Package 'tableone'

November 23, 2018

Type Package

Title Create 'Table 1' to Describe Baseline Characteristics

Version 0.9.3

Date 2018-04-28

Author Kazuki Yoshida, Justin Bohn.

Maintainer Kazuki Yoshida <kazukiyoshida@mail.harvard.edu>

Description Creates 'Table 1', i.e., description of baseline patient characteristics, which is essential in every medical research. Supports both continuous and categorical variables, as well as p-values and standardized mean differences. Weighted data are supported via the 'survey' package. See 'github' for a screen cast. 'tableone' was inspired by descriptive statistics functions in 'Deducer' , a Java-based GUI package by Ian Fellows. This package does not require GUI or Java, and intended for command-line users.

License GPL-2

Imports survey, MASS, e1071, zoo, gmodels, nlme, lmerTest, labelled, methods

Suggests survival, testthat, Matrix, dummies, Matching, reshape2, ggplot2, knitr, geepack, lme4

URL https://github.com/kaz-yos/tableone

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-04-29 04:23:05 UTC

R topics documented:

`tableone-package` *Create "Table 1" to describe baseline characteristics*

Description

Creates "Table 1", i.e., description of baseline patient characteristics, which is essential in every medical research. Supports both continuous and categorical variables, as well as p-values and standardized mean differences. Weighted data are supported via the survey package. See github for a screencast. tableone was inspired by descriptive statistics functions in Deducer , a Java-based GUI package by Ian Fellows. This package does not require GUI or Java, and intended for command-line users. Most important functions are `CreateTableOne` and `svyCreateTableOne`.

Note

Acknowledgement:

Ian Fellows for developing the `deducer` package, which this package is based on.

Hadley Wickham for packaging advice and for creating tools this package was made with (`roxygen2`, `devtools`, `testthat`).

Yoshinobu Kanda for design advice and for integration into `RcmdrPlugin.EZR`.

H.Tachibana and Hiroki Matsui for inputs regarding standardized mean differences.

jomuller, Raja Sriswan Mamidi, Atsushi Shiraishi, and Jacques Ropers for bug reports and/or feature suggestions.

Members of the Facebook Organization of R Users for Medical Statistics in Japan (FORUMS-J) for testing pre-release versions and suggestions.

Developmental repository is on github. Your contributions are appreciated.

`https://github.com/kaz-yos/tableone`

Author(s)

Kazuki Yoshida, Justin Bohn

Maintainer: Kazuki Yoshida <kazukiyoshida@mail.harvard.edu>

See Also

CreateTableOne, svyCreateTableOne, print.TableOne, summary.TableOne, ShowRegTable

Examples

```
## See examples for CreateTableOne and svyCreateTableOne
```

CreateCatTable *Create an object summarizing categorical variables*

Description

Create an object summarizing categorical variables optionally stratifying by one or more startifying variables and performing statistical tests. Usually, CreateTableOne should be used as the universal frontend for both continuous and categorical data.

Usage

```
CreateCatTable(vars, strata, data, includeNA = FALSE, test = TRUE,
  testApprox = chisq.test, argsApprox = list(correct = TRUE),
  testExact = fisher.test, argsExact = list(workspace = 2 * 10^5),
  smd = TRUE)
```

Arguments

vars	Variable(s) to be summarized given as a character vector.
strata	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
data	A data frame in which these variables exist. All variables (both vars and strata) must be in this data frame.
includeNA	If TRUE, NA is handled as a regular factor level rather than missing. NA is shown as the last factor level in the table. Only effective for categorical variables.
test	If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed. Both tests that require the large sample approximation and exact tests are performed. Either one of the result can be obtained from the print method.
testApprox	A function used to perform the large sample approximation based tests. The default is chisq.test. This is not recommended when some of the cell have small counts like fewer than 5.
argsApprox	A named list of arguments passed to the function specified in testApprox. The default is list(correct = TRUE), which turns on the continuity correction for chisq.test.
testExact	A function used to perform the exact tests. The default is fisher.test. If the cells have large numbers, it will fail because of memory limitation. In this situation, the large sample approximation based should suffice.

argsExact	A named list of arguments passed to the function specified in testExact. The default is list(workspace = 2*10^5), which specifies the memory space allocated for fisher.test.
smd	If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Value

An object of class CatTable.

Author(s)

Kazuki Yoshida (based on Deducer::frequencies())

See Also

CreateTableOne, print.CatTable, summary.CatTable

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
head(pbc)

## Create an overall table for categorical variables
catVars <- c("status","ascites","hepato","spiders","edema","stage")
catTableOverall <- CreateCatTable(vars = catVars, data = pbc)

## Simply typing the object name will invoke the print.CatTable method,
## which will show the sample size, frequencies and percentages.
## For 2-level variables, only the higher level is shown for simplicity
## unless the variables are specified in the cramVars argument.
catTableOverall

## If you need to show both levels for some 2-level factors, use cramVars
print(catTableOverall, cramVars = "hepato")

## Use the showAllLevels argument to see all levels for all variables.
print(catTableOverall, showAllLevels = TRUE)

## You can choose form frequencies ("f") and/or percentages ("p") or both.
## "fp" frequency (percentage) is the default. Row names change accordingly.
print(catTableOverall, format = "f")
print(catTableOverall, format = "p")

## To further examine the variables, use the summary.CatTable method,
## which will show more details.
summary(catTableOverall)

## The table can be stratified by one or more variables
```

```
catTableBySexTrt <- CreateCatTable(vars = catVars,
                                   strata = c("sex","trt"), data = pbc)

