A Handbook of Statistical Analyses Using **R**

Brian S. Everitt and Torsten Hothorn

Recursive Partitioning: Large Companies and Glaucoma Diagnosis

8.1 Introduction

8.2 Recursive Partitioning

8.3 Analysis Using **R**

8.3.1 Forbes 2000 Data

For some observations the profit is missing and we first remove those companies from the list

```
R> data("Forbes2000", package = "HSAUR")
R> Forbes2000 <- subset(Forbes2000, !is.na(profits))
```

The `rpart` function from *rpart* can be used to grow a regression tree. The response variable and the covariates are defined by a model formula in the same way as for `lm`, say. By default, a large initial tree is grown.

```
R> library("rpart")
R> forbes_rpart <- rpart(profits ~ assets + marketvalue + sales,
+                        data = Forbes2000)
```

A `print` method for *rpart* objects is available, however, a graphical representation shown in Figure~8.1 is more convenient. Observations which satisfy the condition shown for each node go to the left and observations which don't are element of the right branch in each node. The numbers plotted in the leaves are the mean profit for those observations satisfying the conditions stated above. For example, the highest profit is observed for companies with a market value greater than 89.33 billion US dollars and with more than 91.92 US dollars sales.

To determine if the tree is appropriate or if some of the branches need to be subjected to pruning we can use the `cptable` element of the *rpart* object:

```
R> print(forbes_rpart$cptable)
```

```
          CP nsplit rel error    xerror      xstd
1 0.23748446      0 1.0000000 1.0010339 0.1946331
2 0.04600397      1 0.7625155 0.8397144 0.2174245
3 0.04258786      2 0.7165116 0.8066685 0.2166339
4 0.02030891      3 0.6739237 0.7625940 0.2089684
5 0.01854336      4 0.6536148 0.7842574 0.2093683
6 0.01102304      5 0.6350714 0.7925891 0.2106088
```

```
R> plot(forbes_rpart, uniform = TRUE, margin = 0.1, branch = 0.5,
+       compress = TRUE)
R> text(forbes_rpart)
```
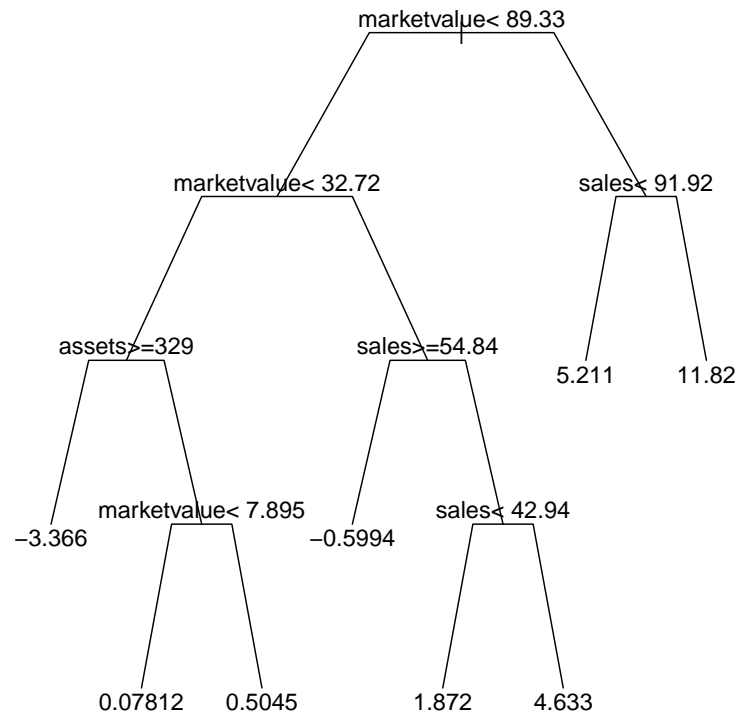

Figure 8.1   Large initial tree for Forbes 2000 data.

```
7 0.01076006        6 0.6240484 0.7931405 0.2128048
8 0.01000000        7 0.6132883 0.7902771 0.2128037

R> opt <- which.min(forbes_rpart$cptable[,"xerror"])
```

The xerror column contains of estimates of cross-validated prediction error for different numbers of splits (nsplit). The best tree has three splits. Now we can prune back the large initial tree using

```
R> cp <- forbes_rpart$cptable[opt, "CP"]
R> forbes_prune <- prune(forbes_rpart, cp = cp)
```

The result is shown in Figure~8.2. This tree is much smaller. From the sample sizes and boxplots shown for each leaf we see that the majority of companies

is grouped together. However, a large market value, more that 32.72 billion US dollars, seems to be a good indicator of large profits.

