Package 'psytabs'

April 12, 2016

Type Package

Title Produce Well-Formatted Tables for Psychological Research

Version 1.0

Date 2016-10-04

Author Johannes Beller

Maintainer Johannes Beller <johannesbeller@gmail.com>

Imports psych, plyr, rtf, R2HTML, mokken, lavaan

Suggests quantreg, semTools

Description Produces tables conforming to ``psychological style'' (i.e. APA style) based on standard R output. The resulting tables can be exported to '.rtf', '.html' or '.doc' format.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2016-04-12 01:24:46

R topics documented:

`psytabs-package` *Produce well-formatted tables for psychological research*

Description

Psytabs produces tables conforming to "psychological style" (i.e. APA style) in the .rtf, .html or .doc format.

Details

Package:	psytabs
Type:	Package
Version:	1.0
Date:	2016-10-04
License:	GPL-2

Author(s)

Johannes Beller

Maintainer: Johannes Beller <johannesbeller@gmail.com>

`addPercent` *Add percent-signs to a table*

Description

Internal function. Should not be used.

Usage

```
addPercent(x)
```

Arguments

x	Table.

Value

A table with added percent-signs.

---

addPercentToCount *Add percent-values to a table.*

Description

Internal function. Should not be used.

Usage

```
addPercentToCount(x, y)
```

Arguments

x	Table containing absolute values.
y	Table containing values in percent.

Value

A table containing both absolute values and values in percent.

---

compareModels *Compare two cfa models fitted by lavaan.*

Description

Internal function. Should not be used.

The function is adapted from the lavaan package.

Usage

```
compareModels(fm0, fm1, scaled = FALSE, fm0.scaling = 1, fm1.scaling = 1)
```

Arguments

fm0	First model.
fm1	Second model.
scaled	Is chisq scaled, i.e. a Satorra-Bentler-scaled chi-squared statistic?
fm0.scaling	Scaling factor of first model.
fm1.scaling	Scaling factor of second model.

Value

Some model comparison statistics.

Examples

```
### to be added
1+1
```

`corTable` *Correlation matrix table.*

Description

Produces a correlation matrix table.

Usage

```
corTable(data, use = "pairwise", method = "pearson", round = 2,
significance = "stars", sd = FALSE, mean.sd.cols = FALSE)
```

Arguments

`data`	data.frame containing the variables for which the correlation matrix should be calculated.
`use`	Which observations should be used? use="pairwise" is the default value and will do pairwise deletion of cases. use="complete" will select only complete cases.
`method`	Which correlation type should be used? method="pearson" is the default value. The alternatives to be passed to cor are "spearman" and "kendall".
`round`	numeric values that denotes to what decimal point should be rounded.
`significance`	character vector that specifies if p-values and/or significance stars should be included in the table. ="NA" displays no significances. ="stars" displays significance stars. ="p-values" displays p-values and =c("stars", "p-values") displays both stars and p-values.
`sd`	logical value that toggles Whether the standard deviation should be displayed in the diagonal of the correlation matrix.
`mean.sd.cols`	logical value that toggles Whether additional mean and standard columns should be included in the table.

Value

A dataframe comprising the correlation matrix table.

Examples

```
data(mydata)
corTable(mydata[,1:4])
corTable(mydata[,1:4], method = "kendall")
corTable(mydata[,1:4], sd = TRUE)
corTable(mydata[,1:4], use = "complete")
corTable(mydata[,1:4], significance = NA)
corTable(mydata[,1:4], significance = c("stars", "p-values"))
corTable(mydata[,1:4], round = 4)
corTable(mydata[,1:4], mean.sd.cols = TRUE)
(cor.tab <- corTable(mydata[,1:4], significance = "stars", mean.sd.cols = TRUE))

#saveTable(cor.tab, "corTab.rtf")
```

---

demographicTable *Demographic table.*

Description

Produces a table of the distribution of demographic characteristics.

Usage

```
demographicTable(hor_fact, ver_fact, count = TRUE, percent = TRUE, header = TRUE)
```

Arguments

hor_fact	factor constituting the columns of the table.
ver_fact	factor constituting the rows of the table.
count	logical value that toggles whether to include the absolute values in the table.
percent	logical value that toggles whether to include the values in percent in the table.
header	logical value that toggles whether to include a header for the row factor.

