Package ‘survcomp’

February 1, 2025

Type Package

Title Performance Assessment and Comparison for Survival Analysis

Version 1.57.0

Date 2021-12-14

Description Assessment and Comparison for Performance of Risk Prediction (Survival) Models.

biocViews GeneExpression, DifferentialExpression, Visualization

Depends survival, prodlim, R (>= 3.4)

Suggests Hmisc, clinfun, xtable, Biobase, BiocManager

Imports ipred, SuppDists, KernSmooth, survivalROC, bootstrap, grid, rmeta, stats, graphics

License Artistic-2.0

URL `http://www.pmgenomics.ca/bhklab/`

git_url https://git.bioconductor.org/packages/survcomp

git_branch devel

git_last_commit d938b84

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-01-31

Author Benjamin Haibe-Kains [aut, cre],
Markus Schroeder [aut],
Catharina Olsen [aut],
Christos Sotiriou [aut],
Gianluca Bontempi [aut],
John Quackenbush [aut],
Samuel Branders [aut],
Zhaleh Safikhani [aut]

Maintainer Benjamin Haibe-Kains `<benjamin.haibe.kains@utoronto.ca>`

Contents

`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:	1.57.0
Date:	2021-12-14
License:	Artistic-2.0

Author(s)

Benjamin Haibe-Kains

- Bioinformatics and Computational Genomics Laboratory, Princess Margaret Cancer Center, University Health Network, Toronto, Ontario, Canada

http://www.pmgenomics.ca/bhklab/

Former labs:

- Computational Biology and Functional Genomics Laboratory, Dana-Farber Cancer Institute, Boston, MA, USA

http://compbio.dfci.harvard.edu/

- Center for Cancer Computational Biology, Dana-Farber Cancer Institute, Boston, MA, USA

http://cccb.dfci.harvard.edu/index.html

- Machine Learning Group (MLG), Universite Libre de Bruxelles, Bruxelles, Belgium

http://www.ulb.ac.be/di/mlg/

- Breast Cancer Translational Research Laboratory (BCTRL), Institut Jules Bordet, Bruxelles, Belgium

http://www.bordet.be/en/services/medical/array/practical.htm

Markus Schroeder

- UCD School of Biomolecular and Biomedical Science, Conway Institute, University College Dublin, Belfield, Dublin, Ireland

http://bioinfo-casl.ucd.ie/cib/people/students/markusschroder

Maintainer:

Benjamin Haibe-Kains

`<benjamin.haibe.kains@utoronto.ca>`

Markus Schroeder

`<markus.schroeder@ucdconnect.ie>`

See Also

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

`balanced.hazard.ratio` *Function to estimate the balanced hazard ratio through Cox regression*

Description

Function to compute the balanced hazard ratio for a risk group prediction.

Usage

```
balanced.hazard.ratio(x, surv.time, surv.event, alpha = 0.05,
method.test = c("logrank", "likelihood.ratio", "wald"),
ties = c("efron", "breslow", "exact"), weights, strat, ...)
```

Arguments

`x`	a vector of risk group predictions.
`surv.time`	a vector of event times.
`surv.event`	a vector of event occurrence indicators.
`alpha`	alpha level to compute confidence interval.
`method.test`	...
`ties`	...
`weights`	...
`strat`	...
`...`	additional parameters to be passed to the `hazard.ratio` and the `coxph` functions.

Details

The balanced hazard ratio is computed using the Cox model.

Value

`balanced.hazard.ratio`	balanced hazard ratio estimate.
`coef`	coefficient (beta) estimated in the cox regression model.
`se`	standard error of the coefficient (beta) estimate.
`lower`	lower bound for the confidence interval.
`upper`	upper bound for the confidence interval.
`p.value`	p-value computed using the score (logrank) test whether the balanced hazard ratio is different from 1.
`n`	number of samples used for the estimation.
`coxm`	`coxph.object` fitted on the survival data and `x`.
`data`	list of data used to compute the balanced hazard ratio (`x`, `surv.time` and `surv.event`).

Author(s)

Samuel Branders

References

Branders, S. and Dupont, P. (2015) "A balanced hazard ratio for risk group evaluation from survival data", *Statistics in Medicine*, **34**(17), pages 2528–2543.

See Also

`hazard.ratio`, `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)
weight <- runif(100, min=0, max=1)
balanced.hazard.ratio(x=age, surv.time=stime, surv.event=sevent, weights=weight,
  strat=strat)
```

`bhr.comp` *Function to statistically compare two balanced hazard ratios*

Description

This function compares two balanced 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 balanced hazard ratios must be computed from the same survival data.

Usage

```
bhr.comp(bhr1, bhr2)
```

Arguments

`bhr1`	first balanced hazard ratio.
`bhr2`	second balanced hazard ratio.

Details

The two balanced 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 beta1 > beta2 (equivalently bhr1 > bhr2)
`bhr1`	value of the first balanced hazard ratio
`bhr2`	value of the second balanced hazard ratio

Author(s)

Samuel Branders, 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.

Branders, S. and Dupont, P. (2015) "A balanced hazard ratio for risk group evaluation from survival data", *Statistics in Medicine*, **34**(17), pages 2528–2543.

See Also

`balanced.hazard.ratio`, `coxph`, `t.test`

Examples

