

# Survival Ensembles

Torsten Hothorn<sup>1,\*</sup>, Peter Bühlmann<sup>2</sup>, Sandrine Dudoit<sup>3</sup>,  
Annette Molinaro<sup>4</sup> and Mark J. van der Laan<sup>3</sup>

<sup>1</sup>Institut für Statistik  
Ludwig-Maximilians-Universität München  
Ludwigstraße 33, D-80539 München, Germany  
Tel: ++49-9131-8522707  
Fax: ++49-9131-8525740  
`Torsten.Hothorn@R-project.org`

<sup>2</sup>Seminar für Statistik, ETH Zürich, CH-8032 Zürich, Switzerland  
`buhlmann@stat.math.ethz.ch`

<sup>3</sup>Division of Biostatistics, University of California, Berkeley  
140 Earl Warren Hall, #7360, Berkeley, CA 94720-7360, USA  
`sandrine@stat.Berkeley.EDU`  
`laan@stat.Berkeley.EDU`

<sup>4</sup>Division of Biostatistics, Epidemiology and Public Health  
Yale University School of Medicine, 206 LEPH  
60 College Street PO Box 208034, New Haven CT 06520-8034  
`annette.molinaro@yale.edu`

## 1 Illustrations and Applications

This document reproduces the data analyses presented in [Hothorn et al. \(2006\)](#). For a description of the theory behind applications shown here we refer to the original manuscript. The results differ slightly due to technical changes or bug-fixes in **mboost** that have been implemented after the paper was printed.

### 1.1 Acute myeloid leukemia

**Data preprocessing** Compute IPC weights, define risk score and set up learning sample:

```
R> ### compute IPC weights
R> AMLw <- IPCweights(Surv(clinical$time, clinical$event))
```

```

R> ### risk score
R> risk <- rep(0, nrow(clinical))
R> rlev <- levels(clinical[, "Cytogenetic.group"])
R> risk[clinical[, "Cytogenetic.group"] %in% rlev[c(7,8,4)]] <- "low"
R> risk[clinical[, "Cytogenetic.group"] %in% rlev[c(5, 9)]] <- "intermediate"
R> risk[clinical[, "Cytogenetic.group"] %in% rlev[-c(4,5, 7,8,9)]] <- "high"
R> risk <- as.factor(risk)
R> ### set-up learning sample
R> AMLlearn <- cbind(clinical[, c("time", "Sex", "Age", "LDH", "WBC",
                                "FLT3.aberration.", "MLL.PTD", "Tx.Group."),
                    risk = risk,
                    iexpressions[, colnames(iexpressions) %in% selgenes[["Clone.ID"]]])
R> cc <- complete.cases(AMLlearn)
R> AMLlearn <- AMLlearn[AMLw > 0 & cc,]
R> AMLw <- AMLw[AMLw > 0 & cc]

```

**Model fitting** Fit random forest for censored data

```

R> ### controls for tree growing
R> ctrl <- ctree_control(testtype = "Teststatistic",
                        teststat = "maximum", mincriterion = .1, minsplit = 5)
R> ### was: cforest_control(mincriterion = 0.1, mtry = 5, minsplit = 5, ntree = 250)
R>
R> ### fit random forest for censored data (warnings are OK here)
R> AMLrf <- cforest(log(time) ~ ., data = AMLlearn, control = ctrl,
                    weights = AMLw, mtry = 5, ntree = 250,
                    perturb = list(replace = TRUE, fraction = 0.632))

```

and  $L_2$ Boosting for censored data

```

R> AMLl2b <- glmboost(I(log(time)) ~ ., data = AMLlearn, weights = AMLw,
                     control = boost_control(mstop = 5000))

```

Compute fitted values

```

R> ### restrict number of boosting iterations and inspect selected variables
R> AMLl2b <- AMLl2b[mstop(aic)]
R> cAML <- coef(AMLl2b)
R> cAML[abs(cAML) > 0]

