Package ‘autoReg’

April 5, 2022

Type Package

Title Automatic Linear and Logistic Regression and Survival Analysis

Version 0.2.6

URL https://github.com/cardiomoon/autoReg, https://cardiomoon.github.io/autoReg/

BugReports https://github.com/cardiomoon/autoReg/issues

Description Make summary tables for descriptive statistics and select explanatory variables automatically in various regression models. Support linear models, generalized linear models and cox-proportional hazard models. Generate publication-ready tables summarizing result of regression analysis and plots. The tables and plots can be exported in ``HTML'', ``pdf('LaTex')'', ``docx('MS Word')'' and ``pptx('MS Powerpoint')'' documents.

LazyData true

Encoding UTF-8

Imports moonBook(>= 0.3.0), nortest, dplyr, crayon, stringr, tidyr, purrr, survival, mice, officer, flextable, rlang, patchwork, ggplot2, boot, broom, tidycmprsk, scales, maxstat, pammtools

Suggests knitr, finalfit, lme4, TH.data, rmarkdown, survminer, asaur, cmprsk, PairedData

RoxygenNote 7.1.2

VignetteBuilder knitr

Depends R (>= 2.10)

NeedsCompilation no

Author Keon-Woong Moon [aut, cre]

Maintainer Keon-Woong Moon <cardiomoon@gmail.com>

Repository CRAN

Date/Publication 2022-04-05 06:20:02 UTC

R topics documented:

`addFitSummary` *Add model summary to an object of class gaze*

Description

Add model summary to an object of class gaze

Usage

```
addFitSummary(df, fit, statsname = "")
```

Arguments

`df`	An object of class "gaze" or "autoReg"
`fit`	An object of class "glm" or "lm" or "crr"
`statsname`	character Name of statistics

Value

addFitSummary returns an object of `gaze` or `autoReg` - the same class as df

Examples

```
require(survival)
require(dplyr)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~rx+age+sex+nodes+obstruct+perfor,data=colon)
df=autoReg(fit,uni=FALSE)
final=fit2final(fit)
df %>% addFitSummary(final,statsname="HR (final)") %>% myft()
```

`addLabelData` *Add labels to data*

Description

Add labels to data

Usage

```
addLabelData(data)
```

Arguments

`data`	A data.frame

Value

A data.frame

Examples

```
addLabelData(data.frame(ph.ecog=0:3,sex=c(1,2,2,2),age=c(20,30,40,70)))
```

`adjustedPlot` *Draw an expected plot*

Description

Draw an expected plot

Usage

```
adjustedPlot(
  fit,
  xnames = NULL,
  pred.values = list(),
  newdata = NULL,
  maxy.lev = 5,
  median = TRUE,
  facet = NULL,
  se = FALSE,
  mark.time = FALSE,
  show.median = FALSE,
  type = "ggplot",
  ...
)
```

Arguments

`fit`	An object of class "coxph" or "survreg"
`xnames`	Character Names of explanatory variable to plot
`pred.values`	A list A list of predictor values
`newdata`	A data.frame or NULL
`maxy.lev`	Integer Maximum unique length of a numeric variable to be treated as categorical variables
`median`	Logical
`facet`	Character Name of facet variable
`se`	logical Whether or not show se
`mark.time`	logical Whether or not mark time
`show.median`	logical
`type`	Character plot type
`...`	further arguments to be passed to plot.survfit

Value

A ggplot or no return value(called for side effects)

Examples

```
library(survival)
fit=coxph(Surv(time,status)~rx+logWBC,data=anderson)
adjustedPlot(fit)
adjustedPlot(fit,xnames="rx",se=TRUE,type="plot")
adjustedPlot(fit,xnames="rx",se=TRUE)
## Not run:
anderson$WBCgroup=ifelse(anderson$logWBC<=2.73,0,1)
anderson$WBCgroup=factor(anderson$WBCgroup,labels=c("low","high"))
anderson$rx=factor(anderson$rx,labels=c("treatment","control"))
fit=coxph(Surv(time,status)~rx,data=anderson)
adjustedPlot(fit,xnames=c("rx"),show.median=TRUE)
fit=coxph(Surv(time,status)~rx*WBCgroup,data=anderson)
adjustedPlot(fit,xnames=c("rx","WBCgroup"),show.median=TRUE)
adjustedPlot(fit,xnames=c("rx","WBCgroup"),facet="WBCgroup",show.median=TRUE)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~rx+strata(sex)+age+differ,data =colon)
adjustedPlot(fit,xnames=c("sex"))
adjustedPlot(fit,xnames=c("sex"),pred.values=list(age=58,differ=3))
adjustedPlot(fit,xnames=c("sex","rx"),facet="sex")
adjustedPlot(fit,xnames=c("rx","sex","differ"),facet=c("sex","rx"),se=TRUE)
fit <- coxph(Surv(start, stop, event) ~ rx + number + size+ cluster(id), data = bladder2)
adjustedPlot(fit,xnames=c("rx","number","size"),facet=c("rx","size"),maxy.lev=8)

## End(Not run)
```

`adjustedPlot.survreg` *Draw predicted survival curve with an object survreg*

Description

Draw predicted survival curve with an object survreg

Usage

```
adjustedPlot.survreg(
  x,
  xnames = NULL,
  pred.values = list(),
  maxy.lev = 5,
  median = TRUE,
  newdata = NULL,
  addCox = FALSE
)
```

Arguments

x	An object of class survreg

`xnames`	Character Names of explanatory variable to plot
`pred.values`	A list A list of predictor values
`maxy.lev`	Integer Maximum unique length of a numeric variable to be treated as categorical variables
`median`	Logical
`newdata`	A data.frame or NULL
`addCox`	logical Whether or not add KM

Value

No return value, called for side effects

Examples

```
library(survival)
x=survreg(Surv(time, status) ~ rx, data=anderson,dist="exponential")
adjustedPlot(x)
adjustedPlot(x,addCox=TRUE)
## Not run:
x=survreg(Surv(time, status) ~ sex, data=lung,dist="weibull")
adjustedPlot(x,addCox=TRUE)
x=survreg(Surv(time, status) ~ rx, data=anderson,dist="exponential")
adjustedPlot(x)
x=survreg(Surv(time, status) ~ ph.ecog + age + sex, data=lung, dist="weibull")
adjustedPlot(x)
adjustedPlot(x,addCox=TRUE)
adjustedPlot(x,pred.values=list(age=c(20,40,60,80),sex=2,ph.ecog=3),addCox=TRUE)
newdata=data.frame(ph.ecog=0:3,sex=c(1,2,2,2),age=c(20,40,60,80))
adjustedPlot(x,newdata=newdata,addCox=TRUE)

## End(Not run)
```

`adjustedPlot2` *Draw a survfitted plot*

Description

Draw a survfitted plot

Usage

```
adjustedPlot2(fit, se = FALSE, mark.time = FALSE)
```

Arguments

`fit`	An object of class coxph or survfit
`se`	logical Whether or not show se
`mark.time`	logical Whether or not mark time

Value

a ggplot

Examples

```
library(survival)
fit=coxph(Surv(time,status)~rx+logWBC,data=anderson)
plot(survfit(fit),conf.int=TRUE)
adjustedPlot2(fit,se=TRUE)
```

`anderson` *Remission survival times of 42 leukemia patients*

Description

A dataset containing survival time of 42 leukemia patients

Usage

```
anderson
```

Format

A data.frame with 42 rows and 5 variables

time survival time in weeks

status censoring status 1=failure 0=censored

sex sex 0=Female 1=Male

logWBC log white blood cell count

rx treatment status 1=control 0 =treatment

Source

David G. Kleinbaum and Mitchel Klein. Survival Analysis. A Self-Learning Text(3rd ed)(Springer,2012) ISBN: 978-1441966452

`anderson1` *Remission survival times of 42 leukemia patients*

Description

A dataset containing survival time of 42 leukemia patients This data is the same data with anderson, but sex and rx variable are factors not numeric

Usage

```
anderson1
```

Format

A data.frame with 42 rows and 5 variables

time survival time in weeks

status censoring status 1=failure 0=censored

sex sex "Female" or "Male

logWBC log white blood cell count

rx treatment status "treatment" or "control"