```
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)
bhr1 <- balanced.hazard.ratio(x=age, surv.time=stime, surv.event=sevent)
bhr2 <- balanced.hazard.ratio(x=size, surv.time=stime, surv.event=sevent)
bhr.comp(bhr1=bhr1, bhr2=bhr2)
```

`breastCancerData` *Sample data containing six datasets for gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from 6 different datasets (see section details). The subsets contain the seven gene signature introduced by Desmedt et al. 2008.

Usage

```
data(breastCancerData)
```

Format

Six ExpressionSets. Example for 'mainz7g': eSet with 7 features and 200 samples, containing:

- `exprs(mainz7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a technology (single-channel, oligonucleotides).
- `fData(mainz7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a.
- `pData(mainz7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(mainz7g)`: MIAME object containing information about the dataset.
- `annotation(mainz7g)`: Name of the affy chip.

Details

This dataset represents subsets of the studies published by Schmidt et al. 2008 [mainz7g], Wang. et al. 2005 and Minn et al. 2007 [vdx7g], Miller et al. 2005 [upp7g], Sotiriou et al. 2006 [unt7g], Desmedt et al. 2007 and TRANSBIG [transbig7g], van't Veer et al. 2002 and van de Vijver et al. 2002 [nki7g]. Each subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as uPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

mainz: `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE11121`

transbig: `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7390`

upp: `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE3494`

unt: `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2990`

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE6532`

vdx: `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2034`

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE5327`

nki: `http://www.rii.com/publications/2002/vantveer.html`

References

Marcus Schmidt, Daniel Boehm, Christian von Toerne, Eric Steiner, Alexander Puhl, Heryk Pilch, Hans-Anton Lehr, Jan G. Hengstler, Hainz Koelbl and Mathias Gehrmann (2008) "The Humoral Immune System Has a Key Prognostic Impact in Node-Negative Breast Cancer", *Cancer Research*, **68**(13):5405-5413

Christine Desmedt, Fanny Piette, Sherene Loi, Yixin Wang, Francoise Lallemand, Benjamin Haibe-Kains, Giuseppe Viale, Mauro Delorenzi, Yi Zhang, Mahasti Saghatchian d Assignies, Jonas Bergh, Rosette Lidereau, Paul Ellis, Adrian L. Harris, Jan G. M. Klijn, John A. Foekens, Fatima Cardoso, Martine J. Piccart, Marc Buyse and Christos Sotiriou (2007) "Strong Time Dependence of the 76-Gene Prognostic Signature for Node-Negative Breast Cancer Patients in the TRANSBIG Multicenter Independent Validation Series", *Clinical Cancer Research*, **13**(11):3207-3214

Lance D. Miller, Johanna Smeds, Joshy George, Vinsensius B. Vega, Liza Vergara, Alexander Ploner, Yudi Pawitan, Per Hall, Sigrid Klaar, Edison T. Liu and Jonas Bergh (2005) "An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival" *Proceedings of the National Academy of Sciences of the United States of America* **102**(38):13550-13555

Christos Sotiriou, Pratyaksha Wirapati, Sherene Loi, Adrian Harris, Steve Fox, Johanna Smeds, Hans Nordgren, Pierre Farmer, Viviane Praz, Benjamin Haibe-Kains, Christine Desmedt, Denis Larsimont, Fatima Cardoso, Hans Peterse, Dimitry Nuyten, Marc Buyse, Marc J. Van de Vijver, Jonas Bergh, Martine Piccart and Mauro Delorenzi (2006) "Gene Expression Profiling in Breast Cancer: Understanding the Molecular Basis of Histologic Grade To Improve Prognosis", *Journal of the National Cancer Institute*, **98**(4):262-272

Y. Wang, J. G. Klijn, Y. Zhang, A. M. Sieuwerts, M. P. Look, F. Yang, D. Talantov, M. Timmermans, M. E. Meijer-van Gelder, J. Yu, T. Jatkoe, E. M. Berns, D. Atkins and J. A. Foekens (2005) "Gene-Expression Profiles to Predict Distant Metastasis of Lymph-Node-Negative Primary Breast Cancer", *Lancet*, **365**:671-679

Andy J. Minn, Gaorav P. Gupta, David Padua, Paula Bos, Don X. Nguyen, Dimitry Nuyten, Bas Kreike, Yi Zhang, Yixin Wang, Hemant Ishwaran, John A. Foekens, Marc van de Vijver and Joan Massague (2007) "Lung metastasis genes couple breast tumor size and metastatic spread", *Proceedings of the National Academy of Sciences*, **104**(16):6740-6745

Laura J. van't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A.M. Hart, Mao Mao, Hans L. Peterse, Karin van der Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, Rene Bernards and Stephen

H. Friend (2002) "Gene expression profiling predicts clinical outcome of breast cancer", *Nature*, **415**:530-536

M. J. van de Vijver, Y. D. He, L. van't Veer, H. Dai, A. M. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend and R. Bernards (2002) "A Gene Expression Signature as a Predictor of Survival in Breast Cancer", *New England Journal of Medicine*, **347**(25):1999-2009

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
########
## Example for the mainz7g dataset
########
## show the first 5 columns of the expression data
exprs(mainz7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(mainz7g))
## show first 20 feature names
featureNames(mainz7g)
## show the experiment data summary
experimentData(mainz7g)
## show the used platform
annotation(mainz7g)
## show the abstract for this dataset
abstract(mainz7g)
```

`censor.time` *Function to artificially censor survival data*

Description

The function censors the survival data at a specific point in time. This is 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 data by using the function `concordance.index`. The statistical test is a Student t test for dependent samples.

