Advances Computational Econometrics. Chapter 5: Tree-based and ensemble methods

Thierry Denoeux

5/2/2022

Exercise 1

Question 1

We start by loading the data, removing variables `Default`, `Exp_Inc`, `Spending` and `Logspend`, and splitting the data set into training and test sets:

```
credit<-read.csv('/Users/Thierry/Documents/R/Data/Economics/Greene/TableF7-3.csv',
                 sep=",",header=TRUE)
credit1<-credit[,-c(2,12:14)]
n<-nrow(credit1)
ntrain<-10000
ntest<-n-ntrain
set.seed(30)
train<-sample(n,ntrain)
```

We also declare the response variable as a factor:

```
credit1$CARDHLDR<-as.factor(credit1$CARDHLDR)
```

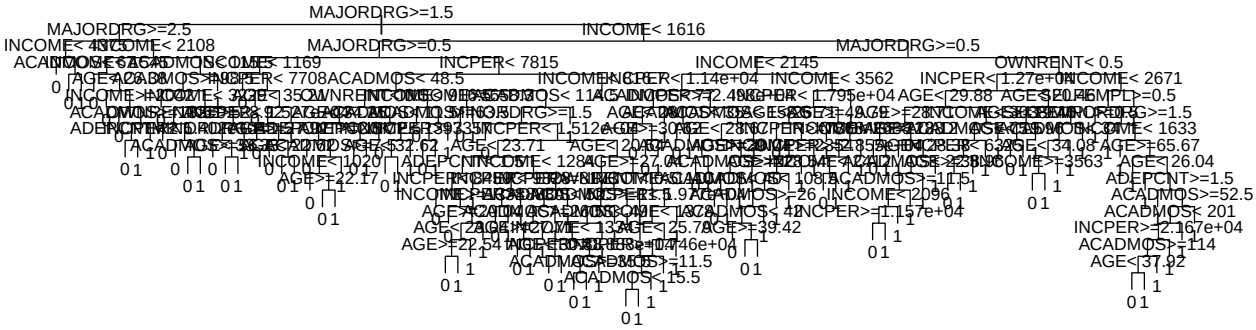
Question 2

We use function `rpart` of package `rpart`:

```
library(rpart)
fit <- rpart(CARDHLDR~.,data=credit1,subset=train,
           method="class",control = rpart.control(xval = 10, minbucket = 10,cp=0.00))
```

We plot the tree:

```
plot(fit,margin = 0.05,compress=TRUE,uniform=TRUE)
text(fit,minlength=1,cex=0.8,splits=TRUE)
```



The tree is too big to be easily plotted and interpretable.

We compute the confusion matrix and the test error rate:

```
yhat <- predict(fit,newdata=credit1[-train,],type='class')
y.test <- credit[-train,"CARDHLDR"]
table(y.test,yhat)
```

```
##       yhat
## y.test    0    1
##      0  318  452
##      1  162 2512
```