Source

David G. Kleinbaum and Mitchel Klein. Survival Analysis. A Self-Learning Text(3rd ed)(Springer,2012) ISBN: 978-1441966452

`anderson2` *Remission survival times of 31 leukemia patients*

Description

This data is subdata of anderson with medium(2.3 < logWBC <= 2.96) and high WBC count(logWBC > 2.96)

Usage

```
anderson2
```

Format

A data.frame with 31 rows and 6 variables

time survival time in weeks

status censoring status 1=failure 0=censored

sex sex 0=Female 1=Male

logWBC log white blood cell count

rx treatment status 1=control 0 =treatment

WBCCAT WBC count group 1=medium 2=high

Details

A dataset containing survival time of 31 leukemia patients

Source

David G. Kleinbaum and Mitchel Klein. Survival Analysis. A Self-Learning Text(3rd ed)(Springer,2012) ISBN: 978-1441966452

`as_printable` *Convert data.frame to printable form*

Description

Calculate character length and apply all data

Usage

```
as_printable(
  data,
  align.first = "left",
  align.chr = "right",
  align.num = "right"
)
```

Arguments

`data`	A data.frame
`align.first`	character Alignment of first variable
`align.chr`	character Alignment of character variable
`align.num`	character Alignment of numeric variable

Value

A data.frame

Examples

```
as_printable(mtcars)
as_printable(iris)
```

---

autoReg *Perform univariable and multivariable regression and stepwise backward regression automatically*

Description

Perform univariable and multivariable regression and stepwise backward regression automatically

Usage

```
autoReg(x, ...)

## S3 method for class 'lm'
autoReg(x, ...)

## S3 method for class 'glm'
autoReg(x, ...)

## S3 method for class 'coxph'
autoReg(x, ...)

## S3 method for class 'survreg'
autoReg(x, ...)
```

Arguments

`x`	An object of class lm, glm or coxph
`...`	Further arguments

Value

autoReg returns an object of class "autoReg" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

attr(*,"model") name of model. One of "lm","glm" or "coxph"

Methods (by class)

- `lm`: S3 method for a class lm
- `glm`: S3 method for a class glm
- `coxph`: S3 method for a class coxph
- `survreg`: S3 method for a class survreg

Examples

```
data(cancer,package="survival")
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
autoReg(fit)
autoReg(fit,uni=FALSE,final=TRUE)
autoReg(fit,uni=FALSE,imputed=TRUE)
fit=lm(mpg~wt*hp+am+I(wt^2),data=mtcars)
autoReg(fit,final=TRUE)
autoReg(fit,imputed=TRUE)
```

`autoRegCox` *perform automatic regression for a class of coxph*

Description

perform automatic regression for a class of coxph

Usage

```
autoRegCox(
  x,
  threshold = 0.2,
  uni = FALSE,
  multi = TRUE,
  final = FALSE,
  imputed = FALSE,
  keepstats = FALSE,
  ...
)
```

Arguments

`x`	An object of class coxph
`threshold`	numeric
`uni`	logical whether or not perform univariable regression
`multi`	logical whether or not perform multivariable regression
`final`	logical whether or not perform stepwise backward elimination
`imputed`	logical whether or not perform multiple imputation
`keepstats`	logical whether or not keep statistic
`...`	Further arguments to be passed to gaze()

Value

autoRegCox returns an object of class "autoReg" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

attr(*,"model") name of model. One of "lm","glm" or "coxph"

Examples

```
require(survival)
require(dplyr)
data(cancer)
fit=coxph(Surv(time,status==2)~log(bili)+age+cluster(edema),data=pbc)
autoReg(fit)
fit=coxph(Surv(time,status)~rx+age+sex+nodes+obstruct+perfor,data=colon)
autoReg(fit)
autoReg(fit,uni=TRUE,threshold=1)
autoReg(fit,uni=TRUE,final=TRUE) %>% myft()
data(colon_s,package="finalfit")
fit=coxph(Surv(time,status)~age.factor+sex.factor+obstruct.factor+perfor.factor,data=colon_s)
autoReg(fit,uni=TRUE,threshold=1)
autoReg(fit,uni=TRUE,imputed=TRUE)
```

`autoRegsurvreg` *perform automatic regression for a class of survreg*

Description

perform automatic regression for a class of survreg

Usage

```
autoRegsurvreg(
  x,
  threshold = 0.2,
  uni = FALSE,
  multi = TRUE,
  final = FALSE,
  imputed = FALSE,
  keepstats = FALSE,
  mode = 1,
  ...
)
```

Arguments

`x`	An object of class survreg
`threshold`	numeric
`uni`	logical whether or not perform univariable regression
`multi`	logical whether or not perform multivariable regression
`final`	logical whether or not perform stepwise backward elimination
`imputed`	logical whether or not perform multiple imputation
`keepstats`	logical whether or not keep statistic
`mode`	integer
`...`	Further arguments to be passed to gaze()

Value

autoRegsurvreg returns an object of class "autoReg" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

attr(*,"model") name of model. One of "lm","glm","coxph" or "survreg"

Examples

```
require(survival)
require(dplyr)
data(cancer)
fit=survreg(Surv(time,status)~rx+age+sex+nodes+obstruct+perfor,data=colon)
autoReg(fit)
autoReg(fit,uni=TRUE,threshold=1)
autoReg(fit,uni=TRUE,final=TRUE)
autoReg(fit,uni=TRUE,final=TRUE) %>% myft()
## Not run:
autoReg(fit,mode=2)
autoReg(fit,uni=TRUE,threshold=1,,mode=2)
autoReg(fit,uni=TRUE,final=TRUE,mode=2)
autoReg(fit,uni=TRUE,final=TRUE,mode=2) %>% myft()
autoReg(fit,final=TRUE,imputed=TRUE) %>% myft()
autoReg(fit,final=TRUE,imputed=TRUE,mode=2) %>% myft()

## End(Not run)
```

`autoReg_sub` *Perform univariable and multivariable regression and stepwise backward regression automatically*

Description

Perform univariable and multivariable regression and stepwise backward regression automatically

Usage

```
autoReg_sub(
  fit,
  threshold = 0.2,
  uni = FALSE,
  multi = TRUE,
  final = FALSE,
  imputed = FALSE,
  keepstats = FALSE,
  showstats = TRUE,
  ...
)
```

Arguments

`fit`	An object of class lm or glm
`threshold`	numeric
`uni`	logical whether or not perform univariate regression
`multi`	logical whether or not perform multivariate regression
`final`	logical whether or not perform stepwise backward elimination
`imputed`	logical whether or not include imputed model
`keepstats`	logical whether or not keep statistics
`showstats`	logical whether or not show descriptive statistics
`...`	Further arguments to be passed to imputedReg()

Value

An object of class "autoReg" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

attr(*,"model") name of model. One of "lm","glm" or "coxph"

`beNumeric` *Whether a string vector can be converted to numeric*

Description

Whether a string vector can be converted to numeric

Usage

```
beNumeric(x)
```

Arguments

`x`	A string vector

Value

A logical vector

`bootPredict` *Bootstrap simulation for model prediction*

Description

Generate model predictions against a specified set of explanatory levels with bootstrapped confidence intervals.

Usage

```
bootPredict(fit, newdata, R = 100, type = "response", ...)
```

Arguments

`fit`	An object of class lm or glm
`newdata`	A data.frame
`R`	Number of simulations. Note default R=100 is very low.
`type`	he type of prediction required, see predict.glm. The default for glm models is on the scale of the response variable. Thus for a binomial model the default predictions are predicted probabilities.
`...`	Further arguments to be passed to boot::boot

Value

An object of class "data.frame"

Examples

```
data(GBSG2,package="TH.data")
fit=glm(cens~horTh+pnodes,data=GBSG2,family="binomial")
newdata=expand.grid(horTh=factor(c(1,2),labels=c("no","yes")),pnodes=1:51)
bootPredict(fit,newdata)
library(survival)
fit=coxph(Surv(time,cens)~age+horTh+progrec+pnodes,data=GBSG2)
```

`countGroups` *Count groups*

Description

Count groups

Usage

```
countGroups(data, yvars)
```

Arguments

`data`	A data.frame
`yvars`	variable names

Value

An object of class "tibble"

Examples

```
library(moonBook)
countGroups(acs,"sex")
countGroups(acs,c("sex","Dx"))
```

`coxzphplot` *Graphical Test of Proportional Hazards*

Description

Tis is a ggplot version of plot.cox.zph. Displays a graph of the scaled Schoenfeld residuals, along with a smooth curve.

