Package 'MDMA'

April 25, 2024

Title Mathijs Deen's Miscellaneous Auxiliaries

Date 2024-04-25

Version 1.1.0

Maintainer Mathijs Deen <dev@mathijsdeen.com>

Description Provides a variety of functions useful for data analysis, selection, manipulation, and graphics.

Depends R (>= 4.2)

Imports stats, MASS, graphics, methods, grDevices, car

License GPL-3

URL https://github.com/mathijsdeen/MDMA

BugReports https://github.com/mathijsdeen/MDMA/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Author Mathijs Deen [aut, cre]

Repository CRAN

Date/Publication 2024-04-25 16:40:02 UTC

R topics documented:

`auc` *Area under the curve*

Description

Calculate the area under the curve.

[Stable]

Usage

```
auc(x, ...)
```

Arguments

`x`	object of class `roc`.
`...`	other arguments (none are used at the moment).

Value

returns the area under the curve for a `roc` class object.

Author(s)

Mathijs Deen

Examples

```
a <- roc(QIDS$QIDS, QIDS$depression, c("Yes","No"), "Yes")
auc(a)
```

`check` *Check model for influential cases*

Description

Perform checks for a linear model regarding influential cases and collinearity numerically and graphically.

[Stable]

Usage

```
check(object, ...)
```

Arguments

`object`	object of class `lm`.
`...`	other parameters (none are used at the moment).

Value

`check` returns a list containing two matrices with statistics regarding influential cases and a vector of variance inflation factors. Furthermore, it produces diagnostics plots.
The return list contains three elements:

- `influence`, a `data.frame`, with observations in the model, and the following variables:

`predicted.value`	The value predicted by the model.
`residual`	The raw residual.
`std.residual`	The standardized residual.
`dfb.<...>`	DFBETAs for the variables in the model.
`dffit`	DFFIT value.
`cov.r`	Covariance ratio, a measure of change in the determinant of the coefficient covariance matrix.
`cook.d`	Cook's distance.
`hat`	Hat values.
`influential`	Determines whether a case is influential on any of the measures `dfb.<...>`, `dffit`, `cov.r`, `cook.d` or `hat`. See `influential cases` for more information.

- `is.infl` is a `data.frame` indicating which influence measure(s) is/are flagged per observation.

- `vifs`, a vector containing variance inflation factors for the variables in the model.

By default, the two data.frames regarding influence measures only give the influence measures for cases that are flagged as being influential. Influence measures for all cases can be queried using print.check.lm.

The generated plots are the plots produced by plot.lm, numbers 1 through 6.

influential cases

For the influence indicators, the following rules are applied to check whether a case is influential:

- any $|\mathrm{dfbeta}| > 1$.
- $|\mathrm{dffit}| > 3\sqrt{\frac{k}{n-k}}$.
- $|1 - \mathrm{cov.r}| > \frac{3k}{n-k}$.
- $F(n, n-k) = \mathrm{cooks.d}\ \ \mathrm{having}\ \ .p > .5$
- $\mathrm{hat} > \frac{3k}{n}$.

These indicators for being an influential case were derived from influence.measures in the stats package.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ disp + wt, data = mtcars)
check(lm.1)
```

---

coefsLogReg *Coefficients for logistic regression analysis*

Description

Show odds ratios and their confidence intervals for logistic regression parameter estimates.

[Stable]

Usage

```
coefsLogReg(model, confint = TRUE, level = 0.95)
```

Arguments

model	object of class glm, with family parameter set to binomial.
confint	indicates whether a confidence interval for the odds ratio should be returned.
level	the confidence level required.

Value

`coefsLogReg` returns the same table as `summary(object)$coefficients`, with the addition of the coefficients' odds ratios and their confidence intervals.

Author(s)

Mathijs Deen

Examples

```
glm(formula = am ~  disp, family = binomial, data = mtcars) |>
 coefsLogReg()
```

`corList` *List of correlation coefficients*

Description

List all correlations in a correlation matrix without duplicates.

[Stable]

Usage

