

Package ‘survcomp’

June 5, 2009

Type Package

Title Performance Assessment and Comparison for Survival Analysis

Version 1.1.2

Date 2009

Description R package providing functions to assess and to compare the performance of risk prediction (survival) models.

Author Benjamin Haibe-Kains, Christos Sotiriou, Gianluca Bontempi

Maintainer Benjamin Haibe-Kains <bhaibeka@ulb.ac.be>

Depends survival, ipred, prodlim, survivalROC, SuppDists, bootstrap, R (>= 2.3.0)

Suggests Hmisc, Design

License GPL (>= 3)

URL <http://www.ulb.ac.be/di/mlg>

Repository CRAN

Date/Publication 2009-06-05 13:40:40

R topics documented:

survcomp-package	2
sensor.time	3
cindex.comp	4
cindex.comp.meta	5
combine.est	6
combine.test	8
concordance.index	9
cvpl	11
D.index	12
dindex.comp	13

dindex.comp.meta	14
fisherz	16
getsurv2	17
hazard.ratio	18
hr.comp	20
hr.comp.meta	21
hr.comp2	22
iauc.comp	24
ibsc.comp	25
km.coxph.plot	27
logpl	28
no.at.risk	29
sbrier.score2proba	30
score2proba	32
td.sens.spec	33
tdrocc	34
test.hetero.est	36
test.hetero.test	37

Index 39

survcomp-package *Performance Assessment and Comparison for Survival Analysis*

Description

Functions to perform the performance assessment and comparison of risk prediction (survival) models.

Details

Package: survcomp
 Type: Package
 Version: 1.1.2
 Date: 2009-04-21
 License: GPL-3

Author(s)

Benjamin Haibe-Kains

- Machine Learning Group (MLG), Universite Libre de Bruxelles, Bruxelles, Belgium <http://www.ulb.ac.be/di/mlg/>

- Functional Genomics Unit (FGU), Institut Jules Bordet, Bruxelles, Belgium <http://www.bordet.be/en/services/medical/array/practical.htm>

Maintainer: Benjamin Haibe-Kains <(bhaibeka@ulb.ac.be)>

See Also

survival, Hmisc, Design, prodlim, survivalROC, ipred, bootstrap

censor.time *Function to artificially censor survival data*

Description

The function censors the survival data at a specific point in time. This is useful if you used datasets having different follow-up periods.

Usage

```
censor.time(surv.time, surv.event, time.cens = 0)
```

Arguments

surv.time vector of times to event occurrence
surv.event vector of indicators for event occurrence
time.cens point in time at which the survival data must be censored

Value

surv.time.cens
 vector of censored times to event occurrence
surv.event.cens
 vector of censored indicators for event occurrence

Author(s)

Benjamin Haibe-Kains

Examples

```
set.seed(12345)  
stime <- rexp(30)  
cens <- runif(30, 0.5, 2)  
sevent <- as.numeric(stime <= cens)  
stime <- pmin(stime, cens)  
censor.time(surv.time=stime, surv.event=sevent, time.cens=1)
```

`cindex.comp`*Function to compare two concordance indices*

Description

This function compares two concordance indices computed from the same survival data by using the function `concordance.index`. The statistical test is a Student t test for dependent samples.

Usage

```
cindex.comp(cindex1, cindex2)
```

Arguments

<code>cindex1</code>	first concordance index as returned by the <code>concordance.index</code> function.
<code>cindex2</code>	second concordance index as returned by the <code>concordance.index</code> function.

Details

The two concordance indices must be computed from the same samples (and corresponding survival data). The function uses a Student t test for dependent samples.

Value

<code>p.value</code>	p-value from the Student t test for the comparison <code>cindex1 > cindex2</code> .
<code>cindex1</code>	value of the first concordance index.
<code>cindex2</code>	value of the second concordance index.

Author(s)

Benjamin Haibe-Kains

References

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[concordance.index](#).

Examples

```
set.seed(12345)
age <- rnorm(100, 50, 10)
size <- rexp(100,1)
stime <- rexp(100)
cens <- runif(100,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
c1 <- concordance.index(x=age, surv.time=stime, surv.event=sevent, method="noether")
c2 <- concordance.index(x=size, surv.time=stime, surv.event=sevent, method="noether")
cindex.comp(c1, c2)
```

cindex.comp.meta *Function to compare two concordance indices*

Description

This function compares two lists of concordance indices computed from the same survival data by using the function `concordance.index`. The statistical test is a Student t test for dependent samples.

Usage

```
cindex.comp.meta(list.cindex1, list.cindex2, hetero = FALSE)
```

Arguments

`list.cindex1` first list of concordance indices as returned by the `concordance.index` function.

`list.cindex2` second list of concordance indices as returned by the `concordance.index` function.

`hetero` if TRUE, a random effect model is use to compute the meta-estimators. Otherwise a fixed effect model is used.

Details

In meta-analysis, we estimate the statistic of interest in several independent datasets. It results a list of estimates such as list of concordance indices. The two lists of concordance indices must be computed from the same samples (and corresponding survival data). The function computes a meta-estimator for the correlations between the two scores and uses a Student t test for dependent samples.

Value

`p.value` p-value from the Student t test for the comparison `cindex1 > cindex2`.

`cindex1` meta-estimator of the first concordance index.

`cindex2` meta-estimator of the second concordance index.

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[concordance.index](#).

Examples

```
#first dataset
set.seed(12345)
age <- rnorm(100, 50, 10)
size <- rexp(100,1)
stime <- rexp(100)
cens <- runif(100,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
c1.1 <- concordance.index(x=age, surv.time=stime, surv.event=sevent, method="noether")
c2.1 <- concordance.index(x=size, surv.time=stime, surv.event=sevent, method="noether")
#second dataset
set.seed(54321)
age <- rnorm(110, 53, 10)
size <- rexp(110,1.1)
stime <- rexp(110)
cens <- runif(110,.55,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
c1.2 <- concordance.index(x=age, surv.time=stime, surv.event=sevent, method="noether")
c2.2 <- concordance.index(x=size, surv.time=stime, surv.event=sevent, method="noether")
cindex.comp.meta(list.cindex1=list("cindex.age1"=c1.1, "cindex.age2"=c1.2), list.cindex2=lis
```

combine.est

Function to combine estimates

Description

The function combines several estimators using meta-analytical formula to compute a meta-estimate.

