatted as Median (Lower, Upper).
decimals	Number of decimal places for values in table. If unspecified, function uses 0 decimal places if the largest median (in magnitude) is in [1,000, Inf), 1 decimal place if [10, 1,000), 2 decimal places if [0.1, 10), 3 decimal places if [0.01, 0.1), 4 decimal places if [0.001, 0.01), 5 decimal places if [0.0001, 0.001), and 6 decimal places if [0, 0.0001).
p.include	If FALSE, statistical test is not performed and p-value is not returned.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
overall.column	If FALSE, column showing median of y in full sample is suppressed.

`n.column` If TRUE, the table will have a column for (unweighted) sample size.

`n.headings` If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name.

`print.html` If TRUE, function prints a .html file to the current working directory.

`html.filename` Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

If x has two levels, a Mann-Whitney U (also known as Wilcoxon rank-sum) test is used to test whether the distribution of the continuous variable (y) differs in the two groups (x). If x has more than two levels, a Kruskal-Wallis test is used to test whether the distribution of y differs across at least two of the x groups.

Both x and y can have missing values. The function drops observations with missing x or y.

Value

A character matrix with the requested table comparing median y across levels of x. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [1].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

tabfreq, tabmeans, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d and drop rows with missing values
data(d)
d <- d[complete.cases(d), ]

# Create labels for group and race
groups <- c("Control", "Treatment")
races <- c("White", "Black", "Mexican American", "Other")

# Compare median BMI in control group vs. treatment group
medtable1 <- tabmedians(x = d$Group, y = d$BMI)

# Repeat, but show first and third quartile rather than IQR in parentheses
medtable2 <- tabmedians(x = d$Group, y = d$BMI, parenth = "q1q3")

# Compare median BMI by race, suppressing overall column and (n = ) part of headings
medtable3 <- tabmedians(x = d$Race, y = d$BMI, overall.column = FALSE, n.headings = FALSE)

# Compare median BMI by quartile of age
medtable4 <- tabmedians(x = d$Age, y = d$BMI, quantiles = 4)

# Create single table comparing median BMI and median age in control vs. treatment group
medtable5 <- rbind(tabmedians(x = d$Group, y = d$BMI), tabmedians(x = d$Group, y = d$Age))

# A (usually) faster way to make the above table is to call the tabmulti function
medtable6 <- tabmulti(dataset = d, xvarname = "Group", yvarnames = c("BMI", "Age"),
                      ymeasures = "median")

# medtable5 and medtable6 are equivalent
all(medtable5 == medtable6)

# Click on medtable1, ... , medtable6 in the Workspace tab of RStudio to see the tables
# that could be copied and pasted into a report or manuscript. Alternatively, setting the
# latex input to TRUE produces tables that can be inserted into LaTeX using the xtable
# package.
```

`tabmedians.svy` *Generate Summary Tables of Median Comparisons for Statistical Reports (Survey Data)*

Description

This function compares the median of a continuous variable across levels of a categorical variable and summarizes the results in a clean table for a statistical report. Similar to tabmeans, but for survey data. Relies heavily on the 'survey' package [1,2].

Usage

```
tabmedians.svy(svy, x, y, latex = FALSE, xlevels = NULL, yname = "Y variable",
            test = "wilcoxon", decimals = 1, p.include = TRUE, p.decimals = c(2, 3),
            p.cuts = 0.01, p.lowerbound = 0.001, p.leading0 = TRUE, p.avoid1 = FALSE,
            n.column = FALSE, n.headings = TRUE, parenth = "iqr", text.label = NULL,
               parenth.sep = "-", bold.colnames = TRUE, bold.varnames = FALSE,
               variable.colname = "Variable")
```

Arguments

svy	Survey design object created by a call to svydesign [1,2].
x	Vector of values for the categorical variable.
y	Vector of values for the continuous variable.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlevels	Optional character vector to label the levels of x. If unspecified, the function uses the values that x takes on.
yname	Optional label for the continuous variable.
test	Controls statistical test. Must be a possible value for the 'test' input of the svyranktest function in the survey package [1,2]: 'wilcoxon' for Mann-Whitney U/Wilcoxon test of whether one group is from distribution that is stochastically greater than the other; 'vanderWaerden' for Van der Waerden test of whether the population distribution functions are equal; 'median' for Mood's test for whether the population medians are equal; and 'KruskalWallis' for Kruskal-Wallis test which is Mann-Whitney U/Wilcoxon generalized to three or more groups.
decimals	Number of decimal places for means and standard deviations or standard errors.
p.include	If FALSE, statistical test is not performed and p-value is not returned.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.

`p.lowerbound` Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.

`p.leading0` If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

`p.avoid1` If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).

`n.column` If TRUE, the table will have a column for (unweighted) sample size.

`n.headings` If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.

`parenth` Controls what values (if any) are placed in parentheses after the medians in each cell. Possible choices are as follows: 'minmax' for minimum and maximum; 'range' for difference between minimum and maximum; 'q1q3' for first and third quartiles; 'iqr' for difference between first and third quartiles; or 'none' for no parentheses at all.

`text.label` Optional text to put after the variable name. For example, if parenth is 'q1q3' and yname is 'BMI' the default label would be 'BMI, Median (Q1-Q3)'. You might prefer to set text.label to something like 'Med (Quartile 1-Quartile 3)' instead.

`parenth.sep` Optional character specifying the separator for the two numbers in parentheses when parenth is set to 'minmax' or 'q1q3'. The default is a dash, so values in the table are formatted as Median (Lower-Upper). If you set parenth.sep to ', ' the values in the table would instead be formatted as Median (Lower, Upper).

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name.

Details

NA

Value

A character matrix with the requested table comparing median y across levels of x. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Lumley T (2012). survey: analysis of complex survey samples. R package version 3.28-2, https://cran.r-project.org/package=survey.

2. Lumley T (2014). Analysis of complex survey samples. Journal of Statistical Software 9(1): 1-19.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

svydesign, svyquantile, svyranktest, tabfreq, tabmeans, tabmulti, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmulti.svy, tabglm.svy

Examples