Value

A dataframe constituting the demographic table.

Examples

```
data(mydata)

tab.1 <- demographicTable(mydata$sex, mydata$age_group7)
tab.1
tab.2 <- demographicTable(mydata$sex, mydata$age_group7, count=FALSE)
tab.2
tab.3 <- demographicTable(mydata$sex, mydata$age_group7, percent=FALSE)
tab.3

#saveTable(tab.1, "demographicTable.rtf")
```

efaTable *Exploratory factor analysis table.*

Description

Produces a table of the results of exploratory factor analysis.

Usage

```
efaTable(fa.res, data, mean.sd = TRUE)
```

Arguments

fa.res	results of exploratory factor analysis as returned by the fa function of the psych package.
data	data.frame on which the EFA was conducted.
mean.sd	logical value that indicates whether means and standard derivitions should be included in the table.

Value

An EFA table.

Examples

```
data(mydata)
library(psych)
res <- fa(r = mydata[,c("item1", "item2", "item3", "item4")], nfactors = 2)
tab.1 <- efaTable(res, mydata[,c("item1", "item2", "item3", "item4")])
tab.1
tab.2 <- efaTable(res, mydata[,c("item1", "item2", "item3", "item4")], mean.sd = FALSE)
tab.2

#saveTable(tab.1, "efaTable.rtf")
```

freq *Calculate frequency.*

Description

Internal function. Should not be used.

Usage

```
freq(x)
```

Arguments

x	factor for which the frequency information should be obtained.

Value

A dataframe comprising the frequency information.

Examples

```
1#1
```

---

frequencyTable *Frequency table.*

Description

Produces a simple frequeny table.

Usage

```
frequencyTable(factors)
```

Arguments

factors	data.frame containing the variables for which the frequency table should be calculated.

Value

A dataframe comprising the frequency table.

Examples

```
data(mydata)
(freq.tab <- frequencyTable(data.frame(Sex=mydata$sex, Age=mydata$age_group7)))

#saveTable(freq.tab, "freqTab.rtf")
```

linearRegressionTable *Regression analysis table.*

Description

Produces a table of the results of a regression analysis.

Usage

```
linearRegressionTable(model.fit)
```

Arguments

model.fit	results of a regression analysis as returned by the lm function. Additionally the robust regression functions of the robust package are supported.

Value

A linear regression table.

Examples

```
data(iris)
fit <- lm(Sepal.Length ~ Petal.Width + Species, data = iris)
(tab <- linearRegressionTable(fit))

#saveTable(tab, "linearRegressionTable.rtf")
```

makeNumeric *Make numeric*

Description

Internal function. Should not be used.

Usage

```
makeNumeric(x)
```

Arguments

x	variable to be made numeric.

Value

A numeric variable.

Examples

```
1#1
```

`meanTable` *Mean table.*

Description

Produces a 2- or 1-dimensional table of means (and a measure of variation) regarding a variable and one or two factors.

Usage

```
meanTable(value, ver.factor, hor.factor = NA, variation="se")
```

Arguments

`value`	numeric vector, for which the table should be created.
`ver.factor`	factor constituting the rows of the table.
`hor.factor`	factor constituting the columns of the table.
`variation`	can be either "se" or "sd". variation="se" computes the standard errors and variation="sd" the standard deviation.

Value

A mean table.

Examples

```
data(mydata)
mydata.sumscore <- rowSums(mydata[,c("item1", "item2", "item3", "item4")])
meanTable(mydata.sumscore, mydata$age_group7)
meanTable(mydata.sumscore, mydata$age_group7, mydata$sex)
meanTable(mydata.sumscore, mydata$sex, mydata$age_group7)
(tab <- meanTable(mydata.sumscore, mydata$age_group7, mydata$sex, variation="sd"))

#saveTable(tab, "meanTable.rtf")
```

`measurementInvarianceTable` *Measurement invariance table.*

Description

Produces a table summarizing the results of a measurement invariance analysis as conducted by the respective function of the lavaan and semTools package.

Usage

```
measurementInvarianceTable(measurement.invariance)
```

Arguments

`measurement.invariance`
: Results returned by the measurementInvariance function of the lavaan or semTools package.

Details

Please note that if the scaled chi-squared statistic is used a special chi-squared difference test is calculated, because the difference between two scaled chi-square statistics does not follow the chi-squared distribution. See also http://www.statmodel.com/chidiff.shtml.