Usage

```
combine.est(x, x.se, hetero = FALSE, na.rm = FALSE)
```

Arguments

x	vector of estimates
x.se	vector of standard errors of the corresponding estimates
hetero	TRUE is the heterogeneity should be taken into account (random effect model), FALSE otherwise (fixed effect model)
na.rm	TRUE if the missing values should be removed from the data, FALSE otherwise

Value

estimate	meta-estimate
se	standard error of the meta-estimate

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.

See Also

test.hetero.est

Examples

```
set.seed(12345)
x1 <- rnorm(100, 50, 10) + rnorm(100, 0, 2)
m1 <- mean(x1)
se1 <- sqrt(var(x1))
x2 <- rnorm(100, 75, 15) + rnorm(100, 0, 5)
m2 <- mean(x2)
se2 <- sqrt(var(x2))

#fixed effect model
combine.est(x=c(m1, m2), x.se=c(se1, se2), hetero=FALSE)
#random effect model
combine.est(x=c(m1, m2), x.se=c(se1, se2), hetero=TRUE)
```

combine.test *Function to combine probabilities*

Description

The function combines several p-value estimated from the same null hypothesis in different studies involving independent data.

Usage

```
combine.test(p, weight, method = c("fisher", "z.transform", "logit"), hetero = FALSE)
```

Arguments

p	vector of p-values
weight	vector of weights (e.g. sample size of each study)
method	fisher for the Fisher's combined probability test, z.transform for the Z-transformed test, logit for the weighted Z-method
hetero	TRUE is the heterogeneity should be taken into account, FALSE otherwise
na.rm	TRUE if the missing values should be removed from the data, FALSE otherwise

Details

The p-values must be one-sided and computed from the same null hypothesis.

Value

p-value

Author(s)

Benjamin Haibe-Kains

References

Whitlock, M. C. (2005) "Combining probability from independent tests: the weighted Z-method is superior to Fisher's approach", *J. Evol. Biol.*, **18**, pages 1368–1373.

See Also

test.hetero.test

Examples

```
p <- c(0.01, 0.13, 0.07, 0.2)
w <- c(100, 50, 200, 30)

#with equal weights
combine.test(p=p, method="z.transform")
#with p-values weighted by the sample size of the studies
combine.test(p=p, weight=w, method="z.transform")
```

concordance.index *Function to compute the concordance index for survival or binary class prediction*

Description

Function to compute the concordance index for a risk prediction, i.e. the probability that, for a pair of randomly chosen comparable samples, the sample with the higher risk prediction will experienced an event before the other sample or belongs to a higher binary class.

Usage

```
concordance.index(x, surv.time, surv.event, cl, weights, strat, alpha = 0.05, outx
```

Arguments

x	a vector of risk predictions.
surv.time	a vector of event times.
surv.event	a vector of event occurrence indicators.
cl	a vector of binary class indicators.
weights	weight of each sample.
strat	stratification indicator.
alpha	alpha level to compute confidence interval.
outx	set to TRUE to not count pairs of observations tied on x as a relevant pair. This results in a Goodman-Kruskal gamma type rank correlation.
method	can take the value conservative, noether or name (see paper Pencina et al. for details).
na.rm	TRUE if missing values should be removed.

Details

Method name is not implemented yet.

Value

<code>c.index</code>	concordance index estimate.
<code>se</code>	standard error of the estimate.
<code>lower</code>	lower bound for the confidence interval.
<code>upper</code>	upper bound for the confidence interval.
<code>p.value</code>	p-value for the statistical test if the estimate is different from 0.5.
<code>n</code>	number of samples used for the estimation.
<code>data</code>	list of data used to compute the index (<code>x</code> , <code>surv.time</code> and <code>surv.event</code> , or <code>cl</code>).

Author(s)

Benjamin Haibe-Kains

References

Harrel Jr, F. E. and Lee, K. L. and Mark, D. B. (1996) "Tutorial in biostatistics: multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing error", *Statistics in Medicine*, **15**, pages 361–387.

Pencina, M. J. and D'Agostino, R. B. (2004) "Overall C as a measure of discrimination in survival analysis: model specific population value and confidence interval estimation", *Statistics in Medicine*, **23**, pages 2109–2123, 2004.

See Also

[rcorr.cens](#)

Examples

```
set.seed(12345)
age <- rnorm(100, 50, 10)
sex <- sample(0:1, 100, replace=TRUE)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
strat <- sample(1:3, 100, replace=TRUE)
cat("survival prediction:\n")
concordance.index(x=age, surv.time=stime, surv.event=sevent, strat=strat, method="noether")
cat("binary class prediction:\n")
concordance.index(x=age, cl=sex, strat=strat, method="noether")
```

`cvpl`*Function to compute the CVPL*

Description

The function computes the cross-validated partial likelihood (CVPL) for the Cox model.

Usage

```
cvpl(x, surv.time, surv.event, strata.cox = NULL, nfold = 1, setseed = NULL, na.rm
```

Arguments

<code>x</code>	data matrix
<code>surv.time</code>	vector of times to event occurrence
<code>surv.event</code>	vector of indicators for event occurrence
<code>strata.cox</code>	stratification variable
<code>nfold</code>	number of folds for the cross-validation
<code>setseed</code>	seed for the random generator
<code>na.rm</code>	TRUE if the missing values should be removed from the data, FALSE otherwise
<code>verbose</code>	verbosity of the function

Details

The function is not fully implemented yet

Value

-1

Author(s)

Benjamin Haibe-Kains

References

~put references to the literature/web site here ~

See Also

[logpl](#), [coxph](#)

D.index

*Function to compute the D index***Description**

Function to compute the D index for a risk prediction, i.e. an estimate of the log hazard ratio comparing two equal-sized prognostic groups. This is a natural measure of separation between two independent survival distributions under the proportional hazards assumption.

