survcomp: a package for performance assessment and comparison for survival analysis

Benjamin Haibe-Kains[1,2,3], Markus Schröder[1,5], John Quackenbush[1,2], Christos Sotiriou[4], and Gianluca Bontempi[3]

[1]*Computational Biology and Functional Genomics Laboratory, Dana-Farber Cancer Institute, Harvard School of Public Health*
[2]*Center for Cancer Computational Biology, Dana-Farber Cancer Institute*
[3]*Machine Learning Group, Université Libre de Bruxelles*
[4]*Breast Cancer Translational Research Laboratory, Institut Jules Bordet, Université Libre de Bruxelles*
[5]*Bioinformatics Resource Facility, Center for Biotechnology, Bielefeld University, Germany*

March 1, 2011

Contents

1 Introduction **1**
1.1 Installation 2
1.2 Further help 2
1.3 Citing 2

2 Functional demonstration **3**
2.1 Overview 3

3 Functions in survcomp **22**

1 Introduction

The *survcomp* package is providing functions to assess and to compare the performance of risk prediction (survival) models. Whilst there are several risk prediction models published, the task to assess the performance This package has been primarily used in gene expression studies and the corresponding results have been published in high impact journals (Bioinformatics, Clinical Cancer Research, Journal of Clinical Oncology, ...). We plan to extend its use to other omics high-throughput data in a close future.

1.1 Installation

survcomp requires that *survival*, *ipred*, *prodlim*, *survivalROC*, *SuppDists*, *bootstrap* and *R* (>= 2.3.0) are installed. These should be installed automatically when you install *survcomp*. To install *survcomp*, source biocLite from bioconductor

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("survcomp")
```

1.2 Further help

To view the *survcomp* description and a summary of all the functions within *survcomp*, type the following:

```
> library(help=survcomp)
```

1.3 Citing

We are delighted if you use this package. Please do email us if you find a bug or have a suggestion. We would be very grateful if you could cite:

B. Haibe-Kains, C. Desmedt, C. Sotiriou and G. Bontempi (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):**2200-2208.

2 Functional demonstration

We will very briefly demonstrate some of the functions in *survcomp*. We use the `sampleData` datafile for demonstration purposes, it includes subsets of the datasets *breastCancerMAINZ*, *breastCancerTRANSBIG*, *breastCancerUPP*, *breastCancerUNT*, *breastCancerVDX* and *breastCancerNKI*, available as experimental datapackages on Bioconductor. The six datasets in `sampleData` contain 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.

2.1 Overview

To use the `ExpressionSet` object we have to load the *Biobase* package.

```
> library(xtable)
> library(Biobase)
```

Loading the `sampleData` object will results in 6 new object. If you execute `ls()` you will see `mainzSample`,`transbigSample`, `uppSample`, `untSample`, `vdxSample` and `nkiSample`. More details about these objects is available in the `sampleData` manpage (`?sampleData`).

```
> data(sampleData)
```

Before we can start the analysis, we have to define the annotation for the mentioned seven genes, the datasets we use and a few help-variables. We define the gene symbol list (`gsList`), the entrez-gene ID list (`gidList`), the probe names for the used Agilent microarray (`probesNKI`), the proben name for the used Affymetrix microarray (`probesAffy`), a list containing the used dataset names (`datasetList`), spaces for displaying the text in the forestplot at the right place (`myspace` and `mybigspace`) and `tc` for setting the censored time to 10 years.

```
> gsList <- c("ESR1", "ERBB2", "AURKA", "PLAU", "VEGF", "STAT1",
+     "CASP3")
> gidList <- c(2099, 2064, 6790, 5328, 7422, 6772, 836)
> probesNKI <- c("NM_000125", "NM_004448", "NM_003600", "NM_002658",
+     "NM_003376", "NM_007315", "NM_004346")
> probesAffy <- c("205225_at", "216836_s_at", "208079_s_at", "211668_s_at",
+     "211527_x_at", "209969_s_at", "202763_at")
> datasetList <- c("MAINZ", "TRANSBIG", "UPP", "UNT", "VDX", "NKI",
```

```
+     "ALL")
> myspace <- " "
> mybigspace <- "    "
> tc <- 10 * 365
```

For computing the concordance index for each gene in each dataset, we have to call the `concordance.index()`) function. See `?concordance.index` for details.