```

| <i>(Intercept)</i> | <i>Age</i>           | <i>WBC</i>          |
|--------------------|----------------------|---------------------|
| 0.56429            | 0.00598              | -0.00562            |
| <i>MLL.PTDyes</i>  | <i>Tx.Group.AUTO</i> | <i>Tx.Group.Ind</i> |
| -0.31539           | 0.45430              | -2.12161            |
| `IMAGE:145643`     | `IMAGE:345601`       | `IMAGE:377560`      |
| 0.10626            | 0.00430              | 0.02757             |
| `IMAGE:2043415`    | `IMAGE:1584563`      | `IMAGE:347035`      |
| 0.05509            | -0.00259             | -0.00848            |
| `IMAGE:262695`     | `IMAGE:26418`        | `IMAGE:950479`      |
| 0.02696            | 0.00802              | 0.03717             |

```
R> ### AIC criterion  
R> plot(aic <- AIC(AML12b))
```

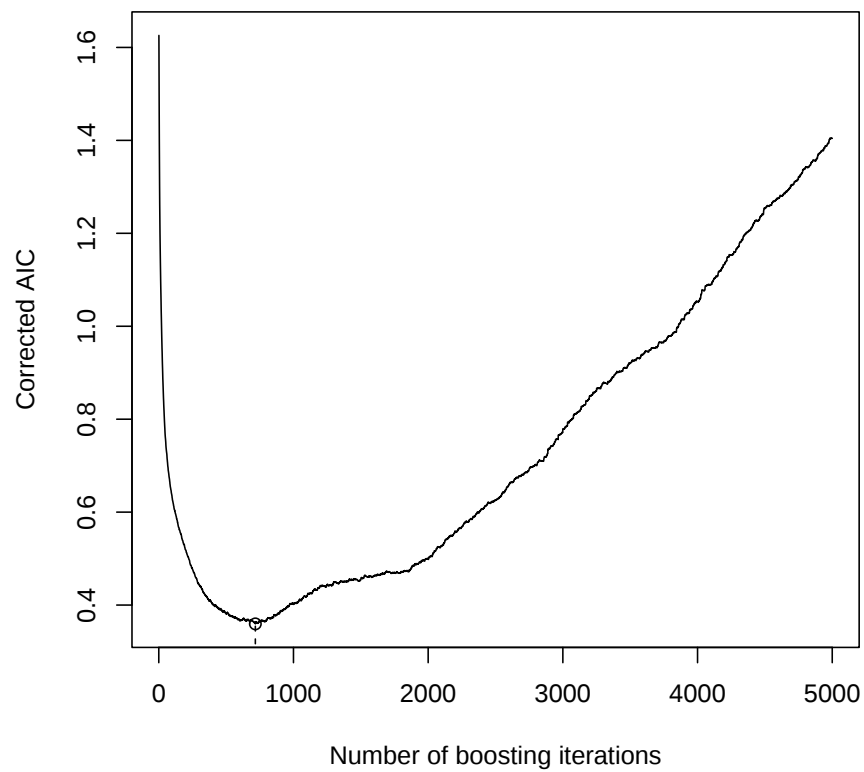


Figure 1: AIC criterion for AML data.

```

`IMAGE:1534700` `IMAGE:1472689` `IMAGE:1526826`
0.02836 0.02256 -0.02784
`IMAGE:786302` `IMAGE:243614` `IMAGE:417884`
0.04493 -0.05667 -0.02489
`IMAGE:1592006` `IMAGE:884333` `IMAGE:133273`
-0.03551 0.01281 0.02579
`IMAGE:950888` `IMAGE:809533` `IMAGE:49389`
0.03485 -0.05835 0.12105
`IMAGE:856174` `IMAGE:435036` `IMAGE:491751`
0.02054 0.06202 0.11555
`IMAGE:782835` `IMAGE:52930` `IMAGE:2545705`
-0.11085 -0.02452 -0.07884
`IMAGE:756405` `IMAGE:129032` `IMAGE:1610168`
0.00853 -0.11582 0.01380
`IMAGE:69002` `IMAGE:2019101` `IMAGE:1456160`
-0.27933 -0.09666 -0.10415
`IMAGE:2566064` `IMAGE:565083` `IMAGE:843028`
0.01547 0.18756 0.06983
`IMAGE:68794` `IMAGE:488505` `IMAGE:291756`
0.07614 0.27846 0.09949
`IMAGE:810801` `IMAGE:1702742` `IMAGE:380462`
0.04659 -0.01045 -0.09573
`IMAGE:154472` `IMAGE:302540` `IMAGE:135221`
-0.14547 0.01888 -0.03668
`IMAGE:1567220`
0.04851