8.3.2 Glaucoma Diagnosis

```
R> data("GlaucomaM", package = "ipred")
R> glaucoma_rpart <- rpart(Class ~ ., data = GlaucomaM,
+                control = rpart.control(xval = 100))
R> glaucoma_rpart$cptable
          CP nsplit rel error    xerror       xstd
1 0.65306122      0 1.0000000 1.5306122 0.06054391
2 0.07142857      1 0.3469388 0.3877551 0.05647630
3 0.01360544      2 0.2755102 0.3775510 0.05590431
4 0.01000000      5 0.2346939 0.4489796 0.05960655
R> opt <- which.min(glaucoma_rpart$cptable[,"xerror"])
R> cp <- glaucoma_rpart$cptable[opt, "CP"]
R> glaucoma_prune <- prune(glaucoma_rpart, cp = cp)
```

As we discussed earlier, the choice of the appropriate sized tree is not a trivial problem. For the glaucoma data, the above choice of three leaves is very unstable across multiple runs of cross-validation. As an illustration of this problem we repeat the very same analysis as shown above and record the optimal number of splits as suggested by the cross-validation runs.

```
R> nsplitopt <- vector(mode = "integer", length = 25)
R> for (i in 1:length(nsplitopt)) {
+      cp <- rpart(Class ~ ., data = GlaucomaM)$cptable
+      nsplitopt[i] <- cp[which.min(cp[,"xerror"]), "nsplit"]
+ }
R> table(nsplitopt)
nsplitopt
 1  2  5
14  7  4
```

Although for 14 runs of cross-validation a simple tree with one split only is suggested, larger trees would have been favored in 11 of the cases. This short analysis shows that we should not trust the tree in Figure~8.3 too much.

