Homework 2 - Decision Trees

Nelson Corrocher

March 16, 2017

Homework

1. Machine Learning with R - Lantz: Review chapter 5 and understand how to use the R tools described
2. Complete the following problems from the Kelleher book (Chapter 4):

- 2
- 3
- 4

Notes:

- Use R for your calculations
- Use a word or pdf file to upload your answers
- Include the R code as an appendix (same file)

Exercise 2

```
ex2data <- read.csv("Ch4Ex2.csv")

# The algorithm C5.0 doesn't work with logical fields, so converting them to factors is necessary as an
ex2data[sapply(ex2data, is.logical)] <- lapply(ex2data[sapply(ex2data, is.logical)], as.factor)

# Create the decision tree
ex2tree <- C5.0(ex2data[-c(1,5)],ex2data$RECIDIVIST)

# First test
# Important note: C5.0 requires the testing set to have the same column names as the training set used
predict(ex2tree, data.frame(GOOD_BEHAVIOR = "FALSE", AGE.30 = "FALSE", DRUG_DEPENDENT="TRUE"))
```

```
## [1] FALSE
## Levels: FALSE TRUE
```

```
# Second test
predict(ex2tree, data.frame(GOOD_BEHAVIOR = "TRUE", AGE.30 = "TRUE", DRUG_DEPENDENT="FALSE"))
```

```
## [1] TRUE
## Levels: FALSE TRUE
```

Exercise 3

```
ex3data <- read.csv("Ch4Ex3.csv")

## Function to calculate the entropy of the set
entropy <- function(set) {
  ent <- 0
  for (elem in unique(set)) {
```

```
    ent <- ent + (t <- sum(set == elem) / length(set)) * (log2(t)) * (-1)
  }
  return (ent)    # Note: the return parameter must always be encased in parentesis
}

gini <- function(set) {
  gin <- 0
  for (elem in unique(set)) {
    gin <- gin + sum(ifelse(set == elem,1,0) / length(set)) ^ 2
  }
  return (1 - gin)
}

remainder <- function()

entropy(ex3data$ANNUAL_INCOME)

gini(ex3data$ANNUAL_INCOME)
```

```
## [1] 0.53125
```

```
# Orders the data frame by AGE
ex3data <- ex3data[with(ex3data, order(AGE)), ]
ex3data
```

```
##   ID AGE   EDUCATION MARITAL_STATUS   OCCUPATION ANNUAL_INCOME
## 3  3  18 high school  never married  agriculture          ?25K
## 6  6  24 high school  never married armed forces          ?25K
## 4  4  28   bachelors        married professional       25K<U+0096>50K
## 5  5  37 high school        married  agriculture       25K<U+0096>50K
## 1  1  39   bachelors  never married    transport       25K<U+0096>50K
## 8  8  40   doctorate        married professional          ?50K
## 2  2  50   bachelors        married professional       25K<U+0096>50K
## 7  7  52 high school       divorced    transport       25K<U+0096>50K
```

As we can see above, ages 24 to 28, 39 to 40 and 40 to 50 have a shift in the target ANNUAL_INCOME so the averages of these ages will be our cuts. The one with the highest *Information Gain* will be the one to be used for the decision tree at the second level.

```
ent <- entropy(ex3data$ANNUAL_INCOME)
# Group 24-28
threshold <- (24 + 28) / 2
threshold
```

```
## [1] 26
```

```
iga <- (-1) * 2 / 8 * (2 / 2 * log2(2 / 2))
igb <- (-1) * 6 / 8 * (5/6 * log2(5/6) + 1/6 * log2(1/6))
ig1 <- ent - (iga + igb)
ig1
```

```
## [1] 0.8112781
```

```
# Group 39-40
threshold <- (39 + 40) / 2
threshold
```

```
## [1] 39.5
```

```
iga <- (-1) * 5 / 8 * (2 / 5 * log2(2/5) + 3/5 * log2(3/5))
igb <- (-1) * 3 / 8 * (1 / 3 * log2(1/3) + 2/3 * log2(2/3))
ig2 <- ent - (iga + igb)
ig2
```

```
## [1] 0.3475899
```

```
# Group 40-50
threshold <- (40 + 50) / 2
threshold
```

```
## [1] 45
```

```
iga <- (-1) * 6 / 8 * (2 / 6 * log2(2/6) + 3/6 * log2(3/6) + 1/6 * log2(1/6))
igb <- (-1) * 2 / 8 * (2 / 2 * log2(2/2))
ig3 <- ent - (iga + igb)
ig3
```

```
## [1] 0.204434
```

