Data Analysis & Graphics Using R, 2^{nd} edn – Solutions to Exercises (December 13, 2006)

Preliminaries

```
> library(DAAG)
> library(rpart)
```

Exercise 1
Refer to the `head.injury` data frame.

(a) Use the default setting in `rpart()` to obtain a tree-based model for predicting occurrence of clinically important brain injury, given the other variables.

(b) How many splits gives the minimum cross-validation error? item Prune the tree using the 1 standard error rule.

(a)
```
> set.seed(29)
> injury.rpart <- rpart(clinically.important.brain.injury ~ .,
+     data = head.injury, method = "class", cp = 1e-04)
> plotcp(injury.rpart)
> printcp(injury.rpart)

Classification tree:
rpart(formula = clinically.important.brain.injury ~ ., data = head.injury,
    method = "class", cp = 1e-04)

Variables actually used in tree construction:
[1] GCS.13                GCS.15.2hours         age.65
[4] amnesia.before        basal.skull.fracture  high.risk
[7] loss.of.consciousness vomiting

Root node error: 250/3121 = 0.080103

n= 3121

      CP nsplit rel error xerror     xstd
1 0.0400      0     1.000  1.000 0.060660
2 0.0360      2     0.920  0.992 0.060438
3 0.0140      3     0.884  0.896 0.057678
4 0.0080      5     0.856  0.904 0.057915
5 0.0001     10     0.816  0.916 0.058268
```

The setting `cp=0.0001` was reached after some experimentation.

(b) The minimum cross-validated relative error is for `nsplit=3`, i.e., for a tree size of 4.

(c) The one-standard-error rule likewise chooses `nsplit=3`, with `cp=0.014`. Setting `cp=0.02`, i.e., larger than `cp` for the next smallest number of splits, will prune the tree back to this size. We have

```
> injury0.rpart <- prune(injury.rpart, cp = 0.02)
```

We plot the tree from (a) that shows the cross-validated relative error, and the tree obtained from (c).

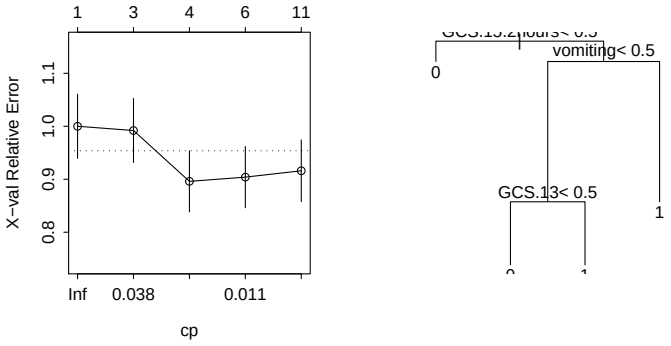

Figure 1: Plots from the `rpart` analysis of the head injury data: (i) cross-validated relative error versus `cp`; and (ii) the tree obtained in (c).

There can be substantial change from one run to the next.

Exercise 2
The data set `mifem` is part of the larger data set in the data frame `monica` that we have included in our *DAAG* package. Use tree-based regression to predict mortality in this larger data set. What is the most immediately striking feature of your analysis? Should this be a surprise?

```
> monica.rpart <- rpart(outcome ~ ., data = monica, method = "class")
```

```
> plot(monica.rpart)
> text(monica.rpart)
```

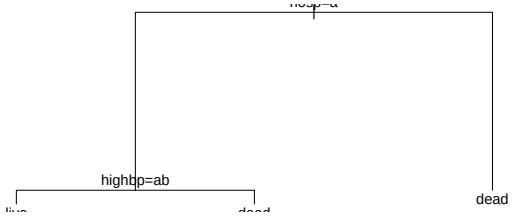

Figure 2: Classification tree for monica data.

Those who were not hospitalised were very likely to be dead! Check by examining the table:

```
> table(monica$hosp, monica$outcome)
```

```
   live dead
y 3522  920
n    3 1922
```

Exercise 3
Use tree-based regression to predict `re78` in the data frame `nsw74pred1` that is in our *DAAG* package. Compare the predictions with the multiple regression predictions in Chapter 6.

In order to reproduce the same results as given here, do:

```
> set.seed(21)
```

Code for the initial calculation is:

```
> nsw.rpart <- rpart(re78 ~ ., data = nsw74psid1, cp = 0.001)
> plotcp(nsw.rpart)
```

It is obvious that `cp=0.002` will be adequate. At this point, the following is a matter of convenience, to reduce the printed output:

```
> nsw.rpart <- prune(nsw.rpart, cp = 0.002)
> printcp(nsw.rpart)

Regression tree:
rpart(formula = re78 ~ ., data = nsw74psid1, cp = 0.001)

Variables actually used in tree construction:
[1] age  educ re74 re75

Root node error: 6.5346e+11/2675 = 244284318

n= 2675

          CP nsplit rel error  xerror     xstd
1  0.3446296      0   1.00000 1.00067 0.046287
2  0.1100855      1   0.65537 0.66461 0.038977
3  0.0409403      2   0.54528 0.55811 0.033004
4  0.0317768      3   0.50434 0.51821 0.035244
5  0.0158188      4   0.47257 0.50636 0.034622
6  0.0105727      5   0.45675 0.49139 0.034688
7  0.0105337      6   0.44618 0.48453 0.034527
8  0.0063341      7   0.43564 0.46901 0.032502
9  0.0056603      8   0.42931 0.46028 0.032969
10 0.0038839      9   0.42365 0.46133 0.033142
11 0.0035516     10   0.41976 0.46238 0.033095
12 0.0031768     11   0.41621 0.47329 0.033838
13 0.0028300     12   0.41304 0.47544 0.033675
14 0.0027221     13   0.41021 0.47495 0.033776
15 0.0023286     15   0.40476 0.47570 0.033783
16 0.0020199     16   0.40243 0.47642 0.033609
17 0.0020000     17   0.40041 0.47715 0.033851
```

The minimum cross-validated relative error is at `nsplit=12`. The one standard error limit is 0.498 (=0.463+0.035). The one standard error rule suggests taking `nsplit=5`.