Usage

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

Arguments

`cindex1`	first concordance index as returned by the `concordance.index` function.
`cindex2`	second concordance index as returned by the `concordance.index` 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

`p.value`	p-value from the Student t test for the comparison cindex1 > cindex2.
`cindex1`	value of the first concordance index.
`cindex2`	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=list("cindex.size1"=c2.1, "cindex.size2"=c2.2))
```

`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, na.rm = 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 experience an event before the other sample or belongs to a higher binary class.

Usage

```
concordance.index(x, surv.time, surv.event, cl, weights, comppairs=10,
strat, alpha = 0.05, outx = TRUE, method = c("conservative", "noether",
"nam"), alternative = c("two.sided", "less", "greater"), na.rm = FALSE)
```

Arguments

`x`	a vector of risk predictions.
`surv.time`	a vector of event times.
`surv.event`	a vector of event occurence indicators.
`cl`	a vector of binary class indicators.
`weights`	weight of each sample.
`comppairs`	threshold for compairable patients.
`strat`	stratification indicator.
`alpha`	apha 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).
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` (concordance index is greater than 0.5) or `"less"` (concordance index is less than 0.5). You can specify just the initial letter.
`na.rm`	`TRUE` if missing values should be removed.

Value

`c.index`	concordance index 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 if different from 0.5.
`n`	number of samples used for the estimation.
`data`	list of data used to compute the index (`x`, `surv.time` and `surv.event`, or `cl`).
`comppairs`	number of compairable pairs.

Note

The "direction" of the concordance index (< 0.5 or > 0.5) is the opposite than the rcorr.cens function in the `Hmisc` package. So you can easily get the same results than rcorr.cens by changing the sign of `x`.

Author(s)

Benjamin Haibe-Kains, Markus Schroeder

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`, `coxphCPE`

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)
weight <- runif(100, min=0, max=1)
comppairs <- 10
cat("survival prediction:\n")
concordance.index(x=age, surv.time=stime, surv.event=sevent, strat=strat,
  weights=weight, method="noether", comppairs=comppairs)
cat("binary class prediction:\n")
## is age predictive of sex?
concordance.index(x=age, cl=sex, strat=strat, method="noether")
```

`concordanceIndexC` *Concordance index C*

Description

Concordance index C

Arguments

`msurv`	...
`ustrat`	...
`x2`	...
`cl2`	...
`st`	...
`se`	...
`weights`	...
`strat`	...
`N`	...
`outx`	...
`ch`	...
`dh`	...
`uh`	...
`rph`	...
`lenS`	...
`lenU`	...

Author(s)

Catharina Colsen, Zhaleh Safikhani, Benjamin Haibe-Kains

`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, nfold = 1, setseed,
  na.rm = FALSE, verbose = FALSE)
```

Arguments

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

Value

`cvpl`	mean cross-validated partial likelihood (lower is better)
`pl`	vector of cross-validated partial likelihoods
`convergence`	vector of booleans reporting the convergence of the Cox model in each fold
`n`	number of observations used to estimate the cross-validated partial likelihood

Author(s)

Benjamin Haibe-Kains

References

Verweij PJM. and van Houwelingen H (1993) "Cross-validation in survival analysis", *Statistics in Medicine*, **12**, pages 2305–2314

van Houwelingen H, Bruinsma T, Hart AA, van't Veer LJ, and Wessels LFA (2006) "Cross-validated Cox regression on microarray gene expression data", *Statistics in Medicine*, **25**, pages 3201–3216.

See Also

`logpl`, `coxph`

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)
cvpl(x=age, surv.time=stime, surv.event=sevent, strata=strat,
  nfold=10, setseed=54321)
```

`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,
method.test = c("logrank", "likelihood.ratio", "wald"), 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`	apha level to compute confidence interval.
`method.test`	Statistical test to use in order to compute the p-values related to a D. index, see summary.coxph for more details.
`na.rm`	`TRUE` if missing values should be removed.
`...`	additional parameters to be passed to the `coxph` 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.

Note that the value D reported in (Royston and Sauerbrei, 2004) is given

Value

`d.index`	The d.index value is the robust hazard ratio (coefficient exponentiated).
`coef`	D index as reported in (Royston and Sauerbrei, 2004), ie. coefficient fitted in the cox regression model.
`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 if different from 0.5.
`n`	number of samples used for the estimation.
`coxm`	`coxph.object` fitted on the survival data and `z` (see below).
`data`	list of data used to compute the index (`x`, `z`, `surv.time` and `surv.event`). The item `z` contains the scaled rankits which are the expected standard Normal order statistics scaled by `kappa`.

