genefu: a package for breast cancer gene expression analysis

Benjamin Haibe-Kains[1], Markus Schröder[2], Christos Sotiriou[3], Gianluca Bontempi[4], and John Quackenbush[5,6]

[1] *Bioinformatics and Computational Genomics Laboratory, Princess Margaret Cancer Center, University Health Network, Toronto, Ontario, Canada*
[2] *UCD School of Biomolecular and Biomedical Science, Conway Institute, University College Dublin, Belfield, Dublin, Ireland*
[3] *Breast Cancer Translational Research Laboratory, Institut Jules Bordet, Université Libre de Bruxelles*
[4] *Machine Learning Group, Université Libre de Bruxelles*
[5] *Computational Biology and Functional Genomics Laboratory, Dana-Farber Cancer Institute, Harvard School of Public Health*
[6] *Center for Cancer Computational Biology, Dana-Farber Cancer Institute*

October 13, 2014

Contents

1 Introduction **1**

2 Case Study 1: comparing risk prediction models **1**

3 Session Info **10**

1 Introduction

The *genefu* package is providing relevant functions for gene expression analysis, especially in breast cancer.

2 Case Study 1: comparing risk prediction models

For computing the risk scores, estimates of the performance of the risk scores, combining the estimates and comparing the estimates we have to load the *genefu* and *survcomp* packages into the workspace. We also load the *xtable* package to display results insight this document.

```
> library(genefu)
> library(xtable)
> library(rmeta)
```

The five data sets that we use in the case study are publicly available as experimental data packages on Bioconductor.org. In particular we used:

breastCancerMAINZ: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerMAINZ.html`

breastCancerUPP: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerUPP.html`

breastCancerUNT: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerUNT.html`

breastCancerNKI: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerNKI.html`

breastCancerTRANSBIG: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerTRANSBIG.html`

We don't use the *breastCancerVDX* experimental package in the case study since it has been used as training data set for GENIUS [**?**]:

breastCancerVDX: `http://www.bioconductor.org/packages/release/data/experiment/html/breastCancerVDX.html`

These experimental data packages can be installed from Bioconductor version 2.8 or higher in R version 2.13.0 or higher. For the experimental data packages the commands for installing the data sets are:

```
> source("http://www.bioconductor.org/biocLite.R")
> biocLite("breastCancerMAINZ")
> biocLite("breastCancerTRANSBIG")
> biocLite("breastCancerUPP")
> biocLite("breastCancerUNT")
> biocLite("breastCancerNKI")
```

And for loading the data sets into the current workspace the commands are:

```
> library(breastCancerMAINZ)
> library(breastCancerTRANSBIG)
> library(breastCancerUPP)
> library(breastCancerUNT)
> library(breastCancerNKI)
```

Table 1: Detailed overview for the data sets used in the case study

Dataset	Patients [#]	ER+ [#]	HER2+ [#]	Age [years]	Grade [1/2/3]	Platform
MAINZ	200	155	23	25-90	29/136/35	HGU133A
TRANSBIG	198	123	35	24-60	30/83/83	HGU133A
UPP	251	175	46	28-93	67/128/54	HGU133AB
UNT	137	94	21	24-73	32/51/29	HGU133AB
NKI	337	212	53	26-62	79/109/149	Agilent
Overall	1123	759	178	25-73	237/507/350	Affy/Agilent

Table 1 shows an overview of the data sets and the patients. From those 1123 breast cancer patients we selected only the patients that are node negative and didn't receive any treatment (except local radiotherapy), which results in 722 patients.