Usage

```
coxzphplot(x, resid = TRUE, se = TRUE, var = NULL, hr = FALSE, add.lm = FALSE)
```

Arguments

`x`	result of the cox.zph function.
`resid`	a logical value, if TRUE the residuals are included on the plot, as well as the smooth fit.
`se`	a logical value, if TRUE, confidence bands at two standard errors will be added.
`var`	The set of variables for which plots are desired. It can be integer or variable names
`hr`	logical If true, plot for hazard ratio, If false, plot for coefficients
`add.lm`	logical If true, add linear regression line

Value

A facetted ggplot

Examples

```
library(survival)
vfit <- coxph(Surv(time,status) ~ trt + factor(celltype) + karno + age, data=veteran, x=TRUE)
x <- cox.zph(vfit)
coxzphplot(x)
coxzphplot(x,var="karno",add.lm=TRUE)
```

`crr2stats` *Extract statistics from an object of class crr*

Description

Extract statistics from an object of class crr

Usage

```
crr2stats(x, digits = 2)
```

Arguments

`x`	an object of class crr
`digits`	integer indication the position of decimal place

Value

An object of class "data.frame"

Examples

```
data(melanoma,package="boot")
melanoma$status_crr=ifelse(melanoma$status==1,1,ifelse(melanoma$status==2,0,2))
x=crrFormula(time+status_crr~age+sex+thickness+ulcer,data=melanoma)
crr2stats(x)
```

`crrFormula` *Competing Risk Regression with Formula*

Description

Competing Risk Regression with Formula

Usage

```
crrFormula(x, data, ...)
```

Arguments

`x`	formula time+status~explanatory variables
`data`	data a data.frame
`...`	Further arguments to be passed to `crr`

Value

An object of class "tidycrr" which is described in `crr`

Examples

```
data(melanoma,package="boot")
melanoma$status_crr=ifelse(melanoma$status==1,1,ifelse(melanoma$status==2,0,2))
crrFormula(time+status_crr~age+sex+thickness+ulcer,data=melanoma)
```

`descNum` *Make description for numeric summary*

Description

Make description for numeric summary

Usage

```
descNum(method = 1, p = NULL)
```

Arguments

`method`	numeric
`p`	A numeric or NULL

Value

A character vector of length 1

`df2flextable` *Convert data.frame to flextable*

Description

Convert data.frame to flextable

Usage

```
df2flextable(
  df,
  vanilla = FALSE,
  fontname = NULL,
  fontsize = 12,
  add.rownames = FALSE,
  even_header = "transparent",
  odd_header = "#5B7778",
  even_body = "#EFEFEF",
  odd_body = "transparent",
  vlines = TRUE,
  colorheader = FALSE,
  digits = 2,
  digitp = 3,
  align_header = "center",
  align_body = "right",
  align_rownames = "left",
  NA2space = TRUE,
  pcol = NULL,
  ...
)
```

Arguments

`df`	A data.frame
`vanilla`	A Logical
`fontname`	Font name
`fontsize`	font size
`add.rownames`	logical. Whether or not include rownames
`even_header`	background color of even_header
`odd_header`	background color of even_header
`even_body`	background color of even_body
`odd_body`	background color of even_body
`vlines`	Logical. Whether or not draw vertical lines
`colorheader`	Logical. Whether or not use color in header
`digits`	integer indicating the number of decimal places
`digitp`	integer indicating the number of decimal places of p values
`align_header`	alignment of header. Expected value is one of 'left', 'right', 'center', 'justify'.
`align_body`	alignment of body. Expected value is one of 'left', 'right', 'center', 'justify'.
`align_rownames`	alignment of rownames. Expected value is one of 'left', 'right', 'center', 'justify'.
`NA2space`	A logical. If true, convert NA value to space

`pcol`	An integer indicating p value. If specified, convert value less than 0.01 to "< 0.001" in given column.
`...`	further arguments to be passed to `flextable`

Value

An object of class "flextable" which is described in `flextable`

`drawline` *draw line character*

Description

draw line character

Usage

```
drawline(n)
```

Arguments

`n`	Numeric

Value

No return value, called for side effects

Examples

```
drawline(10)
```

`expectedPlot` *Draw an adjusted Plot for a numeric predictor*

Description

Select cutpoint for a numeric predictor with maxstat.test() and draw survival plot with this cutpoint

Usage

```
expectedPlot(
  fit,
  xname = NULL,
  no = 2,
  maxy.lev = 5,
  median = TRUE,
  mark.time = FALSE,
  se = FALSE,
  type = "ggplot",
  ...
)
```

Arguments

`fit`	An object of class "coxph"
`xname`	Character Name of explanatory variable to plot
`no`	integer Number of groups to be made
`maxy.lev`	Integer Maximum unique length of a numeric variable to be treated as categorical variables
`median`	Logical
`mark.time`	logical Whether or not mark time
`se`	logical Whether or not show se
`type`	Character "plot" or "ggplot"
`...`	further arguments to be passed to plot.survfit

Value

No return value, called for side effects

Examples

```
library(survival)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~age+sex,data =colon)
expectedPlot(fit,xname="age")
fit=coxph(Surv(time,status)~rx+logWBC,data=anderson)
expectedPlot(fit,xname="logWBC",no=3)
```

`filldown` *filldown vector with lead value*

Description

filldown vector with lead value

Usage

```
filldown(x, what = c("", NA))
```

Arguments

`x`	a vector
`what`	Values to be filled

Value

A vector with the same class as x

Examples

```
x=rep(1:5,each=3)
x=removeDup(x,NA)
filldown(x)
```

`find1stDup` *Find first duplicated position*

Description

Find first duplicated position

Usage

```
find1stDup(x)
```

Arguments

`x`	a vector

Value

A logical vector

Examples

```
x=rep(1:5,each=3)
which(find1stDup(x))
```

`findDup` *Find duplicated term*

Description

Find duplicated term

Usage

```
findDup(x)
```

Arguments

`x`	A vector

Value

A logical vector

Examples

```
x=rep(1:5,each=3)
findDup(x)
x=c(6,x)
findDup(x)
which(!findDup(x))
```

`fit2final` *Make final model using stepwise backward elimination*

Description

Make final model using stepwise backward elimination

Usage

```
fit2final(fit, threshold = 0.2)
```

Arguments

`fit`	An object of class "coxph"
`threshold`	Numeric

Value

An object of class "coxph" which is described in `coxph`

Examples

```
require(survival)
data(cancer)
fit=coxph(Surv(time,status)~age+sex+obstruct+perfor,data=colon)
final=fit2final(fit)
fit2summary(final)
```

`fit2lik` *extract likelihood information with a coxph object*

Description

extract likelihood information with a coxph object

Usage

```
fit2lik(x)
```

Arguments

`x`	An object of class "coxph" or "survreg"

Value

A string

Examples

```
library(survival)
fit=coxph(Surv(time,status) ~rx,data=anderson)
fit2lik(fit)
```

`fit2list` *Make a list of univariable model with multivariable regression model*

Description

Make a list of univariable model with multivariable regression model

Usage

```
fit2list(fit)
```

Arguments

`fit`	An object of class "lm" or "glm"

Value

An object of class "fitlist" which is a list of objects of class "lm" or "glm"

Examples

```
library(survival)
data(cancer)
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
fit2list(fit)
fit=lm(mpg~wt*hp+am,data=mtcars)
fit2list(fit)
```

`fit2model` *Restore fit model data containing AsIs expressions*

Description

Restore fit model data containing AsIs expressions

Usage

```
fit2model(fit)
```

Arguments

`fit`	An object of class "lm", "glm" or "coxph"

Value

An object of class "data.frame"

Examples

```
require(survival)
pbc$status2=ifelse(pbc$status==2,1,0)
fit=coxph(Surv(time,status2)~age+log(bili),data=pbc)
fit2model(fit)
```

`fit2multi` *Make multivariable regression model by selecting univariable models with p.value below threshold*