Value

A measurement invariance table.

Examples

```
#Must have semTools and lavaan installed
library(semTools)
library(lavaan)

#Example taken from the semTools package
HW.model <- ' visual =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed =~ x7 + x8 + x9 '

mi.result <- measurementInvariance(HW.model,
data=HolzingerSwineford1939, group="school")
tab.1 <- measurementInvarianceTable(mi.result)
tab.1

mi.strict.result <- measurementInvariance(HW.model,
data=HolzingerSwineford1939, strict=TRUE, group="school")
tab.2 <- measurementInvarianceTable(mi.strict.result)
tab.2
```

```
mi.robust.result <- measurementInvariance(HW.model,
data=HolzingerSwineford1939, estimator="MLM", group="school")
tab.3 <- measurementInvarianceTable(mi.robust.result)
tab.3

mi.robstrict.result <- measurementInvariance(HW.model,
data=HolzingerSwineford1939, estimator="MLM", strict=TRUE, group="school")
tab.4 <- measurementInvarianceTable(mi.robstrict.result)
tab.4

#saveTable(tab.2, "measurementInvarianceTable.rtf")
```

---

mokkenTable *Mokken table.*

Description

Produces a table summarizing the results of mokken analyses.

Usage

```
mokkenTable(data)
```

Arguments

data	data.frame containing the variables (i.e. items) for which the table should be created.

Value

A mokken table.

Examples

```
data(mydata)
mokkenTable(mydata[,1:4])
#saveTable(mokkenTable(mydata[,1:4]), "mokkentable.rtf")
```

mydata *Simulated data.*

Description

A simulated data frame made with help of the psych package.

Usage

```
data(mydata)
```

Format

A data frame with 500 observations on the following 7 variables.

`item1` a numeric vector

`item2` a numeric vector

`item3` a numeric vector

`item4` a numeric vector

`sex` a factor with levels `Female` `Male`

`age_group7` a factor with levels `< 21` `> 60` `21-30` `31-40` `41-50` `51-60`

`employment` a factor with levels `Employed` `Not employed`

Examples

```
data(mydata)
str(mydata)
summary(mydata)
```

norms *Establish norms.*

Description

Internal function. Should not be used.

Usage

```
norms(sumscores, from, to, statistics = "PR")
```

Arguments

sumscores	The sumscore vector for which norms should be created.
from	numeric value. Lowest possible sumscore as a numeric value.
to	numeric value. Highest possible sumscore as a numeric value.
statistics	character vector that toggles which norm statistics are included in the norm table. Currently Percent ranks "PR", z-Statistic "z", Z-Statistic "Z", IQ-Stiatistc "IQ", T-Statistic "T" and Stanine "Stanine" are supported.

Value

Norms.

Examples

```
###
1+1
```

---

normTable *Norm table.*

Description

Produces a table of norms.

Usage

```
normTable(sumscores, from, to, statistics = "PR", group = NA, as.list = FALSE)
```

Arguments

sumscores	The sumscore vector for which norms should be created.
from	numeric value. Lowest possible sumscore as a numeric value.
to	numeric value. Highest possible sumscore as a numeric value.
statistics	character vector that toggles which norm statistics are included in the norm table. Currently Percent ranks "PR", z-Statistic "z", Z-Statistic "Z", IQ-Stiatistc "IQ", T-Statistic "T" and Stanine "Stanine" are supported.
group	List of subgroups by which the norms should be created.
as.list	logical vector that toggles whether the norm table should rather be returned as a (vertical) list than as a table. This option is useful if you have a lot of subgroups and/or norm statistics and the resulting table would not fit on the page horizontally. Note that currently when you set as.list = TRUE the resulting list cannot be saved by the saveTable function.

Details

The different norm statistics (besides the percent ranks) are created by transforming the z-Statistic. This means that at first the sumscores are z-Transformed and then the other statistics are calculated based on the z-Statistic.

One could also choose to first "normalize" the z-Statistic, i.e. calculate the percent ranks and then assign the z-values to the sumscores according to the percent ranks of the standard normal distribution. This option is—albeit it is sometimes called the scientific standard—debatle because if the population does not follow the normal curve then the normalisation will lead to an artificial spreading or shortening of scores. This leads to the scores having only an ordinal scale quality even when the scores in the population distribution had originally metric scale quality. Thus this option should be used with caution and only on a strong theoretical basis. Therefore this option is currently not implemented.

