Laboration report in Machine Learning

Computer lab 2 block 1

732A99

Simge Cinar
Duc Tran
William Wiik

Division of Statistics and Machine Learning
Department of Computer Science
Linköping University

22 November 2023

Contents

1 Assignment 2. Decision trees and logistic regression for bank marketing.

The data in this assignment is related with direct marketing campaigns of a Portuguese banking institution. Data consists of 21 variables, 20 input variables about the clients and the output variable is if the client has a term deposit (binary variable).

1.1 2.1

Question: Import the data to R, **remove variable "duration"** and divide into training/validation/test as 40/30/30: use data partitioning code specified in Lecture 2a.

Answer: Data is read into R, where all character variables are made into factor variables, and the variable duration was removed.

```
# Index 12 is the variable duration
data <- read.csv2("bank-full.csv", stringsAsFactors = TRUE)[, -12]

# Data partitioning (40/30/30)
# Training data
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=data[id,]
# Validation data
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))
valid=data[id2,]
# Test data
id3=setdiff(id1,id2)
test=data[id3,]
```

After the splitting the number of observations in the data sets are as follows:

- Training: 18 084 observations.
- Validation: 13 563 observations.
- Test: 13 563 observations.

1.2 2.2

Question: Fit decision trees to the training data so that you change the default settings one by one (i.e. not simultaneously):
a. Decision Tree with default settings.
b. Decision Tree with smallest allowed node size equal to 7000.
c. Decision trees minimum deviance to 0.0005.
and report the misclassification rates for the training and validation data. Which model is the best one among these three? Report how changing the deviance and node size affected the size of the trees and explain why.

Answer:

The default setting in tree for smallest allowed node size is 10 and the default for minimum deviance is 0.01. The three different trees are fitted and in table **??**

```
# Tree a: default setting
tree_a <- tree(y ~., data=train)
# Misclassification for training data for tree a
pred_train <- predict(tree_a, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree a
pred_valid <- predict(tree_a, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_a$frame$var == "<leaf>")
# Misclassification for tree a
model_a <- c(miss_train, miss_valid, leaf)



# Tree b: smallest size changed from 10 (default) to 7 000
tree_b <- tree(y ~., data=train, control=tree.control(nobs = nrow(train),
                                                      minsize = 7000))
# Misclassification for training data for tree b
pred_train <- predict(tree_b, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree b
pred_valid <- predict(tree_b, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_b$frame$var == "<leaf>")
# Misclassification for tree b
model_b <- c(miss_train, miss_valid, leaf)


# Tree c: mindev changed from 0.01 (default) to 0.0005
```

```r
tree_c <- tree(y ~., data=train, control=tree.control(nobs = nrow(train),
                                                      mindev = 0.0005))
# Misclassification for training data for tree c
pred_train <- predict(tree_c, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree c
pred_valid <- predict(tree_c, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_c$frame$var == "<leaf>")
# Misclassification for tree c
model_c <- c(miss_train, miss_valid, leaf)

# Summarised results
table <- data.frame(rbind(model_a, model_b, model_c))
colnames(table) <- c("Training error", "Validation error", "Number of leaves")
rownames(table) <- c("Tree a: default setting", "Tree b: smallest node 7000",
                     "Tree c: min deviance 0.0005")

kable(table, digits=6, label="tree",
      caption="Misclassification error for the three trees on training and validation data.")
```

Table 1: Misclassification error for the three trees on training and validation data.

	Training error	Validation error	Number of leaves
Tree a: default setting	0.104844	0.109268	6
Tree b: smallest node 7000	0.104844	0.109268	5
Tree c: min deviance 0.0005	0.094006	0.111922	122

From table **??**, the training error and validation error for tree a and b are the same, however there is a difference in number of leaves where tree a has one more leaf. In tree b, the minimum node size is changed to 7 000 from 10 which forces the tree to have least 7 000 observations in each node. The increase in minimum node size makes that the tree can not split nodes smaller than 7 000 and thus the tree will have less leaves.