```

```

R> ### fitted values
R> AMLprf <- predict(AMLrfr, newdata = AMLlearn)
R> AMLpb <- predict(AMLl2b, newdata = AMLlearn)

```

## 1.2 Node-positive breast cancer

**Data preprocessing** Compute IPC weights and set up learning sample:

```

R> ### attach data
R> data("GBSG2", package = "TH.data")
R> ### IPC weights
R> GBSG2w <- IPCweights(Surv(GBSG2$time, GBSG2$cens))
R> ### set-up learning sample
R> GBSG2learn <- cbind(GBSG2[, -which(names(GBSG2) %in% c("time", "cens"))],
+ ltime = log(GBSG2$time))
R> n <- nrow(GBSG2learn)

```

### Model fitting

```

R> ### linear model
R> LMmod <- lm(ltime ~ ., data = GBSG2learn, weights = GBSG2w)
R> LMerisk <- sum((GBSG2learn$ltime - predict(LMmod))^2 * GBSG2w) / n
R> ### regression tree
R> pos <- GBSG2w > 0

```

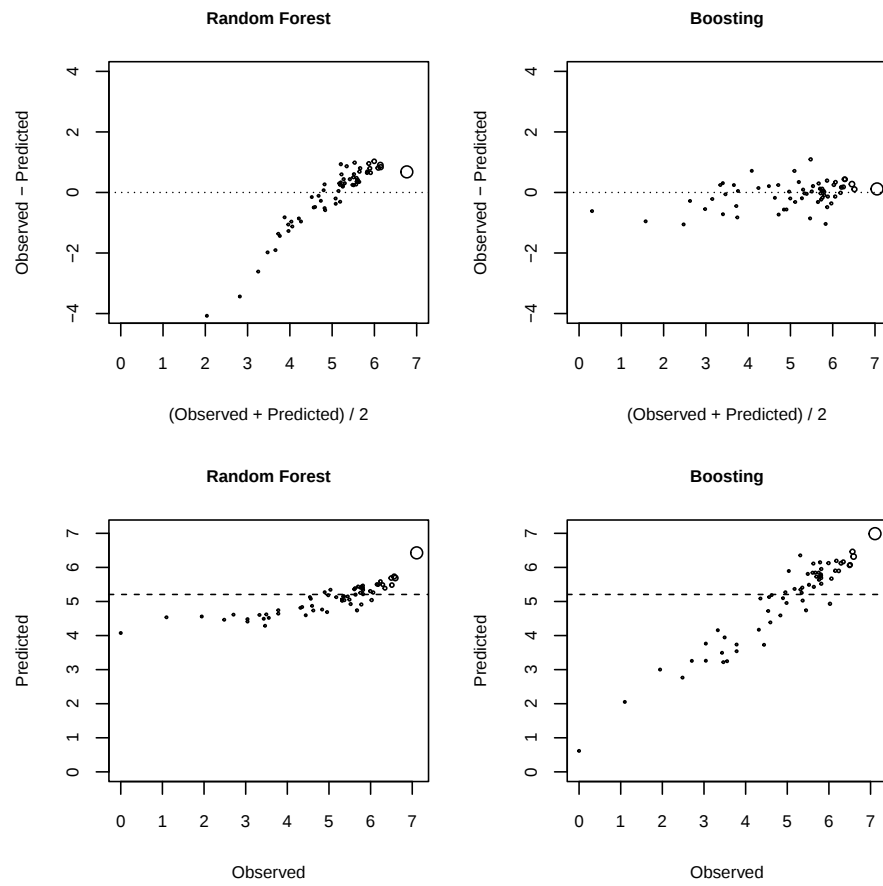


Figure 2: AML data: Reproduction of Figure 1.

```

R> TRmod <- rpart(ltime ~ . , data = GBSG2learn, weights = GBSG2w,
                  subset = pos)
R> TRerisk <- sum((GBSG2learn$ltime[pos] - predict(TRmod))^2*GBSG2w[pos]) / n
R> ### tree controls
R> ctrl <- ctree_control(testtype = "Teststatistic",
                        teststat = "maximum", mincriterion = qnorm(.95),
                        minsplit = 5)
R> ### was: cforest_control(mincriterion = qnorm(0.95), mtry = 5,
R> ###                                minsplit = 5, ntree = 100)
R>
R>
R> ### fit random forest for censored data (warnings are OK here)
R> RFmod <- cforest(ltime ~ . , data = GBSG2learn, weights = GBSG2w,
                   control = ctrl, mtry = 5, ntree = 100,
                   perturb = list(replace = TRUE,
                                fraction = 0.632 * sum(GBSG2w > 0)))
R> ### fit L2 boosting for censored data
R> L2Bmod <- glmboost(ltime ~ . , data = GBSG2learn, weights = GBSG2w,
                    control = boost_control(mstop = 250))
R> ### with Huber loss function
R> L2BHubermod <- glmboost(ltime ~ . , data = GBSG2learn, weights = GBSG2w,
                        family = Huber(d = log(2)))

```

Compute fitted values:

```

R> GBSG2Hp <- predict(L2BHubermod, newdata = GBSG2learn)
R> L2Berisk <- sum((GBSG2learn$ltime - predict(L2Bmod, newdata = GBSG2learn))^2*GBSG2w) / n
R> RFerisk <- sum((GBSG2learn$ltime - predict(RFmod, newdata = GBSG2learn))^2*GBSG2w) / n