```
NA
```

---

tabmulti — *Generate Multi-row Tables Comparing Means/Medians/Frequencies of Multiple Variables Across Levels of One Categorical Variable*

Description

This function basically provides an alternative to making multiple calls to tabmeans, tabmedians, and tabfreq, then using rbind to combine the results into a single table.

Usage

```
tabmulti(dataset, xvarname, yvarnames, ymeasures = NULL, listwise.deletion = TRUE,
         latex = FALSE, xlevels = NULL, ynames = yvarnames, ylevels = NULL,
       quantiles = NULL, quantile.vals = FALSE, parenth.sep = "-", decimals = NULL,
       cell = "n", freq.parenth = NULL, freq.text.label = NULL, freq.tests = "chi",
         means.parenth = "sd", means.text.label = NULL, variance = "unequal",
         medians.parenth = "iqr", medians.text.label = NULL, p.include = TRUE,
       p.decimals = c(2, 3), p.cuts = 0.01, p.lowerbound = 0.001, p.leading0 = TRUE,
       p.avoid1 = FALSE, overall.column = TRUE, n.column = FALSE, n.headings = TRUE,
         compress = FALSE, bold.colnames = TRUE, bold.varnames = FALSE,
       bold.varlevels = FALSE, variable.colname = "Variable", print.html = FALSE,
         html.filename = "table1.html")
```

Arguments

`dataset`	Data frame or matrix containing variables of interest.
`xvarname`	Character string with name of x (column) variable. Should be one of colnames(dataset).
`yvarnames`	Character string or vector of character strings with names of y (row) variables. Each element should be one of colnames(dataset).
`ymeasures`	Character string or vector of character strings indicating whether each y variable should be summarized by mean, median, or frequency. For example, if yvarnames has length three and you wish to display frequencies for the first variable, means for the second, and medians for the third, you would set ymeasures to c("freq", "mean", "median"). If unspecified, function displays frequencies for any factor variable or numeric variable with five or fewer unique values, and means for numeric variables with more than five levels.
`listwise.deletion`	If TRUE, observations with missing values for any y variable are excluded entirely; if FALSE, all available data is used for each comparison. If FALSE, recommend also setting n to TRUE so table shows effective sample size for each comparison.
`latex`	If TRUE, object returned is formatted for printing in LaTeX using xtable [1]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
`xlevels`	Optional character vector to label the levels of x, used in the column headings. If unspecified, the function uses the values that x takes on.
`ynames`	Optional labels for the y variables. If unspecified, y variable names are used.
`ylevels`	Character vector or list of character vectors to label the levels of the categorical y variables.
`quantiles`	If specified, function compares y variables across quantiles of the x variable. For example, if x contains continuous BMI values and y contains continuous HDL and race, setting quantiles to 3 would result in mean HDL and distribution of race being compared across tertiles of BMI.
`quantile.vals`	If TRUE, labels for x show quantile number and corresponding range of the x variable. For example, Q1 [0.00, 0.25). If FALSE, labels for quantiles just show quantile number (e.g. Q1). Only used if xlevels is not specified.
`parenth.sep`	Optional character specifying the separator between first and second numbers in parentheses (e.g. lower and upper bound of confidence intervals, when requested). Usually either "-" or ", " depending on user preference.
`decimals`	Numeric value of vector of numeric values indicating how many decimal places should be used in reporting statistics for each y variable.
`cell`	Controls what values are placed in cells for frequency comparisons. Possible choices are "n" for counts, "tot.percent" for table percentage, "col.percent" for column percentage, "row.percent" for row percentage, "tot.prop" for table proportion, "col.prop" for column proportion, "row.prop" for row proportion, "n/totn" for count/total counts, "n/coln" for count/column count, and "n/rown" for count/row count.

`freq.parenth` Controls what values (if any) are placed in parentheses after the values in each cell for frequency comparisons. By default, if cell is "n", "n/totn", "n/coln", or "n/rown" then the corresponding percentage is shown in parentheses; if cell is "tot.percent", "col.percent", "row.percent", "tot.prop", "col.prop", or "row.prop" then a 95% confidence interval for the requested percentage of proportion is shown in parentheses. Possible values are "none", "se" (for standard error of requested percentage or proportion based on cell), "ci" (for 95% confidence interval for requested percentage of proportion based on cell), "tot.percent", "col.percent", "row.percent", "tot.prop", "col.prop", and "row.prop".