Description

Make multivariable regression model by selecting univariable models with p.value below threshold

Usage

```
fit2multi(fit, threshold = 0.2)
```

Arguments

`fit`	An object of class "coxph"
`threshold`	Numeric

Value

An object of class "coxph"

Examples

```
require(survival)
data(cancer)
fit=coxph(Surv(time,status)~age+sex+obstruct+perfor,data=colon)
fit2multi(fit)
```

`fit2newdata` *Make a new data of mean value or most frequent value*

Description

Make a new data of mean value or most frequent value

Usage

```
fit2newdata(
  fit,
  xnames = NULL,
  pred.values = list(),
  maxy.lev = 5,
  median = TRUE,
  digits = 1
)
```

Arguments

`fit`	An object of class "coxph"
`xnames`	character Names of explanatory variable to plot
`pred.values`	A list A list of predictor values
`maxy.lev`	Integer Maximum unique length of a numeric variable to be treated as categorical variables
`median`	logical If TRUE, select median value for numerical variable. Otherwise select most frequent value
`digits`	integer indicating the number of decimal places

Value

A data.frame

Examples

```
require(survival)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~rx+sex+age,data=colon)
fit=coxph(Surv(time,status)~rx+age+strata(sex),data=colon)
fit=survreg(Surv(time, status) ~ ph.ecog + age + sex, data=lung, dist="weibull")
fit2newdata(fit)
fit2newdata(fit,pred.values=list(sex=0,age=58))
fit2newdata(fit,pred.values=list(age=c(20,40,60,80),sex=2,ph.ecog=3))
```

`fit2stats` *Summarize statistics with a model*

Description

Summarize statistics with a model

Usage

```
fit2stats(fit, method = "default", digits = 2, mode = 1)
```

Arguments

`fit`	An object of class lm or glm or coxph or survreg
`method`	character choices are one of the c("likelihood","wald")
`digits`	integer indicating the number of decimal places
`mode`	integer

Value

An object of class "data.frame"

Examples

```
library(survival)
data(cancer)
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
fit2stats(fit)
fit=lm(mpg~wt*hp+am,data=mtcars)
fit2stats(fit)
fit=survreg(Surv(time,status)~rx+sex+age+obstruct+nodes,data=colon)
fit2stats(fit)
```

`fit2summary` *Summarize statistics with a model or model list*

Description

Summarize statistics with a model or model list

Usage

```
fit2summary(fit, mode = 1, ...)
```

Arguments

`fit`	An object of class "lm" or "glm" or "fitlist" which is a result of `fit2list`
`mode`	integer
`...`	Further argument to be passed to fit2stats

Value

An object of class "data.frame"

Examples

```
library(survival)
data(cancer)
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
fit2summary(fit)
fitlist=fit2list(fit)
fit2summary(fitlist)
fit=survreg(Surv(time,status)~rx+sex+age+obstruct+nodes,data=colon)
fit2summary(fit)
```

`gaze` *Produce table for descriptive statistics*

Description

Produce table for descriptive statistics by groups for several variables easily. Depending on the nature of these variables, different descriptive statistical methods were used(t-test, ANOVA, Kruskal-Wallis, chi-squared, Fisher's,...)

Usage

```
gaze(x, ...)

## S3 method for class 'formula'
gaze(x, ...)

## S3 method for class 'data.frame'
gaze(x, ...)

## S3 method for class 'coxph'
gaze(x, ...)

## S3 method for class 'survreg'
gaze(x, ...)

## S3 method for class 'glm'
gaze(x, ...)

## S3 method for class 'lm'
gaze(x, ...)

## S3 method for class 'tidycrr'
gaze(x, ...)
```

Arguments

`x`	An R object, formula or data.frame
`...`	arguments to be passed to gaze.data.frame or gaze.formula

Value

An object of class "gaze" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

Methods (by class)

- `formula`: S3 method for formula
- `data.frame`: default S3 method
- `coxph`: default S3 method
- `survreg`: default S3 method
- `glm`: default S3 method
- `lm`: default S3 method
- `tidycrr`: default S3 method

Examples

```
library(moonBook)
library(dplyr)
gaze(acs)
gaze(~age+sex,data=acs)
gaze(sex~.,data=acs,digits=1,method=1,show.p=TRUE) %>% myft()

gaze(sex~age+Dx,data=acs)
gaze(EF~.,data=acs) %>% myft()
gaze(sex+Dx~.,data=acs,show.p=TRUE) %>% myft()
gaze(sex+Dx~.,data=acs)
gaze(Dx+sex~cardiogenicShock,data=acs,show.p=TRUE) %>% myft()
gaze(Dx+sex+HBP~cardiogenicShock,data=acs,show.p=TRUE)
gaze(~mpg+cyl,data=mtcars)
gaze(~.,data=mtcars)
gaze(cyl~.,data=mtcars,show.p=TRUE)
gaze(hp~.,data=mtcars)
gaze(cyl+am~.,data=mtcars)

library(survival)
x=coxph(Surv(time,status) ~rx,data=anderson1)
gaze(x)
x=coxph(Surv(time,status) ~rx*logWBC,data=anderson1)
gaze(x)
library(survival)
x=survreg(Surv(time, status) ~ rx, data=anderson,dist="exponential")
gaze(x)
x=survreg(Surv(time, status) ~ ph.ecog + age + sex, lung)
gaze(x)
data(cancer,package="survival")
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
gaze(fit)
fit=lm(mpg~wt*hp+am+I(wt^2),data=mtcars)
gaze(fit)
data(melanoma,package="boot")
melanoma$status_crr=ifelse(melanoma$status==1,1,ifelse(melanoma$status==2,0,2))
fit=crrFormula(time+status_crr~age+sex+thickness+ulcer,data=melanoma)
gaze(fit)
```

`gaze.formula_sub` *Produce table for descriptive statistics*

Description

Produce table for descriptive statistics by groups for several variables easily. Depending on the nature of these variables, different descriptive statistical methods were used(t-test, ANOVA, Kruskal-Wallis, chi-squared, Fisher's,...)

Usage

```
## S3 method for class 'formula_sub'
gaze(x, data, missing = FALSE, ...)
```

Arguments

`x`	An object of class "formula". Left side of ~ must contain the name of one grouping variable or two grouping variables in an additive way(e.g. sex+group~), and the right side of ~ must have variables in an additive way.
`data`	A data.frame
`missing`	logical If true, missing value analysis performed
`...`	Further arguments to be passed to gaze()

Value

An object of class "gaze" which inherits from the class "data.frame" with at least the following attributes:

attr(*,"yvars) character. name of dependent variable

`gazeCat` *Summary function for categorical variable*

Description

Summary function for categorical variable

Usage

```
gazeCat(
  data,
  x,
  y = NULL,
  max.ylev = 5,
  digits = 1,
  show.total = FALSE,
  show.n = FALSE,
  show.missing = FALSE,
  show.stats = TRUE,
  origData = NULL,
  show.p = TRUE,
  method = 1,
  catMethod = 2,
  maxCatLevel = 20,
  ...
)
```

Arguments

`data`	A data frame
`x`	Name of a categorical variable
`y`	Name of a variable, either continuous or categorical
`max.ylev`	max.ylev An integer indicating the maximum number of levels of grouping variable ('y'). If a column have unique values less than max.ylev it is treated as a categorical variable. Default value is 5.
`digits`	Numeric
`show.total`	logical. Whether or not show total column
`show.n`	logical. Whether or not show N column
`show.missing`	logical. Whether or not show missing column
`show.stats`	logical. Whether or not show stats column
`origData`	A data.frame containing original data
`show.p`	logical. Whether or not show p column
`method`	method An integer indicating methods for continuous variables. Possible values in methods are 1 forces analysis as normal-distributed 2 forces analysis as continuous non-normal 3 performs a Shapiro-Wilk test or nortest::ad.test to decide between normal or non-normal Default value is 1.
`catMethod`	An integer indicating methods for categorical variables. Possible values in methods are **0** Perform chisq.test first. If warning present, perform fisher test **1** Perform chisq.test without continuity correction **2** Perform chisq.test with continuity correction **3** perform fisher.test

	4 perform prop.trend test Default value is 2.
`maxCatLevel`	An integer indicating the maximum number of unique levels of categorical variable. If a column have unique values more than maxCatLevel, categorical summarization will not be performed.
`...`	Further arguments