```
corList(x, ...)
```

Arguments

`x`	a numeric vector, matrix or data frame.
`...`	arguments passed to the `cor` function.

Value

`corList` returns a list of correlations

Author(s)

Mathijs Deen

Examples

```
mtcars[,c("mpg","disp", "hp", "drat", "wt", "qsec")] |>
  corList(method="spearman")
```

dPPC2 — *Effect sizes from pretest-posttest-control group designs*

Description

dPPC2 calculates an effect size for studies with pretest and posttest scores for two groups, usually a treatment and a control group. It is based on Morris (2008), who based it on Becker (1988).

[Stable]

Usage

```
dPPC2(preT, posT, preC, posC, correct = TRUE, CIlevel = 0.95)
```

Arguments

preT	pre-scores for treatment group.
posT	post-scores for treatment group.
preC	pre-scores for control group.
posC	post-scores for control group.
correct	indicates whether a correction factor should be calculated (i.e., Hedges' *g* instead of Cohen's *d*).
CIlevel	the confidence level required.

Value

dPPC2 returns a vector of length 6, containing:

d	the effect size estimate.
SE	the standard error of the effect sie estimate.
lower.bound	lower bound of the confidence interval.
upper.bound	upper bound of the confidence interval.
NT	sample size of treatment group.
NC	sample size of control group.

Author(s)

Mathijs Deen

References

- Becker, B.J. (1988). Synthesizing standardized mean-change measures. *British Journal of Mathematical and Statistical Psychology, 41*, 257-278.
- Morris, S.B. (2008). Estimating effect sizes from pretest-posttest-control group designs. *Organizational Research Methods, 11*, 364-386.

Examples

```
library(MASS)
set.seed(1)
treatment <- mvrnorm(n=50, mu=c(50,40), Sigma = matrix(c(100,70,70,100), ncol=2), empirical = TRUE)
control <- mvrnorm(n=50, mu=c(50,45), Sigma = matrix(c(100,70,70,100), ncol=2), empirical = TRUE)
dPPC2(treatment[,1], treatment[,2], control[,1], control[,2])
```

`frequencies` *Display frequency table*

Description

Display frequency table with percentages and cumulative percentages.

[Stable]

Usage

```
frequencies(x)
```

Arguments

`x`	vector of values.

Value

object of type `data.frame` containing frequencies, percentages and cumulative percentages.

Author(s)

Mathijs Deen

Examples

```
frequencies(datasets::mtcars$carb)
```

keep *Save something to an object*

Description

keep saves an object to a new object. This is useful if one wants to save an intermediate result when using pipes.

Usage

```
keep(object, name, pos = 1, envir = as.environment(pos), inherits = FALSE)
```

Arguments

object	the object that is to be saved into name.
name	the name of the new object, containing the value of object.
pos	where to do the assignment. See ?assign for more details.
envir	the environment to use. See ?assign for more details.
inherits	should the enclosing framss of the environment be inspected? See ?assign for more details.

Value

Upon saving object to name, the value of object is returned. This makes it suitable for pipes.

Author(s)

Mathijs Deen

Examples

```
mtcars |>
  lm(mpg ~  disp + hp, data = _) |>
  keep(lm.mpg_disp_hp) |>
  summary()
```

`m` *Mean center*

Description

Mean center a vector or numeric matrix.

[Stable]

Usage

```
m(x)
```

Arguments

`x`	a numeric matrix or vector.

Details

This function resembles `base::scale.default`, with the `scale` argument set to `FALSE`. This, together with the short function name, is especially useful when you want to mean center variables in an analysis (e.g., using `(g)lm`), but you dont want the long form `scale(x, scale=FALSE)` to clutter up the rownames of the parameter estimates or the model anova.

Value

`m` returns a mean centered version of `x`. If `x` is a matrix, the matrix dimensions are preserved.

Author(s)

Mathijs Deen

Examples

```
vals <- matrix(rnorm(24, 15, 10), ncol = 2)
m(vals)
```

`plot.probeInteraction` *plot probed interaction*

Description

Plot the effects of the antecedent as a function of the moderator.

[Stable]

Usage