Since there are duplicated patients in the five data sets, we have to identify the duplicated patients and we subsequently store them in a vector. We then compute the risk scores for NPI, AURKA, GGI and GENIUS with the `npi()`, `sig.score()`, `ggi()` and `genius()` functions within *genefu*, respectively.

```
> dn <- c("transbig", "unt", "upp", "mainz", "nki")
> dn.platform <- c("affy", "affy", "affy", "affy", "agilent")
> res <- ddemo.all <- ddemo.coln <- NULL
> for(i in 1:length(dn)) {
+
+   ## load dataset
+   dd <- get(data(list=dn[i]))
+
+   ddata <- t(exprs(dd))
+   ddemo <- phenoData(dd)@data
+   dannot <- featureData(dd)@data
+   ddemo.all <- c(ddemo.all, list(ddemo))
+   if(is.null(ddemo.coln)) { ddemo.coln <- colnames(ddemo) } else { ddemo.coln <- interse
+   rest <- NULL
+
+   ## NPI
+   ss <- ddemo[ , "size"]
+   gg <- ddemo[ , "grade"]
+   nn <- rep(NA, nrow(ddemo))
+   nn[complete.cases(ddemo[ , "node"]) & ddemo[ , "node"] == 0] <- 1
+   nn[complete.cases(ddemo[ , "node"]) & ddemo[ , "node"] == 1] <- 3
+   names(ss) <- names(gg) <- names(nn) <- rownames(ddemo)
+   rest <- cbind(rest, "NPI"=npi(size=ss, grade=gg, node=nn, na.rm=TRUE)$score)
+
+   ## AURKA
+   ## if affy platform consider the probe published in Desmedt et al., CCR, 2008
```

```
+   if(dn.platform[i] == "affy") { domap <- FALSE } else { domap <- TRUE }
+   modt <- scmgene.robust$mod$AURKA
+   ## if agilent platform consider the probe published in Desmedt et al., CCR, 2008
+   if(dn.platform[i] == "agilent") {
+     domap <- FALSE
+     modt[ , "probe"] <- "NM_003600"
+   }
+   rest <- cbind(rest, "AURKA"=sig.score(x=modt, data=ddata, annot=dannot, do.mapping=dom
+
+   ## GGI
+   if(dn.platform[i] == "affy") { domap <- FALSE } else { domap <- TRUE }
+   rest <- cbind(rest, "GGI"=ggi(data=ddata, annot=dannot, do.mapping=domap)$score)
+   ## GENIUS
+   if(dn.platform[i] == "affy") { domap <- FALSE } else { domap <- TRUE }
+   rest <- cbind(rest, "GENIUS"=genius(data=ddata, annot=dannot, do.mapping=domap)$score)
+   res <- rbind(res, rest)
+ }
> names(ddemo.all) <- dn
```

For further analysis and handling of the data we store all information in one object. We also remove the duplicated patients from the analysis and take only those patients into account, that have complete information for nodal, survival and treatment status.

```
> ddemot <- NULL
> for(i in 1:length(ddemo.all)) {
+   ddemot <- rbind(ddemot, ddemo.all[[i]][ , ddemo.coln, drop=FALSE])
+ }
> res[complete.cases(ddemot[ ,"dataset"]) & ddemot[ ,"dataset"] == "VDX", "GENIUS"] <- NA
> ## select only untreated node-negative patients with all risk predictions
> myx <- complete.cases(res, ddemot[ , c("node", "treatment")]) & ddemot[ , "treatment"] =
> res <- res[myx, , drop=FALSE]
> ddemot <- ddemot[myx, , drop=FALSE]
```

To compare the risk score performances, we compute the concordance index[1]:

```
> cc.res <- complete.cases(res)
> datasetList <- c("MAINZ","TRANSBIG","UPP","UNT","NKI")
> riskPList <- c("NPI", "AURKA", "GGI", "GENIUS")
> setT <- setE <- NULL
> resMatrix <- as.list(NULL)
> for(i in datasetList){
+   dataset.only <- ddemot[,"dataset"] == i
+   patientsAll <- cc.res & dataset.only
+
```

[1]The same analysis could be performed with D index and hazard ratio by using the functions `D.index` and `hazard.ratio` from the *survcomp* respectively

```
+   ## set type of available survival data
+   if(i == "UPP") {
+     setT <- "t.rfs"
+     setE <- "e.rfs"
+   } else {
+     setT <- "t.dmfs"
+     setE <- "e.dmfs"
+   }
+
+   ## cindex computation
+   cindexN <- t(apply(X=t(res[patientsAll,"NPI"]), 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=ddemot[patientsAll,setT], z=ddemot[patientsAll, setE]))
+
+   cindexA <- t(apply(X=t(res[patientsAll,"AURKA"]), 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=ddemot[patientsAll,setT], z=ddemot[patientsAll, setE]))
+
+   cindexM <- t(apply(X=t(res[patientsAll,"GGI"]), 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=ddemot[patientsAll, setT], z=ddemot[patientsAll, setE]))
+
+   cindexG <- t(apply(X=t(res[patientsAll,"GENIUS"]), 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=ddemot[patientsAll, setT], z=ddemot[patientsAll, setE]))
+
+
+   resMatrix[["NPI"]] <- rbind(resMatrix[["NPI"]], cindexN)
+   resMatrix[["AURKA"]] <- rbind(resMatrix[["AURKA"]], cindexA)
+   resMatrix[["GGI"]] <- rbind(resMatrix[["GGI"]], cindexM)
+   resMatrix[["GENIUS"]] <- rbind(resMatrix[["GENIUS"]], cindexG)
+ }
```

Using a random-effects model we combine the dataset-specific performance estimated into overall estimates for each risk prediction model:

```
> for(i in names(resMatrix)){
+   ceData <- combine.est(x=resMatrix[[i]][,"cindex"], x.se=resMatrix[[i]][,"cindex.se"], 
+   cLower <- ceData$estimate + qnorm(0.025, lower.tail=TRUE) * ceData$se
+   cUpper <- ceData$estimate + qnorm(0.025, lower.tail=FALSE) * ceData$se
+
+   cindexO <- cbind("cindex"=ceData$estimate, "cindex.se"=ceData$se, "lower"=cLower, "upp
```

```
+   resMatrix[[i]] <- rbind(resMatrix[[i]], cindexO)
+   rownames(resMatrix[[i]]) <- c(datasetList, "Overall")
+ }
```