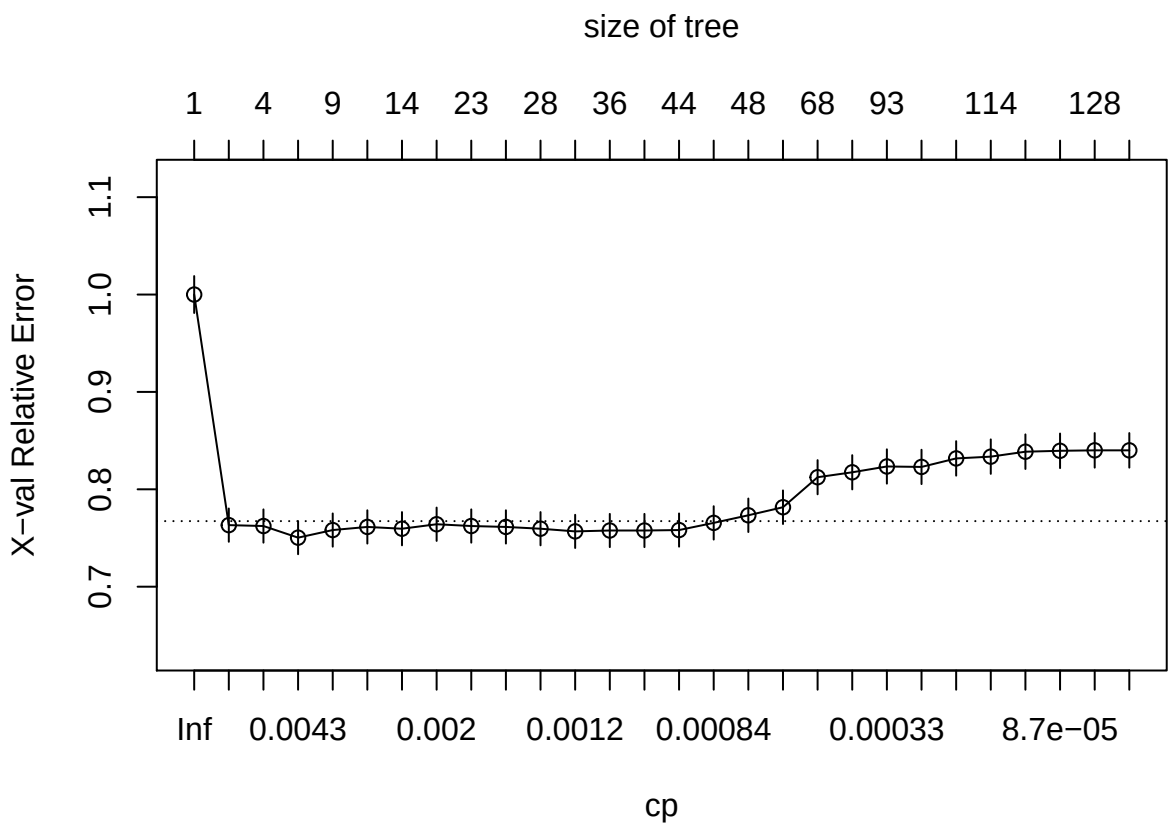
```
err.tree <- 1-mean(y.test==yhat)
print(err.tree)
```

```
## [1] 0.1782811
```

Question 3

We plot the cross-validation error as a function of hyperparameter λ (denoted as `cp` here):

```
plotcp(fit)
```




```
printcp(fit)
```

```
##
## Classification tree:
## rpart(formula = CARDHLDR ~ ., data = credit1, subset = train,
##     method = "class", control = rpart.control(xval = 10, minbucket = 10,
##         cp = 0))
##
## Variables actually used in tree construction:
## [1] ACADMOS  ADEPCNT  AGE      INCOME   INCPER   MAJORDRG MINORDRG OWNRENT
## [9] SELFEMPL
##
## Root node error: 2175/10000 = 0.2175
##
## n= 10000
##
##            CP nsplit rel error  xerror     xstd
## 1  2.3678e-01      0   1.00000 1.00000 0.018968
## 2  6.4368e-03      1   0.76322 0.76322 0.017107
## 3  5.7471e-03      3   0.75034 0.76230 0.017099
## 4  3.2184e-03      5   0.73885 0.75034 0.016991
## 5  2.6054e-03      8   0.72920 0.75816 0.017062
## 6  2.2989e-03     12   0.71816 0.76138 0.017091
## 7  2.0690e-03     13   0.71586 0.75954 0.017074
## 8  1.8391e-03     19   0.70023 0.76414 0.017115
## 9  1.6092e-03     22   0.69471 0.76230 0.017099
## 10 1.3793e-03     26   0.68828 0.76138 0.017091
## 11 1.2261e-03     27   0.68690 0.75954 0.017074
```

```
## 12 1.1494e-03     33   0.67724 0.75678 0.017049
## 13 1.0728e-03     35   0.67494 0.75770 0.017057
## 14 1.0115e-03     38   0.67172 0.75770 0.017057
## 15 9.1954e-04     43   0.66667 0.75816 0.017062
## 16 7.6628e-04     44   0.66575 0.76552 0.017128
## 17 6.8966e-04     47   0.66345 0.77333 0.017197
## 18 4.5977e-04     58   0.65563 0.78161 0.017270
## 19 3.6782e-04     67   0.65149 0.81241 0.017536
## 20 3.4483e-04     83   0.64460 0.81747 0.017579
## 21 3.0651e-04     92   0.64092 0.82345 0.017629
## 22 2.6273e-04     98   0.63908 0.82299 0.017625
## 23 2.2989e-04    105   0.63724 0.83172 0.017698
## 24 1.5326e-04    113   0.63540 0.83356 0.017713
## 25 1.1494e-04    116   0.63494 0.83862 0.017755
## 26 6.5681e-05    120   0.63448 0.83954 0.017763
## 27 3.8314e-05    127   0.63402 0.84000 0.017766
## 28 0.0000e+00    139   0.63356 0.84000 0.017766
```

