Package ‘modtools’

May 28, 2015

Type Package

Title Additional tools for model diagnostic and selection

Version 1.0-9

Date 2014-09-09

Author Pierre BADY <pierre.bady@free.fr>

Maintainer Pierre BADY <pierre.bady@free.fr>

Depends R (>= 3.1.1), boot

Suggests car,epitools,epiR, haplo.ccs,faraway,MASS,irr

Description Additional tools for model diagnostic

License GPL version 2 or newer

URL http://pierre.bady@free.fr

Archs i386, x64

R topics documented:

`modtools-package` *Additional tools for model diagnostic and selection*

Description

This package contains exhaustive regression functions and some tools to evaluate the goodness of fit and the model quality.

Details

Package:	modtools
Type:	Package
Version:	1.0-7
Date:	2009-01-14
License:	GPL version 2 or newer
LazyLoad:	yes
Depends:	R (>= 2.9.0),boot
Suggests:	car

# nothing for the moment

Author(s)

Pierre Bady <pierre.bady@free.fr>
Maintainer: Pierre Bady <pierre.bady@free.fr>

References

Saporta (2006)
Efron et al
Hinkley and Davidson (1998)
etc
to be completed

See Also

`car`, `boot`

Examples

```
# nothing for the moment
```

anscresid *Anscombe's Residuals*

Description

The fonction provides Anscombe's residuals associated with an object 'glm'.

Usage

```
anscresid(object, ...)
```

Arguments

`object`	Object of class inheriting from '"glm"'
`...`	further arguments passed to or from other methods

Details

The formulas to compute the Anscombe'sresiduals are defined as follow:is defined as follow: for gaussian family:

$$r_{ans} = y - \mu$$

for inverse.gaussian family:

$$r_{ans} = (log(y) - log(\mu))/(\mu^{0.5})$$

for binomial family:

$$r_{ans} = \sqrt(m) * (b(y) - b(\mu)) * (\mu * (1 - \mu))^{-1/6}$$

$$b(x) = \int_0^x (x^{-1/3} * (1 - x)^{-1/3}$$

for poisson family:

$$r_{ans} = (3/2) * ((y^{2/3}) * \mu^{-1/6} - \mu^{0.5})$$

for Gamma family:

$$r_{ans} = 3 * ((y/\mu)^{1/3} - 1)$$

Value

The function returns a numerical vector which contains the values of anscombe's residuals for each observation.

References

McCullagh P. and Nelder, J. A. (1989) Generalized Linear Models. London: Chapman and Hall.
Pierce, D. A. and Schafer, D. W. (1986) Residuals in Generalized Linear Models, Journal of the American Statistical Association, **81**,396,977-986.
Abscombe 1953

See Also

glm,residuals.glm

Examples

```
## binomial

## poisson

## Gamma
```

auc *Area Under Curves*

Description

This function compute the area under curves.

Usage

```
auc(x,y)
```

Arguments

x	a numeric vector
y	a numeric vector

Value

The function returns a numeric value corresponding to the area under the curves.

Examples

```
# y <- rpois(1:10)
# d1 <- density(y)
# auc(d1$x,d1$y)
```

bootcoef *~~function to do ... ~~*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
bootcoef(object,data,R=99,...)
```

Arguments

`object`	~~Describe `object` here~~ attention object et different
`data`	~~Describe `data` here~~
`R`	~~Describe `R` here~~
`...`	~~Describe `...` here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

References

Davison, A.C. and Hinkley, D.V. (1997) Bootstrap Methods and Their Application. Cambridge University Press.
Efron et al.
Efron et al.

See Also

~~objects to See Also as `help`, ~~~

Examples

```
# add an example
```

`bootcoef.ci` *~~function to do ... ~~*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
bootcoef.ci(object,type="percent",level=0.05,...)
```

Arguments

`object`	~~Describe `object` here~~ attention object et different
`type`	~~Describe `R` here~~
`level`	~~Describe `R` here~~
`...`	~~Describe `...` here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as `help`, ~~~