Value

An object of class "data.frame" or "tibble"

Examples

```
require(moonBook)
gazeCat(acs,"Dx")
gazeCat(acs,"Dx","smoking")
gazeCat(acs,"sex","Dx",show.p=TRUE)
gazeCat(acs,"Dx","sex",show.p=TRUE)
gazeCat(acs,"Dx","EF")
gazeCat(acs,"sex","EF",method=2)
gazeCat(mtcars,"cyl","hp")
```

`gazeCont` *Summary function for continuous variable*

Description

Summary function for continuous variable

Usage

```
gazeCont(
  data,
  x,
  y = NULL,
  max.ylev = 5,
  digits = 1,
  show.total = FALSE,
  show.n = FALSE,
  show.missing = FALSE,
  show.stats = TRUE,
  show.p = TRUE,
  method = 1,
  origData,
  ...
)
```

Arguments

`data`	A data.frame
`x`	A name of variable
`y`	A name of variable, either continuous or categorical
`max.ylev`	max.ylev An integer indicating the maximum number of levels of grouping variable ('y'). If a column have unique values less than max.ylev it is treated as a categorical variable. Default value is 5.
`digits`	integer indicating the number of decimal places
`show.total`	logical. Whether or not show total column
`show.n`	logical. Whether or not show N column
`show.missing`	logical. Whether or not show missing column
`show.stats`	logical. Whether or not show stats column
`show.p`	logical. Whether or not show p column
`method`	method An integer indicating methods for continuous variables. Possible values in methods are 1 forces analysis as normal-distributed 2 forces analysis as continuous non-normal 3 performs a Shapiro-Wilk test or nortest::ad.test to decide between normal or non-normal Default value is 1.
`origData`	A data.frame containing original data
`...`	Further arguments

Value

An object of class "data.frame" or "tibble"

Examples

```
gazeCont(mtcars,"hp")
gazeCont(mtcars,"hp","mpg")
require(moonBook)
gazeCont(acs,"log(age)")
gazeCont(acs,"age",method=2)
gazeCont(acs,"age","EF",method=2)
gazeCont(acs,"age","Dx",method=1)
gazeCont(acs,"age","Dx",show.p=TRUE,method=3)
```

`gaze_sub` *Summary function for categorical/continuous variable*

Description

Summary function for categorical/continuous variable

Usage

```
gaze_sub(data, xname, y = NULL, max.ylev = 5, ...)
```

Arguments

`data`	A data.frame
`xname`	A name of categorical/continuous vector
`y`	A name of vector, either continuous or categorical
`max.ylev`	max.ylev An integer indicating the maximum number of levels of grouping variable ('y'). If a column have unique values less than max.ylev it is treated as a categorical variable. Default value is 5.
`...`	Further arguments to be passed to gazeCont() or gazeCat()

Value

An object of class "data.frame" or "tibble"

Examples

```
require(moonBook)
gaze_sub(acs,"age")
gaze_sub(acs,"log(age)")
gaze_sub(acs,"I(age^2)")
gaze_sub(acs,"sex")
gaze_sub(acs,"age","EF")
gaze_sub(acs,"sex","EF")
gaze_sub(acs,"age","Dx")
gaze_sub(acs,"sex","Dx")
gaze_sub(iris,"Species","Sepal.Length")
```

`getInteraction` *Get interaction data from data*

Description

Get interaction data from data

Usage

```
getInteraction(name, data)
```

Arguments

`name`	a string with interaction term
`data`	a data.frame

Value

An object of class "data.frame"

Examples

```
data(acs,package="moonBook")
getInteraction("TC:Dx:sex",data=acs)
```

---

getN *Get number of data specified by 'name' and 'desc'*

Description

Get number of data specified by 'name' and 'desc'

Usage

```
getN(name, desc, data)
```

Arguments

`name`	a string with interaction term
`desc`	character
`data`	a data.frame

Value

A numeric vector

Examples

```
data(acs,package="moonBook")
df=getInteraction("TC:Dx:sex",data=acs)
getN(name=df$name,desc=df$desc,data=acs)
```

---

getSigVars *Get explanatory variables of a model with significance level below the threshold*

Description

Get explanatory variables of a model with significance level below the threshold

Usage

```
getSigVars(fit, threshold = 0.2, final = TRUE)
```

Arguments

`fit`	An object of class lm or glm
`threshold`	Numeric
`final`	logical if true, perform stepwise regression using step()

Value

A list containing the following components:

sigVars names of explanatory variables which have significant levels below the threshold in univariable model

finalVars names of explanatory variables included in final model as a result of `step`

Examples

```
library(survival)
data(cancer,package="survival")
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
getSigVars(fit)
fit=lm(mpg~hp*wt+am,data=mtcars)
getSigVars(fit)
```

`ggcmprsk` *Draw Cumulative Incidence Curves for Competing Risks*

Description

Draw Cumulative Incidence Curves for Competing Risks

Usage

```
ggcmprsk(x, data, id = NULL, se = FALSE, strata = NULL, facet = NULL, ...)
```

Arguments

`x`	A formula as time+status~1
`data`	A data.frame
`id`	character vector label for status
`se`	logical whether or not show confidence interval
`strata`	character vector label for strata
`facet`	numeric if facet is not NULL, draw plot with selected facets
`...`	Further arguments to be passed to tidycmprsk::cuminc

Value

An object of class "ggplot"

Examples

```
data(melanoma,package="boot")
melanoma$status1 = ifelse(melanoma$status==1,1,ifelse(melanoma$status==2,0,2))
ggcmprsk(time/365+status1~1,data=melanoma)
ggcmprsk(time/365+status1~1,data=melanoma,id=c("alive","melanoma","other"),se=TRUE)
ggcmprsk(time/365+status1~sex,data=melanoma)
ggcmprsk(time/365+status1~sex,data=melanoma,facet=1)
ggcmprsk(time/365+status1~sex,data=melanoma,
id=c("alive","melanoma","other"),strata=c("female","male"))
ggcmprsk(time/365+status1~sex,data=melanoma,
id=c("alive","melanoma","other"),strata=c("female","male"),facet=1)
```

`ggcmprsk2` *Compare cumulative incidence to th Kaplan-Meier estimate*

Description

Compare cumulative incidence to th Kaplan-Meier estimate

Usage

```
ggcmprsk2(
  x,
  data,
  id = c("disease", "other"),
  se = FALSE,
  xpos = c(2, 2),
  ypos = c(0.25, 0.7),
  ylabs = NULL,
  xlab = NULL,
  label = NULL,
  plot = TRUE
)
```

Arguments

`x`	A formula as time+status~1
`data`	A data.frame
`id`	Character vector of length2
`se`	logical whether or not show confidence interval
`xpos`	numeric x-axis position of label
`ypos`	numeric y-axis position of label
`ylabs`	string vector of length 2. y axis labels
`xlab`	A character. The x-axis label
`label`	string vector of length 2. Label names
`plot`	logical Whether or not print plot

Value

A list containing the following components:

df A long-form data.frame consist of time, est, upper,lower, id, method

df3 A data.frame for label consist of x, y, label, id

p A ggplot object

Examples

```
require(dplyr)
data(prostateSurvival,package="asaur")
prostateHighRisk <- prostateSurvival %>%
  filter(grade=="poor" & stage=="T2",ageGroup=="80+")
ggcmprsk2(survTime/12+status~1,data=prostateHighRisk,
  id=c("prostate cancer","other causes"))
```

`highlight2` *Highlight a data.frame*

Description

Highlight a data.frame

Usage

```
highlight2(x, i = NULL, j = NULL, style = NULL, include.colname = FALSE)
```

Arguments

`x`	A data.frame
`i`	numeric rows to highlight
`j`	numeric columns to hightlight
`style`	A style function or NULL
`include.colname`	logical Whether or not include colname

Value

a data.frame

Examples

```
head(mtcars) %>% highlight2(i=3) %>% printdf()
library(crayon)
head(mtcars) %>% highlight2(i=2) %>% highlight2(j=3,style=blue$bold) %>% printdf()
fit=lm(mpg~wt*hp,data=mtcars)
gaze(fit)
gaze(fit) %>% highlight2(j=4,include.colname=TRUE)
gaze(fit) %>% highlight2(i=2,j=4) %>% highlight2(i=2,j=2:3,style=blue$bold)
gaze(fit) %>% highlight2(i=2) %>% highlight2(j=3,style=blue$bold)
```