One way out of this dilemma is the aggregation of multiple trees via *bagging*. In R, the bagging idea can be implemented by three or four lines of code. Case count or weight vectors representing the bootstrap samples can be drawn from the multinominal distribution with parameters n and $p_1 = 1/n, \ldots, p_n = 1/n$ via the `rmultinom` function. For each weight vector, one large tree is constructed without pruning and the *rpart* objects are stored in a list, here called `trees`:

```
R> trees <- vector(mode = "list", length = 25)
R> n <- nrow(GlaucomaM)
R> bootsamples <- rmultinom(length(trees), n, rep(1, n)/n)
```

```
R> layout(matrix(1:2, nc = 1))
R> plot(forbes_prune, uniform = TRUE, margin = 0.1, branch = 0.5,
+        compress = TRUE)
R> text(forbes_prune)
R> rn <- rownames(forbes_prune$frame)
R> lev <- rn[sort(unique(forbes_prune$where))]
R> where <- factor(rn[forbes_prune$where], levels = lev)
R> n <- tapply(Forbes2000$profits, where, length)
R> boxplot(Forbes2000$profits ~ where, varwidth = TRUE,
+          ylim = range(Forbes2000$profit) * 1.3,
+          pars = list(axes = FALSE),
+          ylab = "Profits in US dollars")
R> abline(h = 0, lty = 3)
R> axis(2)
R> text(1:length(n), max(Forbes2000$profit) * 1.2,
+        paste("n = ", n))
```
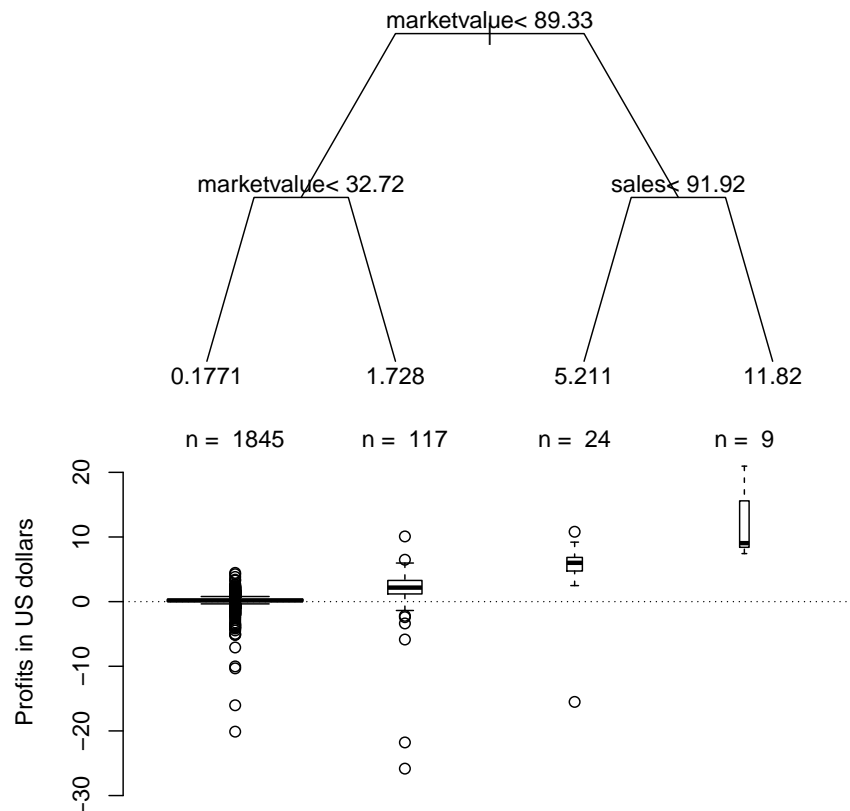

Figure 8.2   Pruned regression tree for Forbes 2000 data with the distribution of the profit in each leaf depicted by a boxplot.

```
R> layout(matrix(1:2, nc = 1))
R> plot(glaucoma_prune, uniform = TRUE, margin = 0.1, branch = 0.5,
+        compress = TRUE)
R> text(glaucoma_prune, use.n = TRUE)
R> rn <- rownames(glaucoma_prune$frame)
R> lev <- rn[sort(unique(glaucoma_prune$where))]
R> where <- factor(rn[glaucoma_prune$where], levels = lev)
R> mosaicplot(table(where, GlaucomaM$Class), main = "", xlab = "",
+             las = 1)
```
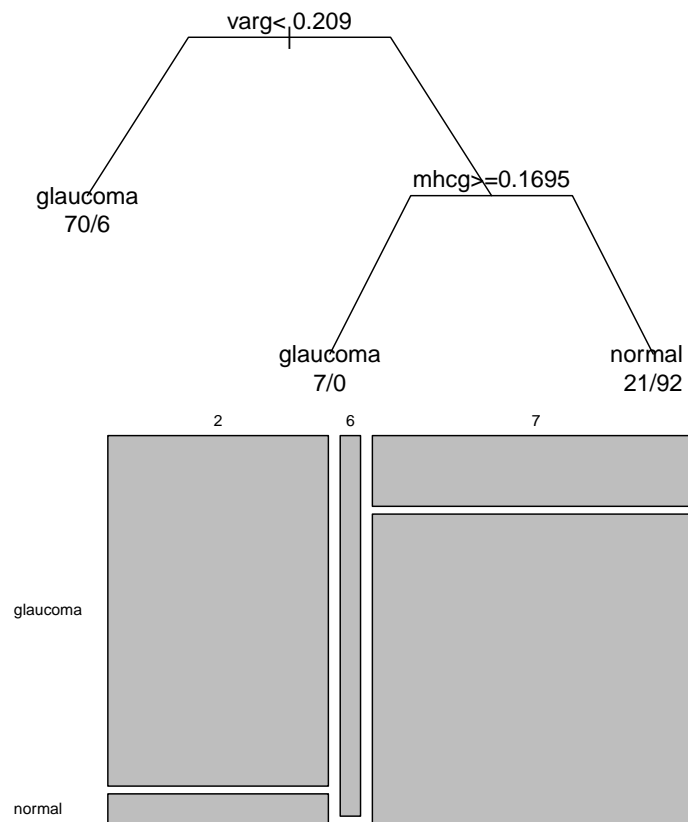