```
> cindexall.mainz.small <- t(apply(X = exprs(mainzSample), MARGIN = 1,
+     function(x, y, z) {
+         tt <- concordance.index(x = x, surv.time = y, surv.event = z,
+             method = "noether", na.rm = TRUE)
+         return(c(cindex = tt$c.index, cindex.se = tt$se, lower = tt$lower,
+             upper = tt$upper))
+     }, y = pData(mainzSample)[, "t.dmfs"], z = pData(mainzSample)[,
+         "e.dmfs"]))
```

For computing the D index for each gene in each dataset, we have to call the `D.index()`) function. See `?D.index` for details.

```
> dindexall.mainz.small <- t(apply(X = exprs(mainzSample), MARGIN = 1,
+     function(x, y, z) {
+         tt <- D.index(x = x, surv.time = y, surv.event = z, na.rm = TRUE)
+         return(c(dindex = tt$d.index, dindex.se = tt$se, lower = tt$lower,
+             upper = tt$upper))
+     }, y = pData(mainzSample)[, "t.dmfs"], z = pData(mainzSample)[,
+         "e.dmfs"]))
```

For computing the hazard ratio for each gene in each dataset, we have to call the `hazard.ratio()`) function. See `?hazard.ratio` for details. Before computing the hazard ratio, we have to rescale the gene expression data for each dataset to a comparable range. Therefor we use the following function with `q=0.05`.

```
> rescale <- function(x, na.rm = FALSE, q = 0) {
+     if (q == 0) {
+         ma <- max(x, na.rm = na.rm)
+         mi <- min(x, na.rm = na.rm)
+     }
+     else {
+         ma <- quantile(x, probs = 1 - (q/2), na.rm = na.rm)
+         mi <- quantile(x, probs = q/2, na.rm = na.rm)
+     }
```

```
+     x <- (x - mi)/(ma - mi)
+     return(x)
+ }


> hratio.mainz.small <- t(apply(X = exprs(mainzSample), MARGIN = 1,
+     function(x, y, z) {
+         tt <- hazard.ratio(x = x, surv.time = y, surv.event = z,
+             na.rm = TRUE)
+         return(c(hratio = tt$hazard.ratio, hratio.se = tt$se,
+             lower = tt$lower, upper = tt$upper))
+     }, y = pData(mainzSample)[, "t.dmfs"], z = pData(mainzSample)[,
+         "e.dmfs"]))
```

To get an overall estimate over all datasets for the concordance index from each gene, we iterate over all the concordance indices of all datasets and combine them with the `combine.est` function from the *genefu* package and recalculate the lower- and upper border accordingly. We do that for the D indices and the hazard ratios in the same way.

The resulting combined concordance indices are:

```
> tt <- as.data.frame(NULL)
> for (i in 1:7) {
+     tt <- rbind(tt, combine.est(x = cbind(cindexall.mainz.small[i,
+         "cindex"], cindexall.transbig.small[i, "cindex"], cindexall.upp.small[i,
+         "cindex"], cindexall.unt.small[i, "cindex"], cindexall.vdx.small[i,
+         "cindex"], cindexall.nki.small[i, "cindex"]), x.se = cbind(cindexall.mainz.sm
+         "cindex.se"], cindexall.transbig.small[i, "cindex.se"],
+         cindexall.upp.small[i, "cindex.se"], cindexall.unt.small[i,
+             "cindex.se"], cindexall.vdx.small[i, "cindex.se"],
+         cindexall.nki.small[i, "cindex.se"]), ))
+ }
> tt$lower <- tt$estimate + qnorm(0.025, lower.tail = TRUE) * tt$se
> tt$upper <- tt$estimate + qnorm(0.025, lower.tail = FALSE) *
+     tt$se
> rownames(tt) <- gsList
> colnames(tt) <- c("cindex", "cindex.se", "lower", "upper")
> ccindex <- tt