`imputedReg` *Make a multiple imputed model*

Description

Make a multiple imputed model

Usage

```
imputedReg(fit, data = NULL, m = 20, seed = 1234, digits = 2, mode = 1, ...)
```

Arguments

`fit`	An object of class lm, glm, coxph or survreg
`data`	a data.frame
`m`	Number of multiple imputations. The default is m=20.
`seed`	An integer that is used as argument by the set.seed() for offsetting the random number generator.
`digits`	Integer indicating the number of decimal place
`mode`	integer indicating summary mode of class survreg
`...`	Further argument to be passed to mice

Value

An object of class "imputedReg" which inherits from the class "data.frame"

Examples

```
data(cancer,package="survival")
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
imputedReg(fit)

library(survival)
fit=coxph(Surv(time,status)~rx+age+sex+nodes+obstruct+perfor,data=colon)
imputedReg(fit)
fit=survreg(Surv(time,status)~rx+age+sex+nodes+obstruct+perfor,data=colon)
imputedReg(fit)
imputedReg(fit,mode=2)
```

`is.mynumeric` *Decide whether a vector can be treated as a numeric variable*

Description

Decide whether a vector can be treated as a numeric variable

Usage

```
is.mynumeric(x, maxy.lev = 5)
```

Arguments

`x`	A vector
`maxy.lev`	An integer indicating the maximum number of unique values of a numeric variable be treated as a categorical variable

Value

A logical value

Examples

```
x=1:5
is.mynumeric(x)
x=1:13
is.mynumeric(x)
```

`loglogplot` *Draw log-log plot*

Description

Draw log-log plot

Usage

```
loglogplot(
  fit,
  xnames = NULL,
  main = NULL,
  labels = NULL,
  no = 3,
  add.loess = FALSE,
  add.lm = TRUE,
  type = "l",
```

```
  se = TRUE,
  what = "surv",
  legend.position = NULL,
  ...
)
```

Arguments

`fit`	An object of class "coxph" or "survfit"
`xnames`	character Names of explanatory variable to plot
`main`	String Title of plot
`labels`	String vector Used as legend in legend
`no`	Numeric The number of groups to be converted
`add.loess`	logical If true, add loess regression line
`add.lm`	logical If true, add linear regression line
`type`	character "l" or "p"
`se`	logical If true, add se
`what`	character One of c("surv","survOdds","failureOdds")
`legend.position`	legend position. One of c("left","top","bottom","right") or numeric vector of length 2.
`...`	Furhter arguments to be passed to plot()

Value

A ggplot or no return value, called for side effects

Examples

```
require(survival)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~x,data=leukemia)
loglogplot(fit)
fit=survfit(Surv(time,status)~1,data=anderson)
loglogplot(fit)
fit=survfit(Surv(time,status)~sex,data=anderson)
loglogplot(fit)
fit=survfit(Surv(time,status)~logWBC,data=anderson)
loglogplot(fit)
fit=survfit(Surv(time,status)~logWBC+rx,data=anderson)
loglogplot(fit,no=2)
fit=survfit(Surv(time,status)~rx,data=anderson)
loglogplot(fit,type="p")
fit=survfit(Surv(time,status)~WBCCAT,data=anderson2)
loglogplot(fit,type="p",what="survOdds")
loglogplot(fit,type="p",what="failureOdds")
```

`maxnchar` *Return maximum character number except NA*

Description

Return maximum character number except NA

Usage

```
maxnchar(x)
```

Arguments

`x`	a vector

Value

A numeric vector of length 1

Examples

```
x=c(1,2,"sadf",NA)
maxnchar(x)
data(acs,package="moonBook")
lapply(acs,maxnchar)
```

`modelPlot` *Draw coefficients/odds ratio/hazard ratio plot*

Description

Draw coefficients/odds ratio/hazard ratio plot

Usage

```
modelPlot(
  fit,
  widths = NULL,
  change.pointsize = TRUE,
  show.OR = TRUE,
  show.ref = TRUE,
  bw = TRUE,
  legend.position = "top",
  ...
)
```

Arguments

`fit`	An object of class glm
`widths`	Numeric vector
`change.pointsize`	logical Whether or not change point size
`show.OR`	logical Whether or not show odds ratio
`show.ref`	logical Whether or not show reference
`bw`	logical If true, use grey scale
`legend.position`	legend position default value is 'top'
`...`	Further arguments to be passed to autoReg()

Value

modelPlot returns an object of class "modelPlot" An object of class modelPlot is a list containing at least of the following components:

tab1 The first table containing names

tab2 The 2nd table containing levels

tab3 The 3rd table containing coefficients or odds ratio or hazards ratio

p A ggplot

widths the widths of the tables and the ggplot

Examples

```
fit=lm(mpg~wt*hp+am,data=mtcars)
modelPlot(fit,widths=c(1,0,2,3))
modelPlot(fit,uni=TRUE,threshold=1,widths=c(1,0,2,3))
fit=lm(Sepal.Width~Sepal.Length*Species,data=iris)
modelPlot(fit)
modelPlot(fit,uni=TRUE,change.pointsize=FALSE)

data(cancer,package="survival")
fit=glm(status~rx+age+sex+nodes+obstruct+perfor,data=colon,family="binomial")
modelPlot(fit)
modelPlot(fit,uni=TRUE,multi=TRUE,threshold=1)
modelPlot(fit,multi=TRUE,imputed=TRUE,change.pointsize=FALSE)
data(colon_s,package="finalfit")
fit=glm(mort_5yr~age.factor+sex.factor+obstruct.factor+perfor.factor,data=colon_s,family="binomial")
modelPlot(fit)
modelPlot(fit,uni=TRUE,multi=TRUE,threshold=1)
modelPlot(fit,uni=TRUE,multi=TRUE)
modelPlot(fit,uni=TRUE,multi=TRUE,threshold=1,show.ref=FALSE)
library(survival)
fit=coxph(Surv(time,status)~rx+age+sex+obstruct+perfor,data=colon)
modelPlot(fit)
modelPlot(fit,uni=TRUE,threshold=1)
modelPlot(fit,multi=FALSE,final=TRUE,threshold=1)
```

```
fit=coxph(Surv(time,status)~age.factor+sex.factor+obstruct.factor+perfor.factor,data=colon_s)
modelPlot(fit)
modelPlot(fit,uni=TRUE,threshold=1)
modelPlot(fit,uni=TRUE,threshold=1,show.ref=FALSE)
modelPlot(fit,imputed=TRUE)
```

`modelsSummary` *Makes table summarizing list of models*

Description

Makes table summarizing list of models

Usage

```
modelsSummary(fitlist, show.lik = FALSE)
```

Arguments

`fitlist`	A list of objects of class "coxph"
`show.lik`	logical Whether or not show likelihood test results

Value

No return value, called for side effects

Examples

```
library(survival)
fit1=coxph(Surv(time,status) ~rx,data=anderson)
fit2=coxph(Surv(time,status) ~rx+logWBC,data=anderson)
fit3=coxph(Surv(time,status) ~rx*logWBC,data=anderson)
fitlist=list(fit1,fit2,fit3)
modelsSummary(fitlist)
```

`modelsSummaryTable` *Makes flextable summarizing list of models*

Description

Makes flextable summarizing list of models

Usage

```
modelsSummaryTable(fitlist, labels = NULL, show.lik = FALSE)
```

Arguments

`fitlist`	A list of objects of class "coxph"
`labels`	character labels of models
`show.lik`	logical Whether or not show likelihood test results

Value

A flextable

Examples

```
library(survival)
fit1=coxph(Surv(time,status) ~rx,data=anderson)
fit2=coxph(Surv(time,status) ~rx+logWBC,data=anderson)
fit3=coxph(Surv(time,status) ~rx*logWBC,data=anderson)
fitlist=list(fit1,fit2,fit3)
modelsSummaryTable(fitlist)
```

`my.chisq.test2` *Statistical test for categorical variables Statistical test for categorical variables*