Usage

```
D.index(x, surv.time, surv.event, weights, strat, alpha = 0.05, na.rm = FALSE, ...)
```

Arguments

<code>x</code>	a vector of risk predictions.
<code>surv.time</code>	a vector of event times.
<code>surv.event</code>	a vector of event occurrence indicators.
<code>weights</code>	weight of each sample.
<code>strat</code>	stratification indicator.
<code>alpha</code>	alpha level to compute confidence interval.
<code>na.rm</code>	TRUE if missing values should be removed.
<code>...</code>	additional parameters to be passed to the <code>coxph</code> function.

Details

The D index is computed using the Cox model fitted on the scaled rankits of the risk scores instead of the risk scores themselves. The scaled rankits are the expected standard Normal order statistics scaled by $\kappa = \sqrt{8/\pi}$. See (Royston and Sauerbrei, 2004) for details.

Value

<code>d.index</code>	D index estimate.
<code>se</code>	standard error of the estimate.
<code>lower</code>	lower bound for the confidence interval.
<code>upper</code>	upper bound for the confidence interval.
<code>p.value</code>	p-value for the statistical test if the estimate is different from 0.5.
<code>n</code>	number of samples used for the estimation.
<code>coxm</code>	<code>coxph.object</code> fitted on the survival data and <code>z</code> (see below).
<code>data</code>	list of data used to compute the index (<code>x</code> , <code>z</code> , <code>surv.time</code> and <code>surv.event</code>). The item <code>z</code> contains the scaled rankits which are the expected standard Normal order statistics scaled by κ .

Author(s)

Benjamin Haibe-Kains

References

Royston, P. and Sauerbrei, W. (2004) "A new measure of prognostic separation in survival data", *Statistics in Medicine*, **23**, pages 723–748.

See Also

[coxph](#), [coxph.object](#), [normOrder](#)

Examples

```
set.seed(12345)
age <- rnorm(100, 50, 10)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
strat <- sample(1:3, 100, replace=TRUE)
D.index(x=age, surv.time=stime, surv.event=sevent, strat=strat)
```

dindex.comp

Function to compare two D indices

Description

This function compares two D indices from their betas and standard errors as computed by a Cox model for instance. The statistical test is a Student t test for dependent samples. The two D indices must be computed using the [D.index](#) function from the same survival data.

Usage

```
dindex.comp(dindex1, dindex2)
```

Arguments

dindex1	first D index
dindex2	second D index

Details

The two D indices must be computed using the [D.index](#) function from the same samples (and corresponding survival data). The function uses a Student t test for dependent samples.

Value

p.value	p-value from the Wilcoxon rank sum test for the comparison <code>dindex1 > dindex2</code>
dindex1	value of the first D index
dindex2	value of the second D index

Author(s)

Benjamin Haibe-Kains

References

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[D.index](#), [t.test](#)

Examples

```
set.seed(12345)
age <- rnorm(100, 50, 10)
size <- rexp(100, 1)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
d1 <- D.index(x=age, surv.time=stime, surv.event=sevent)
d2 <- D.index(x=size, surv.time=stime, surv.event=sevent)
dindex.comp(d1, d2)
```

dindex.comp.meta *Function to compare two concordance indices*

Description

This function compares two lists of D indices computed from the same survival data by using the function `D.index`. The statistical test is a Student t test for dependent samples.

Usage

```
dindex.comp.meta(list.dindex1, list.dindex2, hetero = FALSE)
```

Arguments

`list.dindex1` first list of D indices as returned by the `D.index` function.
`list.dindex2` second list of D indices as returned by the `D.index` function.
`hetero` if TRUE, a random effect model is used to compute the meta-estimators. Otherwise a fixed effect model is used.

Details

In meta-analysis, we estimate the statistic of interest in several independent datasets. It results a list of estimates such as list of D indices. The two lists of D indices must be computed from the same samples (and corresponding survival data). The function computes a meta-estimator for the correlations between the two scores and uses a Student t test for dependent samples.

Value

`p.value` p-value from the Student t test for the comparison `dindex1 > dindex2`.
`dindex1` meta-estimator of the first D index.
`dindex2` meta-estimator of the second D index.

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.
 Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[concordance.index](#).

Examples

```
#first dataset
set.seed(12345)
age <- rnorm(100, 50, 10)
size <- rexp(100,1)
stime <- rexp(100)
cens <- runif(100,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
d1.1 <- D.index(x=age, surv.time=stime, surv.event=sevent)
d2.1 <- D.index(x=size, surv.time=stime, surv.event=sevent)
#second dataset
set.seed(54321)
```

```

age <- rnorm(110, 53, 10)
size <- rexp(110, 1.1)
stime <- rexp(110)
cens <- runif(110, .55, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
d1.2 <- D.index(x=age, surv.time=stime, surv.event=sevent)
d2.2 <- D.index(x=size, surv.time=stime, surv.event=sevent)
dindex.comp.meta(list.dindex1=list("dindex.age1"=d1.1, "dindex.age2"=d1.2), list.dindex2=lis

```

fisherz

Function to compute Fisher z transformation

Description

The function computes the Fisher z transformation useful to calculate the confidence interval of Pearson's correlation coefficient.

Usage

```
fisherz(x, inv = FALSE, eps = 1e-10)
```

Arguments

x	value, e.g. Pearson's correlation coefficient
inv	TRUE for inverse Fisher z transformation, FALSE otherwise
eps	tolerance for extreme cases, i.e. <i>latex</i>
	when inv = FALSE and <i>latex</i>
	when inv = TRUE

Details

The sampling distribution of Pearson's *latex* is not normally distributed. R. A. Fisher developed a transformation now called "Fisher's z transformation" that converts Pearson's *latex* to the normally distributed variable z. The formula for the transformation is

$$latex$$

Two attributes of the distribution of the z statistic: (1) It is normally distributed and (2) it has a known standard error of

$$latex$$

where *latex* is the number of samples.

Fisher's z is used for computing confidence intervals on Pearson's correlation and for confidence intervals on the difference between correlations.

Value

Fisher's z statistic

Author(s)

Benjamin Haibe-Kains

References

R. A. Fisher (1915) "Frequency distribution of the values of the correlation coefficient in samples of an indefinitely large population". *Biometrika*, **10**, pages 507–521.