```

```
R> plot(aic <- AIC(L2Bmod))
```

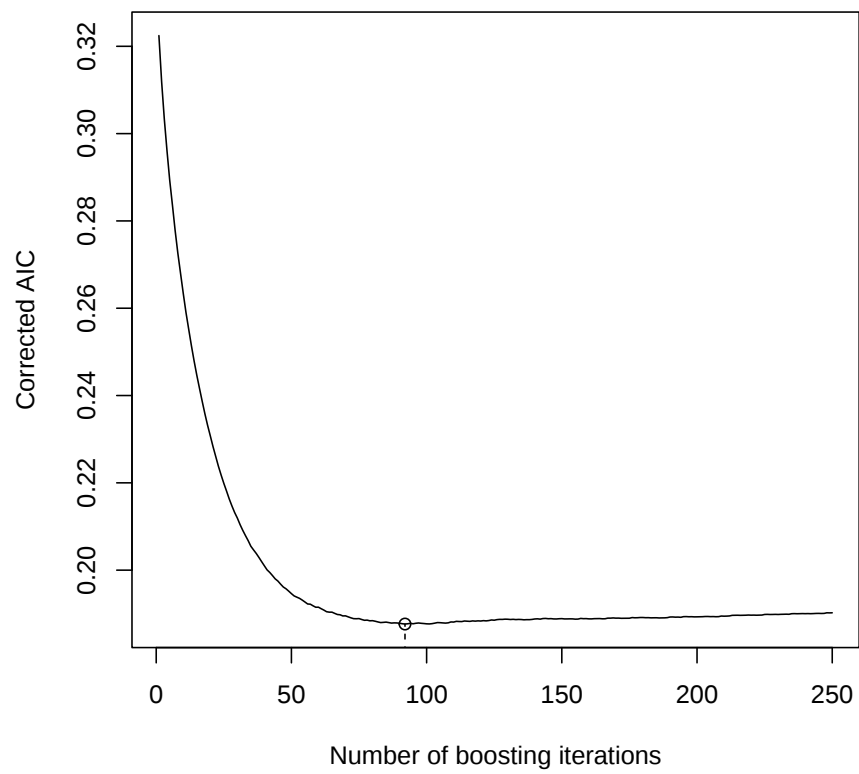


Figure 3: AIC criterion for GBSG2 data.

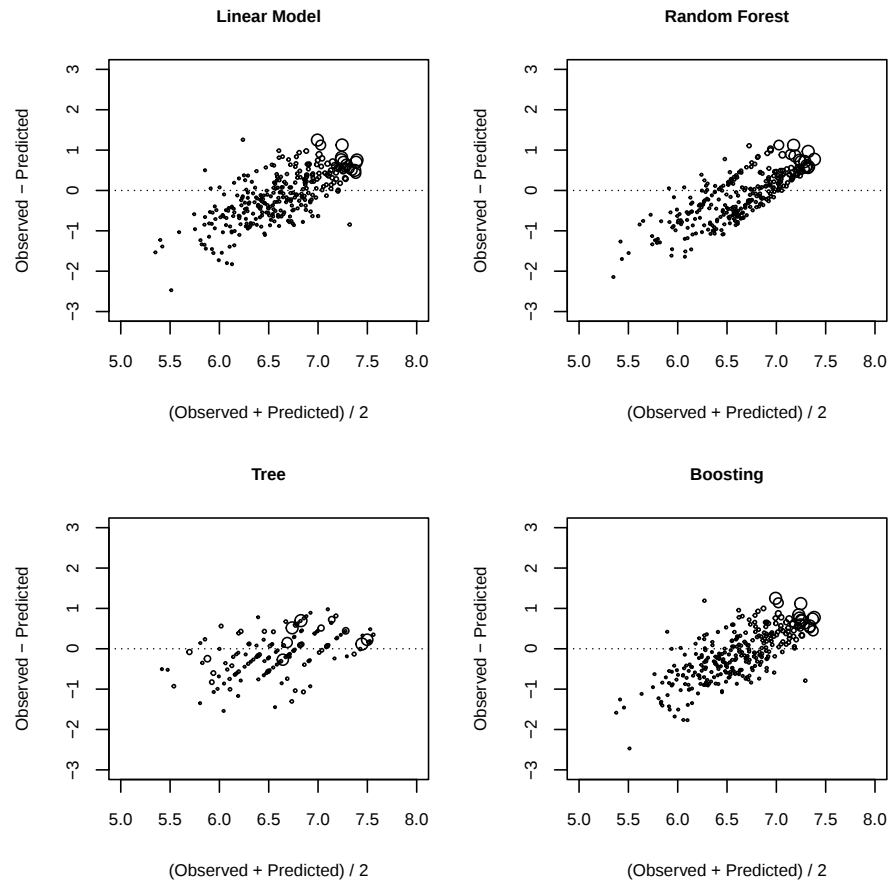


Figure 4: GBSG-2 data: Reproduction of Figure 3.