> xtable(ccindex)
```

The resulting combined D indices are:

	cindex	cindex.se	lower	upper
ESR1	0.46	0.02	0.43	0.49
ERBB2	0.50	0.02	0.47	0.53
AURKA	0.64	0.01	0.62	0.67
PLAU	0.52	0.01	0.49	0.55
VEGF	0.56	0.01	0.53	0.59
STAT1	0.53	0.01	0.51	0.56
CASP3	0.52	0.01	0.50	0.55

	dindex	dindex.se	lower	upper
ESR1	0.89	0.08	0.73	1.05
ERBB2	1.06	0.08	0.91	1.22
AURKA	1.95	0.08	1.79	2.10
PLAU	1.18	0.08	1.02	1.34
VEGF	1.37	0.08	1.21	1.52
STAT1	1.14	0.08	0.99	1.29
CASP3	1.14	0.08	0.99	1.30

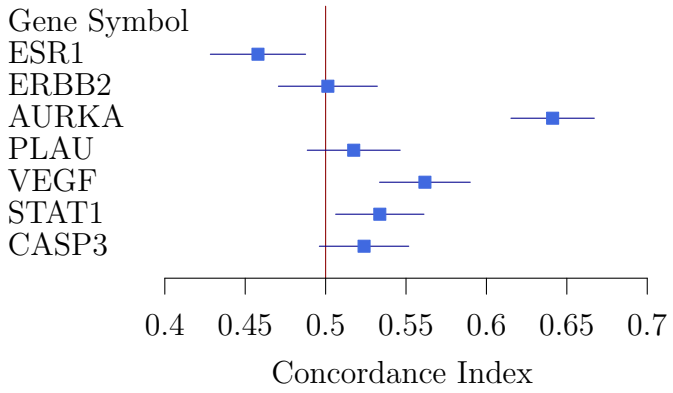
The resulting combined hazard ratios are:

	hratio	hratio.se	lower	upper
ESR1	0.98	0.03	0.93	1.03
ERBB2	1.06	0.04	0.98	1.14
AURKA	1.57	0.06	1.46	1.68
PLAU	1.07	0.06	0.95	1.19
VEGF	1.10	0.04	1.02	1.18
STAT1	1.06	0.05	0.96	1.16
CASP3	1.41	0.15	1.12	1.71

For displaying the combined concordance indices for each genes over all datasets, we use the `forestplot()` function. The resulting forestplot for all concordance indices is:

```
> labeltext <- cbind(c("Gene Symbol", gsList), c(rep(myspace, 8)))
> bs <- rep(0.5, nrow(labeltext))
> r.mean <- c(NA, ccindex$cindex)
> r.lower <- c(NA, ccindex$lower)
> r.upper <- c(NA, ccindex$upper)
> forestplot.surv(labeltext = labeltext, mean = r.mean, lower = r.lower,
+     upper = r.upper, zero = 0.5, align = c("l"), graphwidth = unit(2,
```

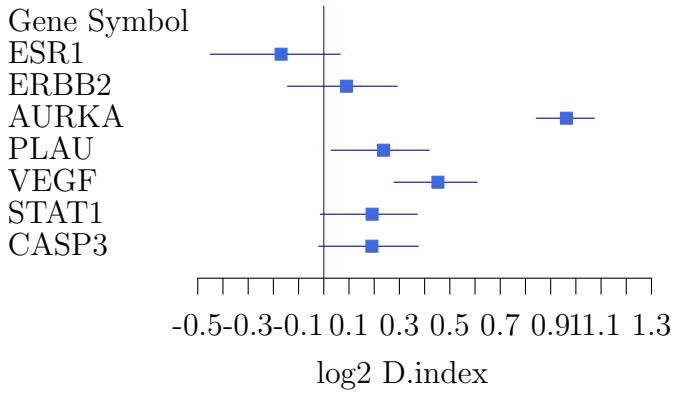
```
+         "inches"), x.ticks = seq(0.4, 0.7, 0.05), xlab = paste("Concordance
Index",
+         myspace, sep = ""), col = meta.colors(box = "royalblue",
+         line = "darkblue", zero = "darkred"), box.size = bs,
+     clip = c(0.4, 1))
```



The resulting forestplot for all D indices is:

```
> labeltext <- cbind(c("Gene Symbol", gsList), c(rep(myspace, 8)))
> bs <- rep(0.5, nrow(labeltext))
> tt <- log2(cdindex)
> forestplot.surv(labeltext = labeltext, mean = c(NA, tt$dindex),
+     lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0,
```

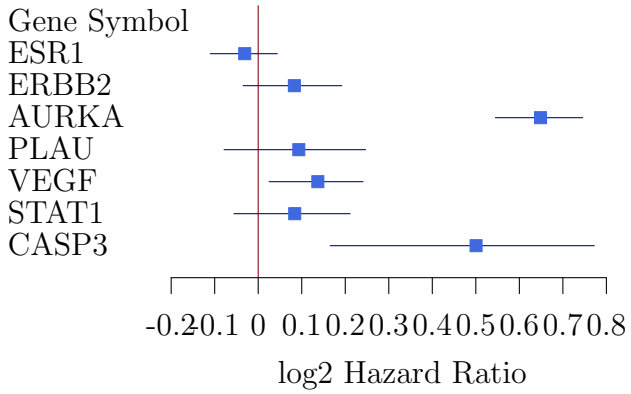
```
+     align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(-0.5,
+         1.3, 0.1), xlab = paste("log2 D.index", myspace, sep = ""),
+     col = meta.colors(box = "royalblue", line = "darkblue", zero = "dark-
red"),
+     box.size = bs, clip = c(-0.5, 1.3))
```



The resulting forestplot for all hazard ratios is:

```
> labeltext <- cbind(c("Gene Symbol", gsList), c(rep(myspace, 8)))
> bs <- rep(0.5, nrow(labeltext))
> tt <- log2(chratio)
> forestplot.surv(labeltext = labeltext, mean = c(NA, tt$hratio),
+     lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0,
```

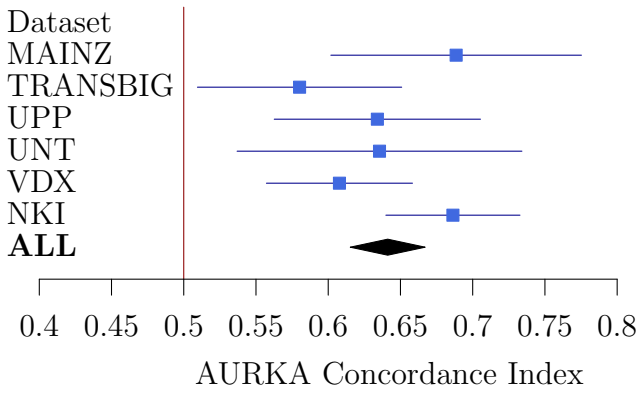
```
+     align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(-0.2,
+         0.8, 0.1), xlab = paste("log2 Hazard Ratio", myspace,
+         sep = ""), col = meta.colors(box = "royalblue", line = "darkblue",
+         zero = "darkred"), box.size = bs, clip = c(-0.2, 0.8))
```



Taking a more specific look, e.g. at the gene AURKA, we create the forestplot the same was as before, showing the concordance indices for the gene AURKA in each dataset and the combined estimation over all datasets.

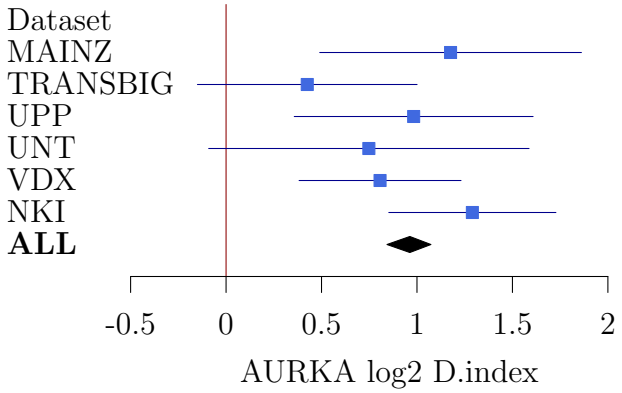
```
> tt <- rbind(cindexall.mainz.small[3, ], cindexall.transbig.small[3,
+     ], cindexall.upp.small[3, ], cindexall.unt.small[3, ], cindexall.vdx.small[3,
+     ], cindexall.nki.small[3, ], as.numeric(ccindex[3, ]))
> rownames(tt) <- datasetList
```