If we go with the one standard error rule, we have a residual variance equal to $244284318 \times 0.49177 = 120131699$.

For the estimate of residual variance from the calculations of Section 6.x, we do the following.

```
> attach(nsw74psid1)
> here <- age <= 40 & re74 <= 5000 & re75 <= 5000 & re78 < 30000
> nsw74psidA <- nsw74psid1[here, ]
> detach(nsw74psid1)
> A1.lm <- lm(re78 ~ trt + (age + educ + re74 + re75) + (black +
+     hisp + marr + nodeg), data = nsw74psidA)
> summary(A1.lm)$sigma^2

[1] 40177577
```

The variance estimate is 40177577. This is about a third of the variance estimate that was obtained with tree-based regression.

Exercise 4
Copy down the email spam data set from the web site given in Section 10.2. Carry out a tree-based regression using all 57 available explanatory variables. Determine the change in the cross-validation estimate of predictive accuracy.

We set the random number seed to 21, to allow users to reproduce our results. In most other contexts, it will be best not to set a seed. The file **spam.shortnames** is available for copying from the web address `http://wwwmaths.anu.edu.au/~johnm/r-book/xtra-data`. The data frame `spam` is created thus:

```
> spam <- read.table("spambase.data", header = FALSE, sep = ",")
> nam <- scan("spam.shortnames", what = "")
> names(spam) <- nam
```

Now load *rpart* and proceed with the calculations.

```
> set.seed(21)
> spam.rpart <- rpart(yesno ~ ., data = spam, cp = 1e-04, method = "class")
> printcp(spam.rpart)

Classification tree:
rpart(formula = yesno ~ ., data = spam, method = "class", cp = 1e-04)

Variables actually used in tree construction:
 [1] address    bang       crl.av     crl.long   crl.tot    data
 [7] dollar     edu        email      font       free       george
[13] hp         internet   leftparen  money      n1999      n650
[19] our        over       re         remove     semicolon  technology
[25] will       you        your

Root node error: 1813/4601 = 0.39404

n= 4601

           CP nsplit rel error  xerror      xstd
1  0.47655819      0   1.00000 1.00000 0.0182819
2  0.14892443      1   0.52344 0.55819 0.0154972
3  0.04302261      2   0.37452 0.46001 0.0144131
4  0.03088803      4   0.28847 0.32212 0.0124547
5  0.01047987      5   0.25758 0.27910 0.0117052
```

```
6  0.00827358      6   0.24710 0.26586 0.0114576
7  0.00717044      7   0.23883 0.26089 0.0113626
8  0.00529509      8   0.23166 0.25317 0.0112121
9  0.00441258     14   0.19581 0.23552 0.0108559
10 0.00358522     15   0.19140 0.22835 0.0107060
11 0.00275786     19   0.17705 0.22559 0.0106475
12 0.00257400     22   0.16878 0.22228 0.0105767
13 0.00220629     25   0.16106 0.22228 0.0105767
14 0.00211436     27   0.15665 0.21897 0.0105052
15 0.00165472     33   0.14396 0.21180 0.0103477
16 0.00110314     36   0.13900 0.19857 0.0100476
17 0.00082736     43   0.13127 0.19581 0.0099834
18 0.00055157     47   0.12796 0.19581 0.0099834
19 0.00036771     53   0.12466 0.20077 0.0100985
20 0.00010000     62   0.12135 0.20077 0.0100985
```

Figure 3 shows the graph that is obtained by plotting this tree. For making a decision on the size of tree however, it is convenient to work from the information given by the function `printcp()`.

```
> plotcp(spam.rpart)
```

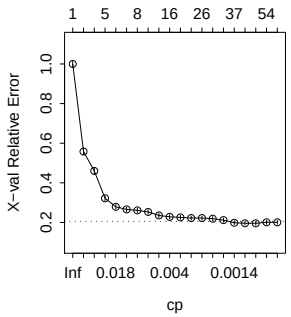

Figure 3: Plot of cross-validated relative error versus `cp`, for the full `spam` data set.

Setting `cp=0.0001` ensures, when the random number seed is set to 21, that the cross-validated relative error reaches a minimum, of 0.1958, at `nsplit=43`. Pruning to get the tree that is likely to have best predictive power can use `cp=0.001`. Adding the SE to the minimum cross-validated relative error gives 0.2. The smallest tree with an SE smaller than this is at `nsplit=36`; setting `cp=0.0012` will give this tree.

Here then are the two prunings:

```
> spam.rpart1 <- prune(spam.rpart, cp = 0.001)
> spam.rpart2 <- prune(spam.rpart, cp = 0.0012)
```

Additional Exercises A number of additional exercises are included in the laboratory exercises that are available from the web page `http:www.maths.anu.edu.au/~johnm/courses/dm`