Description

Statistical test for categorical variables Statistical test for categorical variables

Usage

```
my.chisq.test2(x, y, catMethod = 2, all = FALSE)
```

Arguments

`x`	a vector
`y`	a vector
`catMethod`	An integer indicating methods for categorical variables. Possible values in methods are **0** Perform chisq.test first. If warning present, perform fisher test **1** Perform chisq.test without continuity correction **2** Perform chisq.test with continuity correction **3** perform fisher.test **4** perform prop.trend test Default value is 2.
`all`	A logical

Value

A numeric vector of length 1

Examples

```
library(moonBook)
x=acs$sex
y=acs$Dx
my.chisq.test2(x,y)
```

`my.t.test2` *Statistical test for continuous variables*

Description

Statistical test for continuous variables

Usage

```
my.t.test2(y, x, method = 1, all = FALSE)
```

Arguments

`y`	a categorical vector
`x`	a numeric vector
`method`	method An integer indicating methods for continuous variables. Possible values in methods are 1 forces analysis as normal-distributed 2 forces analysis as continuous non-normal 3 performs a Shapiro-Wilk test or nortest::ad.test to decide between normal or non-normal Default value is 1.
`all`	A logical

Value

A numeric vector of length 1

Examples

```
library(moonBook)
y=acs$sex
x=acs$height
my.t.test2(y,x)
```

---

mycphSimple *Fit Simple Proportional Hazards Regression Model*

Description

Fit Simple Proportional Hazards Regression Model

Usage

```
mycphSimple(fit, threshold = 0.2, digits = 2)
```

Arguments

`fit`	An object of class coxph
`threshold`	numeric p-value threshold to enter multiple model
`digits`	integer indicating the position decimal place

Value

An object of class "data.frame"

Examples

```
require(survival)
data(cancer)
fit=coxph(Surv(time,status)~age+sex+obstruct+perfor,data=colon)
mycphSimple(fit)
```

---

myformat *Convert data.frame to printable format*

Description

Convert data.frame to printable format

Usage

```
myformat(x, showid = FALSE, digits = 3)
```

Arguments

`x`	A data.frame
`showid`	logical if TRUE, show id
`digits`	Integer indicating the number of decimal places

Value

A data.frame

Examples

```
fit=lm(mpg~wt*hp,data=mtcars)
gaze(fit) %>% myformat()
```

`myft` *Convert data.frame into flextable*

Description

Convert data.frame into flextable

Usage

```
myft(x, vanilla = TRUE, fontsize = 10, digits, showid = FALSE, ...)
```

Arguments

`x`	A data.frame
`vanilla`	logical
`fontsize`	Numeric
`digits`	integer indicating the position of decimal place
`showid`	logical if TRUE, show id
`...`	Further arguments to be passed to df2flextable()

Value

An object of class `flextable`

Examples

```
data(acs,package="moonBook")
library(dplyr)
gaze(acs) %>% myft()
gaze(sex~.,acs) %>% myft()
fit=lm(mpg~hp*wt,data=mtcars)
gaze(fit) %>% myft()
library(survival)
fit=coxph(Surv(time,status) ~rx,data=anderson1)
gaze(fit) %>% myft()

gaze(sex+Dx~.,data=acs,show.p=TRUE,show.total=TRUE,show.n=TRUE,shiw.missing=TRUE) %>% myft()
gaze(Dx+sex~cardiogenicShock,data=acs,show.p=TRUE) %>% myft()
gaze(Dx+sex+HBP~cardiogenicShock,data=acs,show.p=TRUE) %>% myft()
```

`mysurvregSimple` *Fit Simple AFT Model*

Description

Fit Simple AFT Model

Usage

```
mysurvregSimple(fit, threshold = 0.2, digits = 2, mode = 1)
```

Arguments

`fit`	An object of class survreg
`threshold`	numeric p-value threshold to enter multiple model
`digits`	integer indicating the position decimal place
`mode`	integer

Value

An object of class "data.frame"

Examples

```
require(survival)
data(cancer)
fit=survreg(Surv(time,status)~rx+age+strata(sex)+obstruct+perfor,data=colon)
mysurvregSimple(fit)
```

`num2factor` *Convert a numeric column in a data.frame to a factor*

Description

Convert a numeric column in a data.frame to a factor

Usage

```
num2factor(data, call, name, no = 3)
```

Arguments

`data`	A data.frame
`call`	a function call
`name`	character Name of numeric column
`no`	numeric

Value

A data.frame

Examples

```
num2factor(anderson,name="logWBC")
library(survival)
fit=coxph(Surv(time,status)~logWBC+rx,data=anderson)
num2factor(anderson,call=fit$call,name="logWBC",no=2)
```

`num2stat` *Summarize numeric vector to statistical summary*

Description

Summarize numeric vector to statistical summary

Usage

```
num2stat(x, digits = 1, method = 1, p = NULL)
```

Arguments

`x`	A numeric vector
`digits`	integer indicating the number of decimal places
`method`	An integer indicating methods for continuous variables. Possible values in methods are 1 forces analysis as normal-distributed 2 forces analysis as continuous non-normal 3 performs a Shapiro-Wilk test or nortest::ad.test to decide between normal or non-normal Default value is 1.
`p`	A numeric

Value

A character vector of length 1

Examples

```
library(moonBook)
num2stat(acs$age)
num2stat(acs$age,method=2)
```

`OEplot` *Draw an Observed vs Expected plot*

Description

Draw an Observed vs Expected plot

Usage

```
OEplot(fit, xnames = NULL, no = 3, maxy.lev = 5, median = TRUE)
```

Arguments

`fit`	An object of class "coxph"
`xnames`	Character Names of explanatory variable to plot
`no`	integer Number of groups to be made
`maxy.lev`	Integer Maximum unique length of a numeric variable to be treated as categorical variables
`median`	logical

Value

No return value, called for side effects

Examples

```
library(survival)
data(cancer,package="survival")
fit=coxph(Surv(time,status)~rx+age+sex,data=colon)
OEplot(fit)
OEplot(fit,xnames="sex")
## Not run:
fit=coxph(Surv(time,status)~age,data=colon)
OEplot(fit)
fit=coxph(Surv(time,status)~logWBC,data=anderson)
OEplot(fit)

## End(Not run)
```

`p2character2` *Change p value to string*

Description

Change p value to string

Usage

```
p2character2(x, digits = 3, add.p = TRUE)
```

Arguments

`x`	a numeric
`digits`	integer indicating decimal place
`add.p`	logical

Value

A character vector

`print.autoReg` *S3 method print for an object of class autoReg*

Description

S3 method print for an object of class autoReg

Usage

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

Arguments

`x`	An object of class autoReg
`...`	Further arguments

Value

No return value, called for side effects

Examples

```
data(cancer,package="survival")
fit=glm(status~rx+sex+age+obstruct+nodes,data=colon,family="binomial")
autoReg(fit)
```

`print.gaze` *S3 method print for an object of class gaze*

Description

S3 method print for an object of class gaze

Usage

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

Arguments

`x`	An object of class gaze
`...`	Further arguments

Value

No return value, called for side effects

Examples

```
data(acs,package="moonBook")
x=gaze(acs,show.n=TRUE,show.missing=TRUE)
gaze(sex~.,acs,show.p=TRUE,show.n=TRUE,show.missing=TRUE,show.total=TRUE)

gaze(Dx+sex~.,acs,show.p=TRUE)
gaze(sex+Dx+HBP~.,acs,show.p=TRUE)
```

`print.modelPlot` *S3 method for an class modelPlot*

Description

S3 method for an class modelPlot

Usage

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

Arguments

`x`	An object of class modelPlot
`...`	Further arguments to be passed to plot()

`printdf` *Print function for data.frame*

Description

Print function for data.frame

Usage

```
printdf(x)
```

Arguments

`x`	A data.frame

Value

No return value, called for side effects

Examples

```
x=mtcars[1:5,1:5]
printdf(x)
```

`removeDup` *Remove duplicated term*

Description

Remove duplicated term

Usage

```
removeDup(x, replacement = "")
```

Arguments

`x`	A vector
`replacement`	A character to be replaced or NA

Value

A vector with the same class as x

Examples

```
x=rep(1:5,each=3)
removeDup(x)
```

`residualNull` *Make a residual plot of NULL model*

Description

Make a residual plot of NULL model

Usage

```
residualNull(x, add.log = TRUE, type = "martingale")
```

Arguments

`x`	An object of calss coxph
`add.log`	logical If true, log of predictor varaibles are added
`type`	character type of residuals