## print now includes p-values which are by default calculated by chisq.test.
## It is formatted at the decimal place specified by the pDigits argument
## (3 by default). It is formatted like <0.001 if very small.
catTableBySexTrt

## The exact argument toggles the p-values to the exact test result from
## fisher.test. It will show which ones are from exact tests.
print(catTableBySexTrt, exact = "ascites")

## summary now includes both types of p-values
summary(catTableBySexTrt)

## If your work flow includes copying to Excel and Word when writing manuscripts,
## you may benefit from the quote argument. This will quote everything so that
## Excel does not mess up the cells.
print(catTableBySexTrt, exact = "ascites", quote = TRUE)

## If you want to center-align values in Word, use noSpaces option.
print(catTableBySexTrt, exact = "ascites", quote = TRUE, noSpaces = TRUE)
```

`CreateContTable` *Create an object summarizing continous variables*

Description

Create an object summarizing continous variables optionally stratifying by one or more startifying variables and performing statistical tests. Usually, `CreateTableOne` should be used as the universal frontend for both continuous and categorical data.

Usage

```
CreateContTable(vars, strata, data, funcNames = c("n", "miss", "p.miss",
  "mean", "sd", "median", "p25", "p75", "min", "max", "skew", "kurt"),
  funcAdditional, test = TRUE, testNormal = oneway.test,
  argsNormal = list(var.equal = TRUE), testNonNormal = kruskal.test,
  argsNonNormal = list(NULL), smd = TRUE)
```

Arguments

`vars`	Variable(s) to be summarized given as a character vector.
`strata`	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
`data`	A data frame in which these variables exist. All variables (both vars and strata) must be in this data frame.
`funcNames`	The functions to give the group size, number with missing values, proportion with missing values, mean, standard deviations, median, 25th percentile, 75th percentile, minimum, maximum, skewness (same definition as in SAS), kurtosis (same definition as in SAS). All of them can be seen in the summary method

output. The print method uses subset of these. You can choose subset of them or reorder them. They are all configure to omit NA values (`na.rm = TRUE`).

`funcAdditional`
: Additional functions can be given as a named list. For example, `list(sum = sum)`.

`test`
: If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed. Both tests that assume normality and tests that do not are performed. Either one of the result can be obtained from the print method.

`testNormal`
: A function used to perform the normal assumption based tests. The default is `oneway.test`. This is equivalent of the t-test when there are only two groups.

`argsNormal`
: A named list of arguments passed to the function specified in `testNormal`. The default is `list(var.equal = TRUE)`, which makes it the ordinary ANOVA that assumes equal variance across groups.

`testNonNormal`
: A function used to perform the nonparametric tests. The default is `kruskal.test` (Kruskal-Wallis rank sum test). This is equivalent of the wilcox.test (Man-Whitney U test) when there are only two groups.

`argsNonNormal`
: A named list of arguments passed to the function specified in `testNonNormal`. The default is `list(NULL)`, which is just a placeholder.

`smd`
: If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Value

An object of class `ContTable`.

Author(s)

Kazuki Yoshida (based on `Deducer::descriptive.table()`)

See Also

`CreateTableOne`, `print.ContTable`, `summary.ContTable`

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
head(pbc)

## Create an overall table for continuous variables
contVars <- c("time","age","bili","chol","albumin","copper",
              "alk.phos","ast","trig","platelet","protime")
contTableOverall <- CreateContTable(vars = contVars, data = pbc)

## Simply typing the object name will invoke the print.ContTable method,
## which will show the sample size, means and standard deviations.
```

```
contTableOverall

## To further examine the variables, use the summary.ContTable method,
## which will show more details.
summary(contTableOverall)

## c("age","chol","copper","alk.phos","trig","protime") appear highly skewed.
## Specify them in the nonnormal argument, and the display changes to the median,
## and the [25th, 75th] percentile.
nonNormalVars <- c("age","chol","copper","alk.phos","trig","protime")
print(contTableOverall, nonnormal = nonNormalVars)

## To show median [min,max] for nonnormal variables, use minMax = TRUE
print(contTableOverall, nonnormal = nonNormalVars, minMax = TRUE)

## The table can be stratified by one or more variables
contTableBySexTrt <- CreateContTable(vars = contVars,
                                     strata = c("sex","trt"), data = pbc)

## print now includes p-values which are by default calculated by oneway.test (t-test
## equivalent in the two group case). It is formatted at the decimal place specified
## by the pDigits argument (3 by default). It does <0.001 for you.
contTableBySexTrt

## The nonnormal argument toggles the p-values to the nonparametric result from
## kruskal.test (wilcox.test equivalent for the two group case).
print(contTableBySexTrt, nonnormal = nonNormalVars)

## summary now includes both types of p-values
summary(contTableBySexTrt)

## If your work flow includes copying to Excel and Word when writing manuscripts,
## you may benefit from the quote argument. This will quote everything so that
## Excel does not mess up the cells.
print(contTableBySexTrt, nonnormal = nonNormalVars, quote = TRUE)

## If you want to center-align values in Word, use noSpaces option.
print(contTableBySexTrt, nonnormal = nonNormalVars, quote = TRUE, noSpaces = TRUE)
```

`CreateTableOne` *Create an object summarizing both continuous and categorical variables*

Description

Create an object summarizing all baseline variables (both continuous and categorical) optionally stratifying by one or more startifying variables and performing statistical tests. The object gives a table that is easy to use in medical research papers.