Training error is lowest for tree c, but the validation error is also highest for tree c. This indicates that tree c is more overfitted on training data compared to the tree a and b. The difference for tree c is that minimum deviance is changed from 0.01 to0.0005. Minimum deviance is how much within-node deviance must be at least for a node to be able to split. By having a smaller number, the nodes can have smaller deviance and still be able split. This allows the tree to have more leaves.

Amongst tree a, b, and c the model b is best since it has lowest validation error but also has the least amounts of leaves. By having less leaves the model is less complex and is easier to interpret.

1.3 2.3

Question: Use training and validation sets to choose the optimal tree depth in the model 2c: study the trees up to 50 leaves. Present a graph of the dependence of deviances for the training and the validation data on the number of leaves and interpret this graph in terms of bias-variance tradeoff. Report the optimal amount of leaves and which variables seem to be most important for decision making in this tree. Interpret the information provided by the tree structure (not everything but most important findings).

Answer: Tree c is used where the tree is post-pruned to trees with between 2 and 50 leaves. The deviance for training data and validation data are calculated and the code is as follows:

```
fit <- tree(y ~., data=train,
            control=tree.control(nobs = nrow(train),
                                 mindev = 0.0005))
trainScore=rep(0,50)
testScore=rep(0,50)
for(i in 2:50){
  prunedTree=prune.tree(fit,best=i)
  pred=predict(prunedTree, newdata=valid,
               type="tree")
  trainScore[i]=deviance(prunedTree)
  testScore[i]=deviance(pred)
}
```

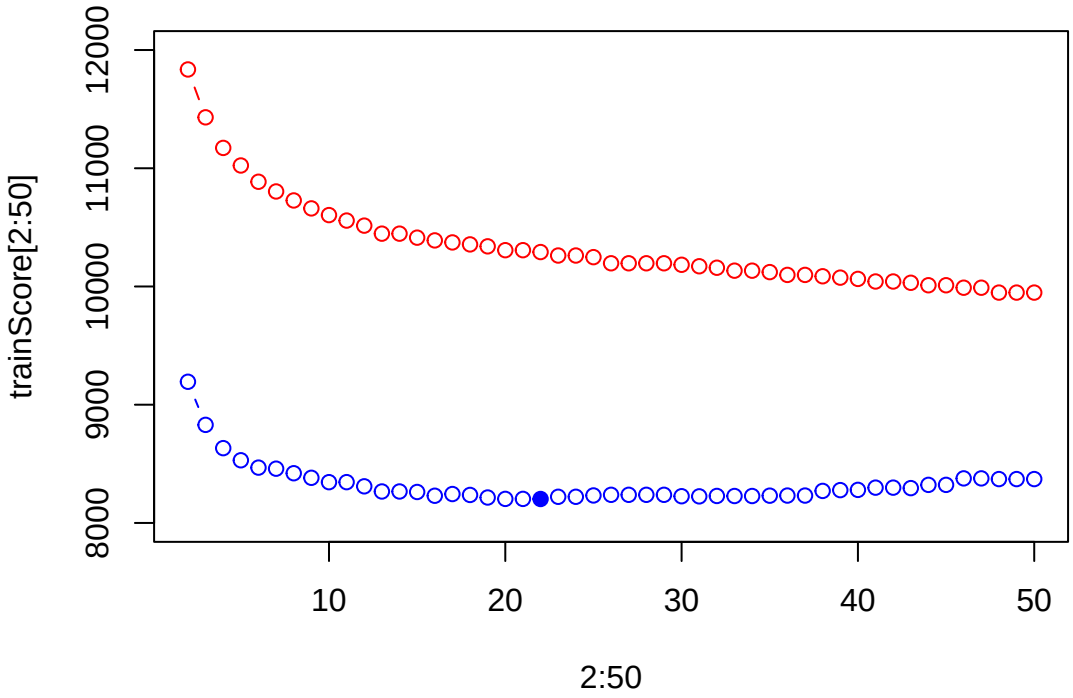
The tree with the lowest validation error had 22 leaves.

```
# Finds min, add +1 since index 1 is tree with 2 leaves.
which(min(testScore[2:50]) == testScore[2:50])+1
```

```
## [1] 22
```