Figure 8.3   Pruned classification tree of the glaucoma data with class distribution
in the leaves depicted by a mosaicplot.

```
R> mod <- rpart(Class ~ ., data = GlaucomaM,
+               control = rpart.control(xval = 0))
R> for (i in 1:length(trees))
+     trees[[i]] <- update(mod, weights = bootsamples[,i])
```

The `update` function re-evaluates the call of `mod`, however, with the weights
being altered, i.e., fits a tree to a bootstrap sample specified by the weights.
It is interesting to have a look at the structures of the multiple trees. For
example, the variable selected for splitting in the root of the tree is not unique
as can be seen by

```
R> table(sapply(trees, function(x) as.character(x$frame$var[1])))
```

```
phcg varg vari vars
   1   14    9    1
```

Although `varg` is selected most of the time, other variables such as `vari` occur
as well – a further indication that the tree in Figure~8.3 is questionable and
that hard decisions are not appropriate for the glaucoma data.

   In order to make use of the ensemble of trees in the list `trees` we estimate
the conditional probability of suffering from glaucoma given the covariates for
each observation in the original data set by

```
R> classprob <- matrix(0, nrow = n, ncol = length(trees))
R> for (i in 1:length(trees)) {
+     classprob[,i] <- predict(trees[[i]],
+                              newdata = GlaucomaM)[,1]
+     classprob[bootsamples[,i] > 0,i] <- NA
+ }
```

Thus, for each observation we get 25 estimates. However, each observation has
been used for growing one of the trees with probability 0.632 and thus was
not used with probability 0.368. Consequently, the estimate from a tree where
an observation was not used for growing is better for judging the quality of
the predictions and we label the other estimates with `NA`.

   Now, we can average the estimates and we vote for glaucoma when the
average of the estimates of the conditional glaucoma probability exceeds 0.5.
The comparison between the observed and the predicted classes does not suffer
from overfitting since the predictions are computed from those trees for which
each single observation was *not* used for growing.

```
R> avg <- rowMeans(classprob, na.rm = TRUE)
R> predictions <- factor(ifelse(avg > 0.5, "glaucoma", "normal"))
R> predtab <- table(predictions, GlaucomaM$Class)
R> predtab
```

```
predictions glaucoma normal
   glaucoma        77     12
   normal          21     86
```

Thus, an honest estimate of the probability of a glaucoma prediction when
the patient is actually suffering from glaucoma is

```
R> round(predtab[1,1] / colSums(predtab)[1] * 100)
```

```
glaucoma
      79
```

per cent. For

```
R> round(predtab[2,2] / colSums(predtab)[2] * 100)
```

```
normal
    88
```

per cent of normal eyes, the ensemble does not predict a glaucomateous damage.

The *bagging* procedure is a special case of a more general approach called *random forest* (Breiman, 2001). The package *randomForest* (Breiman et˜al., 2006) can be used to compute such ensembles via

```
R> library("randomForest")
R> rf <- randomForest(Class ~ ., data = GlaucomaM)
```

and we obtain out-of-bag estimates for the prediction error via

```
R> table(predict(rf), GlaucomaM$Class)
```

```
          glaucoma normal
 glaucoma       80     11
 normal         18     87
```