`freq.text.label` Optional text to put after the y variable name for frequency comparisons, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on cell and freq.parenth settings. Set to "none" for no text labels.

`freq.tests` Character string or vector of character strings indicating what statistical tests should be used to compare distributions of each categorical row variable across levels of the column variable. Elements can be "chi" for Pearson's chi-squared test, which is valid only in large samples; 'fisher' for Fisher's exact test, which is valid in small or large samples; 'z' for z test without continuity correction; or 'z.continuity' for z test with continuity correction. 'z' and 'z.continuity' can only be used for binary column and row variables.

`means.parenth` Controls what values (if any) are placed in parentheses after the means in each cell for mean comparisons. Possible values are "none", "sd" for standard deviation, "se" for standard error, "t.ci" for 95% confidence interval for population mean based on t distribution, and "z.ci" for 95% confidence interval for population mean based on z distribution.

`means.text.label` Optional text to put after the y variable name for mean comparisons, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on parenth, e.g. M (SD) if means.parenth is "sd". Set to "none" for no text labels.

`variance` Controls whether equal variance t-test or unequal variance t-test is used for mean comparisons when x has two levels. Possible values are "equal" for equal variance, "unequal" for unequal variance, or "ftest" for F test to determine which version of the t-test to use. Note that unequal variance t-test is less restrictive than equal variance t-test, and the F test is only valid when y is normally distributed in both x groups.

`medians.parenth` Controls what values (if any) are placed in parentheses after the medians in each cell for median comparisons. Possible values are "none", "iqr" for difference between first and third quartiles, "range" for difference between minimum and maximum, "minmax" for minimum and maximum, and "q1q3" for first and third quartiles.

`medians.text.label` Optional text to put after the y variable name for median comparisons, identifying what cell values and parentheses indicate in the table. If unspecified, function uses default labels based on parenth, e.g. Median (IQR) if medians.parenth is "iqr". Set to "none" for no text labels.

p.include	If FALSE, statistical test is not performed and p-value is not returned.
p.decimals	Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.
p.cuts	Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.
p.lowerbound	Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.
p.leading0	If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.
p.avoid1	If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).
overall.column	If FALSE, column showing frequencies/means/medians for y in full sample is suppressed.
n.column	If TRUE, the table will have a column for (unweighted) sample size.
n.headings	If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.
compress	Logical indicating whether categorical y variables with two levels should be compressed into a single row rather than two rows for the table.
bold.colnames	If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.
bold.varnames	If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.
bold.varlevels	If TRUE, levels of categorical y variables are printed in bold font. Only applies if latex = TRUE.
variable.colname	Character string with desired heading for first column of table, which shows the y variable name and levels.
print.html	If TRUE, function prints a .html file to the current working directory.
html.filename	Character string indicating the name of the .html file that gets printed if print.html is set to TRUE.

Details

See help files for tabmeans, tabmedians, and tabfreq for details on statistical tests.

Value

A character matrix comparing mean/medians/frequencies of row variables across levels of the column variable. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [1].

Note

In older versions of RStudio, it was easier to copy tables from the Viewer and paste them directly into a text editor. The Viewer changed a few versions ago, and now it seems to work better if you paste into Microsoft Excel, and then copy again and paste into Microsoft Word. This is a little clumsy, so I recently added the new option to print a .html file with the table to your current working directory (see function inputs print.html and html.filename). Copying and pasting from the table from the .html file into a text editor seems to work well.

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

tabfreq, tabmeans, tabmedians, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabmulti.svy, tabglm.svy

Examples