```
## S3 method for class 'probeInteraction'
plot(
  x,
  ...,
  col.JN = "red",
  lty.JN = "dotted",
  col.CI = rgb(red = 0.5, green = 0.5, blue = 0.5, alpha = 0.2),
  lty.CI = "longdash",
  lty.0 = "dotted"
)
```

Arguments

`x`	object of class `probeInteraction`.
`...`	other arguments (none are used).
`col.JN`	color for Johnson-Neyman cut-off line(s).
`lty.JN`	linetype for Johnson-Neyman cut-off line(s).
`col.CI`	color of the shade for the confidence interval.
`lty.CI`	linetype for confidence interval boundaries.
`lty.0`	linetype for the horizontal line where the effect of the focal predictor on the outcome equals 0.

Value

`plot.probeInteraction` returns a combined plot with p value on the first y axis and effect of the antecedent variable.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ hp * wt, data = mtcars)
pI.1 <- probeInteraction(lm.1, hp, wt, JN=TRUE, n.interval.moderator = 3,
                         quantile.moderator = c(0.1,0.9), values.moderator = 2)
plot(pI.1)
lm.2 <- lm(mpg ~ qsec * drat, data = mtcars)
pI.2 <- probeInteraction(lm.2, qsec, drat, JN=TRUE, n.interval.moderator = 30,
                         quantile.moderator = c(0.1,0.9), values.moderator = 2)
plot(pI.2)
```

plot.roc *plot roc curve*

Description

Plot an ROC curve.

[Stable]

Usage

```
## S3 method for class 'roc'
plot(
  x,
  y,
  which = 1:3,
  orientation = c("horizontal", "vertical"),
  cutoffs.1 = NULL,
  cutoffs.2 = NULL,
  cutoffs.3 = NULL,
  xlab.3 = NULL,
  labels.3 = NULL,
  xlim.3 = NULL,
  ylim.3 = c(0, 10),
  pos.legend.2 = "right",
  pos.legend.3 = "topright",
  ...
)
```

Arguments

`x`	object of class `roc`.
`y`	argument for generic `plot` function, not used here.
`which`	which plots to show (see Details).
`orientation`	indicate whether the plots should be arranged horizontally or vertically.
`cutoffs.1`	cutoff value(s) to be shown in the first plot.
`cutoffs.2`	cutoff value(s) to be shown in the second plot.
`cutoffs.3`	cutoff value(s) to be shown in the third plot.
`xlab.3`	lable for x axis in third plot.
`labels.3`	legend labels for third plot.
`xlim.3`	xlim for third plot.
`ylim.3`	ylim for third plot.
`pos.legend.2`	legend position for second plot.
`pos.legend.3`	legend position for third plot.
`...`	other arguments for generic `plot` function, none are used here.

Value

`plot.roc` provides three plots:

- The first plot contains the ROC curve.
- The second plot contains curves for the sensitivity and the specificity for all threshold values.
- The third plot contains density plots for the two classification groups.

Author(s)

Mathijs Deen

Examples

```
a <- roc(QIDS$QIDS, QIDS$depression, c("Yes","No"), "Yes")
plot(a, ylim.3 = c(0,.2), xlab.3= "QIDS value", cutoffs.1 = 14.5,
     cutoffs.2 = 14.5, cutoffs.3 = 14.5)
```

`plotDistribution` *Plot a probability distribution*

Description

Plot the density function of certain probability distributions.

[Stable]

Usage

```
plotDistribution(
  distribution = c("normal", "t", "chi2", "F"),
  xRange = c(0, 5),
  xColArea = NULL,
  xAreaCol = NULL,
  mean = 0,
  sd = 1,
  df,
  df1,
  df2,
  ncp,
  ...
)
```

Arguments

`distribution`	the probability distribution for which a plot should be drawn. Currently, the options are `"normal"`, `"t"`, `"chi2"` and `"F"`.
`xRange`	Range of x axis over which the distribution should be drawn.
`xColArea`	Optional, a matrix with two columns, where each row contains lower and upper bounds for intervals that should be colored under the pdf curve.
`xAreaCol`	Optional, should contain (a) color(s) for the interval colors in `xColArea`. Defaults to `"red"`. Should either be length 1 or length `length(xColArea)`.
`mean`	mean for the normal distribution.
`sd`	sd for the normal distribution.
`df`	df for the t distribution.
`df1`	first df for the F distribution.
`df2`	second df for the F distribution.
`ncp`	non-centrality parameter
`...`	other arguments to be forwarded to the `plot` function.