See Also

[cor](#)

Examples

```
set.seed(12345)
x1 <- rnorm(100, 50, 10)
x2 <- runif(100, .5, 2)
cc <- cor(x1, x2)
z <- fisherz(x=cc, inv=FALSE)
z.se <- 1 / sqrt(100 - 3)
fisherz(z, inv=TRUE)
```

getsurv2

Function to retrieve the survival probabilities at a specific point in time

Description

The function retrieves the survival probabilities from a `survfit` object, for a specific point in time.

Usage

```
getsurv2(sf, time, which.est = c("point", "lower", "upper"))
```

Arguments

<code>sf</code>	survfit object
<code>time</code>	time at which the survival probabilities must be retrieved
<code>which.est</code>	which estimation to be returned? <code>point</code> for the point estimate, <code>lower</code> for the lower bound and <code>upper</code> for the upper bound

Details

The survival probabilities are estimated through the `survfit` function.

Value

vector of survival probabilities

Author(s)

Benjamin Haibe-Kains

See Also

[survfit](#)

Examples

```
require(survival)
set.seed(12345)
age <- rnorm(30, 50, 10)
stime <- rexp(30)
cens <- runif(30, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
sf <- survfit(Surv(stime, sevent) ~ 1)
getsurv2(sf, time=1)
```

hazard.ratio *Function to compute the D index*

Description

Function to compute the hazard ratio for a risk prediction.

Usage

```
hazard.ratio(x, surv.time, surv.event, weights, strat, alpha = 0.05, na.rm = FALSE,
```

Arguments

x	a vector of risk predictions.
surv.time	a vector of event times.
surv.event	a vector of event occurrence indicators.
weights	weight of each sample.
strat	stratification indicator.
alpha	alpha level to compute confidence interval.
na.rm	TRUE if missing values should be removed.
...	additional parameters to be passed to the coxph function.

Details

The hazard ratio is computed using the Cox model.

Value

hazard.ratio	hazard ratio estimate.
se	standard error of the estimate.
lower	lower bound for the confidence interval.
upper	upper bound for the confidence interval.
p.value	p-value for the statistical test if the estimate is different from 0.5.
n	number of samples used for the estimation.
coxm	coxph.object fitted on the survival data and x (see below).
data	list of data used to compute the hazard ratio (x , <code>surv.time</code> and <code>surv.event</code>).

Author(s)

Benjamin Haibe-Kains

References

Cox, D. R. (1972) "Regression Models and Life Tables", *Journal of the Royal Statistical Society Series B*, **34**, pages 187–220.

See Also

[coxph](#), [coxph.object](#)

Examples

```
set.seed(12345)
age <- rnorm(100, 50, 10)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
strat <- sample(1:3, 100, replace=TRUE)
hazard.ratio(x=age, surv.time=stime, surv.event=sevent, strat=strat)
```

`hr.comp`*Function to statistically compare two hazard ratios*

Description

This function compares two hazard ratios from their betas and standard errors as computed by a Cox model for instance. The statistical test is a Student t test for dependent samples. The two hazard ratios must be computed from the same survival data.

Usage

```
hr.comp(hr1, hr2)
```

Arguments

<code>hr1</code>	first hazard ratio.
<code>hr2</code>	second hazard ratio.

Details

The two hazard ratios must be computed from the same samples (and corresponding survival data). The function uses a Student t test for dependent samples.

Value

<code>p.value</code>	p-value from the Student t test for the comparison $\beta_1 > \beta_2$ (equivalently $hr_1 > hr_2$)
<code>hr1</code>	value of the first hazard ratio
<code>hr2</code>	value of the second hazard ratio

Author(s)

Benjamin Haibe-Kains

References

Student 1908 "The Probable Error of a Mean", *Biometrika*, **6**, 1, pages 1–25.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[coxph](#), [t.test](#)

Examples

```
require(survival)
set.seed(12345)
age <- as.numeric(rnorm(100, 50, 10) >= 50)
size <- as.numeric(rexp(100,1) > 1)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
hr1 <- hazard.ratio(x=age, surv.time=stime, surv.event=sevent)
hr2 <- hazard.ratio(x=size, surv.time=stime, surv.event=sevent)
hr.comp(hr1=hr1, hr2=hr2)
```

hr.comp.meta *Function to compare two concordance indices*

Description

This function compares two lists of hazard ratios computed from the same survival data by using the function `hazard.ratio`. The statistical test is a Student t test for dependent samples.

Usage

```
hr.comp.meta(list.hr1, list.hr2, hetero = FALSE)
```

Arguments

<code>list.hr1</code>	first list of D indices as returned by the <code>hazard.ratio</code> function.
<code>list.hr2</code>	second list of D indices as returned by the <code>hazard.ratio</code> function.
<code>hetero</code>	if TRUE, a random effect model is use to compute the meta-estimators. Otherwise a fixed effect model is used.

Details

In meta-analysis, we estimate the statistic of interest in several independent datasets. It results a list of estimates such as list of hazard ratios. The two lists of hazrd ratios must be computed from the same samples (and corresponding survival data). The function computes a meta-estimator for the correlations between the two scores and uses a Student t test for dependent samples.

Value

<code>p.value</code>	p-value from the Student t test for the comparison $hr1 > hr2$.
<code>hr1</code>	meta-estimator of the first D index.
<code>hr2</code>	meta-estimator of the second D index.

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[concordance.index](#).

Examples

```
#first dataset
set.seed(12345)
age <- rnorm(100, 50, 10)
size <- rexp(100,1)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
h1.1 <- hazard.ratio(x=age, surv.time=stime, surv.event=sevent)
h2.1 <- hazard.ratio(x=size, surv.time=stime, surv.event=sevent)
#second dataset
set.seed(54321)
age <- rnorm(110, 53, 10)
size <- rexp(110,1.1)
stime <- rexp(110)
cens <- runif(110, .55, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
h1.2 <- hazard.ratio(x=age, surv.time=stime, surv.event=sevent)
h2.2 <- hazard.ratio(x=size, surv.time=stime, surv.event=sevent)
hr.comp.meta(list.hr1=list("hr.age1"=h1.1, "hr.age2"=h1.2), list.hr2=list("hr.size1"=h2.1, "
```

hr.comp2

Function to statistically compare two hazard ratios (alternative interface)

Description

This function compares two hazard ratios from their betas and standard errors as computed by a Cox model for instance. The statistical test is a Student t test for dependent samples. The two hazard ratios must be computed from the same survival data.