Examples

```
# add an example
```

`bootvalid` *Validation procedure based on bootstrap for object from 'glm' or 'lm'.*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
bootvalid(object, ...)
## Default S3 method:
bootvalid(object, data, cost = costMSE,
R = 99,method = "raw",...)
## S3 method for class 'bootcorrected'
summary(object,display=TRUE,...)
```

Arguments

`object`	Object of class inheriting from '"glm"'
`data`	~~Describe `data` here~~
`cost`	~~Describe `cost` here~~
`R`	~~Describe `R` here~~
`method`	'raw' or 'corrected'
`display`	a logical value (by default, display=TRUE)
`...`	further arguments passed to or from other methods

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

References

Davison, A.C. and Hinkley, D.V. (1997) Bootstrap Methods and Their Application. Cambridge University Press.
Efron et al.
Efron et al.
Harrel 2001

See Also

~~objects to See Also as `help`, ~~~

Examples

```
## glm1

## bootvalid

## histogram of results
```

ckappa *Kappa's index*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
ckappa(x)
```

Arguments

x	a matrix

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

comp1	Description of 'comp1'
comp2	Description of 'comp2'

...

Note

~~further notes~~

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as help, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x)
{
    if (ncol(x) != nrow(x))
        stop("non covenient dimension !")
    N <- sum(x)
    sxii <- sum(diag(x))
    sxip <- apply(x, 2, sum)
```

```
    sxpi <- apply(x, 1, sum)
    k <- (N * sxii - sum(sxip * sxpi))/(N * N - sum(sxip * sxpi))
    return(k)
  }
```

---

cost *Cost functions*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
costAVER(y, yhat = 0)
costMAE(y, yhat = 0)
costMSE(y, yhat = 0)
costRMSE(y, yhat = 0)
costSlope(y, yhat)
costOri(y, yhat)
costR2(y, yhat)
costBIN(y, mu = 0, cutoff = 0.5)
costKappa(y, yhat, cutoff = 0.5)
costGoodCl(y,yhat, cutoff = 0.5)
```

Arguments

`y`	a numerical vector corresponding to the obsevred values.
`yhat`	a numerical vector corresponding to the expected values.
`mu`	a numerical vector corresponding to the expected values.
`cutoff`	a numerical value in the range [0,1] corresponding to the threshold to transform the values 'mu' into binary vector.

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~

add descriptions and formula

References

Davidson and Hinkley, Saporta 2006

See Also

SRM,bootvalid,cv.glm

Examples

```
x <- rnorm(20,2,5)
y <- -6+x*3+rnorm(20)
lm1 <- lm(y~x)
costRMSE(y,lm1$fitted)
```

---

goodclassif *Good classification*

Description

This function gives the percentage of good classification in confusion matrix.

Usage

```
goodclassif(x)
```

Arguments

x	a matrix or data.frame

Value

~Describe the value returned If it is a LIST, use

comp1	Description of 'comp1'
comp2	Description of 'comp2'

...

Note

~~further notes~~ + description des matrices de confusion

sum de la diagonale

References

~put references to the literature/web site here ~

See Also

kappa,roc

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x)
{
    if (ncol(x) != nrow(x))
        stop("non covenient dimension !")
    N <- sum(x)
    sxii <- sum(diag(x))
    return(sxii/N)
  }
```

`hist.boot` *Graphical representation of object ’boot’*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
## S3 method for class 'boot'
hist(x, nclass = 10, coeff = 1, mfrow = NULL, which.par = 1:length(x$t0),
sub = NULL, ...)
```

Arguments

`x` An object of class ’"boot"’ containing the output of a bootstrap calculation.

`nclass` numeric (integer).’nclass’is equivalent to ’breaks’ for a scalar or character argument.

`coeff` ~~Describe `coeff` here~~

`mfrow` ~~Describe `mfrow` here~~

`which.par` ~~Describe `which.par` here~~

`sub` ~~Describe `sub` here~~

`...` further graphical parameters passed to ’plot.histogram’

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1` Description of ’comp1’

`comp2` Description of ’comp2’

...