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)
weight <- runif(100, min=0, max=1)
D.index(x=age, surv.time=stime, surv.event=sevent, weights=weight, 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 dindex1 > dindex2
`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 D 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 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 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=list("dindex.size1"=d2.1, "dindex.size2"=d2.2))
```

`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-16)
```

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. $latex$ when inv = `FALSE` and $latex$ 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)
```

`forestplot.surv` *Forest plots enables to display performance estimates of survival models*

Description

Draw a forest plot together with a table of text.

Usage

```
forestplot.surv(labeltext, mean, lower, upper, align = NULL,
is.summary = FALSE, clip = c(-Inf, Inf), xlab = "", zero = 0,
graphwidth = unit(2, "inches"), col, xlog = FALSE,
box.size = NULL, x.ticks = NULL, ...)
```

Arguments

`labeltext`	Matrix of strings or `NA`s for blank spaces
`mean`	Vector of centers of confidence intervals (or `NA`s for blank space)
`lower`	Vector of lower ends of confidence intervals
`upper`	Vector of upper ends of confidence intervals
`align`	Vector giving alignment (`l`,`r`,`c`) for columns of table
`is.summary`	Vector of logicals. Summary lines have bold text and diamond confidence intervals.
`clip`	Lower and upper limits for clipping confidence intervals to arrows
`xlab`	x-axis label
`zero`	x-axis coordinate for zero line
`graphwidth`	Width of confidence interval graph
`col`	See `meta.colors`
`xlog`	If `TRUE`, x-axis tick marks are exponentiated
`box.size`	Override the default box size based on precision
`x.ticks`	Optional user-specified x-axis tick marks. Specify `NULL` to use the defaults, `numeric(0)` to omit the x-axis.
`...`	Not used.

Details

This function is more flexible than `metaplot` and the `plot` methods for meta-analysis objects, but requires more work by the user.

In particular, it allows for a table of text, and clips confidence intervals to arrows when they exceed specified limits.

Value

None

References

rmeta package, CRAN, Thomas Lumley <tlumley@u.washington.edu>. Functions for simple fixed and random effects meta-analysis for two-sample comparisons and cumulative meta-analyses. Draws standard summary plots, funnel plots, and computes summaries and tests for association and heterogeneity.

See Also

`metaplot`, `forestplot`

Examples

```
require(rmeta)
myspace <- "    "
labeltext <- cbind(c("Gene Symbol", "AAA", "BBB", "CCC"),c(rep(myspace,4)))
bs <- rep(0.5, nrow(labeltext))
r.mean <- c(NA, 0.35, 0.5, 0.65)
r.lower <- c(NA, 0.33, 0.4, 0.6)
r.upper <- c(NA, 0.37, 0.6, 0.7)

forestplot.surv(labeltext=labeltext, mean=r.mean, lower=r.lower, upper=r.upper,
zero=0.5, align=c("l"), graphwidth=grid::unit(2, "inches"), x.ticks=seq(0.3,0.8,0.1),
xlab=paste( "Forestplot Example", myspace, sep=""), col=meta.colors(box="royalblue",
line="darkblue", zero="darkred"), box.size=bs, clip=c(0.3,0.8))
```

`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

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

```
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)
```

`get_concordanceIndex_onevariable`

Get concordance index

Description

Get the concordance index

Arguments

`Rmsurv`	...
`Rustrat`	...
`Rx2`	...
`Rcl2`	...
`Rst`	...
`Rse`	...
`Rweights`	...
`Rstrat`	...
`RN`	...
`Routx`	...
`RlenS`	...
`RlenU`	...

Author(s)

Catharina Colsen

`hazard.ratio` *Function to estimate the hazard ratio through Cox regression*

Description

Function to compute the hazard ratio for a risk prediction.

Usage

```
hazard.ratio(x, surv.time, surv.event, weights, strat, alpha = 0.05,
method.test = c("logrank", "likelihood.ratio", "wald"), 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`	apha level to compute confidence interval.
`method.test`	Statistical test to use in order to compute the p-values related to a D. index, see summary.coxph for more details.
`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.
`coef`	coefficient (beta) estimated in the cox regression model.
`se`	standard error of the coefficient (beta) estimate.
`lower`	lower bound for the confidence interval.
`upper`	upper bound for the confidence interval.
`p.value`	p-value computed using the likelihood ratio test whether the hazard ratio is different from 1.
`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`, `surv.time` and `surv.event`).

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)
weight <- runif(100, min=0, max=1)
hazard.ratio(x=age, surv.time=stime, surv.event=sevent, weights=weight,
  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

`hr1`	first hazard ratio.
`hr2`	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

`p.value`	p-value from the Student t test for the comparison beta1 > beta2 (equivalently hr1 > hr2)
`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

```
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

`list.hr1`	first list of D indices as returned by the `hazard.ratio` function.
`list.hr2`	second list of D indices as returned by the `hazard.ratio` 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 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

`p.value`	p-value from the Student t test for the comparison hr1 > hr2.
`hr1`	meta-estimator of the first D index.
`hr2`	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.size2"=h2.2))
```

`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 beta1 > beta2 (equivalently hr1 > hr2)
`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

```
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)
coxm1 <- coxph(Surv(stime, sevent) ~ age)
coxm2 <- coxph(Surv(stime, sevent) ~ size)
hr.comp2(x1=age, beta1=coxm1$coefficients, se1=drop(sqrt(coxm1$var)),
  x2=size, beta2=coxm2$coefficients, se2=drop(sqrt(coxm2$var)), n=100)
```

`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

`auc1`	vector of AUCs computed from the first time-dependent ROC curves for some points in time
`auc2`	vector of AUCs computed from the second time-dependent ROC curves for some points in time
`time`	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

