Messina Experiment 2: Messina vs Classical on APGI

March 8, 2015

1 Preparation

```
library(messina)

## Loading required package:  survival
## Loading required package:  splines
## Loading required package:  methods

library(plyr)
library(reshape2)
library(ggplot2)
```

2 Data preparation

```
load("../biosurv/data/07_data_for_SIS.rda")
x = x.diag_dsd
y = y.diag_dsd
samps = samps.diag_dsd

temp = NA
temp = ls()
rm(list = temp[!(temp %in% c("x", "y", "samps"))])
```

3 Detectors

```
# For ncuts = 1, this equates to median cut.
detector_multicut = function(x, y, ncuts = 10, correct = "none")
{
        if (ncuts == 1) { correct = "none" }
        aaply(x, 1, function(x1) {
                cutpoints = quantile(x1, probs = (1:ncuts)/(ncuts + 1))
                pvals = sapply(cutpoints, function(c) {
                        x1c = x1 > c
                        test = survdiff(y ~ x1c)
                        pval = pchisq(test$chisq, df = 1, lower.tail = FALSE)
                        pval
                })
```

```
                pvals = p.adjust(pvals, correct)
                pvals[is.na(pvals)] = 1
                c(min(pvals), cutpoints[which.min(pvals)])
        }, .parallel = TRUE)
}


# A 'best-approach' to all-cutoff testing
detector_maxstat = function(x, y, pmethod = "HL")
{
        require(maxstat)

        aaply(x, 1, function(x1) {
                temp.data = data.frame(x1 = x1, time = y[,1], event = y[,2])
                test = try(maxstat.test(Surv(time, event) ~ x1, data = temp.data, smethod = "LogRank", p
                if (class(test) == "try-error")
                {
                        return(NA, NA)
                }
                c(test$p.value, test$estimate)
        }, .parallel = TRUE)
}
```

4 The Experiment

```
library(doMC)

## Loading required package:  foreach
## Loading required package:  iterators
## Loading required package:  parallel

registerDoMC(32)

fit.messina = messinaSurv(x, y, messinaSurvObj.CoxCoef(log(2)), parallel = TRUE)

## Performance bootstrapping...
## Final training...

fit.medcut = detector_multicut(x, y, ncuts = 1)
fit.10cutHolm = detector_multicut(x, y, ncuts = 10, correct = "holm")
fit.maxstat = detector_maxstat(x, y)

## Loading required package:  maxstat

det.messina = fit.messina@fits@summary$passed
det.medcut = fit.medcut[,1] < 0.05 & !is.na(fit.medcut[,1])
det.10cutHolm = fit.10cutHolm[,1] < 0.05 & !is.na(fit.medcut[,1])
det.maxstat = fit.maxstat[,1] < 0.05 & !is.na(fit.medcut[,1])

thresh.messina = fit.messina@fits@summary$threshold
thresh.medcut = fit.medcut[,2]
thresh.10cutHolm = fit.10cutHolm[,2]
thresh.maxstat = fit.maxstat[,2]
```

```
thresh.messina[det.messina == FALSE] = NA
thresh.medcut[det.medcut == FALSE] = NA
thresh.10cutHolm[det.10cutHolm == FALSE] = NA
thresh.maxstat[det.maxstat == FALSE] = NA
```

```
load("../biosurv/data/15_validation.rda")
```

```
save.image("03_surv_exp2.rda")
```