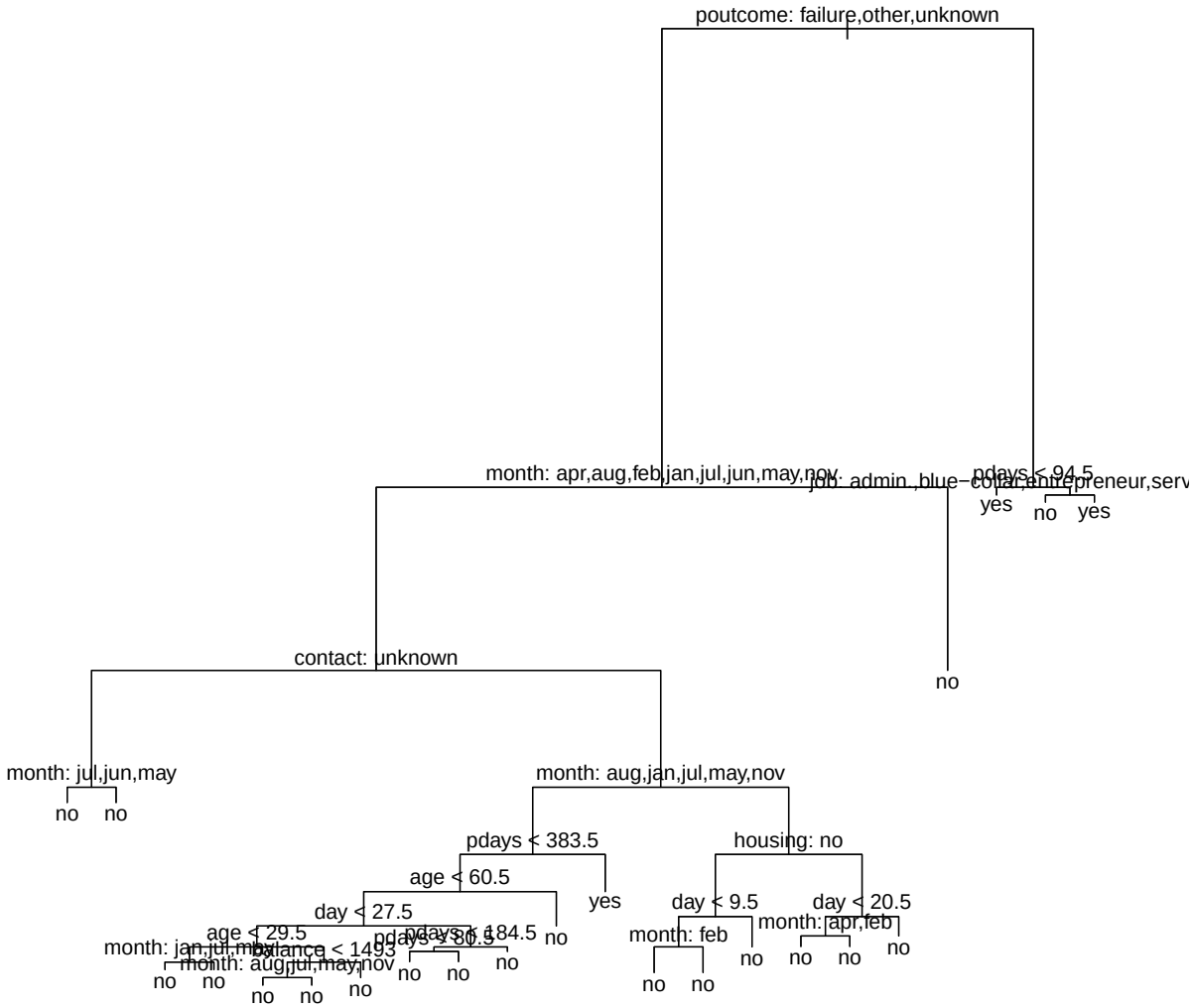
A visual

```
plot(2:50, trainScore[2:50], type="b", col="red",ylim=c(8000,12000))
points(2:50, testScore[2:50], type="b", col="blue")
# Fill the point with the min value on validation
points(22, testScore[22], pch=16, col="blue")
```



Best tree

```
best_fit <- prune.tree(fit, best=22)
plot(best_fit)
text(best_fit, pretty=0)
```



1.4 2.4

Question: Answer:

1.5 2.5

Question: Answer:

1.6 2.6

Question: Answer:

2 Statement of Contribution

We worked on the assignment individually for the computer labs (to be more efficient when asking questions), William on task 1, Duc on task 2, and Simge on task 3. We later solved all assignment individually and compared and discussed our solutions before dividing the task of writing the laboration report.