Usage

```
hr.comp2(x1, beta1, se1, x2, beta2, se2, n)
```

Arguments

x1	risk score used to estimate the first hazard ratio.
beta1	beta estimate for the first hazard ratio.
se1	standard error of beta estimate for the first hazard ratio.
x2	risk score used to estimate the second hazard ratio.
beta2	beta estimate for the second hazard ratio.
se2	standard error of beta estimate for the first hazard ratio.
n	number of samples from which the hazard ratios were estimated.

Details

The two hazard ratios must be computed from the same samples (and corresponding survival data). The function uses a Student t test for dependent samples.

Value

p.value	p-value from the Student t test for the comparison $\beta_1 > \beta_2$ (equivalently $hr_1 > hr_2$)
hr1	value of the first hazard ratio
hr2	value of the second hazard ratio

Author(s)

Benjamin Haibe-Kains

References

Student 1908) "The Probable Error of a Mean", *Biometrika*, **6**, 1, pages 1–25.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[coxph](#), [t.test](#)

Examples

```
require(survival)
set.seed(12345)
age <- as.numeric(rnorm(100, 50, 10) >= 50)
size <- as.numeric(rexp(100,1) > 1)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
coxml <- coxph(Surv(stime, sevent) ~ age)
cox2 <- coxph(Surv(stime, sevent) ~ size)
hr.comp2(x1=age, beta1=coxml$coefficients, se1=drop(sqrt(coxml$var)), x2=size, beta2=cox2$
```

`iauc.comp`*Function to compare two IAUCs through time-dependent ROC curves*

Description

This function compares two integrated areas under the curves (IAUC) through the results of time-dependent ROC curves at some points in time. The statistical test is a Wilcoxon rank sum test for dependent samples.

Usage

```
iauc.comp(auc1, auc2, time)
```

Arguments

<code>auc1</code>	vector of AUCs computed from the first time-dependent ROC curves for some points in time
<code>auc2</code>	vector of AUCs computed from the second time-dependent ROC curves for some points in time
<code>time</code>	vector of points in time for which the AUCs are computed

Details

The two vectors of AUCs must be computed from the same samples (and corresponding survival data) and for the same points in time. The function uses a Wilcoxon rank sum test for dependent samples.

Value

<code>p.value</code>	p-value from the Wilcoxon rank sum test for the comparison <code>iauc1 > iauc2</code>
<code>iauc1</code>	value of the IAUC for the first set of time-depdent ROC curves
<code>iauc2</code>	value of the IAUC for the second set of time-depdent ROC curves

Author(s)

Benjamin Haibe-Kains

References

Wilcoxon, F. (1945) "Individual comparisons by ranking methods", *Biometrics Bulletin*, **1**, pages 80–83.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also[tdrocc, wilcox.test](#)**Examples**

```

set.seed(12345)
age <- rnorm(30, 50, 10)
size <- rexp(30,1)
stime <- rexp(30)
cens <- runif(30,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
##time-dependent ROC curves
tt <- unique(sort(stime[sevent == 1]))
##size
mytdroc1 <- NULL
for(i in 1:length(tt)) {
  rr <- tdrocc(x=size, surv.time=stime, surv.event=sevent, time=tt[i], na.rm=TRUE, verb=0)
  mytdroc1 <- c(mytdroc1, list(rr))
}
auc1 <- unlist(lapply(mytdroc1, function(x) { return(x$AUC) }))
##age
mytdroc2 <- NULL
for(i in 1:length(tt)) {
  rr <- tdrocc(x=age, surv.time=stime, surv.event=sevent, time=tt[i], na.rm=TRUE, verb=0)
  mytdroc2 <- c(mytdroc2, list(rr))
}
auc2 <- unlist(lapply(mytdroc2, function(x) { return(x$AUC) }))
iauc.comp(auc1=auc1, auc2=auc2, time=tt)

```

ibsc.comp

*Function to compare two IBSCs***Description**

This function compares two integrated Briers scores (IBSC) through the estimation of the Brier scores (BSC) at some points in time. The statistical test is a Wilcoxon rank sum test for dependent samples.

Usage

```
ibsc.comp(bsc1, bsc2, time)
```

Arguments

bsc1	vector of BSCs computed from the first predicted probabilities for some points in time
bsc2	vector of BSCs computed from the second predicted probabilities for some points in time
time	vector of points in time for which the BSCs are computed

Details

The two vectors of BSCs must be computed from the same samples (and corresponding survival data) and for the same points in time. The function uses a Wilcoxon rank sum test for dependent samples.

Value

p.value	p-value from the Wilcoxon rank sum test for the comparison <code>ibsc1 < ibsc2</code>
ibsc1	value of the IBSC for the first set of BSCs
ibsc2	value of the IBSC for the second set of BSCs

Author(s)

Benjamin Haibe-Kains

References

Wilcoxon, F. (1945) "Individual comparisons by ranking methods", *Biometrics Bulletin*, **1**, pages 80–83.

Haibe-Kains, B. and Desmedt, C. and Sotiriou, C. and Bontempi, G. (2008) "A comparative study of survival models for breast cancer prognostication based on microarray data: does a single gene beat them all?", *Bioinformatics*, **24**, 19, pages 2200–2208.

See Also

[sbrier.score2proba](#), [sbrier](#)

Examples

```
set.seed(12345)
age <- rnorm(30, 50, 10)
size <- rexp(30,1)
stime <- rexp(30)
cens <- runif(30,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
##Brier scores
##size
dd <- data.frame("time"=stime, "event"=sevent, "score"=size)
bsc1 <- sbrier.score2proba(data.tr=dd, data.ts=dd, method="cox")
##size
dd <- data.frame("time"=stime, "event"=sevent, "score"=age)
bsc2 <- sbrier.score2proba(data.tr=dd, data.ts=dd, method="cox")
if(!all(bsc1$time == bsc2$time)) { stop("the two vector of BSCs must be computed for the same time")
ibsc.comp(bsc1=bsc1$bsc, bsc2=bsc2$bsc, time=bsc1$time)
```

km.coxph.plot *Function to plot several Kaplan-Meier survival curves*

Description

Function to plot several Kaplan-Meier survival curves with number of individuals at risk at some time points.