```
# Load in sample dataset d
data(d)

# Compare age, sex, race, and BMI in control vs. treatment group
# data for each comparison
table1 <- tabmulti(dataset = d, xvarname = "Group",
                   yvarnames = c("Age", "Sex", "Race", "BMI"))

# Repeat, but use all available data for each comparison (as opposed to listwise deletion)
table2 <- tabmulti(dataset = d, xvarname = "Group", n.column = TRUE, n.headings = FALSE,
                  yvarnames = c("Age", "Sex", "Race", "BMI"), listwise.deletion = FALSE)

# Same as table1, but compare medians rather than means for BMI
table3 <- tabmulti(dataset = d, xvarname = "Group",
                   yvarnames = c("Age", "Sex", "Race", "BMI"),
                   ymeasures = c("mean", "freq", "freq", "median"))

# Click on table1, table2, or table3 in the Workspace tab of RStudio to see the tables
# that could be copied and pasted into a report or manuscript. Alternatively, setting
# the latex input to TRUE produces tables that can be inserted into LaTeX using the
# xtable package.
```

tabmulti.svy *Generate Multi-row Tables Comparing Means/Medians/Frequencies of Multiple Variables Across Levels of One Categorical Variable (Survey Data)*

Description

This function basically provides an alternative to making multiple calls to tabmeans.svy, tabmedians.svy, and tabfreq.svy, then using rbind to combine the results into a single table. Similar to tabmulti, but for survey data. Relies heavily on the 'survey' package [1,2].

Usage