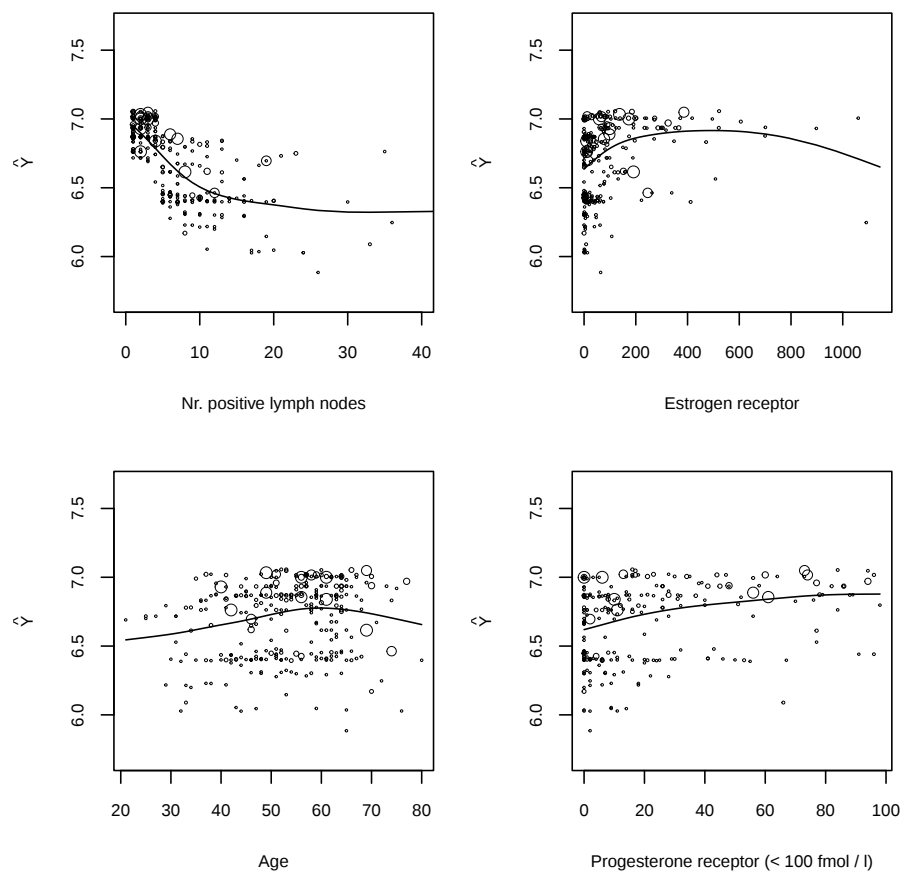


Figure 5: GBSG-2 data: Reproduction of Figure 5.