Usage

```
km.coxph.plot(formula.s, data.s, sub.s = "all", x.label, y.label, main.title, sub.t
```

Arguments

formula.s	formula composed of a <code>Surv</code> object and a strata variable (i.e. stratification).
data.s	data frame composed of the variables used in the formula.
sub.s	vector of booleans specifying if only a subset of the data should be considered. Default value is "all".
x.label	label for the y-axis.
y.label	label for the x-axis.
main.title	main title at the top of the plot.
sub.title	subtitle at the bottom of the plot.
leg.text	text in the legend.
leg.pos	the location may also be specified by setting 'x' to a single keyword from the list "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". This places the legend on the inside of the plot frame at the given location.
leg.inset	inset distance from the margins as a fraction of the plot region. Default value is 0.05.
o.text	plot the logrank p-value.
v.line	x coordinate(s) for vertical line(s).
h.line	y coordinate(s) for horizontal line(s).
.col	vector of colors for the different survival curves.
.lty	vector of line types for the different survival curves
.lwd	vector of line widths for the different survival curves.
show.n.risk	if TRUE, show the numbers of samples at risk for each time step.
n.risk.step	vector specifying the time to be the steps for displaying the number of individuals at risk.
n.risk.cex	size of the number of individuals at risk. Default value is 0.85.
verbose	verbosity level (TRUE or FALSE). Default value is TRUE.
...	additional parameters to be passed to the <code>plot</code> function.

Details

The original version of this function was kindly provided by Dr Christos Hatzis (January, 17th 2006).

Author(s)

Christos Hatzis, Benjamin Haibe-Kains

See Also

[survfit](#), [coxph](#)

Examples

```
require(survival)
set.seed(12345)
stime <- rexp(100) * 10
cens   <- runif(100,.5,2) * 10
sevent <- as.numeric(stime <= cens)
stime  <- pmin(stime, cens)
strat  <- sample(1:3, 100, replace=TRUE)
dd <- data.frame("surv.time"=stime, "surv.event"=sevent, "strat"=strat)

km.coxph.plot(formula.s=Surv(surv.time, surv.event) ~ strat, data.s=dd, sub.s="all", x.label=)
```

logpl

Function to compute the log partial likelihood of a Cox model

Description

The function computes the log partial likelihood of a set of coefficients given some survival data.

Usage

```
logpl(surv.time, surv.event, beta, x, strata.cox = NULL, na.rm = FALSE, verbose = F)
```

Arguments

<code>surv.time</code>	vector of times to event occurrence
<code>surv.event</code>	vector of indicators for event occurrence
<code>beta</code>	vector of coefficients fitted by a Cox model for instance
<code>x</code>	data matrix
<code>strata.cox</code>	stratification variable
<code>na.rm</code>	TRUE if the missing values should be removed from the data, FALSE otherwise
<code>verbose</code>	verbosity of the function

Value

vector of two elements: `logpl` and `event` for the estimation of the log partial likelihood and the number of events, respectively

Author(s)

Benjamin Haibe-Kains

References

Cox, D. R. (1972) "Regression Models and Life Tables", *Journal of the Royal Statistical Society Series B*, **34**, pages 187–220.

See Also

[coxph](#), [cvpl](#)

Examples

```
require(survival)
set.seed(12345)
age <- rnorm(100, 50, 10)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
##Cox model
coxmodel <- coxph(Surv(stime, sevent) ~ age)
##log partial likelihood of the null model
logpl(surv.time=stime, surv.event=sevent, beta=0, x=age)
##log partial likelihood of the Cox model
logpl(surv.time=stime, surv.event=sevent, beta=coxmodel$coefficients, x=age)
##equivalent to
coxmodel$loglik
```

no.at.risk

Function to compute the number of individuals at risk

Description

Function to compute the number of individuals at risk at certain time points, as used in the Kaplan-Meier estimator for instance, depending on stratification.

Usage

```
no.at.risk(formula.s, data.s, sub.s = "all", t.step, t.end)
```

Arguments

<code>formula.s</code>	formula composed of a <code>Surv</code> object and a strata variable (i.e. stratification).
<code>data.s</code>	data frame composed of the variables used in the formula.
<code>sub.s</code>	vector of booleans specifying if only a subset of the data should be considered.
<code>t.step</code>	time step at which the number of individuals at risk is computed.
<code>t.end</code>	maximum time to be considered.

Details

The original version of this function was kindly provided by Dr Christos Hatzis (January, 17th 2006).

Value

number of individuals at risk at each time step specified in `t.step` up to `t.end`.

Author(s)

Christos Hatzis, Benjamin Haibe-Kains

See Also

[survfit, km.coxph.plot](#)

Examples

```
require(survival)
set.seed(12345)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
strat <- sample(1:3, 100, replace=TRUE)
dd <- data.frame("surv.time"=stime, "surv.event"=sevent, "strat"=strat)
no.at.risk(formula.s=Surv(surv.time,surv.event) ~ strat, data.s=dd, sub.s="all", t.step=0.05)
```

`sbrier.score2proba` *Function to compute the BSCs from a risk score, for all the times of event occurrence*

Description

The function computes all the Brier scores (BSC) and the corresponding integrated Briser score (IBSC) from a risk score, for all the times of event occurrence. The risk score is first transformed in survival probabilities using either a Cox model or the product-limit estimator.

Usage

```
sbrier.score2proba(data.tr, data.ts, method = c("cox", "prodlim"))
```

Arguments

<code>data.tr</code>	the data frame for the training set. This data frame must contain three columns for the times, the event occurrence and the risk score. These columns are called "time", "event" and "score" respectively.
<code>data.ts</code>	the data frame for the test set. This data frame must contain three columns for the times, the event occurrence and the risk score. These columns are called "time", "event" and "score" respectively.
<code>method</code>	method for survival probabilities estimation using either a Cox model or the product-limit estimator

Value

<code>time</code>	vector of points in time
<code>bsc</code>	vector of Brier scores (BSC) at ome points in time
<code>bsc.integrated</code>	value of the integrated Brier score (IBSC)

Author(s)

Benjamin Haibe-Kains

References

Brier, G. W. (1950) "Verification of forecasts expressed in terms of probabilities", *Monthly Weather Review*, **78**, pages 1–3.