As we can see above, the greatest information gain from AGE comes from the threshold of >= 26.

```
# Information gain from Education, Marital Status and Occupation
ent <- entropy(ex3data$ANNUAL_INCOME)

# Education
ed_hs <- (-1) * (4/8) * ((2/4) * log(2/4,2) + 2/4 * log(2/4,2))
ed_ba <- (-1) * (3/8) * ((3/3) * log(3/3,2))
ed_do <- (-1) * (1/8) * ((1/3) * log(1/3,2))
ed_remainder <- ed_hs + ed_ba + ed_do
ed_ig <- ent - ed_remainder
ed_ig
```

```
## [1] 0.7327548
```

```
# Marital Status
ms_nm <- (-1) * (3/8) * ((1/3) * log(1/3,2) + 2/3 * log(2/3,2))
ms_ma <- (-1) * (4/8) * ((3/4) * log(3/4,2) + 1/4 * log(1/4,2))
ms_di <- (-1) * (1/8) * ((1/1) * log(1/1,2))
ms_remainder <- ms_nm + ms_ma + ms_di
ms_ig <- ent - ms_remainder
ms_ig
```

```
## [1] 0.5487949
```

```
# Occupation
oc_tr <- (-1) * (2/8) * ((2/2) * log(2/2,2))
oc_pr <- (-1) * (3/8) * ((2/3) * log(2/3,2) + 1/3 * log(1/3,2))
oc_ag <- (-1) * (2/8) * ((1/2) * log(1/2,2) + 1/2 * log(1/2,2))
oc_ar <- (-1) * (1/8) * ((1/1) * log(1/1,2))
oc_remainder <- oc_tr + oc_pr + oc_ag + oc_ar
oc_ig <- ent - oc_remainder
oc_ig
```

```
## [1] 0.704434
```

```
# Information gain ratio from Education, Marital Status and Occupation
# Education
```

```r
igr_ed <- ed_ig / entropy(ex3data$EDUCATION)
igr_ed
```

```
## [1] 0.5212966
```

```r
# Marital Status
igr_ms <- ms_ig / entropy(ex3data$MARITAL_STATUS)
igr_ms
```

```
## [1] 0.3904238
```

```r
# Occupation
igr_oc <- oc_ig / entropy(ex3data$OCCUPATION)
igr_oc
```

```
## [1] 0.3696576
```

```r
# Information gain ratio using Gini Index from Education, Marital Status and Occupation
g <- gini(ex3data$ANNUAL_INCOME)
g
```

```
## [1] 0.53125
```

```r
# Education
ed_hs_gini_rem <- (4/8) * (1 - ((2/4)^2 + (2/4)^2))
ed_ba_gini_rem <- (3/8) * (1 - ((3/3)^2))
ed_do_gini_rem <- (1/8) * (1 - ((1/1)^2))
ig_ed_gini <- g - (ed_hs_gini_rem + ed_ba_gini_rem + ed_do_gini_rem)
ig_ed_gini
```

```
## [1] 0.28125
```

```r
# Marital Status
ms_nm_gini_rem <- (3/8) * (1 - ((2/3)^2 + (1/3)^2))
ms_ma_gini_rem <- (4/8) * (1 - ((3/4)^2 + (1/4)^2))
ms_di_gini_rem <- (1/8) * (1 - ((1/1)^2))
ig_ms_gini <- g - (ms_nm_gini_rem + ms_ma_gini_rem + ms_di_gini_rem)
ig_ms_gini
```

```
## [1] 0.1770833
```

```r
# Occupation
oc_ag_gini_rem <- (2/8) * (1 - ((2/2)^2))
oc_ar_gini_rem <- (1/8) * (1 - ((1/1)^2))
oc_pr_gini_rem <- (3/8) * (1 - ((2/3)^2 + (1/3)^2))
oc_tr_gini_rem <- (2/8) * (1 - ((2/2)^2))
ig_oc_gini <- g - (oc_ag_gini_rem + oc_ar_gini_rem + oc_pr_gini_rem + oc_tr_gini_rem)
ig_oc_gini
```

```
## [1] 0.3645833
```

Exercise 4

Since the point of this exercise is to follow the algorithm, no point in coding something to do the work in R. So, starting from bottom-up, left-right:

Blood Pressure errors (CHEST PAIN FALSE): 0 errors in 3 data points Left-Node Errors (CHEST PAIN False, BLOOD PRESSURE High) : 2 errors on 2 data points Right-Node Errors (CHEST PAIN False, BLOOD PRESSURE False) : 0 errors in 1 data point

We convert the BLOOD PRESSURE subtree into a leaf.

Now we have a tree that shows CHEST PAIN, Left Node (False) is False, Right Node(True) is True. Following the errors: CHEST PAIN FALSE: 3 errors in 5 data points LEFT NODE (False): 0 errors in 3 data points RIGHT NODE (True): 0 errors in 2 data points.

So we don't prune any other leafts.