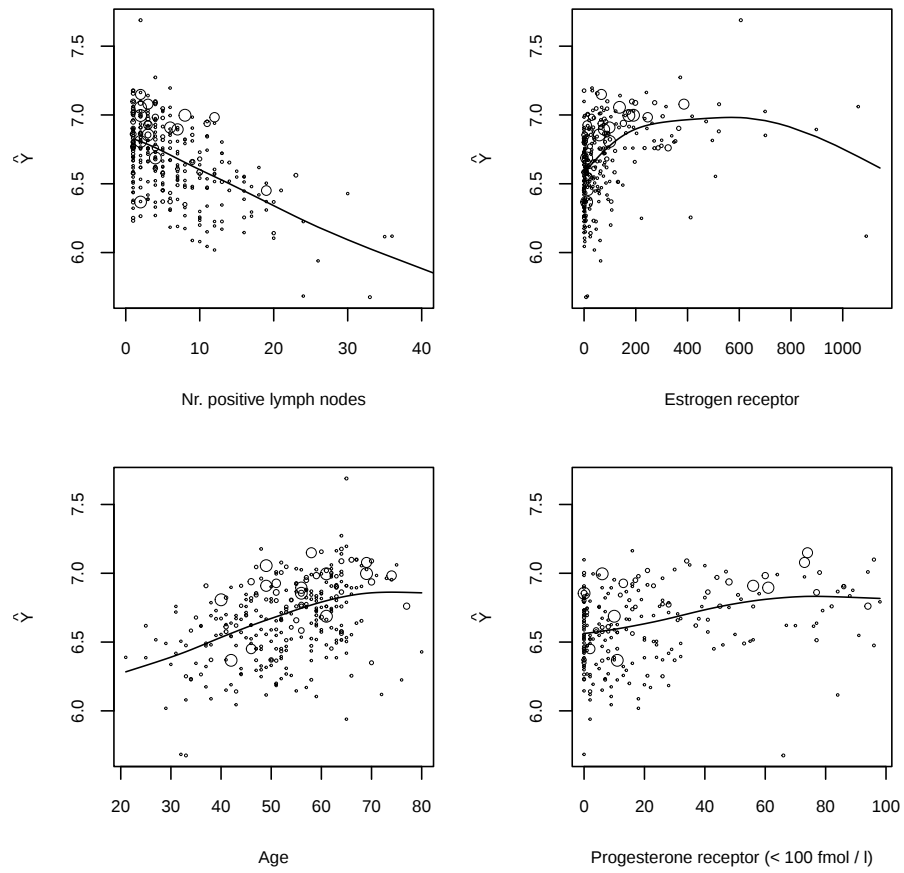


Figure 6: GBSG-2 data: Reproduction of Figure 6.

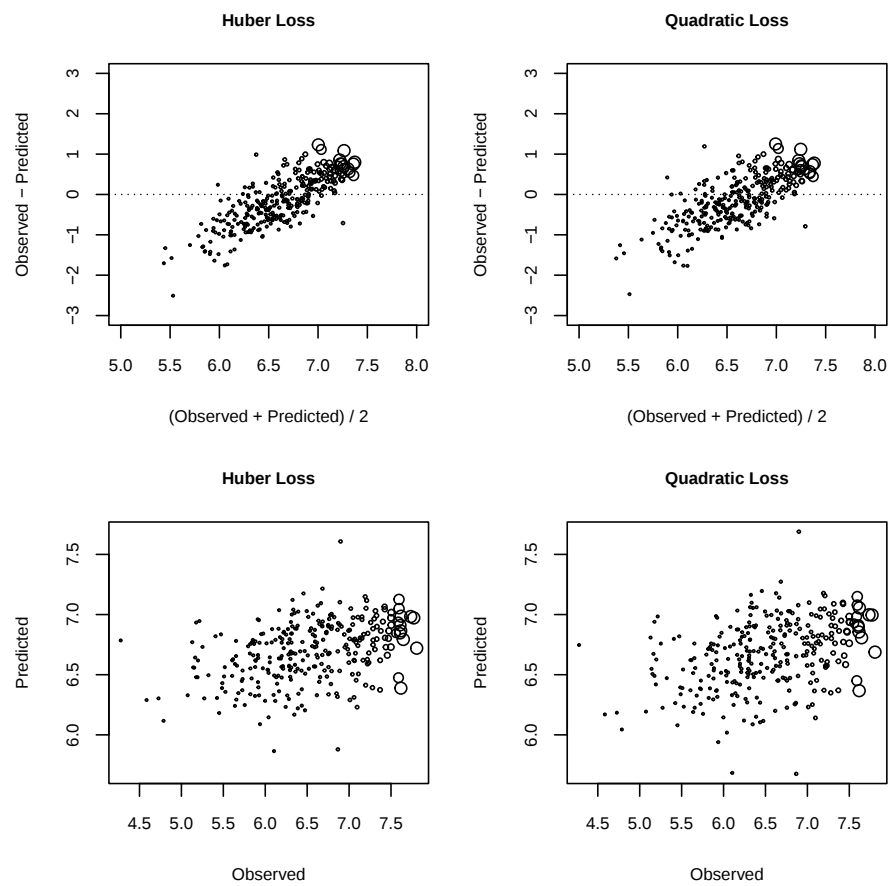


Figure 7: GBSG-2 data: Reproduction of Figure 7.

## References

Torsten Hothorn, Peter Bühlmann, Sandrine Dudoit, Annette Molinaro, and Mark J. van der Laan. Survival ensembles. *Biostatistics*, 7(3):355–373, 2006. doi: 10.1093/biostatistics/kxj011.