`p.value`	p-value from the Wilcoxon rank sum test for the comparison iauc1 > iauc2
`iauc1`	value of the IAUC for the first set of time-depdent ROC curves
`iauc2`	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, verbose=FALSE)
 mytdroc1 <- c(mytdroc1, list(rr))
}
auc1 <- unlist(lapply(mytdroc1, function(x) { return(x$AUC) }))
##age
mytdroc2 <- NULL
for(i in 1:length(tt)) {
 rr <- tdrocc(x=age, surv.time=stime, surv.event=sevent, time=tt[i],
    na.rm=TRUE, verbose=FALSE)
 mytdroc2 <- c(mytdroc2, list(rr))
}
auc2 <- unlist(lapply(mytdroc2, function(x) { return(x$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 ibsc1 < ibsc2
`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 points in 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, weight.s, x.label, y.label, main.title, sub.title,
leg.text, leg.pos = "bottomright", leg.bty = "o", leg.inset = 0.05, o.text, v.line,
h.line, .col = 1:4, .lty = 1, .lwd = 1, show.n.risk = FALSE, n.risk.step,
n.risk.cex = 0.85, verbose = TRUE, ...)
```

Arguments

`formula.s`	formula composed of a `Surv` object and a strata variable (i.e. stratification).
`data.s`	data frame composed of the variables used in the formula.
`weight.s`	vector of weights of length nrow(data.s).
`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.bty`	the type of box to be drawn around the legend. The allowed values are `"o"` (the default) and `"n"`.
`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 `plot` function.

Details

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

Value

Several Kaplan-Meier survival curves with number of individuals at risk at some time points.

Author(s)

Christos Hatzis, Benjamin Haibe-Kains

See Also

`survfit`, `coxph`

Examples

```
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)
ddweights <- array(1, dim=nrow(dd))

km.coxph.plot(formula.s=Surv(surv.time, surv.event) ~ strat, data.s=dd,
  weight.s=ddweights, x.label="Time (years)", y.label="Probability of survival",
  main.title="", leg.text=paste(c("Low", "Intermediate", "High"), "   ", sep=""),
  leg.pos="topright", leg.inset=0, .col=c("darkblue", "darkgreen", "darkred"),
  .lty=c(1,1,1), show.n.risk=TRUE, n.risk.step=2, n.risk.cex=0.85, verbose=FALSE)
```

`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(pred, surv.time, surv.event, strata, na.rm = FALSE, verbose = FALSE)
```

Arguments

`surv.time`	vector of times to event occurrence
`surv.event`	vector of indicators for event occurrence
`pred`	linear predictors computed using the Cox model
`strata`	stratification variable
`na.rm`	`TRUE` if the missing values should be removed from the data, `FALSE` otherwise
`verbose`	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

```
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)
dd <- data.frame("stime"=stime, "sevent"=sevent, "age"=age)
```

```
##Cox model
coxm <- coxph(Surv(stime, sevent) ~ age, data=dd)
##log partial likelihood of the null model
logpl(pred=rep(0, nrow(dd)), surv.time=stime, surv.event=sevent)
##log partial likelihood of the Cox model
logpl(pred=predict(object=coxm, newdata=dd), surv.time=stime, surv.event=sevent)
##equivalent to
coxm$loglik
```

`mainz7g` *Subset of MAINZ dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the MAINZ datasets (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 200 samples, containing:

- `exprs(mainz7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a technology (single-channel, oligonucleotides).
- `fData(mainz7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a.
- `pData(mainz7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(mainz7g)`: MIAME object containing information about the dataset.
- `annotation(mainz7g)`: Name of the affy chip.

Details

This dataset represents a subset of the study published by Schmidt et al. 2008. The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

mainz:

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE11121`

References

Marcus Schmidt, Daniel Boehm, Christian von Toerne, Eric Steiner, Alexander Puhl, Heryk Pilch, Hans-Anton Lehr, Jan G. Hengstler, Hainz Koelbl and Mathias Gehrmann (2008) "The Humoral Immune System Has a Key Prognostic Impact in Node-Negative Breast Cancer", *Cancer Research*, **68**(13):5405-5413

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(mainz7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(mainz7g))
## show first 20 feature names
featureNames(mainz7g)
## show the experiment data summary
experimentData(mainz7g)
## show the used platform
annotation(mainz7g)
## show the abstract for this dataset
abstract(mainz7g)
```

`metaplot.surv` *Meta-analysis plot (forest plot)*

Description

Plot confidence intervals with boxes indicating the sample size/precision and optionally a diamond indicating a summary confidence interval. This function is usually called by `plot` methods for meta-analysis objects. Additional, you can specifiy your own lower and upper boarder from the confidence interval.

Usage