Note

~~further notes~~

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as help, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x, nclass = 10, coeff = 1, mfrow = NULL, which.par = 1:length(x$t0),
    sub = NULL, ...)
{
    if (!inherits(x, "boot"))
        stop("non convenient argument")
    opar <- par(ask = par("ask"), mfrow = par("mfrow"))
    on.exit(par(opar))
    if (is.null(mfrow))
        mfrow <- n2mfrow(length(which.par))
    par(mfrow = mfrow)
    if (length(which.par) > prod(mfrow))
        par(ask = TRUE)
    for (i in which.par) {
        if (is.null(sub))
            sub <- paste("t", i, "*", sep = "")
        obs <- x$t0[i]
        sim <- x$t[, i]
        r0 <- c(sim, obs)
        h0 <- hist(sim, plot = FALSE, nclass = nclass)
        y0 <- max(h0$counts)
        l0 <- max(sim) - min(sim)
        w0 <- l0/(log(length(sim), base = 2) + 1)
        w0 <- w0 * coeff
        xlim0 <- range(r0) + c(-w0, w0)
        hist(sim, plot = TRUE, nclass = nclass, xlim = xlim0,
            col = grey(0.8), main = sub, ...)
        lines(c(obs, obs), c(y0/2, 0))
        points(obs, y0/2, pch = 18, cex = 2)
    }
    invisible()
  }
```

`histsim` *Graphical represenation of simulation results*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
histsim(sim, obs, nclass = 10, coeff = 1, ...)
```

Arguments

`sim`	~~Describe `sim` here~~
`obs`	~~Describe `obs` here~~
`nclass`	~~Describe `nclass` here~~
`coeff`	~~Describe `coeff` here~~
`...`	further graphical parameters passed to 'plot.histogram'

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as `help`, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (sim, obs, nclass = 10, coeff = 1, ...)
{
    r0 <- c(sim, obs)
    h0 <- hist(sim, plot = FALSE, nclass = nclass)
    y0 <- max(h0$counts)
    l0 <- max(sim) - min(sim)
    w0 <- l0/(log(length(sim), base = 2) + 1)
    w0 <- w0 * coeff
    xlim0 <- range(r0) + c(-w0, w0)
    hist(sim, plot = TRUE, nclass = nclass, xlim = xlim0, col = grey(0.8),
        ...)
    lines(c(obs, obs), c(y0/2, 0))
    points(obs, y0/2, pch = 18, cex = 2)
    invisible()
  }
```

`intervals` *confidence and prediction/tolerance intervals for glm*

Description

This method gives confidence and prediction/tolerance intervals for the expected values from a generalized linear model (object of class ’glm’).

Usage

```
intervals(object, ...)
## S3 method for class 'glm'
intervals(object, newdata, type = "response", interval = "confidence",
method = 1, level = 0.05, ...)
```

Arguments

`object`	Object of class inheriting from ’"glm"’
`newdata`	An optional data frame in which to look for variables with which to predict. If omitted, the data values are used.
`type`	the type of prediction required (by defaut type="response")
`interval`	Type of interval calculation.
`method`	a numerical values (by default method=1). The option ’method = 1’ gives intervals based on the carry-over of the extreme values. The option ’method = 2’ provides "direct" interval (see the section note for more details.
`level`	Tolerance/confidence level
`...`	further arguments passed to or from other methods

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~ Several procedure can provide confidence (or prediction/tolerance) intervals. In the function intervals.glm, we propose the two following procedures: method 1: confidence and prediction intervals based on the carry-over of the extreme values. This method is an extrapolation of the results obtained in the linear model. for confidence intervals

$$sigma = \sqrt{x^t VCOV x}$$

where for prediction intervals

$$sigma = \sqrt{1 + x^t VCOV x}$$

where method 2: "direct" confidence intervals

$$\hat{y} + -epsilon_{alpha} var(\hat{y}) \sqrt{x^t VCOV x}$$

where $var(\hat{y}) = psivar()$

References

~put references to the literature/web site here ~

See Also

`glm`,`predict.glm`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (object, ...)
{
    UseMethod("intervals")
  }
```

modperf *Model performance*

Description

Model performance

Usage

```
modperf(x, ...)
modperf.boot(x, data, cost = costRMSE, R = 99, ...)
modperf.cv(x, data, cost = costRMSE, K = 10, ...)
modperf.binary(x,...)
## Default S3 method:
modperf(x, data, cost = costRMSE, R = 99, ...)
## S3 method for class 'lm'
modperf(x, ...)
## S3 method for class 'glm'
modperf(x, ...)
```