Usage

```
CreateTableOne(vars, strata, data, factorVars, includeNA = FALSE,
  test = TRUE, testApprox = chisq.test, argsApprox = list(correct = TRUE),
```

```
  testExact = fisher.test, argsExact = list(workspace = 2 * 10^5),
  testNormal = oneway.test, argsNormal = list(var.equal = TRUE),
  testNonNormal = kruskal.test, argsNonNormal = list(NULL), smd = TRUE)
```

Arguments

vars	Variables to be summarized given as a character vector. Factors are handled as categorical variables, whereas numeric variables are handled as continuous variables. If empty, all variables in the data frame specified in the data argument are used.
strata	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
data	A data frame in which these variables exist. All variables (both vars and strata) must be in this data frame.
factorVars	Numerically coded variables that should be handled as categorical variables given as a character vector. Do not include factors, unless you need to relevel them by removing empty levels. If omitted, only factors are considered categorical variables. The variables specified here must also be specified in the `vars` argument.
includeNA	If TRUE, NA is handled as a regular factor level rather than missing. NA is shown as the last factor level in the table. Only effective for categorical variables.
test	If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed.
testApprox	A function used to perform the large sample approximation based tests. The default is `chisq.test`. This is not recommended when some of the cell have small counts like fewer than 5.
argsApprox	A named list of arguments passed to the function specified in testApprox. The default is `list(correct = TRUE)`, which turns on the continuity correction for `chisq.test`.
testExact	A function used to perform the exact tests. The default is `fisher.test`. If the cells have large numbers, it will fail because of memory limitation. In this situation, the large sample approximation based should suffice.
argsExact	A named list of arguments passed to the function specified in testExact. The default is `list(workspace = 2*10^5)`, which specifies the memory space allocated for `fisher.test`.
testNormal	A function used to perform the normal assumption based tests. The default is `oneway.test`. This is equivalent of the t-test when there are only two groups.
argsNormal	A named list of arguments passed to the function specified in `testNormal`. The default is `list(var.equal = TRUE)`, which makes it the ordinary ANOVA that assumes equal variance across groups.
testNonNormal	A function used to perform the nonparametric tests. The default is `kruskal.test` (Kruskal-Wallis Rank Sum Test). This is equivalent of the wilcox.test (Man-Whitney U test) when there are only two groups.
argsNonNormal	A named list of arguments passed to the function specified in `testNonNormal`. The default is `list(NULL)`, which is just a placeholder.
smd	If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Details

The definitions of the standardized mean difference (SMD) are available in Flury *et al* 1986 for the univariate case and the multivariate case (essentially the square root of the Mahalanobis distance). Extension to binary variables is discussed in Austin 2009 and extension to multinomival variables is suggested in Yang *et al* 2012. This multinomial extesion treats a single multinomial variable as multiple non-redundant dichotomous variables and use the Mahalanobis distance. The off diagonal elements of the covariance matrix on page 3 have an error, and need negation. In weighted data, the same definitions can be used except that the mean and standard deviation estimates are weighted estimates (Li *et al* 2013 and Austin *et al* 2015). In tableone, all weighted estimates are calculated by weighted estimation functions in the `survey` package.

Value

An object of class `TableOne`, which is a list of three objects.

`ContTable`	object of class `ContTable`, containing continuous variables only
`CatTable`	object of class `CatTable`, containing categorical variables only
`MetaData`	list of metadata regarding variables

Author(s)

Kazuki Yoshida, Justin Bohn

References

Flury, BK. and Riedwyl, H. (1986). Standard distance in univariate and multivariate analysis. *The American Statistician*, **40**, 249-251.

Austin, PC. (2009). Using the Standardized Difference to Compare the Prevalence of a Binary Variable Between Two Groups in Observational Research. *Communications in Statistics - Simulation and Computation*, **38**, 1228-1234.

Yang, D. and Dalton, JE. (2012). A unified approach to measuring the effect size between two groups using SAS. SAS Global Forum 2012, Paper 335-2012.

Li, L. and Greene, T. (2013). A weighting analogue to pair matching in propensity score analysis. *International Journal of Biostatistics*, **9**, 215-234.

Austin, PC. and Stuart, EA. (2015). Moving towards best practice when using inverse probability of treatment weighting (IPTW) using the propensity score to estimate causal treatment effects in observational studies. *Statistics in Medicine*, Online on August 3, 2015.

See Also

`print.TableOne`, `summary.TableOne`

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
```

```
head(pbc)

## Make categorical variables factors
varsToFactor <- c("status","trt","ascites","hepato","spiders","edema","stage")
pbc[varsToFactor] <- lapply(pbc[varsToFactor], factor)

## Create a variable list
dput(names(pbc))
vars <- c("time","status","age","sex","ascites","hepato",
          "spiders","edema","bili","chol","albumin",
          "copper","alk.phos","ast","trig","platelet",
          "protime","stage")

## Create Table 1 stratified by trt
tableOne <- CreateTableOne(vars = vars, strata = c("trt"), data = pbc)

## Just typing the object name will invoke the print.TableOne method
tableOne

## Specifying nonnormal variables will show the variables appropriately,
## and show nonparametric test p-values. Specify variables in the exact
## argument to obtain the exact test p-values. cramVars can be used to
## show both levels for a 2-level categorical variables.
print(tableOne, nonnormal = c("bili","chol","copper","alk.phos","trig"),
      exact = c("status","stage"), cramVars = "hepato", smd = TRUE)

## Use the summary.TableOne method for detailed summary
summary(tableOne)

## See the categorical part only using $ operator
tableOne$CatTable
summary(tableOne$CatTable)

## See the continuous part only using $ operator
tableOne$ContTable
summary(tableOne$ContTable)

## If your work flow includes copying to Excel and Word when writing manuscripts,
## you may benefit from the quote argument. This will quote everything so that
## Excel does not mess up the cells.
print(tableOne, nonnormal = c("bili","chol","copper","alk.phos","trig"),
      exact = c("status","stage"), quote = TRUE)