In order to compare the different risk prediction models we compute one-sided p-values of the meta-estimates:

```
> pv <- sapply(resMatrix, function(x) { return(x["Overall", c("cindex","cindex.se")]) })
> pv <- apply(pv, 2, function(x) { return(pnorm((x[1] - 0.5) / x[2], lower.tail=x[1] < 0.5
> printPV <- matrix(pv,ncol=4)
> rownames(printPV) <- "P-value"
> colnames(printPV) <- names(pv)

> xtable(printPV, digits=c(0, rep(-1,ncol(printPV))))
```

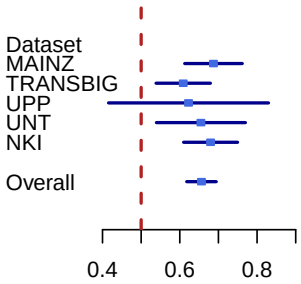
	NPI	AURKA	GGI	GENIUS
P-value	5.5E-16	1.6E-08	2.2E-15	2.9E-24

The following five figures represent the risk score performances measured by the concordance index for NPI, AURKA, GGI and GENIUS. The last figure represents the overall estimates.

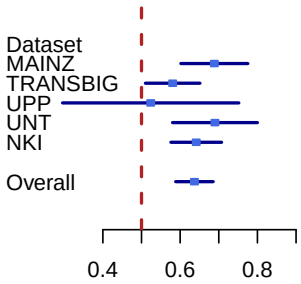
```
> par(mfrow=c(2,2))
> datasetListF <- c("MAINZ","TRANSBIG","UPP","UNT","NKI", NA, "Overall")
> myspace <- "   "
> ## NPI Forestplot
> tt <- rbind(resMatrix[["NPI"]][1:5,],
+           "NA"=NA,
+           "Overall"=resMatrix[["NPI"]][6,])
> tt <- as.data.frame(tt)
> labeltext <- cbind(c("Dataset", datasetListF))
> r.mean <- c(NA,tt$cindex)
> r.lower <- c(NA,tt$lower)
> r.upper <- c(NA,tt$upper)
> metaplot.surv(mn=r.mean, lower=r.lower, upper=r.upper, labels=labeltext, xlim=c(0.4,0.9)
> #@
> #
> #<<forestplotAURKA,fig=TRUE,echo=FALSE>>=
> ## AURKA Forestplot
> tt <- rbind(resMatrix[["AURKA"]][1:5,],
+           "NA"=NA,
+           "Overall"=resMatrix[["AURKA"]][6,])
> tt <- as.data.frame(tt)
> labeltext <- cbind(c("Dataset", datasetListF))
> r.mean <- c(NA,tt$cindex)
> r.lower <- c(NA,tt$lower)
```

```
> r.upper <- c(NA,tt$upper)
> metaplot.surv(mn=r.mean, lower=r.lower, upper=r.upper, labels=labeltext, xlim=c(0.4,0.9)
> #@
> #
> #<<forestplotGGI,fig=TRUE,echo=FALSE>>=
> ## GGI Forestplot
> tt <- rbind(resMatrix[["GGI"]][1:5,],
+           "NA"=NA,
+           "Overall"=resMatrix[["GGI"]][6,])
> tt <- as.data.frame(tt)
> labeltext <- cbind(c("Dataset", datasetListF))
> r.mean <- c(NA,tt$cindex)
> r.lower <- c(NA,tt$lower)
> r.upper <- c(NA,tt$upper)
> metaplot.surv(mn=r.mean, lower=r.lower, upper=r.upper, labels=labeltext, xlim=c(0.4,0.9)
> #@
> #
> #<<forestplotGENIUS,fig=TRUE,echo=FALSE>>=
> ## GENIUS Forestplot
> tt <- rbind(resMatrix[["GENIUS"]][1:5,],
+           "NA"=NA,
+           "Overall"=resMatrix[["GENIUS"]][6,])
> tt <- as.data.frame(tt)
> labeltext <- cbind(c("Dataset", datasetListF))
> r.mean <- c(NA,tt$cindex)
> r.lower <- c(NA,tt$lower)
> r.upper <- c(NA,tt$upper)
> metaplot.surv(mn=r.mean, lower=r.lower, upper=r.upper, labels=labeltext, xlim=c(0.4,0.9)
```