Graf, E. and Schmoor, C. and Sauerbrei, W. and Schumacher, M. (1999) "Assessment and comparison of prognostic classification schemes for survival data ", *Statistics in Medicine*, **18**, pages 2529–2545.

Cox, D. R. (1972) "Regression Models and Life Tables", *Journal of the Royal Statistical Society Series B*, **34**, pages 187–220.

Andersen, P. K. and Borgan, O. and Gill, R. D. and Keiding, N. (1993) "Statistical Models Based on Counting Processes", *Springer*.

See Also

[sbrier](#), [cox](#), [prodlim](#)

Examples

```
set.seed(12345)
age <- rnorm(30, 50, 10)
stime <- rexp(30)
cens <- runif(30, .5, 2)
sevent <- as.numeric(stime <= cens)
```

```

stime <- pmin(stime, cens)
dd <- data.frame("time"=stime, "event"=sevent, "score"=age)

#Cox's model
sbrier.score2proba(data.tr=dd, data.ts=dd, method="cox")
#product-limit estimator
sbrier.score2proba(data.tr=dd, data.ts=dd, method="prodlim")

```

score2proba

Function to compute the survival probabilities from a risk score

Description

the function uses either a Cox model or the product-limit estimator to compute the survival probabilities from a risk score for a specific point in time.

Usage

```
score2proba(data.tr, score, yr, method = c("cox", "prodlim"), conf.int = 0.95, which
```

Arguments

<code>data.tr</code>	the data frame for the training set. This data frame must contain three columns for the times, the event occurrence and the risk score. These columns are called "time", "event" and "score" respectively
<code>score</code>	risk score for the test set
<code>yr</code>	a point in time for which the survival probabilities must be computed
<code>method</code>	method for survival probabilities estimation, either <code>cox</code> or <code>prodlim</code> for the Cox model or the product-limit estimator, respectively
<code>conf.int</code>	value in [0,1]. Default at 0.95
<code>which.est</code>	which estimation to be returned? <code>point</code> for the point estimate, <code>lower</code> for the lower bound and <code>upper</code> for the upper bound

Value

vector of predicted survival probabilities

Author(s)

Benjamin Haibe-Kains

References

Cox, D. R. (1972) "Regression Models and Life Tables", *Journal of the Royal Statistical Society Series B*, **34**, pages 187–220.

Andersen, P. K. and Borgan, O. and Gill, R. D. and Keiding, N. (1993) "Statistical Models Based on Counting Processes", *Springer*.

See Also[cox](#), [prodlim](#)**Examples**

```

set.seed(12345)
age <- rnorm(30, 50, 10)
stime <- rexp(30)
cens <- runif(30, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
dd <- data.frame("time"=stime, "event"=sevent, "score"=age)

#Cox's model
score2proba(data.tr=dd, score=dd$score, yr=1, method="cox")
#product-limit estimator
score2proba(data.tr=dd, score=dd$score, yr=1, method="prodlim")

```

td.sens.spec	<i>Function to compute sensitivity and specificity for a binary classification of survival data</i>
--------------	---

Description

The function is a wrapper for the [survivalROC.C](#) function in order to compute sensitivity and specificity for a binary classification of survival data.

Usage

```
td.sens.spec(cl, surv.time, surv.event, time, span = 0, sampling = FALSE, na.rm = F
```

Arguments

cl	vector of binary classes.
surv.time	vector of times to event occurrence.
surv.event	vector of event occurrence indicators.
time	time point for sensitivity and specificity estimations.
span	Span for the NNE. Default value is 0.
sampling	jackknife procedure to estimate the standard error of sensitivity and specificity estimations.
na.rm	TRUE if the missing values should be removed from the data, FALSE otherwise.
...	additional arguments to be passed to the survivalROC function.

Details

Only NNE method is used to estimate sensitivity and specificity (see [survivalROC.C](#)). The standard error for sensitivity and specificity is estimated through jackknife procedure (see [jackknife](#)).

Value

<code>sens</code>	sensitivity estimate
<code>sens.se</code>	standard error for sensitivity estimate
<code>spec</code>	specificity estimate
<code>spec.se</code>	standard error for specificity estimate

Author(s)

Benjamin Haibe-Kains

References

Heagerty, P. J. and Lumley, T. L. and Pepe, M. S. (2000) "Time-Dependent ROC Curves for Censored Survival Data and a Diagnostic Marker", *Biometrics*, **56**, pages 337–344.

Efron, B. and Tibshirani, R. (1986). "The Bootstrap Method for standard errors, confidence intervals, and other measures of statistical accuracy", *Statistical Science*, **1** (1), pages 1–35.

See Also

[survivalROC](#)

Examples

```
set.seed(12345)
gender <- sample(c(0,1), 100, replace=TRUE)
stime <- rexp(100)
cens <- runif(100,.5,2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
mysenspec <- td.sens.spec(cl=gender, surv.time=stime, surv.event=sevent, time=1, span=0, na.rm=TRUE)
```

tdrocc

Function to compute time-dependent ROC curves

Description

The function is a wrapper for the [survivalROC](#) function in order to compute the time-dependent ROC curves.

Usage

```
tdrocc(x, surv.time, surv.event, surv.entry = NULL, time, cutpts = NA, na.rm = FALSE)
```

Arguments

<code>x</code>	vector of risk scores.
<code>surv.time</code>	vector of times to event occurrence.
<code>surv.event</code>	vector of event occurrence indicators.
<code>surv.entry</code>	entry time for the subjects.
<code>time</code>	time point for the ROC curve.
<code>cutpts</code>	cut points for the risk score.
<code>na.rm</code>	TRUE if the missing values should be removed from the data, FALSE otherwise.
<code>verbose</code>	verbosity of the function.
<code>span</code>	Span for the NNE, need either lambda or span for NNE.
<code>lambda</code>	smoothing parameter for NNE.
<code>...</code>	additional arguments to be passed to the survivalROC function.