Value

`plotDistribution` returns a probability density plot.

Author(s)

Mathijs Deen

Examples

```
plotDistribution(distribution = "normal",
                 xRange       = c(-5, 5),
                 xColArea     = matrix(data  = c(-5, -1.96,
                                                 1.96, 5),
                                       ncol  = 2,
                                       byrow = TRUE),
                 xAreaCol     = c("green", "blue"),
                 mean         = 0,
                 sd           = 1,
                 yaxt         = "n",
                 ylab         = "")
```

`posteriorModelOdds` *Posterior model odds*

Description

Calculate the posterior model odds for a set of models.

[Stable]

Usage

```
posteriorModelOdds(...)
```

Arguments

`...`	objects of class `(g)lm`, given as separate arguments.

Details

Posterior model odds are calculated for every model i as

$$\text{pMO}_i = \frac{\exp\Big[-\frac{1}{2}\Delta_i\text{BIC}\Big]}{\sum_{j=1}^{K}\exp\Big[-\frac{1}{2}\Delta_j\text{BIC}\Big]},$$

where the minimal BIC value is subtracted from all BICs. In other words: the model with the lowest BIC has $\Delta\text{BIC} = 0$.

Value

`posteriorModelOdds` returns to posterior model odds for the models provided.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ hp + wt, data = mtcars)
lm.2 <- lm(mpg ~ hp * wt, data = mtcars)
lm.3 <- lm(mpg ~ hp * wt + gear, data = mtcars)
posteriorModelOdds(lm.1, lm.2, lm.3)
```

print.check.lm *Print lm check*

Description

Print the check of lm object

[Stable]

Usage

```
## S3 method for class 'check.lm'
print(x, which.infl = c("influential", "all"), ...)
```

Arguments

x	an object used to select a method.
which.infl	Indicate whether only influential cases (influential, the default) or all cases (all) should be printed.
...	further arguments passed to or from other methods (none are used).

Value

prints the check.lm object.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ disp + wt, data = mtcars)
chk.lm.1 <- check(lm.1)
print(chk.lm.1, which.infl="all")
```

print.probeInteraction *Print effects of probed interaction*

Description

Print the effects from a probed interaction.

[Stable]

Usage

```
## S3 method for class 'probeInteraction'
print(x, ...)
```

Arguments

`x`	object of class `probeInteraction`.
`...`	other parameters (none are used).

Value

`print.probeInteraction` prints the effects table of a `probeInteraction` object.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ hp * wt, data = mtcars)
pI <- probeInteraction(lm.1, hp, wt, JN=TRUE, n.interval.moderator = 3,
                       quantile.moderator = c(0.1,0.9), values.moderator = 2)
print(pI)
```

`print.tTest` *Print t test*

Description

Print the output of a t test.

[Stable]

Usage

```
## S3 method for class 'tTest'
print(x, ...)
```

Arguments

`x`	an object used to select a method.
`...`	further arguments passed to or from other methods.

Value

prints the `tTest` object as a `htest` object.

Author(s)

Mathijs Deen

Examples

```
x1 <- QIDS$QIDS[QIDS$depression == "Yes"]
x2 <- QIDS$QIDS[QIDS$depression == "No"]
tt <- tTest(x1, x2)
print(tt)
```

`probeInteraction` *Probe interaction models*

Description

Probe the effect of a moderator on an X/antecedent variable in a linear model.

[Stable]

Usage