Value

A dataframe constituting the norm table.

Examples

```
data(mydata)
mydata.sumscore <- rowSums(mydata[,c("item1", "item2", "item3", "item4")])

tab.1 <- normTable(mydata.sumscore, from = 0, to = 12, statistics=c("PR"))
tab.1
tab.2 <- normTable(mydata.sumscore, from = 0,
to = 12, statistics=c("PR", "T", "Stanine"))
tab.2
tab.3 <- normTable(mydata.sumscore, from = 0,
to = 12, statistics=c("PR"), group=mydata$sex)
tab.3
tab.4 <- normTable(mydata.sumscore, from = 0,
to = 12, statistics=c("PR", "T"), group=mydata$employment)
tab.4
tab.5 <- normTable(mydata.sumscore, from = 0,
to = 12, statistics=c("T"), group=list(mydata$sex, mydata$employment))
tab.5
list.5 <- normTable(mydata.sumscore, from = 0,
to = 12, statistics=c("PR", "T", "Z", "z"),
group=list(mydata$sex, mydata$employment), as.list=TRUE)
list.5
#saveTable(tab.2, "normTable.rtf")
```

---

quantregTable *Quantile regression table.*

Description

Produces a quantile regression table.

Usage

```
quantregTable(x, digits = 2, significance="none")
```

Arguments

x	summary for quantile regression results as returned when calling summary on an quantile regression object.
digits	number of digits to display.
significance	factor that toggles whether beside the standard error significance should be made visible. Can be either "none" nothing additional is displayed, "stars" significance stars are displayed and "bold" signicant values are bold when saved by the saveTable function.

Value

A quantreg table.

Examples

```
#must have quantreg installed
library(quantreg)
data(stackloss)
y <- stack.loss
x <- stack.x
res <- summary(rq(y ~ x, tau=c(0.25, 0.5, 0.75)), se="boot")
quantregTable(res)
quantregTable(res, significance="stars")
tab <- quantregTable(res, significance="bold")
```

---

regressionTable *Regression analysis table.*

Description

Produces a table of the results of a regression analysis.

Usage

```
regressionTable(model.fit)
```

Arguments

model.fit	results of a regression analysis as returned by the lm or glm functions. Additionally the robust regression functions of the robust package are supported.

Value

A regression table.

Examples

```
data(iris)
fit <- lm(Sepal.Length ~ Petal.Width + Species, data = iris)
(tab <- regressionTable(fit))

#Must have the robust package instaled
#fit.rob <- lmrob(Sepal.Length ~ Petal.Width + Species, data = iris)
#regressionTable(fit.rob)

#saveTable(tab, "regressionTable.rtf")

#Logisitc Regression currently works not correctly.
# data(anorexia, package="MASS")
# anorexia.logistic <- anorexia[anorexia$Treat != "FT",]
# fit <- glm(Treat ~ Postwt, family = binomial(), data = anorexia.logistic)
#
# regressionTable(fit)
```

---

saveTable *Save a table.*

Description

Saves a table either in rtf-format (recommended) or in HTML-format.

Usage

```
saveTable(x, file, HTML = FALSE, post.editing = TRUE)
```

Arguments

x	data.frame as generated by the different xTable functions in this package (for example by the mokkenTable function).
file	string; filename the table should be saved as.
HTML	logical value that toggles whether the file format should be HTML.
post.editing	logical value that toggles whether the tables should be post-edited. If true, hard-coded changes to the final table are made. See the details section for further informations.

Details

If post-editing is set to true then the code, which is generated by the RTF package, is changed a posteriori.

Value

None

Examples

```
data(mydata)
test <- demographicTable(mydata$sex, mydata$age_group7, percent=FALSE)
test

#saveTable(test, "test.rtf")
```

`standarderror` *Standard error.*

Description

Computes the standard error of a numeric vector.

Usage

```
standarderror(x)
```

Arguments

`x`	a numerical vector.

Value

The standard error.

Examples

```
data(mydata)
mydata.sumscore <- rowSums(mydata[,c("item1", "item2", "item3", "item4")])
standarderror(mydata.sumscore)
```

Index