```
metaplot.surv(mn, se=NULL, lower=NULL, upper=NULL, nn=NULL,
    labels=NULL, conf.level = .95, xlab = "", ylab = "", xlim = NULL,
    summn = NULL, sumse = NULL, sumlower = NULL, sumupper = NULL,
    sumnn = NULL, summlabel = "Summary", logeffect = FALSE,
    lwd = 2, boxsize = 1, zero = as.numeric(logeffect),
    colors, xaxt="s", logticks=TRUE, ... )
```

Arguments

`mn`	point estimates from studies
`se`	standard errors of `mn`

`lower`	Vector of lower ends of confidence intervals
`upper`	Vector of upper ends of confidence intervals
`nn`	precision: box ares is proportional to this. `1/se^2` is the default
`labels`	labels for each interval
`conf.level`	Confidence level for confidence intervals
`xlab`	label for the point estimate axis
`ylab`	label for the axis indexing the different studies
`xlim`	the range for the x axis.
`summn`	summary estimate
`sumse`	standard error of summary estimate
`sumlower`	lower end of confidence intervals of summary estimate
`sumupper`	upper end of confidence intervals of summary estimate
`sumnn`	precision of summary estimate
`summlabel`	label for summary estimate
`logeffect`	`TRUE` to display on a log scale
`lwd`	line width
`boxsize`	Scale factor for box size
`zero`	"Null" effect value
`xaxt`	use `"n"` for no x-axis (to add a customised one)
`logticks`	if `TRUE` and `logscale`, have tick values approximately equally spaced on a log scale
.	
`colors`	see `meta.colors`
`...`	Other graphical parameters

Value

This function is used for its side-effect.

See Also

`forestplot.surv` for more flexible plots

Examples

```
metaplot.surv(mn=c(0.4,0.5,0.6), lower=c(0.35,0.4,0.57), upper=c(0.45,0.6,0.63),
labels=c("A","B","C"), xlim=c(0.2,0.8), boxsize=0.5, zero=0.5,
col=rmeta::meta.colors(box="royalblue",line="darkblue",zero="firebrick"))
```

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

Description

Function to compute the minimum redundancy - maximum relevance (mRMR) ranking for a risk prediction or a binary classification task based on the concordance index. The mRMR feature selection has been adapted to use the concordance index to estimate the correlation between a variable and the output (binary or survival) data.

Usage

```
mrmr.cindex(x, surv.time, surv.event, cl, weights, comppairs=10, strat,
alpha = 0.05, outx = TRUE, method = c("conservative", "noether", "nam"),
alternative = c("two.sided", "less", "greater"), na.rm = FALSE)
```

Arguments

`x`	a vector of risk predictions.
`surv.time`	a vector of event times.
`surv.event`	a vector of event occurence indicators.
`cl`	a vector of binary class indicators.
`weights`	weight of each sample.
`comppairs`	threshold for compairable patients.
`strat`	stratification indicator.
`alpha`	apha 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).
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` (concordance index is greater than 0.5) or `"less"` (concordance index is less than 0.5). You can specify just the initial letter.
`na.rm`	`TRUE` if missing values should be removed.

Value

A mRMR ranking

Note

The "direction" of the concordance index (< 0.5 or > 0.5) is the opposite than the rcorr.cens function in the `Hmisc` package. So you can easily get the same results than rcorr.cens by changing the sign of `x`.

Author(s)

Benjamin Haibe-Kains, Markus Schroeder

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`, `coxphCPE`

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)
weight <- runif(100, min=0, max=1)
comppairs <- 10
xx <- data.frame("age"=age, "sex"=sex)
cat("survival prediction:\n")
mrmr.cindex(x=xx, surv.time=stime, surv.event=sevent, strat=strat, weights=weight,
method="noether", comppairs=comppairs)
```

`mrmr.cindex.ensemble` *Function to compute the concordance index for survival or binary class prediction*

Description

Function to compute the minimum redundancy - maximum relevance (mRMR) ranking for a risk prediction or a binary classification task based on the concordance index. The mRMR feature selection has been adapted to use the concordance index to estimate the correlation between a variable and the output (binary or survival) data.

Usage

```
mrmr.cindex.ensemble(x, surv.time, surv.event, cl, weights, comppairs=10,
strat, alpha = 0.05, outx = TRUE, method = c("conservative", "noether",
"nam"), alternative = c("two.sided", "less", "greater"), maxparents,
maxnsol, nboot = 200, na.rm = FALSE)
```

Arguments

`x`	a vector of risk predictions.
`surv.time`	a vector of event times.
`surv.event`	a vector of event occurence indicators.
`cl`	a vector of binary class indicators.
`weights`	weight of each sample.
`comppairs`	threshold for compairable patients.
`strat`	stratification indicator.
`alpha`	apha 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).
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` (concordance index is greater than 0.5) or `"less"` (concordance index is less than 0.5). You can specify just the initial letter.
`maxparents`	maximum number of candidate variables to be added in the ranking solutions tree.
`maxnsol`	maximum number of ranking solutions to be considered.
`nboot`	number of bootstraps to compute standard error of a ranking solution.
`na.rm`	`TRUE` if missing values should be removed.

Value

A mRMR ranking

Note

The "direction" of the concordance index (< 0.5 or > 0.5) is the opposite than the rcorr.cens function in the `Hmisc` package. So you can easily get the same results than rcorr.cens by changing the sign of `x`.

Author(s)

Benjamin Haibe-Kains, Markus Schroeder

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`, `coxphCPE`

`mrmr_cIndex` *mRMR cIndex*

Description

mRMR cIndex

Arguments

`Rdata`	...
`Rnamat`	...
`RcIndex`	...
`Rnvar`	...
`Rnsample`	...
`Rthreshold`	...