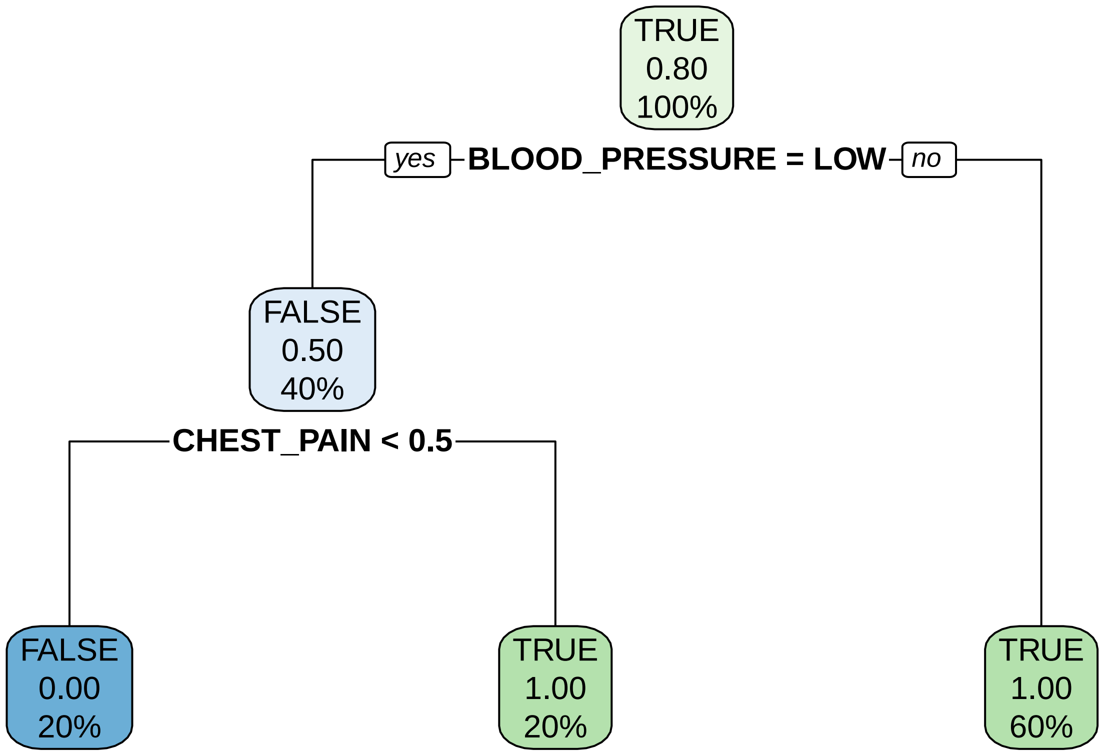
```
valid <- data.frame(CHEST_PAIN = c(FALSE,TRUE,FALSE,TRUE,FALSE), BLOOD_PRESSURE = c("HIGH","LOW","LOW",
valid
```

```
##   CHEST_PAIN BLOOD_PRESSURE HEART_DISEASE
## 1      FALSE           HIGH         FALSE
## 2       TRUE            LOW          TRUE
## 3      FALSE            LOW         FALSE
## 4       TRUE           HIGH          TRUE
## 5      FALSE           HIGH         FALSE
```

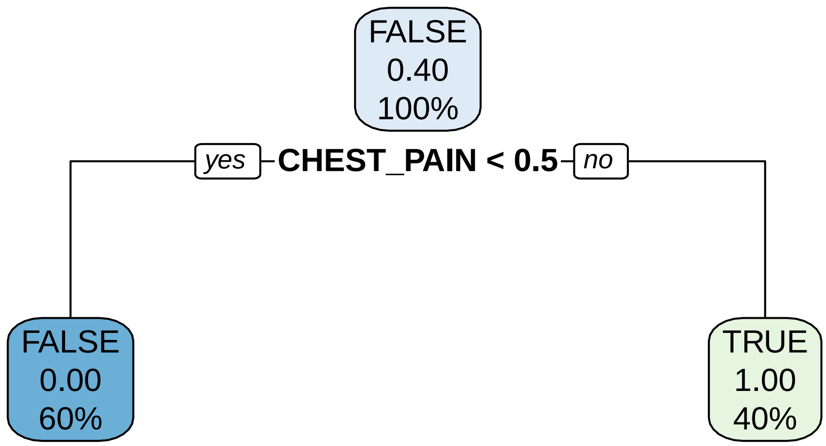
```
tree <- data.frame(CHEST_PAIN = c(FALSE,TRUE,FALSE,TRUE,FALSE), BLOOD_PRESSURE = c("HIGH","LOW","LOW","
tree
```

```
##   CHEST_PAIN BLOOD_PRESSURE HEART_DISEASE
## 1      FALSE           HIGH          TRUE
## 2       TRUE            LOW          TRUE
## 3      FALSE            LOW         FALSE
## 4       TRUE           HIGH          TRUE
## 5      FALSE           HIGH          TRUE
```

```
x <- rpart(HEART_DISEASE ~ BLOOD_PRESSURE + CHEST_PAIN, data = tree, method="class",control = rpart.con
rpart.plot::rpart.plot(x)
```




```r
y <- rpart(HEART_DISEASE ~ BLOOD_PRESSURE + CHEST_PAIN, data = valid, method="class",control = rpart.co
rpart.plot::rpart.plot(y)
```


FALSE
0.40
100%
yes
CHEST_PAIN < 0.5
no
FALSE
0.00
60%
TRUE
1.00
40%