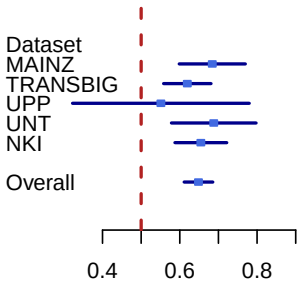
NPI Concordance Index



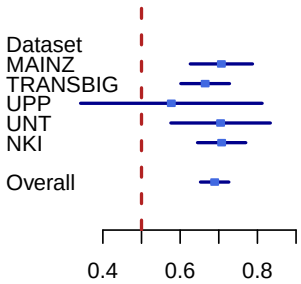
AURKA Concordance Index

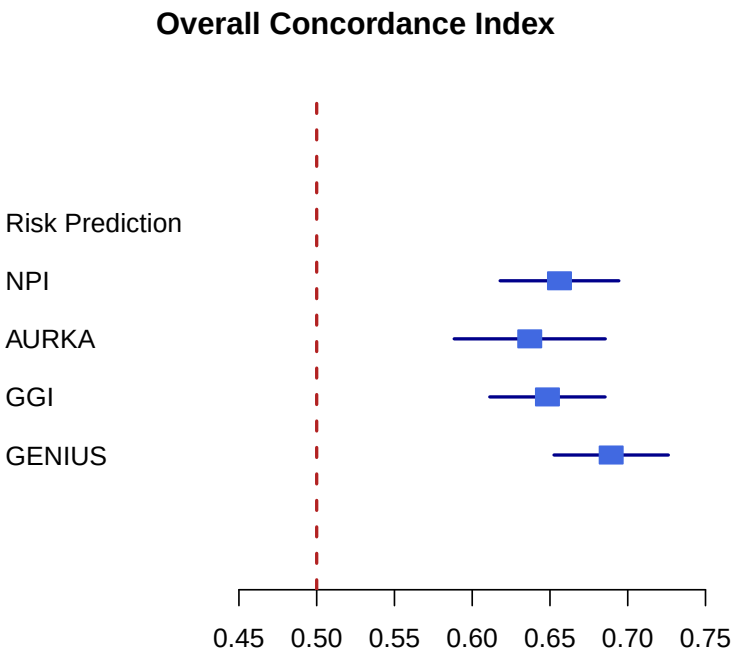


GGI Concordance Index



GENIUS Concordance Index





In order to assess the difference between the risk scores, we compute the concordance indices with their p-values and compare the estimates with the `cindex.comp.meta` with a paired student t test.

Table 2 displays the for multiple testing uncorrected p-values for the comparison of the different methods:

```
> xtable(ccmData, digits=c(0, rep(-1,ncol(ccmData))))
```

	NPI	AURKA	GGI	GENIUS
NPI	1.0E+00	2.5E-01	3.7E-01	9.1E-01
AURKA	7.5E-01	1.0E+00	7.0E-01	9.8E-01
GGI	6.3E-01	3.0E-01	1.0E+00	9.7E-01
GENIUS	9.0E-02	2.4E-02	3.1E-02	1.0E+00

We correct the p-value with Holms method:

```
> ccmDataPval <- matrix(p.adjust(data.matrix(ccmData), method="holm"),ncol=4,dimnames=list
```

Table 3 displays the corrected p-values:

```
> xtable(ccmDataPval, digits=c(0, rep(-1,ncol(ccmDataPval))))
```

	NPI	AURKA	GGI	GENIUS
NPI	1.0E+00	1.0E+00	1.0E+00	1.0E+00
AURKA	1.0E+00	1.0E+00	1.0E+00	1.0E+00
GGI	1.0E+00	1.0E+00	1.0E+00	1.0E+00
GENIUS	1.0E+00	3.8E-01	4.7E-01	1.0E+00

3 Session Info

- R version 3.1.1 Patched (2014-09-25 r66681), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, splines, stats, utils
- Other packages: Biobase 2.26.0, BiocGenerics 0.12.0, biomaRt 2.22.0, breastCancerMAINZ 1.3.0, breastCancerNKI 1.3.0, breastCancerTRANSBIG 1.3.0, breastCancerUNT 1.3.0, breastCancerUPP 1.3.0, genefu 1.16.0, mclust 4.4, prodlim 1.4.5, rmeta 2.16, survcomp 1.16.0, survival 2.37-7, xtable 1.7-4
- Loaded via a namespace (and not attached): AnnotationDbi 1.28.0, DBI 0.3.1, GenomeInfoDb 1.2.0, IRanges 2.0.0, KernSmooth 2.23-13, RCurl 1.95-4.3, RSQLite 0.11.4, S4Vectors 0.4.0, SuppDists 1.1-9.1, XML 3.98-1.1, amap 0.8-12, bootstrap 2014.4, lava 1.2.6, stats4 3.1.1, survivalROC 1.0.3, tools 3.1.1