Value

<code>spec</code>	specificity estimates
<code>sens</code>	sensitivity estimates
<code>rule</code>	rule to compute the predictions at each cutoff
<code>cuts</code>	cutoffs
<code>time</code>	time point at which the time-dependent ROC is computed
<code>survival</code>	overall survival at the time point
<code>AUC</code>	Area Under the Curve (AUC) of the time-dependent ROC curve
<code>data</code>	survival data and risk score used to compute the time-dependent ROC curve

Author(s)

Benjamin Haibe-Kains

References

Heagerty, P. J. and Lumley, T. L. and Pepe, M. S. (2000) "Time-Dependent ROC Curves for Censored Survival Data and a Diagnostic Marker", *Biometrics*, **56**, pages 337–344.

See Also

[survivalROC](#)

Examples

```

set.seed(12345)
age <- rnorm(100, 50, 10)
stime <- rexp(100)
cens <- runif(100, .5, 2)
sevent <- as.numeric(stime <= cens)
stime <- pmin(stime, cens)
tdroc <- tdrocc(x=age, surv.time=stime, surv.event=sevent, time=1, na.rm=TRUE, verbose=FALSE)
##plot the time-dependent ROC curve
plot(1-tdroc$spec, tdroc$sens, type="l", xlab="1 - specificity", ylab="sensitivity")
lines(x=c(0,1), y=c(0,1), lty=3, col="red")

```

test.hetero.est *Function to test the heterogeneity of set of probabilities*

Description

The function tests whether a set of p-values are heterogeneous.

Usage

```
test.hetero.est(x, x.se, na.rm = FALSE)
```

Arguments

x	vector of estimates
x.se	vector of standard errors of the corresponding estimates
na.rm	TRUE if the missing values should be removed from the data, FALSE otherwise

Details

The heterogeneity test is known to be very conservative. Consider a p-value < 0.1 as significant.

Value

Q	Q statistic
p.value	p-value of the heterogeneity test

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.

See Also`combine.test`**Examples**

```
set.seed(12345)
x1 <- rnorm(100, 50, 10) + rnorm(100, 0, 2)
m1 <- mean(x1)
se1 <- sqrt(var(x1))
x2 <- rnorm(100, 75, 15) + rnorm(100, 0, 5)
m2 <- mean(x2)
se2 <- sqrt(var(x2))

test.hetero.est(x=c(m1, m2), x.se=c(se1, se2))
```

`test.hetero.test` *Function to test the heterogeneity of set of probabilities*

Description

The function tests whether a set of p-values are heterogeneous.

Usage

```
test.hetero.test(p, weight, na.rm = FALSE)
```

Arguments

<code>p</code>	vector of p-values
<code>weight</code>	vector of weights (e.g. sample size of each study)
<code>na.rm</code>	TRUE if the missing values should be removed from the data, FALSE otherwise

Details

The p-values should be one-sided and computed from the same null hypothesis.

Value

<code>Q</code>	Q statistic
<code>p.value</code>	p-value of the heterogeneity test

Author(s)

Benjamin Haibe-Kains

References

Cochrane, W. G. (1954) "The combination of estimates from different experiments", *Biometrics*, **10**, pages 101–129.

Whitlock, M. C. (2005) "Combining probability from independent tests: the weighted Z-method is superior to Fisher's approach", *J. Evol. Biol.*, **18**, pages 1368–1373.

See Also

combine.test

Examples

```
p <- c(0.01, 0.13, 0.07, 0.2)
w <- c(100, 50, 200, 30)

#with equal weights
test.hetero.test(p=p)
#with p-values weighted by the sample size of the studies
test.hetero.test(p=p, weight=w)
```

Index

*Topic **htest**

- cindex.comp, 3
- cindex.comp.meta, 4
- dindex.comp, 12
- dindex.comp.meta, 13
- hr.comp, 19
- hr.comp.meta, 20
- hr.comp2, 21
- iauc.comp, 23
- ibsc.comp, 24
- survcomp-package, 1
- test.hetero.est, 35
- test.hetero.test, 36

*Topic **survival**

- cancel.time, 2
- cindex.comp, 3
- cindex.comp.meta, 4
- concordance.index, 8
- cvpl, 10
- D.index, 11
- dindex.comp, 12
- dindex.comp.meta, 13
- getsurv2, 16
- hazard.ratio, 17
- hr.comp, 19
- hr.comp.meta, 20
- hr.comp2, 21
- iauc.comp, 23
- ibsc.comp, 24
- km.coxph.plot, 26
- logpl, 27
- no.at.risk, 28
- sbrier.score2proba, 29
- score2proba, 31
- survcomp-package, 1
- td.sens.spec, 32
- tdrocc, 33

*Topic **univar**

- combine.est, 6

- combine.test, 7
- concordance.index, 8
- D.index, 11
- fisherz, 15
- hazard.ratio, 17
- km.coxph.plot, 26
- no.at.risk, 28
- sbrier.score2proba, 29

- cancel.time, 2
- cindex.comp, 3
- cindex.comp.meta, 4
- combine.est, 6
- combine.test, 7
- concordance.index, 4, 5, 8, 14, 21
- cor, 16
- cox, 30, 32
- coxph, 10–12, 17–19, 22, 27, 28
- coxph.object, 11, 12, 18
- cvpl, 10, 28

- D.index, 11, 12, 13
- dindex.comp, 12
- dindex.comp.meta, 13

- fisherz, 15

- getsurv2, 16

- hazard.ratio, 17
- hr.comp, 19
- hr.comp.meta, 20
- hr.comp2, 21

- iauc.comp, 23
- ibsc.comp, 24

- jackknife, 32

- km.coxph.plot, 26, 29

- logpl, 10, 27

no.at.risk, 28
normOrder, 12

plot, 26
prodlim, 30, 32

rcorr.cens, 9

sbrier, 25, 30
sbrier.score2proba, 25, 29
score2proba, 31
survcomp (*survcomp-package*), 1
survcomp-package, 1
survfit, 16, 17, 27, 29
survivalROC, 32–34
survivalROC.C, 32

t.test, 13, 19, 22
td.sens.spec, 32
tdrocc, 24, 33
test.hetero.est, 35
test.hetero.test, 36

wilcox.test, 24