```
probeInteraction(
  object,
  antecedent,
  moderator,
  alpha = 0.05,
  JN = TRUE,
  n.interval.moderator,
  quantile.moderator,
  values.moderator
)
```

Arguments

`object`	object of class `lm`.
`antecedent`	antecedent (or x) variable in `object`.
`moderator`	moderator variable in `object`.
`alpha`	desired alpha level for Johnson-Neyman procedure.
`JN`	indicate whether Johnson-Neyman procedure should be carried out.
`n.interval.moderator`	number of intervals in the moderator variable to probe.
`quantile.moderator`	quantile values in the moderator variable to probe.
`values.moderator`	raw values in the moderator variable to probe.

Details

the arguments `n.interval.moderator`, `quantile.moderator` and `values.moderator` can be combined. All unique values from these methods combined, together with the values from the Johnson-Neyman procedure (if specified) will be part of the probing procedure.

Value

`probeInteraction` returns a data frame containing values of the moderator in a linear model, the effect of the antecedent at that value of the moderator, standard errors, t values, p values and a confidence interval.

Author(s)

Mathijs Deen

Examples

```
lm.1 <- lm(mpg ~ hp * wt, data = mtcars)
probeInteraction(lm.1, hp, wt, JN=TRUE, n.interval.moderator = 3,
                 quantile.moderator = c(0.1,0.9), values.moderator = 2)
```

`QIDS` *QIDS depression data*

Description

The `QIDS` dataframe consists of 100 observations of people that were clinically diagnosed with a major depressive disorder and who filled out the QIDS-SR questionnaire. The data were simulated.

[Stable]

Usage

```
QIDS
```

Format

the following variables are available:

- `QIDS`: QIDS-SR total score.
- `Depression`: an indicator whether the individual was diagnosed with a depression or not.

Author(s)

Mathijs Deen

rci — *Reliable change index*

Description

`rci` computes the reliable change index according to Jacobson and Truax (1992).

[Stable]

Usage

```
rci(x1, x2, rxx)
```

Arguments

`x1`	prescore.
`x2`	postscore, same length as `x1`.
`rxx`	internal consistency statistic.

Value

`rci` returns a vector of `length(x1)` with reliable change index scores.

Author(s)

Mathijs Deen

References

- Jacobson, N.S., & Truax, P. (1992). Clinical significance: a statistical approach to defining meaningful change in psychotherapy research. *Journal of Consulting and Clinical Psychology, 59*, 12-19.

Examples

```
library(MASS)
set.seed(1)
q <- mvrnorm(n=120, mu=c(40, 50), Sigma = matrix(c(56.25,45,45,56.25), ncol = 2), empirical = TRUE)
cbind(q, rci(q[,1], q[,2], .8), rci(q[,1], q[,2], .8) > 1.96)
```

`roc` *Receiver operator characteristic*

Description

Calculate ROC curve statistics.

[Stable]

Usage

```
roc(response, group, levels, state)
```

Arguments

`response`	response variable for which thresholds will be calculated.
`group`	group variable.
`levels`	relevant levels of `group` variable. Should have length 2.
`state`	state level of `levels`..

Value

Returns a list with the following elements:

`data`	`data.frame` with two columns, containing the response and group variable for each level in `levels` without missings.
`rdf`	ROC dataframe. This is a `data.frame` containing sensitivity and specificity values for all threshold values.
`auc`	Area under the ROC curve.
`response`	Response variable from input data.
`group`	Group variable from the input data.
`levels`	Used levels.
`state`	State level.

Author(s)

Mathijs Deen

Examples

```
roc(QIDS$QIDS, QIDS$depression, c("No","Yes"), "Yes") |>
  plot(ylim.3=c(0,.2))
```

`summary.tTest` *Summarize outcome of a t test*

Description

Summarize the outcome of a t test

[Stable]

Usage

```
## S3 method for class 'tTest'
summary(object, rnd = 3L, ...)
```

Arguments

`object`	object of class `htest` (i.e., the result of `mdma::tTest` or `stats::t.test`).
`rnd`	number of decimal places. Should have length 1 or 3. One value specifies the rounding value for the degrees of freedom, t statistic and p value all at once, while specifying three values gives the rounding values for the three statistics respectively.
`...`	other arguments of the summary generic (none are used).

Value

`summary.htest` returns a typical APA-like output (without italics) for a t-test.

Author(s)

Mathijs Deen

Examples