## If you want to center-align values in Word, use noSpaces option.
print(tableOne, nonnormal = c("bili","chol","copper","alk.phos","trig"),
      exact = c("status","stage"), quote = TRUE, noSpaces = TRUE)

## If SMDs are needed as numericals, use ExtractSmd()
ExtractSmd(tableOne)
```

`ExtractSmd` *Extract standardized mean differences from a (svy)TableOne object*

Description

Extracts standardized mean differences data as a vector or matrix from a (svy)TableOne object

Usage

```
ExtractSmd(x, varLabels = FALSE)
```

Arguments

x	A stratified (svy)TableOne object containing standardized mean differences.
varLabels	Whether to replace variable names with variable labels obtained from `labelled::var_label()` function.

Value

A vector or matrix containing the average standardized mean differences (if more than two contrasts exist) as well as the all possible pairwise standardized mean differences. Variables are ordered in the same order as the printed table.

Author(s)

Kazuki Yoshida

See Also

CreateTableOne, svyCreateTableOne

Examples

```
## See examples for CreateTableOne and svyCreateTableOne
```

---

print.CatTable *Format and print* CatTable *class objects*

Description

`print` method for the `CatTable` class objects created by `CreateCatTable` function.

Usage

```
## S3 method for class 'CatTable'
print(x, digits = 1, pDigits = 3, quote = FALSE,
  missing = FALSE, explain = TRUE, printToggle = TRUE, noSpaces = FALSE,
  format = c("fp", "f", "p", "pf")[1], showAllLevels = FALSE,
  cramVars = NULL, dropEqual = FALSE, test = TRUE, exact = NULL,
  smd = FALSE, CrossTable = FALSE, ...)
```

Arguments

x	Object returned by CreateCatTable function.
digits	Number of digits to print in the table.
pDigits	Number of digits to print for p-values (also used for standardized mean differences).
quote	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
missing	Whether to show missing data information.
explain	Whether to add explanation to the variable names, i.e., (%) is added to the variable names when percentage is shown.
printToggle	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
noSpaces	Whether to remove spaces added for alignment. Use this option if you prefer to align numbers yourself in other software.
format	The default is "fp" frequency (percentage). You can also choose from "f" frequency only, "p" percentage only, and "pf" percentage (frequency).
showAllLevels	Whether to show all levels. FALSE by default, i.e., for 2-level categorical variables, only the higher level is shown to avoid redundant information.
cramVars	A character vector to specify the two-level categorical variables, for which both levels should be shown in one row.
dropEqual	Whether to drop " = second level name" description indicating which level is shown for two-level categorical variables.
test	Whether to show p-values. TRUE by default. If FALSE, only the numerical summaries are shown.
exact	A character vector to specify the variables for which the p-values should be those of exact tests. By default all p-values are from large sample approximation tests (chisq.test).
smd	Whether to show standardized mean differences. FALSE by default. If there are more than one contrasts, the average of all possible standardized mean differences is shown. For individual contrasts, use summary.
CrossTable	Whether to show the cross table objects held internally using gmodels::CrossTable function. This will give an output similar to the PROC FREQ in SAS.
...	For compatibility with generic. Ignored.

Value

A matrix object containing what you see is also invisibly returned. This can be assinged a name and exported via write.csv.

Author(s)

Kazuki Yoshida

See Also

CreateTableOne, CreateCatTable, summary.CatTable

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
head(pbc)

## Create an overall table for categorical variables
catVars <- c("status","ascites","hepato","spiders","edema","stage")
catTableOverall <- CreateCatTable(vars = catVars, data = pbc)

## Simply typing the object name will invoke the print.CatTable method,
## which will show the sample size, frequencies and percentages.
## For 2-level variables, only the higher level is shown for simplicity.
catTableOverall

## If you need to show both levels for some 2-level factors, use cramVars
print(catTableOverall, cramVars = "hepato")

## Use the showAllLevels argument to see all levels for all variables.
print(catTableOverall, showAllLevels = TRUE)

## You can choose form frequencies ("f") and/or percentages ("p") or both.
## "fp" frequency (percentage) is the default. Row names change accordingly.
print(catTableOverall, format = "f")
print(catTableOverall, format = "p")

## To further examine the variables, use the summary.CatTable method,
## which will show more details.
summary(catTableOverall)

## The table can be stratified by one or more variables
catTableBySexTrt <- CreateCatTable(vars = catVars,
                                   strata = c("sex","trt"), data = pbc)

## print now includes p-values which are by default calculated by chisq.test.
## It is formatted at the decimal place specified by the pDigits argument
## (3 by default). It does <0.001 for you.
catTableBySexTrt

## The exact argument toggles the p-values to the exact test result from
## fisher.test. It will show which ones are from exact tests.
print(catTableBySexTrt, exact = "ascites")

## summary now includes both types of p-values
summary(catTableBySexTrt)

## If your work flow includes copying to Excel and Word when writing manuscripts,
## you may benefit from the quote argument. This will quote everything so that
## Excel does not mess up the cells.
print(catTableBySexTrt, exact = "ascites", quote = TRUE)
```

```
## If you want to center-align values in Word, use noSpaces option.
print(catTableBySexTrt, exact = "ascites", quote = TRUE, noSpaces = TRUE)
```

`print.ContTable` *Format and print* `ContTable` *class objects*

Description

`print` method for the `ContTable` class objects created by `CreateContTable` function.