```
tabmulti.svy(svy, xvarname, yvarnames, ymeasures = NULL, listwise.deletion = FALSE,
             latex = FALSE, xlevels = NULL, ynames = yvarnames, ylevels = NULL,
             mean.tests = "Wald", median.tests = "wilcoxon", freq.tests = "F",
            decimals = 1, p.include = TRUE, p.decimals = c(2, 3), p.cuts = 0.01,
          p.lowerbound = 0.001, p.leading0 = TRUE, p.avoid1 = FALSE, n.column = FALSE,
             n.headings = TRUE, se = FALSE, compress = FALSE, parenth = "iqr",
             text.label = NULL, parenth.sep = "-", bold.colnames = TRUE,
          bold.varnames = FALSE, bold.varlevels = FALSE, variable.colname = "Variable")
```

Arguments

svy	Survey design object created by a call to svydesign [1,2].
xvarname	Character string with name of column variable. Should be one of colnames(dataset).
yvarnames	Character string or vector of character strings with names of row variables. Each element should be one of colnames(dataset).
ymeasures	Character string or vector of character strings indicating whether each row variable should be summarized by mean, median, or frequency. For example, if yvarnames has length three and you wish to display frequencies for the first variable, means for the second, and medians for the third, you would set ymeasures to c("freq", "mean", "median"). If unspecified, function displays frequencies for any factor variable or numeric variable with five or fewer unique values, and means for numeric variables with more than five levels.
listwise.deletion	If TRUE, observations with missing values for any row variable are excluded entirely; if FALSE, all available data is used for each comparison. If FALSE, recommend also setting n to TRUE so table shows effective sample size for each comparison.
latex	If TRUE, object returned is formatted for printing in LaTeX using xtable [3]; if FALSE, formatted for copy-and-pasting from RStudio into a word processor.
xlevels	Optional character vector to label the levels of x. If unspecified, the function uses the values that x takes on.
ynames	Optional labels for the row variables.

`ylevels` Character vector or list of character vectors to label the levels of the categorical row variables.

`mean.tests` Character string or vector of character strings indicating what statistical tests should be used to compare means for each row variable for which a comparison of means is requested. Elements should be 'Wald' for Wald test or 'LRT' for likelihood ratio test.

`median.tests` Character string or vector of character strings indicating what statistical tests should be used to compare medians for each row variable for which a comparison of medians is requested. Elements should be possible values for the 'test' input of the svyranktest function in the survey package [1,2]: 'wilcoxon' for Mann-Whitney U/Wilcoxon test of whether one group is from distribution that is stochastically greater than the other; 'vanderWaerden' for Van der Waerden test of whether the population distribution functions are equal; 'median' for Mood's test for whether the population medians are equal; and 'KruskalWallis' for Kruskal-Wallis test which is Mann-Whitney U/Wilcoxon generalized to three or more groups.

`freq.tests` Character string or vector of character strings indicating what statistical tests should be used to compare distributions of each categorical row variable across levels of the column variable. Elements should be possible values for the 'statistic' input of the svychisq function in the survey package [1,2]: 'F', 'Chisq', 'Wald', 'adjWald', 'lincom', or 'saddlepoint'.

`decimals` Number of decimal places for various cell entries, such as means and percentages. Does not affect p-values.

`p.include` If FALSE, statistical test is not performed and p-value is not returned.

`p.decimals` Number of decimal places for p-values. If a vector is provided rather than a single value, number of decimal places will depend on what range the p-value lies in. See p.cuts.

`p.cuts` Cut-point(s) to control number of decimal places used for p-values. For example, by default p.cuts is 0.1 and p.decimals is c(2, 3). This means that p-values in the range [0.1, 1] will be printed to two decimal places, while p-values in the range [0, 0.1) will be printed to three decimal places.

`p.lowerbound` Controls cut-point at which p-values are no longer printed as their value, but rather <lowerbound. For example, by default p.lowerbound is 0.001. Under this setting, p-values less than 0.001 are printed as <0.001.

`p.leading0` If TRUE, p-values are printed with 0 before decimal place; if FALSE, the leading 0 is omitted.

`p.avoid1` If TRUE, p-values rounded to 1 are not printed as 1, but as >0.99 (or similarly depending on values for p.decimals and p.cuts).

`n.column` If TRUE, the table will have a column for (unweighted) sample size.

`n.headings` If TRUE, the table will indicate the (unweighted) sample size overall and in each group in parentheses after the column headings.

`se` If TRUE, the table will present mean (standard error) rather than mean (standard deviation) for continuous row variables.

`compress` If TRUE, categorical row variables with two levels will have a single row for n (percent) for the higher level. For example, if a row variable is sex, with 0 for females and 1 for males, setting compress = TRUE would result in the sex row showing n (percent) for males only. If FALSE, the table would have two rows for sex, one showing n (percent) for males and another sohwing n (percent) for females.

`parenth` For median comparisons, controls what values (if any) are placed in parentheses after the medians in each cell. Possible choices are as follows: 'minmax' for minimum and maximum; 'range' for difference between minimum and maximum; 'q1q3' for first and third quartiles; 'iqr' for difference between first and third quartiles; or 'none' for no parentheses at all.

`text.label` For median comparisons, optional text to put after the variable name. For example, if parenth is 'q1q3' and yname is 'BMI' the default label would be 'BMI, Median (Q1-Q3)'. You might prefer to set text.label to something like 'Med (Quartile 1-Quartile 3)' instead.

`parenth.sep` For median comparisons, optional character specifying the separator for the two numbers in parentheses when parenth is set to 'minmax' or 'q1q3'. The default is a dash, so values in the table are formatted as Median (Lower-Upper). If you set parenth.sep to ', ' the values in the table would instead be formatted as Median (Lower, Upper).

`bold.colnames` If TRUE, column headings are printed in bold font. Only applies if latex = TRUE.

`bold.varnames` If TRUE, variable name in the first column of the table is printed in bold font. Only applies if latex = TRUE.

`bold.varlevels` If TRUE, levels of categorical y variables are printed in bold font. Only applies if latex = TRUE.

`variable.colname` Character string with desired heading for first column of table, which shows the y variable name and levels.

Details

Please see help files for tabmeans.svy, tabmedians.svy, and tabfreq.svy for details on statistical tests.

Value

A character matrix comparing mean/medians/frequencies of row variables across levels of the column variable. If you click on the matrix name under "Data" in the RStudio Workspace tab, you will see a clean table that you can copy and paste into a statistical report or manuscript. If latex is set to TRUE, the character matrix will be formatted for inserting into an Sweave or Knitr report using the xtable package [3].

Note

If you have suggestions for additional options or features, or if you would like some help using any function in the package tab, please e-mail me at vandomed@gmail.com. Thanks!

Author(s)

Dane R. Van Domelen

References

1. Lumley T (2012). survey: analysis of complex survey samples. R package version 3.28-2, https://cran.r-project.org/package=survey.

2. Lumley T (2014). Analysis of complex survey samples. Journal of Statistical Software 9(1): 1-19.

3. Dahl DB (2013). xtable: Export tables to LaTeX or HTML. R package version 1.7-1, https://cran.r-project.org/package=xtable.

Acknowledgment: This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

See Also

svydesign, svyglm, svychisq, svyquantile, svyranktest, tabfreq, tabmeans, tabmedians, tabglm, tabcox, tabgee, tabfreq.svy, tabmeans.svy, tabmedians.svy, tabglm.svy

Examples

```
NA
```

Index