```
x1 <- QIDS$QIDS[QIDS$depression == "Yes"]
x2 <- QIDS$QIDS[QIDS$depression == "No"]
tt <- tTest(x1, x2)
summary(tt, rnd = c(1,2,3))
```

`tTest` *t Test*

Description

perform t tests with the possibility of inputting group statistics.

[Stable]

Usage

```
tTest(
  x,
  y = NULL,
  sdx = NULL,
  sdy = NULL,
  nx = length(na.omit(x)),
  ny = length(na.omit(y)),
  alternative = c("two.sided", "greater", "less"),
  mu = 0,
  paired = FALSE,
  rxy = NULL,
  var.equal = FALSE,
  conf.level = 0.95
)
```

Arguments

`x`	a numeric vector. Can be of length 1 for a group mean.
`y`	a numeric vector. Should be `NULL` for a one-sample t-test.
`sdx`	standard deviation for `x`, when this reflects a group mean.
`sdy`	standard deviation for `y`, when this reflects a group mean.
`nx`	sample size for `x`, when this reflects a group mean.
`ny`	sample size for `y`, when this reflects a group mean.
`alternative`	a character string specifying the alternative hypothesis, must be one of "`two.sided`" (default), "`greater`" or "`less`". You can specify just the initial letter.
`mu`	a number indicating the true value of the mean (or difference in means) if you are performing an independent samples t-test).
`paired`	a logical indicating whether you want a paired t-test.
`rxy`	correlation between two paired samples.
`var.equal`	a logical variable indicating whether to treat the two variances as being equal. If `TRUE` then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.
`conf.level`	level of the confidence interval.

Value

tTest performs a t-test (independent samples, paired samples, one sample) just like base-R t.test, but with the extended possibility to enter group statistics instead of raw data.

Author(s)

Mathijs Deen

Examples

```
library(MASS)
set.seed(1)
ds <- mvrnorm(n=50, mu = c(50,55),
              Sigma = matrix(c(100,0,0,81),
                             ncol = 2),
              empirical = TRUE) |>
  data.frame() |>
  setNames(c("x1","x2"))
t.test(ds$x1, ds$x2)
tTest(x   = ds$x1,
      y   = 55,
      sdy = 9,
      ny  = 50)
```

`%inRange%` *inRange*

Description

Return which values are in a certain range `%inRange%` indicates which values are in a certain range, including the boundaries of the range.

[Experimental]

Usage

```
lhs %inRange% rhs
```

Arguments

`lhs`	numeric vector.
`rhs`	numeric vector of length 2 with the bounds of the range.

Value

`%inRange%` returns a logical vector of `length(lhs)`, indicating which values of `lhs` are and are not in range `rhs`. Boundaries of `rhs` are included.

Author(s)

Mathijs Deen

See Also

`%withinRange%`

Examples

```
a <- seq(0, 100, 5)
r <- c(40, 70)
cbind(a,
      'a %inRange% r' = a %inRange% r,
      'a %withinRange% r' = a %withinRange% r)
```

`%ni%` *Inverse value matching*

Description

Evaluates whether the left hand side argument is not in the right hand side argument.

[Stable]

Usage

```
lhs %ni% rhs
```

Arguments

`lhs`	left hand side.
`rhs`	right hand side.

Details

The `%ni%` function negates the `%in%` operator.

Value

The function returns a vector of the same length as `lhs`.

Author(s)

Mathijs Deen

Examples

```
c(1,2,3) %ni% c(1,2)
```

`%withinRange%` *withinRange*

Description

Return which values are within a certain range

`%withinRange%` indicates which values are in a certain range, excluding the boundaries of the range.

[Experimental]

Usage

```
lhs %withinRange% rhs
```

Arguments

`lhs`	numeric vector.
`rhs`	numeric vector of length 2 with the bounds of the range.

Value

`%withinRange%` returns a logical vector of `length(lhs)`, indicating which values of `lhs` are and are not in range `rhs`. Boundaries of `rhs` are excluded.

Author(s)

Mathijs Deen

See Also

`%inRange%`

Examples

```
a <- seq(0, 100, 5)
r <- c(40, 70)
cbind(a,
      'a %inRange% r' = a %inRange% r,
      'a %withinRange% r' = a %withinRange% r)
```

Index