Usage

```
## S3 method for class 'ContTable'
print(x, digits = 2, pDigits = 3, quote = FALSE,
  missing = FALSE, explain = TRUE, printToggle = TRUE, noSpaces = FALSE,
  nonnormal = NULL, minMax = FALSE, insertLevel = FALSE, test = TRUE,
  smd = FALSE, ...)
```

Arguments

`x`	Object returned by `CreateContTable` function.
`digits`	Number of digits to print in the table.
`pDigits`	Number of digits to print for p-values (also used for standardized mean differences).
`quote`	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
`missing`	Whether to show missing data information.
`explain`	Whether to add explanation to the variable names, i.e., (mean (sd) or median [IQR]) is added to the variable names.
`printToggle`	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
`noSpaces`	Whether to remove spaces added for alignment. Use this option if you prefer to align numbers yourself in other software.
`nonnormal`	A character vector to specify the variables for which the p-values should be those of nonparametric tests. By default all p-values are from normal assumption-based tests (oneway.test).
`minMax`	Whether to use [min,max] instead of [p25,p75] for nonnormal variables. The default is FALSE.
`insertLevel`	Whether to add an empty level column to the left of strata.
`test`	Whether to show p-values. TRUE by default. If FALSE, only the numerical summaries are shown.
`smd`	Whether to show standardized mean differences. FALSE by default. If there are more than one contrasts, the average of all possible standardized mean differences is shown. For individual contrasts, use `summary`.
`...`	For compatibility with generic. Ignored.

Value

A matrix object containing what you see is also invisibly returned. This can be assinged a name and exported via `write.csv`.

Author(s)

Kazuki Yoshida

See Also

`CreateTableOne`, `CreateContTable`, `summary.ContTable`

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
head(pbc)

## Create an overall table for continuous variables
contVars <- c("time","age","bili","chol","albumin","copper",
              "alk.phos","ast","trig","platelet","protime")
contTableOverall <- CreateContTable(vars = contVars, data = pbc)

## Simply typing the object name will invoke the print.ContTable method,
## which will show the sample size, means and standard deviations.
contTableOverall

## To further examine the variables, use the summary.ContTable method,
## which will show more details.
summary(contTableOverall)

## c("age","chol","copper","alk.phos","trig","protime") appear highly skewed.
## Specify them in the nonnormal argument, and the display changes to the median,
## and the [25th, 75th] percentile.
nonNormalVars <- c("age","chol","copper","alk.phos","trig","protime")
print(contTableOverall, nonnormal = nonNormalVars)

## To show median [min,max] for nonnormal variables, use minMax = TRUE
print(contTableOverall, nonnormal = nonNormalVars, minMax = TRUE)

## The table can be stratified by one or more variables
contTableBySexTrt <- CreateContTable(vars = contVars,
                                     strata = c("sex","trt"), data = pbc)

## print now includes p-values which are by default calculated by oneway.test (t-test
## equivalent in the two group case). It is formatted at the decimal place specified
## by the pDigits argument (3 by default). It does <0.001 for you.
contTableBySexTrt

## The nonnormal argument toggles the p-values to the nonparametric result from
```

```
## kruskal.test (wilcox.test equivalent for the two group case).
print(contTableBySexTrt, nonnormal = nonNormalVars)

## The minMax argument toggles whether to show median [range]
print(contTableBySexTrt, nonnormal = nonNormalVars, minMax = TRUE)

## summary now includes both types of p-values
summary(contTableBySexTrt)

## If your work flow includes copying to Excel and Word when writing manuscripts,
## you may benefit from the quote argument. This will quote everything so that
## Excel does not mess up the cells.
print(contTableBySexTrt, nonnormal = nonNormalVars, quote = TRUE)

## If you want to center-align values in Word, use noSpaces option.
print(contTableBySexTrt, nonnormal = nonNormalVars, quote = TRUE, noSpaces = TRUE)
```

`print.svyCatTable` *Format and print* `svyCatTable` *class objects*

Description

`print` method for the `svyCatTable` class objects created by `svyCreateCatTable` function.

Usage

```
## S3 method for class 'svyCatTable'
print(x, digits = 1, pDigits = 3, quote = FALSE,
  missing = FALSE, explain = TRUE, printToggle = TRUE, noSpaces = FALSE,
  format = c("fp", "f", "p", "pf")[1], showAllLevels = FALSE,
  cramVars = NULL, dropEqual = FALSE, test = TRUE, exact = NULL,
  smd = FALSE, CrossTable = FALSE, ...)
```

Arguments

`x`	The result of a call to the `svyCreateCatTable` function.
`digits`	Number of digits to print in the table.
`pDigits`	Number of digits to print for p-values (also used for standardized mean differences).
`quote`	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
`missing`	Whether to show missing data information.
`explain`	Whether to add explanation to the variable names, i.e., (%) is added to the variable names when percentage is shown.
`printToggle`	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
`noSpaces`	Whether to remove spaces added for alignment. Use this option if you prefer to align numbers yourself in other software.

format	The default is "fp" frequency (percentage). You can also choose from "f" frequency only, "p" percentage only, and "pf" percentage (frequency).
showAllLevels	Whether to show all levels. FALSE by default, i.e., for 2-level categorical variables, only the higher level is shown to avoid redundant information.
cramVars	A character vector to specify the two-level categorical variables, for which both levels should be shown in one row.
dropEqual	Whether to drop " = second level name" description indicating which level is shown for two-level categorical variables.
test	Whether to show p-values. TRUE by default. If FALSE, only the numerical summaries are shown.
exact	This option is not available for tables from weighted data.
smd	Whether to show standardized mean differences. FALSE by default. If there are more than one contrasts, the average of all possible standardized mean differences is shown. For individual contrasts, use summary.
CrossTable	Whether to show the cross table objects held internally using gmodels::CrossTable function. This will give an output similar to the PROC FREQ in SAS.
...	For compatibility with generic. Ignored.

Value

A matrix object containing what you see is also invisibly returned. This can be assinged a name and exported via write.csv.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, svyCreateCatTable, summary.svyCatTable

Examples