2.1 Question 1

Text written by William.

2.2 Question 2

Text written by Duc.

2.3 Question 3

Text written by Simge.

3 Appendix

The code used in this laboration report are summarised in the code as follows:

```
library(ggplot2)
library(tree)
library(knitr)
knitr::opts_chunk$set(
  echo = TRUE,
  fig.width = 4.5,
  fig.height = 3)
# Index 12 is the variable duration
data <- read.csv2("bank-full.csv", stringsAsFactors = TRUE)[, -12]

# Data partitioning (40/30/30)
# Training data
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=data[id,]
# Validation data
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))
valid=data[id2,]
# Test data
id3=setdiff(id1,id2)
```

```r
test=data[id3,]
# Tree a: default setting
tree_a <- tree(y ~., data=train)
# Misclassification for training data for tree a
pred_train <- predict(tree_a, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree a
pred_valid <- predict(tree_a, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_a$frame$var == "<leaf>")
# Misclassification for tree a
model_a <- c(miss_train, miss_valid, leaf)



# Tree b: smallest size changed from 10 (default) to 7 000
tree_b <- tree(y ~., data=train, control=tree.control(nobs = nrow(train),
                                                      minsize = 7000))
# Misclassification for training data for tree b
pred_train <- predict(tree_b, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree b
pred_valid <- predict(tree_b, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_b$frame$var == "<leaf>")
# Misclassification for tree b
model_b <- c(miss_train, miss_valid, leaf)


# Tree c: mindev changed from 0.01 (default) to 0.0005
tree_c <- tree(y ~., data=train, control=tree.control(nobs = nrow(train),
                                                      mindev = 0.0005))
# Misclassification for training data for tree c
pred_train <- predict(tree_c, newdata=train, type="class")
table_train <- table(train$y, pred_train)
miss_train <- 1 - (sum(diag(table_train)) / sum(table_train))
# Misclassification for validation data for tree c
pred_valid <- predict(tree_c, newdata=valid, type="class")
table_valid <- table(valid$y, pred_valid)
miss_valid <- 1 - (sum(diag(table_valid)) / sum(table_valid))
# Number of leaves
leaf <- sum(tree_c$frame$var == "<leaf>")
# Misclassification for tree c
```

```r
model_c <- c(miss_train, miss_valid, leaf)

# Summarised results
table <- data.frame(rbind(model_a, model_b, model_c))
colnames(table) <- c("Training error", "Validation error", "Number of leaves")
rownames(table) <- c("Tree a: default setting", "Tree b: smallest node 7000",
                     "Tree c: min deviance 0.0005")

kable(table, digits=6, label="tree",
      caption="Misclassification error for the three trees on training and validation data.")
fit <- tree(y ~., data=train,
            control=tree.control(nobs = nrow(train),
                                 mindev = 0.0005))
trainScore=rep(0,50)
testScore=rep(0,50)
for(i in 2:50){
  prunedTree=prune.tree(fit,best=i)
  pred=predict(prunedTree, newdata=valid,
               type="tree")
  trainScore[i]=deviance(prunedTree)
  testScore[i]=deviance(pred)
}
# Finds min, add +1 since index 1 is tree with 2 leaves.
which(min(testScore[2:50]) == testScore[2:50])+1
plot(2:50, trainScore[2:50], type="b", col="red",ylim=c(8000,12000))
points(2:50, testScore[2:50], type="b", col="blue")
# Fill the point with the min value on validation
points(22, testScore[22], pch=16, col="blue")
best_fit <- prune.tree(fit, best=22)
plot(best_fit)
text(best_fit, pretty=0)
```