Examples

```
library(survival)
data(pharmacoSmoking,package="asaur")
pharmacoSmoking$priorAttemptsT=pharmacoSmoking$priorAttempts
pharmacoSmoking$priorAttemptsT[pharmacoSmoking$priorAttemptsT>20]=20
x=coxph(Surv(ttr,relapse)~age+priorAttemptsT+longestNoSmoke,data=pharmacoSmoking)
residualNull(x)
```

`residualPlot` *Draw a residual plot with an object of class coxph*

Description

Draw a residual plot with an object of class coxph

Usage

```
residualPlot(
  fit,
  type = "martingale",
  vars = NULL,
  ncol = 2,
  show.point = TRUE,
  se = TRUE,
  topn = 5,
  labelsize = 4
)
```

Arguments

`fit`	An object of class coxph or survreg
`type`	character One of the c("martingale","deviance","score","schoenfeld", "dfbeta","dfbetas","scaledsch","part Default value is "martingale".
`vars`	character Names of variables to plot. default value is NULL
`ncol`	numeric number of columns
`show.point`	logical Whether or not show point
`se`	logical Whether or not show se
`topn`	numeric number of data to be labelled
`labelsize`	numeric size of label

Value

A patchwork object

Examples

```
require(survival)
data(cancer)
fit=coxph(Surv(time,status==2)~log(bili)+age+cluster(edema),data=pbc)
residualPlot(fit)
residualPlot(fit,vars="age")
fit=coxph(Surv(time,status==2)~age,data=pbc)
residualPlot(fit)
residualPlot(fit,"partial")
fit=coxph(Surv(time,status)~rx+sex+logWBC,data=anderson)
residualPlot(fit,ncol=3)
## Not run:
data(pharmacoSmoking,package="asaur")
fit=coxph(Surv(ttr,relapse)~grp+employment+age,data=pharmacoSmoking)
residualPlot(fit)
residualPlot(fit,var="age")
residualPlot(fit,type="dfbeta")
residualPlot(fit,type="dfbeta",var="age")
residualPlot(fit,type="dfbeta",var="employment")
residualPlot(fit,type="dfbeta",var="employmentother")
pharmacoSmoking$ttr[pharmacoSmoking$ttr==0]=0.5
fit=survreg(Surv(ttr,relapse)~grp+age+employment,data=pharmacoSmoking,dist="weibull")
residualPlot(fit,type="response")
residualPlot(fit,type="deviance")
residualPlot(fit,type="dfbeta",vars="age")
fit=survreg(Surv(time,status)~ph.ecog+sex*age,data=lung,dist="weibull")
residualPlot(fit,"dfbeta")
residualPlot(fit,"deviance")

## End(Not run)
```

`restoreData` *restore data with factor in column name*

Description

restore data with factor in column name

Usage

```
restoreData(data)
```

Arguments

`data`	An object of class "data.frame"

Value

An object of class "data.frame"

`restoreData2` *restore data with I() in column name*

Description

restore data with I() in column name

Usage

```
restoreData2(df)
```

Arguments

`df`	An object of class "data.frame"

Value

An object of class "data.frame"

`restoreData3` *restore data with operator in column name*

Description

restore data with operator in column name

Usage

```
restoreData3(df, changeLabel = FALSE)
```

Arguments

`df`	An object of class "data.frame"
`changeLabel`	logical

Value

An object of class "data.frame"

`revOperator` *get opposite arithmetic operator*

Description

get opposite arithmetic operator

Usage

```
revOperator(operator)
```

Arguments

`operator`	A character

Value

A character

`roundDf` *Convert numeric columns of data.frame to character*

Description

Convert numeric columns of data.frame to character

Usage

```
roundDf(df, digits = 2)
```

Arguments

`df`	a data.frame
`digits`	integer indicating the number of decimal places

Value

An object of class "data.frame"

`setLabel` *Add label to a vector*

Description

Add label to a vector

Usage

```
setLabel(x, label = "")
```

Arguments

`x`	a vector
`label`	string

Value

a labelled vector

`shorten` *Shorten an object of class gaze*

Description

Shorten an object of class gaze

Usage

```
shorten(x, xname = NULL, ref = 1)
```

Arguments

`x`	an object of class gaze
`xname`	A variable name
`ref`	Numeric Th number to be used as reference

Value

An object of class "gaze" which is described in `gaze`

Examples

```
data(acs,package="moonBook")
x=gaze(sex~.,data=acs)
shorten(x)
```

`showEffect` *Show effects of covariates*

Description

Show effects of covariates

Usage

```
showEffect(
  fit,
  x = NULL,
  color = NULL,
  facet = NULL,
  pred.values = list(),
  se = TRUE,
  logy = TRUE,
  collabel = label_both,
  rowlabel = label_both
)
```

Arguments

`fit`	An object of class survreg
`x`	character name of x-axis variable
`color`	character name of color variable
`facet`	character name of facet variable
`pred.values`	list list of values of predictor variables
`se`	logical whether or not show se
`logy`	logical WHether or not draw y-axis on log scale
`collabel`	labeller for column
`rowlabel`	labeller for row

Value

A ggplot

Examples

```
library(survival)
library(ggplot2)
fit=survreg(Surv(time,status)~ph.ecog+sex*age,data=lung,dist="weibull")
showEffect(fit)
fit=survreg(Surv(time,status)~rx+sex+age+obstruct+adhere,data=colon,dist="weibull")
showEffect(fit)
showEffect(fit,rowlabel=label_value)
fit=survreg(Surv(time,status)~ph.ecog+sex,data=lung,dist="weibull")
showEffect(fit)
fit=survreg(Surv(time,status)~ph.ecog+age,data=lung,dist="weibull")
showEffect(fit)
fit=survreg(Surv(time,status)~sex*age,data=lung,dist="weibull")
showEffect(fit)
fit=survreg(Surv(time,status)~age,data=lung,dist="weibull")
showEffect(fit)
```

`strata2df` *Convert a character vector to a data.frame*

Description

Convert a character vector to a data.frame

Usage

```
strata2df(strata)
```

Arguments

`strata`	A character vector

Value

A data.frame

`survfit2df` *Extract survival data from an object of class "survfit"*

Description

Extract survival data from an object of class "survfit"

Usage

```
survfit2df(fit, labels = NULL)
```

Arguments

`fit`	An object of class "survfit"
`labels`	Character

Value

A data.frame

Examples

```
library(survival)
data(cancer,package="survival")
fit=survfit(coxph(Surv(time,status)~sex+age+strata(rx),data=colon))
survfit2df(fit)
## Not run:
fit=coxph(Surv(time,status)~sex+age+strata(rx),data=colon)
fit=survfit(as.formula(deparse(fit$terms)),data=fit2model(fit))
survfit2df(fit)
fit=survfit(Surv(time,status)~rx+sex+age,data=colon)
survfit2df(fit)
fit=survfit(Surv(time,status)~1,data=colon)
survfit2df(fit)

## End(Not run)
```

`survreg2final` *Make final model using stepwise backward elimination*

Description

Make final model using stepwise backward elimination

Usage

```
survreg2final(fit, threshold = 0.2)
```

Arguments

`fit`	An object of class "survreg"
`threshold`	Numeric

Value

An object of class "survreg" which is described in `survreg`

Examples

```
require(survival)
data(cancer)
fit=survreg(Surv(time,status)~rx+age+sex+obstruct+perfor,data=colon)
survreg2final(fit)
```

`survreg2multi` *Make multivariable regression model by selecting univariable models with p.value below threshold*

Description

Make multivariable regression model by selecting univariable models with p.value below threshold

Usage

```
survreg2multi(fit, threshold = 0.2)
```

Arguments

`fit`	An object of class "survreg"
`threshold`	Numeric

Value

An object of class "survreg"

Examples

```
require(survival)
data(cancer)
fit=survreg(Surv(time,status)~rx+age+sex+obstruct+perfor,data=colon)
survreg2multi(fit)
```

Index