```
## See the examples for svyCreateTableOne()
```

---

print.svyContTable *Format and print* svyContTable *class objects*

Description

print method for the svyContTable class objects created by CreateContTable function.

Usage

```
## S3 method for class 'svyContTable'
print(x, digits = 2, pDigits = 3, quote = FALSE,
  missing = FALSE, explain = TRUE, printToggle = TRUE, noSpaces = FALSE,
  nonnormal = NULL, minMax = FALSE, insertLevel = FALSE, test = TRUE,
  smd = FALSE, ...)
```

Arguments

x	Object returned by svyCreateContTable function.
digits	Number of digits to print in the table.
pDigits	Number of digits to print for p-values (also used for standardized mean differences).
quote	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
missing	Whether to show missing data information.
explain	Whether to add explanation to the variable names, i.e., (mean (sd) or median [IQR]) is added to the variable names.
printToggle	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
noSpaces	Whether to remove spaces added for alignment. Use this option if you prefer to align numbers yourself in other software.
nonnormal	A character vector to specify the variables for which the p-values should be those of nonparametric tests. By default all p-values are from normal assumption-based tests (oneway.test).
minMax	Whether to use [min,max] instead of [p25,p75] for nonnormal variables. The default is FALSE.
insertLevel	Whether to add an empty level column to the left of strata.
test	Whether to show p-values. TRUE by default. If FALSE, only the numerical summaries are shown.
smd	Whether to show standardized mean differences. FALSE by default. If there are more than one contrasts, the average of all possible standardized mean differences is shown. For individual contrasts, use summary.
...	For compatibility with generic. Ignored.

Value

A matrix object containing what you see is also invisibly returned. This can be assinged a name and exported via write.csv.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, svyCreateCatTable, summary.svyCatTable

Examples

```
## See the examples for svyCreateTableOne()
```

print.TableOne *Format and print* TableOne *class objects*

Description

print method for the TableOne class objects created by CreateTableOne function.

Usage

```
## S3 method for class 'TableOne'
print(x, catDigits = 1, contDigits = 2, pDigits = 3,
  quote = FALSE, missing = FALSE, explain = TRUE, printToggle = TRUE,
  test = TRUE, smd = FALSE, noSpaces = FALSE, padColnames = FALSE,
  varLabels = FALSE, format = c("fp", "f", "p", "pf")[1],
  showAllLevels = FALSE, cramVars = NULL, dropEqual = FALSE,
  exact = NULL, nonnormal = NULL, minMax = FALSE, ...)
```

Arguments

x	Object returned by CreateTableOne function.
catDigits	Number of digits to print for proportions. Default 1.
contDigits	Number of digits to print for continuous variables. Default 2.
pDigits	Number of digits to print for p-values (also used for standardized mean differences). Default 3.
quote	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
missing	Whether to show missing data information.
explain	Whether to add explanation to the variable names, i.e., (%) is added to the variable names when percentage is shown.
printToggle	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
test	Whether to show p-values. TRUE by default. If FALSE, only the numerical summaries are shown.
smd	Whether to show standardized mean differences. FALSE by default. If there are more than one contrasts, the average of all possible standardized mean differences is shown. For individual contrasts, use summary.
noSpaces	Whether to remove spaces added for alignment. Use this option if you prefer to align numbers yourself in other software.
padColnames	Whether to pad column names with spaces to center justify. The default is FALSE. It is not conducted if noSpaces = TRUE.
varLabels	Whether to replace variable names with variable labels obtained from labelled::var_label() function.
format	The default is "fp" frequency (percentage). You can also choose from "f" frequency only, "p" percentage only, and "pf" percentage (frequency).

`showAllLevels`
: Whether to show all levels. FALSE by default, i.e., for 2-level categorical variables, only the higher level is shown to avoid redundant information.

`cramVars`
: A character vector to specify the two-level categorical variables, for which both levels should be shown in one row.

`dropEqual`
: Whether to drop " = second level name" description indicating which level is shown for two-level categorical variables.

`exact`
: A character vector to specify the variables for which the p-values should be those of exact tests. By default all p-values are from large sample approximation tests (chisq.test).

`nonnormal`
: A character vector to specify the variables for which the p-values should be those of nonparametric tests. By default all p-values are from normal assumption-based tests (oneway.test).

`minMax`
: Whether to use [min,max] instead of [p25,p75] for nonnormal variables. The default is FALSE.

`...`
: For compatibility with generic. Ignored.

Value

A matrix object containing what you see is also invisibly returned. This can be assinged a name and exported via `write.csv`.

Author(s)

Kazuki Yoshida, Justin Bohn

See Also

`CreateTableOne`, `CreateTableOne`, `summary.TableOne`

Examples

```
## See examples for CreateTableOne and svyCreateTableOne
```

ShowRegTable *Format regression results in medically decent format*

Description

It shows the regression result in the HR [95% CI] p-value format, which is usually the form used in medical research papers.

Usage