Arguments

`x`

`data`

`cost`

`R`

`K`

`...`

Note

The function ’modperf.binary’ is based on the function ’perf.binary’. this function returns the following values: sensitivity,specificty, positive predictive value, negative predictive value and prevalence.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x, ...)
{
    UseMethod("modperf")
  }
```

modplot *Plot Diagnostics for glm and lm Objects*

Description

Ten plots are currently available for model assessment: Observed vs Expected values, Normal QQ-plot of residuals,etc...

Usage

```
#
plotBoot(object,cost=costBIN,R=200,nclass=13,
sub=paste("Bootstrap (R=",R,")",sep=""),plot=TRUE,...)
#
plotCovPat(object,sub="DX2 and Dbeta",plot=TRUE,...)
#
plotEtaResfunction(object,type="pearson",sub="Residual structure")
#
plotHalfnorm(object,sub="Half-normal plot",type="deviance",env=TRUE,...)
#
plotLeverage(object,type="pearson",sub="Leverage",cex=1.5,pch=20,...)
#
lotObsExp(object,sub="Expected vs observed values",...)
plotObsExpCat(object,sub="Expected vs observed values",horizontal=TRUE,...)
#
plotParRes(object,mgraph=NULL,...)
#
plotQQres(object,type="pearson",sub="QQ-norm for residuals")
#
plotResDens(object,type="pearson",sub="Residuals histogram",nclass=13,...)
```

Arguments

`object`	an object of class inheriting from 'lm' or glm
`cost`	
`R`	
`nclass`	
`plot`	
`sub`	
`env`	
`type`	
`cex`	
`pch`	
`horizontal`	
`mgraph`	
`...`	further arguments passed to or from other methods.

Details

The selected plots are drawn on a graphics device.
plotBoot
plotCovPat
lotEtaRes
plotHalfnorm
plotLeverage
plotObsExp: Observed vs Expected values
plotObsExpCat: Observed vs Expected values
plotParRes
plotQQres: Normal QQ-plot of residuals
plotResDens

Value

x is invisibly returned.

References

ted

See Also

`qqnorm`, `plot.glm`, `plot.lm`

Examples

```
### plot
```

`pseudoR2` *Pseudo-R2 for object 'glm'*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
pseudoR2(mod0, mod, ...)
## S3 method for class 'glm'
pseudoR2(mod0, mod, option = "phi", ...)
```

Arguments

`mod0`	~~Describe `mod0` here~~
`mod`	~~Describe `mod` here~~
`option`	~~Describe `option` here~~
`...`	~~Describe ... here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~

References

Mcfadden 1973, estrella 1998

See Also

~~objects to See Also as `help`, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (mod0, mod, ...)
{
    UseMethod("pseudoR2")
  }
```

`roc` *ROC functions*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
prep.roc(obs,pred,nbval=20,method="max",subset,...)
## S3 method for class 'roc'
plot(x, type = "curve", sub,posi = c(0.8, 0.2),...)
## S3 method for class 'roc'
print(x, ...)
## S3 method for class 'roc'
summary(object, rnd = 3, type = "curve", display = TRUE, ...)
```

Arguments

`obs`	~~Describe `obs` here~~
`pred`	~~Describe `pred` here~~
`nbval`	~~Describe `nbval` here~~
`subset`	~~Describe `subset` here~~
`x`	~~Describe `x` here~~
`method`	~~Describe `method` here~~
`sub`	~~Describe `sub` here~~
`object`	~~Describe `object` here~~
`type`	~~Describe `method` here~~
`posi`	information position
`rnd`	~~Describe `rnd` here~~
`display`	~~Describe `display` here~~
`...`	~~Describe `...` here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

With the option 'estim', the function prep.roc used the function OptimCut to define the optimal cut-off. this one can be based on several criteria:
fgoodclassif:
fkappa
fSpecSens
fSpecSens2