Author(s)

Catharina Colsen

`mrmr_cIndex_ensemble_remove` *mRMR cIndex Ensemble Remove*

Description

mRMR cIndex Ensemble Remove

Arguments

`Rdata`	...
`Rnamat`	...
`Rmaxparents`	...
`Rnvar`	...
`Rnsample`	...
`Rpredn`	...
`Rnpredn`	...
`Rrep_boot`	...
`Rmaxnsol`	...
`Rthreshold`	...
`Rmsurv`	...
`Rustrat`	...
`Rcl2`	...
`Rst`	...
`Rse`	...
`Rweights`	...
`Rstrat`	...
`RN`	...
`Routx`	...
`RlenS`	...
`RlenU`	...

Author(s)

Catharina Colsen

`nki7g` *Subset of NKI dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the NKI datasets (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 337 samples, containing:

- `exprs(nki7g)`: Matrix containing gene expressions as measured by Agilent technology (dual-channel, oligonucleotides).
- `fData(nki7g)`: AnnotatedDataFrame containing annotations of Agilent microarray platform.
- `pData(nki7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(nki7g)`: MIAME object containing information about the dataset.
- `annotation(nki7g)`: Name of the agilent chip.

Details

This dataset represents a subset of the study published by van't Veer et al. 2002 and van de Vijver et al. 2002. The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

nki:

`http://www.rii.com/publications/2002/vantveer.html`

References

Laura J. van't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A.M. Hart, Mao Mao, Hans L. Peterse, Karin van der Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, Rene Bernards and Stephen H. Friend (2002) "Gene expression profiling predicts clinical outcome of breast cancer", *Nature*, **415**:530-536

M. J. van de Vijver, Y. D. He, L. van't Veer, H. Dai, A. M. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend and R. Bernards (2002) "A Gene Expression Signature as a Predictor of Survival in Breast Cancer", *New England Journal of Medicine*, **347**(25):1999-2009

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(nki7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(nki7g))
## show first 20 feature names
```

```
featureNames(nki7g)
## show the experiment data summary
experimentData(nki7g)
## show the used platform
annotation(nki7g)
## show the abstract for this dataset
abstract(nki7g)
```

`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

`formula.s`	formula composed of a `Surv` 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.
`t.step`	time step at which the number of individuals at risk is computed.
`t.end`	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

```
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, t.end=1)
```

`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

`data.tr`	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.
`data.ts`	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.
`method`	method for survival probabilities estimation using either a Cox model or the product-limit estimator

Value

`time`	vector of points in time
`bsc`	vector of Brier scores (BSC) at ome points in time
`bsc.integrated`	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`, `coxph`, `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.est= c ("point", "lower", "upper"))
```

Arguments

`data.tr`	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
`score`	risk score for the test set
`yr`	a point in time for which the survival probabilities must be computed
`method`	method for survival probabilities estimation, either `cox` or `prodlim` for the Cox model or the product-limit estimator, respectively
`conf.int`	value in [0,1]. Default at 0.95
`which.est`	which estimation to be returned? `point` for the point estimate, `lower` for the lower bound and `upper` 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

`coxph`, `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` *Function to compute sensitivity and specificity for a binary classification of survival data*

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 = FALSE, ...)
```

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

`sens`	sensitivity estimate
`sens.se`	standard error for sensitivity estimate
`spec`	specificity estimate
`spec.se`	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=FALSE)
```

`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, verbose = FALSE, span = 0, lambda = 0, ...)
```

Arguments

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

Value

`spec`	specificity estimates
`sens`	sensitivity estimates
`rule`	rule to compute the predictions at each cutoff
`cuts`	cutoffs
`time`	time point at which the time-dependent ROC is computed
`survival`	overall survival at the time point
`AUC`	Area Under the Curve (AUC) of teh time-dependent ROC curve
`data`	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(x=1-tdroc$spec, y=tdroc$sens, type="l", xlab="1 - specificity",
  ylab="sensitivity", xlim=c(0, 1), ylim=c(0, 1))
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

`p`	vector of p-values
`weight`	vector of weights (e.g. sample size of each study)
`na.rm`	`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

`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.

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)
```

`transbig7g` *Subset of the TRANSBIG dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the TRANSBIG dataset (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 198 samples, containing:

- `exprs(transbig7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a technology (single-channel, oligonucleotides).
- `fData(transbig7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a.
- `pData(transbig7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(transbig7g)`: MIAME object containing information about the dataset.
- `annotation(transbig7g)`: Name of the affy chip.

Details

This dataset represents a subset of the study published by Desmedt et al. 2007. The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

transbig:

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7390`

References

Christine Desmedt, Fanny Piette, Sherene Loi, Yixin Wang, Francoise Lallemand, Benjamin Haibe-Kains, Giuseppe Viale, Mauro Delorenzi, Yi Zhang, Mahasti Saghatchian d Assignies, Jonas Bergh, Rosette Lidereau, Paul Ellis, Adrian L. Harris, Jan G. M. Klijn, John A. Foekens, Fatima Cardoso, Martine J. Piccart, Marc Buyse and Christos Sotiriou (2007) "Strong Time Dependence of the 76-Gene Prognostic Signature for Node-Negative Breast Cancer Patients in the TRANSBIG Multicenter Independent Validation Series", *Clinical Cancer Research*, **13**(11):3207-3214