```
ShowRegTable(model, exp = TRUE, digits = 2, pDigits = 3,
  printToggle = TRUE, quote = FALSE, ciFun = confint)
```

Arguments

model	Regression model result objects that have the summary and confint methods.
exp	TRUE by default. You need to specify exp = FALSE if your model is has the indentity link function (linear regression, etc).
digits	Number of digits to print for the main part.
pDigits	Number of digits to print for the p-values.
printToggle	Whether to print the output. If FALSE, no output is created, and a matrix is invisibly returned.
quote	Whether to show everything in quotes. The default is FALSE. If TRUE, everything including the row and column names are quoted so that you can copy it to Excel easily.
ciFun	Function used for calculation. `confint` is the default. For generalized linear models this gives the profile likelihood-based calculation, which may take too much time for large models, use `confint.default` for simple normal approximation method (+/- 1.96 * standard error).

Value

A matrix containing what you see is returned invisibly. You can capture it by assignment to an object.

Author(s)

Kazuki Yoshida

Examples

```
## Load
library(tableone)

## Load Mayo Clinic Primary Biliary Cirrhosis Data
library(survival)
data(pbc)
## Check variables
head(pbc)

## Fit a Cox regression model
objCoxph <- coxph(formula = Surv(time, status == 2) ~ trt + age + albumin + ascites,
                  data    = pbc)

## Show the simple table
ShowRegTable(objCoxph)

## Show with quote to ease copy and paste
ShowRegTable(objCoxph, quote = TRUE)
```

summary.CatTable *Shows all results in a* CatTable *class object*

Description

Shows all data a CatTable class object has. This includes the (optionally stratified) part with summary statistics and, if available, p-values from the approximation method test (chisq.test by default) and exact method test (fisher.test by default) and standardized mean differences of all possible pairwise contrasts.

Usage

```
## S3 method for class 'CatTable'
summary(object, digits = 1, ...)
```

Arguments

object	An object that has the CatTable class to be shown.
digits	Number of digits to print.
...	For compatibility with generic. Ignored.

Value

None. Results are printed.

Author(s)

Kazuki Yoshida

See Also

CreateTableOne, CreateCatTable, print.CatTable,

Examples

```
## See examples for CreateTableOne
```

summary.ContTable *Shows all results in a* ContTable *class object*

Description

Shows all data a ContTable class object has. This includes the (optionally stratified) part with summary statistics and, if available, p-values from the normal assupmtion-based test (oneway.test by default) and nonparametric test (kruskal.test by default) and standardized mean differences of all possible pairwise contrasts.

Usage

```
## S3 method for class 'ContTable'
summary(object, digits = 2, ...)
```

Arguments

object	An object that has the ContTable class to be shown.
digits	Number of digits to print.
...	For compatibility with generic. Ignored.

Value

None. Results are printed.

Author(s)

Kazuki Yoshida

See Also

CreateTableOne, CreateContTable, print.ContTable

Examples

```
## See examples for CreateTableOne
```

---

summary.svyCatTable

Shows all results in a svyCatTable *class object*

Description

Shows all data a svyCatTable class object has. This includes the (optionally stratified) part with summary statistics and, if available, p-values from the approximation method test (chisq.test by default) and exact method test (fisher.test by default) and standardized mean differences of all possible pairwise contrasts.

Usage

```
## S3 method for class 'svyCatTable'
summary(object, digits = 1, ...)
```

Arguments

object	An object that has the svyCatTable class to be shown.
digits	Number of digits to print.
...	For compatibility with generic. Ignored.

Value

None. Results are printed.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, svyCreateCatTable, print.svyCatTable

Examples

```
## See the examples for svyCreateTableOne()
```

---

summary.svyContTable

Shows all results in a svyContTable *class object*

Description

Shows all data a svyContTable class object has. This includes the (optionally stratified) part with summary statistics and, if available, p-values from the normal assupmtion-based test (regTermTest with svyglm by default) and nonparametric test (svyranktest by default) and standardized mean differences of all possible pairwise contrasts.

Usage

```
## S3 method for class 'svyContTable'
summary(object, digits = 2, ...)
```

Arguments

object	An object that has the svyContTable class to be shown.
digits	Number of digits to print.
...	For compatibility with generic. Ignored.

Value

None. Results are printed.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, svyCreateContTable, print.svyContTable

Examples

```
## See the examples for svyCreateTableOne()
```

`summary.TableOne` *Shows all results in a* `(svy)TableOne` *class object*

Description

Shows all data a `(svy)TableOne` class object has. This includes the (optionally stratified) part with summary statistics and p-values and/or standardized mean differences.

Usage

```
## S3 method for class 'TableOne'
summary(object, digits = 1, ...)
```

Arguments

`object`	An object that has the `(svy)TableOne` class to be shown.
`digits`	Number of digits to print.
`...`	For compatibility with generic. Ignored.

Value

None. Results are printed.

Author(s)

Kazuki Yoshida

See Also

`CreateTableOne`, `svyCreateCatTable`

Examples

```
## See examples for CreateTableOne and svyCreateTableOne
```

svyCreateCatTable *Create an object summarizing categorical variables for weighted data*

Description

Create an object summarizing categorical variables optionally stratifying by one or more startifying variables and performing statistical tests. Usually, svyCreateTableOne should be used as the universal frontend for both continuous and categorical data.

Usage

```
svyCreateCatTable(vars, strata, data, includeNA = FALSE, test = TRUE,
  testApprox = svyTestChisq, argsApprox = NULL, smd = TRUE)
```

Arguments

vars	Variable(s) to be summarized given as a character vector.
strata	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
data	A survey design object in which these variables exist. All variables (both vars and strata) must be in this survey design object. It is created with the svydesign function in the survey package.
includeNA	If TRUE, NA is handled as a regular factor level rather than missing. NA is shown as the last factor level in the table. Only effective for categorical variables.
test	If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed. Both tests that require the large sample approximation and exact tests are performed. Either one of the result can be obtained from the print method.
testApprox	A function used to perform the large sample approximation based tests. The default is svychisq.
argsApprox	A named list of arguments passed to the function specified in testApprox.
smd	If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Value

An object of class svyCatTable.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, print.svyCatTable, summary.svyCatTable,

Examples