```
> tt <- as.data.frame(tt)
> labeltext <- cbind(c("Dataset", datasetList), c(rep(mybigspace,
+     8)))
> bs <- rep(0.5, nrow(labeltext))
> forestplot.surv(labeltext = labeltext, mean = c(NA, tt$cindex),
+     lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0.5,
+     align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(0.4,
+         0.8, 0.05), xlab = paste("AURKA Concordance Index", myspace,
+         sep = ""), col = meta.colors(box = "royalblue", line = "darkblue",
+         zero = "darkred"), box.size = bs, clip = c(0.5, 1), is.summary =
(c(rep(FALSE,
+         7), TRUE)))
```



And the same with the D indices for the gene AURKA in each dataset and the combined estimation over all datasets.

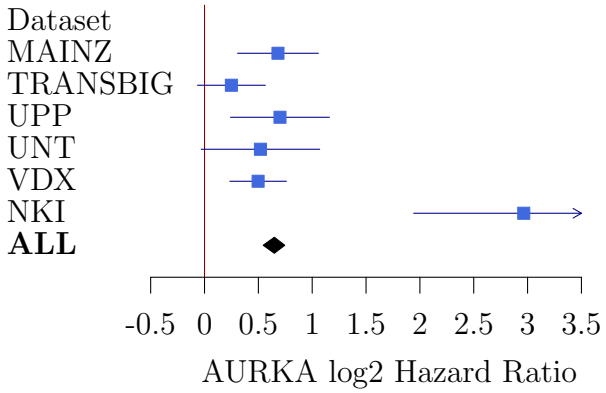
```
> tt <- rbind(dindexall.mainz.small[3, ], dindexall.transbig.small[3,
+     ], dindexall.upp.small[3, ], dindexall.unt.small[3, ], dindexall.vdx.small[3,
+     ], dindexall.nki.small[3, ], as.numeric(cdindex[3, ]))
> rownames(tt) <- datasetList
> tt <- as.data.frame(tt)
> tt <- log2(tt)
> labeltext <- cbind(c("Dataset", datasetList), c(rep(mybigspace,
+     8)))
> bs <- rep(0.5, nrow(labeltext))
> forestplot.surv(labeltext = labeltext, mean = c(NA, tt$dindex),
+     lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0,
+     align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(-0.5,
+         2, 0.5), xlab = paste("AURKA log2 D.index", myspace,
+         sep = ""), col = meta.colors(box = "royalblue", line = "darkblue",
+         zero = "darkred"), box.size = bs, clip = c(-0.25, 2.5),
+     is.summary = (c(rep(FALSE, 7), TRUE)))
```



And at last the hazard ratio for the gene AURKA in each dataset and the combined estimation over all datasets.