For the glaucoma data, such a *conditional inference tree* can be computed using the ctree function

```
R> library("party")
R> glaucoma_ctree <- ctree(Class ~ ., data = GlaucomaM)
```

and a graphical representation is depicted in Figure˜8.5 showing both the cutpoints and the p-values of the associated independence tests for each node. The first split is performed using a cutpoint defined with respect to the volume of the optic nerve above some reference plane, but in the inferior part of the eye only (vari).

```
R> library("lattice")
R> gdata <- data.frame(avg = rep(avg, 2),
+       class = rep(as.numeric(GlaucomaM$Class), 2),
+       obs = c(GlaucomaM[["varg"]], GlaucomaM[["vari"]]),
+       var = factor(c(rep("varg", nrow(GlaucomaM)),
+                      rep("vari", nrow(GlaucomaM)))))
R> panelf <- function(x, y) {
+             panel.xyplot(x, y, pch = gdata$class)
+             panel.abline(h = 0.5, lty = 2)
+          }
R> print(xyplot(avg ~ obs | var, data = gdata,
+          panel = panelf,
+          scales = "free", xlab = "",
+          ylab = "Estimated Class Probability Glaucoma"))
```
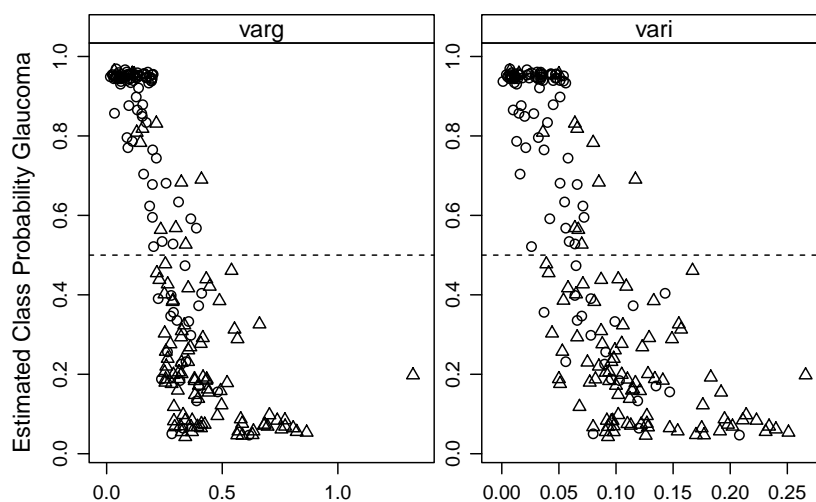

Figure 8.4   Glaucoma data: Estimated class probabilities depending on two important variables. The 0.5 cut-off for the estimated glaucoma probability is depicted as horizontal line. Glaucomateous eyes are plotted as circles and normal eyes are triangles.

```
R> plot(glaucoma_ctree)
```
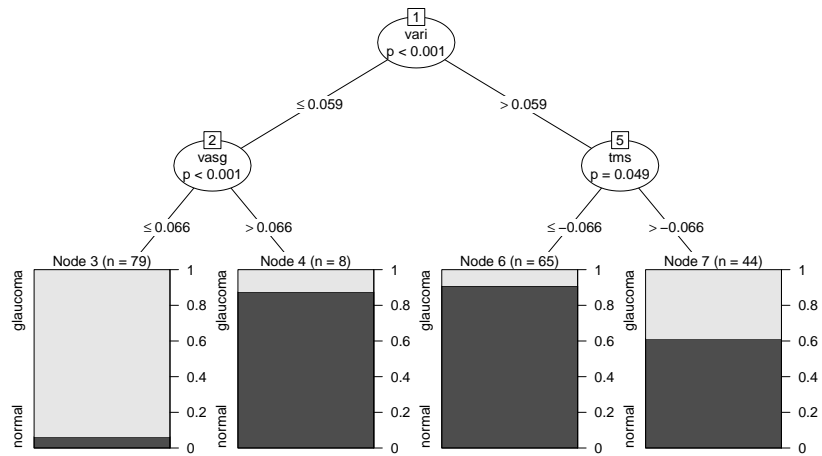

Figure 8.5   Glaucoma data: Conditional inference tree with the distribution of glaucomateous eyes shown for each terminal leaf.

Bibliography

Breiman, L. (2001), "Random forests," *Machine Learning*, 45, 5–32.

Breiman, L., Cutler, A., Liaw, A., and Wiener, M. (2006), *randomForest: Breiman and Cutler's Random Forests for Classification and Regression*, URL http://stat-www.berkeley.edu/users/breiman/RandomForests, R package version 4.5-18.