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(transbig7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(transbig7g))
## show first 20 feature names
featureNames(transbig7g)
## show the experiment data summary
experimentData(transbig7g)
## show the used platform
annotation(transbig7g)
## show the abstract for this dataset
abstract(transbig7g)
```

`unt7g` *Subset of UNT dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the UNT datasets (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 137 samples, containing:

- `exprs(unt7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a and hgu133b technology (single-channel, oligonucleotides).
- `fData(unt7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a and hgu133b.
- `pData(unt7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(unt7g)`: MIAME object containing information about the dataset.
- `annotation(unt7g)`: Name of the affy chip.

Details

This dataset represents a subset of the study published by Sotiriou et al. 2006. The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

unt:

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2990`

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE6532`

References

Christos Sotiriou, Pratyaksha Wirapati, Sherene Loi, Adrian Harris, Steve Fox, Johanna Smeds, Hans Nordgren, Pierre Farmer, Viviane Praz, Benjamin Haibe-Kains, Christine Desmedt, Denis Larsimont, Fatima Cardoso, Hans Peterse, Dimitry Nuyten, Marc Buyse, Marc J. Van de Vijver, Jonas Bergh, Martine Piccart and Mauro Delorenzi (2006) "Gene Expression Profiling in Breast Cancer: Understanding the Molecular Basis of Histologic Grade To Improve Prognosis", *Journal of the National Cancer Institute*, **98**(4):262-272

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(unt7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(unt7g))
## show first 20 feature names
featureNames(unt7g)
## show the experiment data summary
experimentData(unt7g)
## show the used platform
annotation(unt7g)
## show the abstract for this dataset
abstract(unt7g)
```

`upp7g` *Subset of UPP dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the UPP datasets (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 251 samples, containing:

- `exprs(upp7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a and hgu133b technology (single-channel, oligonucleotides).
- `fData(upp7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a and hgu133b.
- `pData(upp7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(upp7g)`: MIAME object containing information about the dataset.
- `annotation(upp7g)`: Name of the affy chip.

Details

This dataset represents a subset of the study published by Miller et al. 2005. The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

upp:

`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE3494`

References

Lance D. Miller, Johanna Smeds, Joshy George, Vinsensius B. Vega, Liza Vergara, Alexander Ploner, Yudi Pawitan, Per Hall, Sigrid Klaar, Edison T. Liu and Jonas Bergh (2005) "An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival" *Proceedings of the National Academy of Sciences of the United States of America* **102**(38):13550-13555

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(upp7g)[ ,1:5]
## show the first 6 rows of the phenotype data
```

```
head(pData(upp7g))
## show first 20 feature names
featureNames(upp7g)
## show the experiment data summary
experimentData(upp7g)
## show the used platform
annotation(upp7g)
## show the abstract for this dataset
abstract(upp7g)
```

`vdx7g` *Subset of VDX dataset containing gene expression, annotations and clinical data.*

Description

This dataset contains a subset of the gene expression, annotations and clinical data from the VDX datasets (see section details). The subset contains the seven genes introduced by Desmedt et al. 2008

Format

ExpressionSet with 7 features and 344 samples, containing:

- `exprs(vdx7g)`: Matrix containing gene expressions as measured by Affymetrix hgu133a technology (single-channel, oligonucleotides).
- `fData(vdx7g)`: AnnotatedDataFrame containing annotations of Affy microarray platform hgu133a.
- `pData(vdx7g)`: AnnotatedDataFrame containing Clinical information of the breast cancer patients whose tumors were hybridized.
- `experimentalData(vdx7g)`: MIAME object containing information about the dataset.
- `annotation(vdx7g)`: Name of the affy chip.

Details

This dataset represents a subset of the study published by Wang. et al. 2005 and Minn et al. 2007 . The subset contains the genes AURKA (also known as STK6, STK7, or STK15), PLAU (also known as UPA), STAT1, VEGF, CASP3, ESR1, and ERBB2, as introduced by Desmedt et al. 2008. The seven genes represent the proliferation, tumor invasion/metastasis, immune response, angiogenesis, apoptosis phenotypes, and the ER and HER2 signaling, respectively.

Source

vdx:

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2034

References

Y. Wang, J. G. Klijn, Y. Zhang, A. M. Sieuwerts, M. P. Look, F. Yang, D. Talantov, M. Timmermans, M. E. Meijer-van Gelder, J. Yu, T. Jatkoe, E. M. Berns, D. Atkins and J. A. Foekens (2005) "Gene-Expression Profiles to Predict Distant Metastasis of Lymph-Node-Negative Primary Breast Cancer", *Lancet*, **365**:671-679

Andy J. Minn, Gaorav P. Gupta, David Padua, Paula Bos, Don X. Nguyen, Dimitry Nuyten, Bas Kreike, Yi Zhang, Yixin Wang, Hemant Ishwaran, John A. Foekens, Marc van de Vijver and Joan Massague (2007) "Lung metastasis genes couple breast tumor size and metastatic spread", *Proceedings of the National Academy of Sciences*, **104**(16):6740-6745

Examples

```
## load Biobase package
library(Biobase)
## load the dataset
data(breastCancerData)
## show the first 5 columns of the expression data
exprs(vdx7g)[ ,1:5]
## show the first 6 rows of the phenotype data
head(pData(vdx7g))
## show first 20 feature names
featureNames(vdx7g)
## show the experiment data summary
experimentData(vdx7g)
## show the used platform
annotation(vdx7g)
## show the abstract for this dataset
abstract(vdx7g)
```

Index