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as `help`, ~~~

Examples

```
x <- rnorm( 100 )
z <- rnorm( 100 )
w <- rnorm( 100 )
tigol <- function( x ) 1 - ( 1 + exp( x ) )^(-1)
y <- rbinom( 100, 1, tigol( 0.3 + 3*x + 5*z + 7*w ) )
# need update
# ROC( form = y ~ x + z, plot="ROC" )
glm1 <- glm(y ~ x + z,family=binomial)
roc1 <- prep.roc(glm1$y,glm1$fitted)
plot(roc1)
```

scorevalid *Performance curves*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
scorevalid(y, score, ...)
## Default S3 method:
scorevalid(y, score, recal = FALSE,
  qth=seq(0, 1, length = 10),tol=1e-04,...)
## S3 method for class 'scorevalid'
print(x,digits = getOption("digits"),...)
## S3 method for class 'scorevalid'
plot(x,mgraph = NULL,...)
```

Arguments

y	~~Describe y here~~
score	~~Describe score here~~
recal	~~Describe recal here~~
x	~~Describe x here~~
tol	1e-04
qth	seq(0, 1, length = 10)
mgraph	graphic organisation
...	~~Describe ... here~~

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

comp1	Description of 'comp1'
comp2	Description of 'comp2'

...

Note

The functions is based on R code of the excellent document proposed by Pierre-André Cornillon (see the section 'references'). for more details, you can consult the following links:
http://www.uhb.fr/sc_sociales/labstats/PAC/doc/score.pdf
http://www.uhb.fr/sc_sociales/labstats/PAC/

References

bardos (2001)
Cornillon P-A (200x) Discrimination et Scores, MASS course - Rennes 2:
http://www.uhb.fr/sc_sociales/labstats/PAC/doc/score.pdf

See Also

roc

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (y, score, ...)
{
    UseMethod("scorevalid")
  }
```

---

sefit *Standard error of prediction (or fitted values) from glm*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
sefit(object, ...)
## S3 method for class 'glm'
sefit(object, newdata, interval = "confidence", dispersion = TRUE,m=1,...)
```

Arguments

object	Object of class inheriting from '"glm"'
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the data values are used.
interval	a logical value ...
dispersion	Type of interval calculation.
m	a numerical value corresponding to observation number
...	further arguments passed to or from other methods

Details

~~ If necessary, more details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Author(s)

~~who you are~~

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as `help`, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (object, ...)
{
    UseMethod("sefit")
  }
```

`SRM` *Structural Risk Minimization*

Description

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
SRM(y, yhat, h, cost = costRMSE, alpha = 0.05)
```

Arguments

`y`	~~Describe `y` here~~
`yhat`	~~Describe `yhat` here~~
`h`	~~Describe `h` here~~
`cost`	~~Describe `cost` here~~
`alpha`	~~Describe `alpha` here~~

Details

+ fonction de cout

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Note

~~further notes~~ + formule Remp <- cost(y, yhat) n <- length(y) Comp <- sqrt((h * (log(2 * n/h) + 1) - log(alpha/4))/n) Remp + Comp

$$SRM = R_{emp} + \sqrt{\frac{(h * (log(2 * n/h) + 1) - log(alpha/4))}{n}}$$

where n = number of elements, h = complexity measure, alpha = . R_{emp} is given by the cost function.

References

Freeman and Hastie
Vapnik
Saporta, G. (2006) Probabilités, analyses des données et statistiques, Second édition, Editions Technip, 622.

See Also

~~objects to See Also as `help`, ~~~

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (y, yhat, h, cost = costRMSE, alpha = 0.05)
{
    Remp <- cost(y, yhat)
    n <- length(y)
    Comp <- sqrt((h * (log(2 * n/h) + 1) - log(alpha/4))/n)
    return(Remp + Comp)
  }
```

training.dataset *building training and test dataset*

Description

building training and test dataset

Usage

```
training.dataset(x, cluster = rep(1, length(x)), ratio = 1/4)
```

Arguments

`x`

`cluster`

`ratio`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (x, cluster = rep(1, length(x)), ratio = 1/4)
{
    test <- unlist(tapply(x, cluster, function(j) sample(j, round(length(j) *
        ratio))))
    res <- data.frame(x = x, test = x %in% test, training = !(x %in%
        test), cluster = cluster)
    attr(res, "ratio") <- ratio
    attr(res, "N") <- length(x)
    attr(res, "Ntest") <- sum(res$test)
    attr(res, "Ntraining") <- sum(res$training)
    return(res)
  }
```

Index