```
> tt <- rbind(hratio.mainz.small[3, ], hratio.transbig.small[3,
+     ], hratio.upp.small[3, ], hratio.unt.small[3, ], hratio.vdx.small[3,
+     ], hratio.nki.small[3, ], as.numeric(chratio[3, ]))
> rownames(tt) <- datasetList
> tt <- as.data.frame(tt)
> tt <- log2(tt)
> labeltext <- cbind(c("Dataset", datasetList), c(rep(myspace,
+     8)))
> bs <- rep(0.5, nrow(labeltext))
```

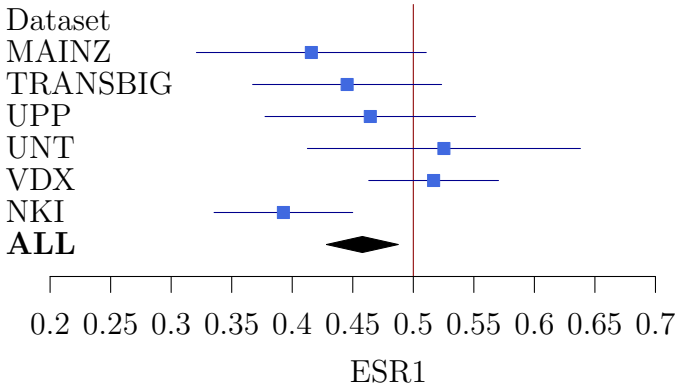
```
> forestplot.surv(labeltext = labeltext, mean = c(NA, tt$hratio),
+     lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0,
+     align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(-0.5,
+         3.5, 0.5), xlab = paste("AURKA log2 Hazard Ratio", myspace,
+         sep = ""), col = meta.colors(box = "royalblue", line = "darkblue",
+         zero = "darkred"), box.size = bs, clip = c(-0.5, 3.5),
+     is.summary = (c(rep(FALSE, 7), TRUE)))
```

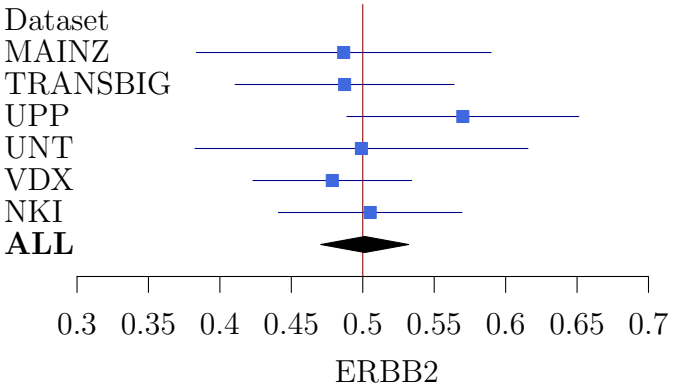


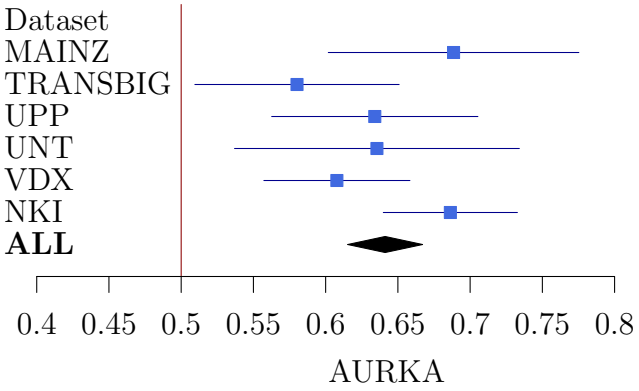
The following small loop shows an easy way for creating forestplots showing the concordance indices for a single gene for all datasets and the combined estimation over all datasets. The same can be done for the D indices and hazard ratios.

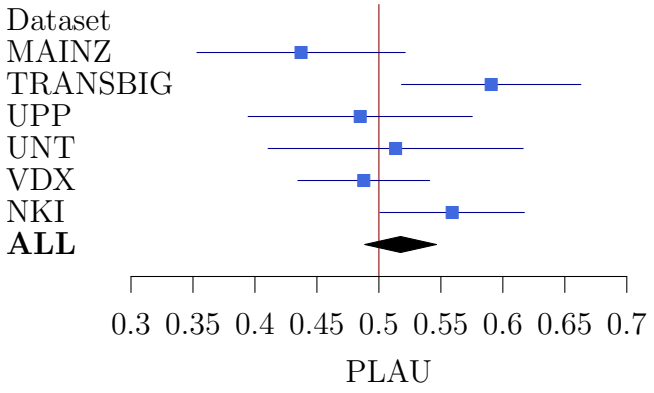
```
> for (i in 1:length(gsList)) {
```

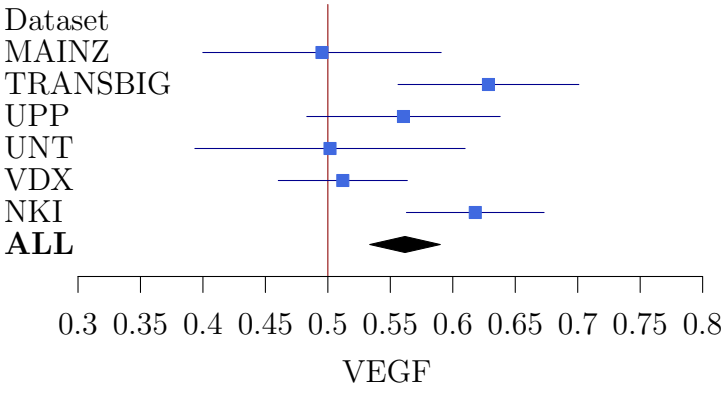
```
+     ## par(mfrow=c(3,3))
+     myspace <- " "
+     tt <- rbind(cindexall.mainz.small[i, ], cindexall.transbig.small[i,
+         ], cindexall.upp.small[i, ], cindexall.unt.small[i, ],
+         cindexall.vdx.small[i, ], cindexall.nki.small[i, ], as.numeric(ccindex[i,
+             ]))
+     rownames(tt) <- datasetList
+     tt <- as.data.frame(tt)
+     labeltext <- cbind(c("Dataset", datasetList), c(rep(myspace,
+         8)))
+     bs <- rep(0.5, nrow(labeltext))
+     r.mean <- c(NA, tt$cindex)
+     r.lower <- c(NA, tt$cindex + qnorm(0.025, lower.tail = TRUE) *
+         tt$cindex.se)
+     r.upper <- c(NA, tt$cindex + qnorm(0.025, lower.tail = FALSE) *
+         tt$cindex.se)
+     x.ticks.lower <- (floor((min(r.mean, na.rm = TRUE) - 0.1) *
+         10)/10)
+     x.ticks.upper <- (floor((max(r.mean, na.rm = TRUE) + 0.2) *
+         10)/10)
+     forestplot.surv(labeltext = labeltext, mean = c(NA, tt$cindex),
+         lower = c(NA, tt$lower), upper = c(NA, tt$upper), zero = 0.5,
+         align = c("l"), graphwidth = unit(2, "inches"), x.ticks = seq(x.ticks.lower,
+             x.ticks.upper, 0.05), xlab = paste(gsList[i], myspace,
+             sep = ""), col = meta.colors(box = "royalblue", line = "dark-
blue",