```
## See the examples for svyCreateTableOne()
```

---

svyCreateContTable *Create an object summarizing continous variables for weighted dataset*

Description

Create an object summarizing continous variables optionally stratifying by one or more startifying variables and performing statistical tests. Usually, svyCreateTableOne should be used as the universal frontend for both continuous and categorical data.

Usage

```
svyCreateContTable(vars, strata, data, test = TRUE,
  testNormal = svyTestNormal, argsNormal = list(method = "Wald"),
  testNonNormal = svyTestNonNormal, argsNonNormal = NULL, smd = TRUE)
```

Arguments

vars	Variable(s) to be summarized given as a character vector.
strata	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
data	A survey design object in which these variables exist. All variables (both vars and strata) must be in this survey design object. It is created with the svydesign function in the survey package.
test	If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed. Both tests that assume normality and tests that do not are performed. Either one of the result can be obtained from the print method.
testNormal	A function used to perform the normal assumption based tests. The default is multiple degrees of freedom test using svyglm and regTermTest. This is equivalent of the svyttest when there are only two groups.
argsNormal	A named list of arguments passed to the function specified in testNormal.
testNonNormal	A function used to perform the nonparametric tests. The default is svyranktest.
argsNonNormal	A named list of arguments passed to the function specified in testNonNormal.
smd	If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Value

An object of class svyContTable.

Author(s)

Kazuki Yoshida

See Also

svyCreateTableOne, print.svyContTable, summary.svyContTable,

Examples

```
## See the examples for svyCreateTableOne()
```

---

svyCreateTableOne *Create an object summarizing both continuous and categorical variables for weighted data*

Description

Create an object summarizing all baseline variables (both continuous and categorical) optionally stratifying by one or more startifying variables and performing statistical tests. The object gives a table that is easy to use in medical research papers.

Usage

```
svyCreateTableOne(vars, strata, data, factorVars, includeNA = FALSE,
  test = TRUE, testApprox = svyTestChisq, argsApprox = NULL,
  testNormal = svyTestNormal, argsNormal = list(method = "Wald"),
  testNonNormal = svyTestNonNormal, argsNonNormal = NULL, smd = TRUE)
```

Arguments

vars	Variables to be summarized given as a character vector. Factors are handled as categorical variables, whereas numeric variables are handled as continuous variables. If empty, all variables in the survey design object specified in the data argument are used.
strata	Stratifying (grouping) variable name(s) given as a character vector. If omitted, the overall results are returned.
data	A survey design object in which these variables exist. All variables (both vars and strata) must be in this survey design object. It is created with the svydesign function in the survey package.
factorVars	Numerically coded variables that should be handled as categorical variables given as a character vector. Do not include factors, unless you need to relevel them by removing empty levels. If omitted, only factors are considered categorical variables. The variables specified here must also be specified in the vars argument.
includeNA	If TRUE, NA is handled as a regular factor level rather than missing. NA is shown as the last factor level in the table. Only effective for categorical variables.
test	If TRUE, as in the default and there are more than two groups, groupwise comparisons are performed.
testApprox	A function used to perform the large sample approximation based tests. The default is svychisq.

argsApprox	A named list of arguments passed to the function specified in testApprox.
testNormal	A function used to perform the normal assumption based tests. The default is multiple degrees of freedom test using svyglm and regTermTest. This is equivalent of the svyttest when there are only two groups.
argsNormal	A named list of arguments passed to the function specified in testNormal.
testNonNormal	A function used to perform the nonparametric tests. The default is svyranktest.
argsNonNormal	A named list of arguments passed to the function specified in testNonNormal.
smd	If TRUE, as in the default and there are more than two groups, standardized mean differences for all pairwise comparisons are calculated.

Details

See the details for CreateTableOne.

Value

An object of class svyTableOne, which is a list of three objects.

ContTable	an object of class svyContTable, containing continuous variables only
CatTable	an object of class svyCatTable, containing categorical variables only
MetaData	list of metadata regarding variables

Author(s)

Kazuki Yoshida

See Also

print.TableOne, summary.TableOne

Examples

```
## Load packages
library(tableone)
library(survey)

## Create a weighted survey design object
data(nhanes)
nhanesSvy <- svydesign(ids = ~ SDMVPSU, strata = ~ SDMVSTRA, weights = ~ WTMEC2YR,
                       nest = TRUE, data = nhanes)

## Create a table object
## factorVars are converted to factors; no need for variables already factors
## strata will stratify summaries; leave it unspecified for overall summaries
tab1 <- svyCreateTableOne(vars = c("HI_CHOL","race","agecat","RIAGENDR"),
                          strata = "RIAGENDR", data = nhanesSvy,
                          factorVars = c("race","RIAGENDR"))

## Detailed output
summary(tab1)
```

```
## Default formatted printing
tab1

## nonnormal specifies variables to be shown as median [IQR]
print(tab1, nonnormal = "HI_CHOL", contDigits = 3, catDigits = 2,
      pDigits = 4, smd = TRUE)

## minMax changes it to median [min, max]
print(tab1, nonnormal = "HI_CHOL", minMax = TRUE, contDigits = 3,
      catDigits = 2, pDigits = 4, smd = TRUE)

## showAllLevels can be used tow show levels for all categorical variables
print(tab1, showAllLevels = TRUE, smd = TRUE)

## To see all printing options
?print.TableOne

## To examine categorical variables only
tab1$CatTable

## To examine continuous variables only
tab1$ContTable

## If SMDs are needed as numericals, use ExtractSmd()
ExtractSmd(tab1)
```

Index