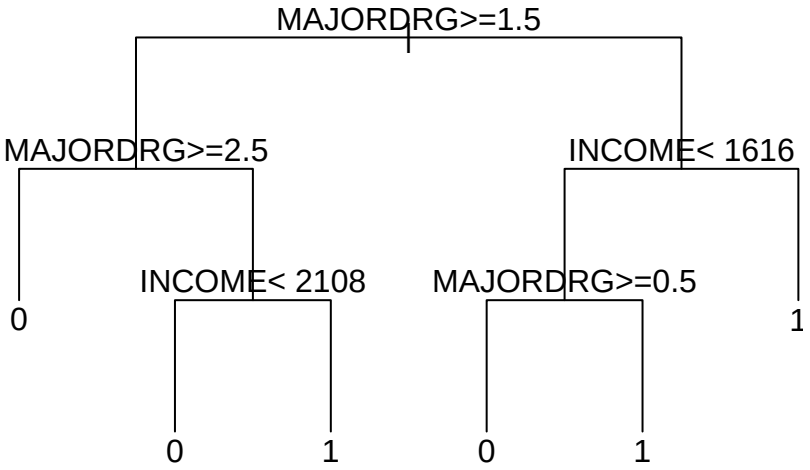
We can select the value of λ with minimum cross-validation error:

```
i.min<-which.min(fit$cptable[,4])
cp.opt<-fit$cptable[i.min,1]
print(cp.opt)
```

```
## [1] 0.003218391
```

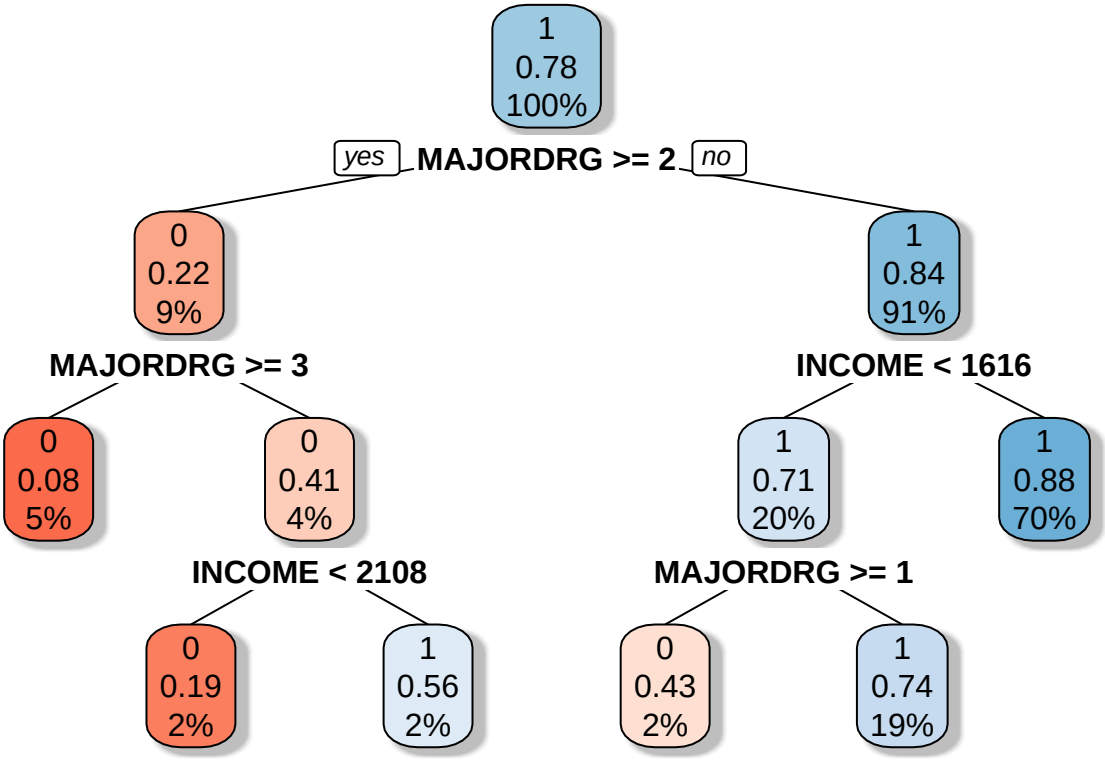
We plot the tree:

```
pruned_tree<-prune(fit,cp=cp.opt)
plot(pruned_tree,margin = 0.1,compress=TRUE,uniform=TRUE)
text(pruned_tree,pretty=0)
```