+             zero = "darkred"), box.size = bs, clip = c(0.3, 0.8),
+         is.summary = (c(rep(FALSE, 7), TRUE)))
+     ## title(paste('cindex forestplot, ', gsList[i]))
+ }
```

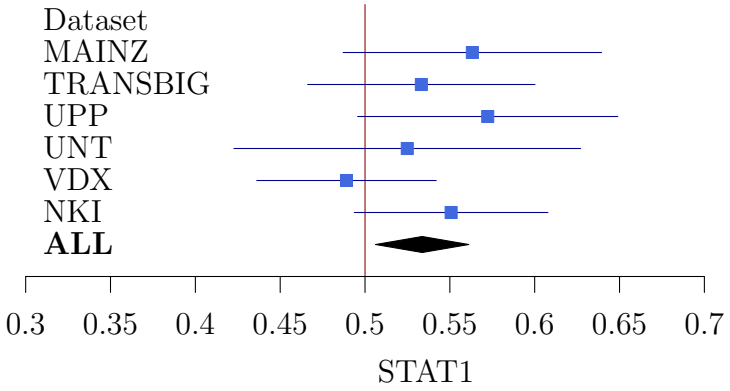

Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.2 0.25 0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7
ESR1

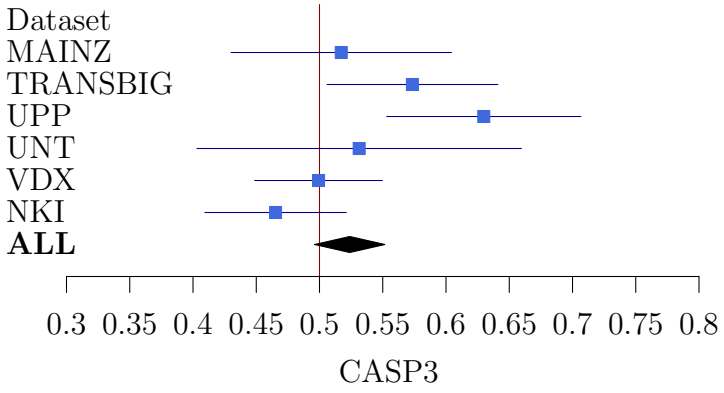

Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.3
0.35
0.4
0.45
0.5
0.55
0.6
0.65
0.7
ERBB2


Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.4
0.45
0.5
0.55
0.6
0.65
0.7
0.75
0.8
AURKA


Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7
PLAU


Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7 0.75 0.8
VEGF


Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.3
0.35
0.4
0.45
0.5
0.55
0.6
0.65
0.7
STAT1


Dataset
MAINZ
TRANSBIG
UPP
UNT
VDX
NKI
ALL
0.3 0.35 0.4 0.45 0.5 0.55 0.6 0.65 0.7 0.75 0.8
CASP3

3 Functions in survcomp

Function	Description
D.index	Function to compute the D index
censor.time	Function to artificially censor survival data
cindex.comp	Function to compare two concordance indices
cindex.comp.meta	Function to compare two concordance indices
combine.est	Function to combine estimates
combine.test	Function to combine probabilities
concordance.index	Function to compute the concordance index for survival or binary class prediction
cvpl	Function to compute the CVPL
dindex.comp	Function to compare two D indices
dindex.comp.meta	Function to compare two concordance indices
fisherz	Function to compute Fisher z transformation
forestplot.surv	Function to create a Forest Plot
getsurv2	Function to retrieve the survival probabilities at a specific point in time
hazard.ratio	Function to estimate the hazard ratio through Cox regression
hr.comp	Function to statistically compare two hazard ratios
hr.comp.meta	Function to compare two concordance indices
hr.comp2	Function to statistically compare two hazard ratios (alternative interface)
iauc.comp	Function to compare two IAUCs through time-dependent ROC curves
ibsc.comp	Function to compare two IBSCs
km.coxph.plot	Function to plot several Kaplan-Meier survival curves
logpl	Function to compute the log partial likelihood of a Cox model
no.at.risk	Function to compute the number of individuals at risk
sbrier.score2proba	Function to compute the BSCs from a risk score, for all the times of event occurrence
score2proba	Function to compute the survival probabilities from a risk score
survcomp-package	Performance Assessment and Comparison for Survival Analysis
td.sens.spec	Function to compute sensitivity and specificity for a binary classification of survival data