Package `rpart.plot` has a function to draw nicer plots:

```
library(rpart.plot)
rpart.plot(pruned_tree, box.palette="RdBu", shadow.col="gray",fallen.leaves=FALSE)
```



We compute the confusion matrix and the error rate for the pruned tree:

```
yhat<-predict(pruned_tree,newdata=credit1[-train,],type='class')
CM<-table(y.test,yhat)
err.pruned<-1-mean(y.test==yhat)
print(err.pruned)
```

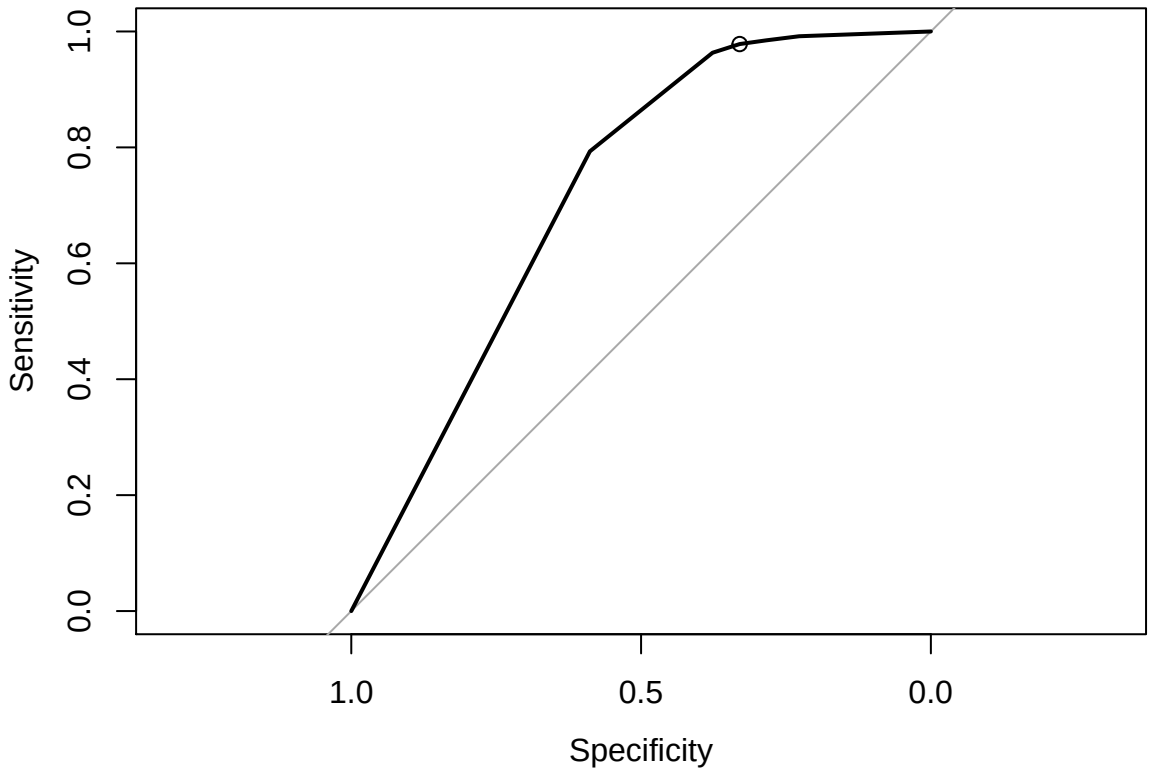
```
## [1] 0.1666667
```

The error rate has decreased slightly but, above all, the tree is more interpretable.

Question 4

We now plot the ROC curve using function `roc` of package `pROC`. For that, we need a discriminant function. We use the estimated posterior probability, which can be computed by function `predict.rpart` with the argument `type='prob'`. We will also plot the point on the ROC curve corresponding to the maximum a posteriori decision rule:

```
library(pROC)
prob<-predict(pruned_tree,newdata=credit[-train,],type='prob')
roc_tree<-roc(y.test,prob[,2])
plot(roc_tree)
TPR<-CM[2,2]/rowSums(CM)[2]
FPR<-CM[1,2]/rowSums(CM)[1]
points(1-FPR,TPR)
```


Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1.0
0.5
0.0
Specificity