tdrocc	Function to compute time-dependent ROC curves
test.hetero.est	Function to test the heterogeneity of set of probabilities
test.hetero.test	Function to test the heterogeneity of set of probabilities

References

[1] Cochrane, W. G.: The combination of estimates from different experiments. *Biometrics*, **10**, 101-129. 1954.

[2] Whitlock, M. C.: Combining probability from independent tests: the weighted Z-method is superior to Fisher's approach. *J. Evol. Biol.*, **18**, 1368-1373. 2005.

[3] Heagerty, P. J. and Lumley, T. L. and Pepe, M. S.: Time-Dependent ROC Curves for Censored Survival Data and a Diagnostic Marker. *Biometrics*, **56**, 337-344. 2000.

[4] Efron, B. and Tibshirani, R.: The Bootstrap Method for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, **1**, 1-35. 1986.

[5] Becker, R. A., Chambers, J. M. and Wilks, A. R.: The New S Language. *Wadsworth & Brooks/Cole*, 1988.

[6] Cox, D. R.: Regression Models and Life Tables. *Journal of the Royal Statistical Society Series B*, **34**, 187-220. 1972.

[7] Andersen, P. K. and Borgan, O. and Gill, R. D. and Keiding, N.: Statistical Models Based on Counting Processes *Springer*, 1993.

[8] Brier, G. W.: Verification of forecasts expressed in terms of probabilities. *Monthly Weather Review*, **78**, 1-3. 1950.

[9] Graf, E. and Schmoor, C. and Sauerbrei, W. and Schumacher, M.: Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine*, **18**, 2529-2545. 1999.

[10] Wilcoxon, F.: Individual comparisons by ranking methods. *Biometrics Bulletin*, **1**, 80-83. 1945.

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

[12] Student: The Probable Error of a Mean. *Biometrika*, **6**, 1-25. 1908.

[13] R. A. Fisher: Frequency distribution of the values of the correlation coefficient in samples of an indefinitely large population. *Biometrika*, **10**, 507-521. 1915.

[14] Verweij PJM. and van Houwelingen H: Cross-validation in survival analysis. *Statistics in Medicine*, **12**, 2305-2314. 1993.

[15] van Houwelingen H, Bruinsma T, Hart AA, van't Veer LJ and Wessels LFA: Cross-validated Cox regression on microarray gene expression data. *tatistics in Medicine*, **25**, 3201-3216. 2006.

[16] Harrel Jr, F. E. and Lee, K. L. and Mark, D. B.: Tutorial in biostatistics: multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine*, **15**, 361-387. 1996.

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

[18] Royston, P. and Sauerbrei, W.: A new measure of prognostic separation in survival data. *Statistics in Medicine*, **23**, 723-748. 2004.