# Homework 4.1

## DecisionTree algorithms, C5.0 - Identifying Risky Bank Loans

STEP 1: Loading the data The original datset is available at the UCI Machine Learning Data Repository. The data was donated by Hans Hofmann of the University of Hamburg, and contains loan information from a credit agency in Germany.The original dataset was modified by the authors of the Machine Learning with R book to eliminate some preprocessing steps. The dataset contains 1000 observations with 17 variables.

# Reading data into R  
credit <- read.csv("credit.csv")

STEP 2: Exploring and preparing the data As we can see below, some of the variables are factors, some are numeric (integers). The output variable "Default" has two levels - "yes" and "no".

str(credit)

## 'data.frame': 1000 obs. of 17 variables:  
## $ checking\_balance : Factor w/ 4 levels "< 0 DM","> 200 DM",..: 1 3 4 1 1 4 4 3 4 3 ...  
## $ months\_loan\_duration: int 6 48 12 42 24 36 24 36 12 30 ...  
## $ credit\_history : Factor w/ 5 levels "critical","good",..: 1 2 1 2 4 2 2 2 2 1 ...  
## $ purpose : Factor w/ 6 levels "business","car",..: 5 5 4 5 2 4 5 2 5 2 ...  
## $ amount : int 1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...  
## $ savings\_balance : Factor w/ 5 levels "< 100 DM","> 1000 DM",..: 5 1 1 1 1 5 4 1 2 1 ...  
## $ employment\_duration : Factor w/ 5 levels "< 1 year","> 7 years",..: 2 3 4 4 3 3 2 3 4 5 ...  
## $ percent\_of\_income : int 4 2 2 2 3 2 3 2 2 4 ...  
## $ years\_at\_residence : int 4 2 3 4 4 4 4 2 4 2 ...  
## $ age : int 67 22 49 45 53 35 53 35 61 28 ...  
## $ other\_credit : Factor w/ 3 levels "bank","none",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ housing : Factor w/ 3 levels "other","own",..: 2 2 2 1 1 1 2 3 2 2 ...  
## $ existing\_loans\_count: int 2 1 1 1 2 1 1 1 1 2 ...  
## $ job : Factor w/ 4 levels "management","skilled",..: 2 2 4 2 2 4 2 1 4 1 ...  
## $ dependents : int 1 1 2 2 2 2 1 1 1 1 ...  
## $ phone : Factor w/ 2 levels "no","yes": 2 1 1 1 1 2 1 2 1 1 ...  
## $ default : Factor w/ 2 levels "no","yes": 1 2 1 1 2 1 1 1 1 2 ...

# Looking closely at "Default" (output) variable  
table(credit$default)

##   
## no yes   
## 700 300

Our next step is to create training and test sets. To perform randomized sampling, we use the function sample(), which utilizes mathematical function called a pseudorandom number generator. Since we have 1000 observations, we allocate 900 observations (90%) to the training set and 100 observations (10%) to the test set. At the end of this step, we want to make sure that proportions of records belonging to the classes "Defaulted"("yes") and "Did Not Default"("no") inside of the training and test sets are very similar to the proroptions of the entire dataset.

# Setting the seed to get the reproducable results  
set.seed(123)  
# Identifying the size of the training set  
train\_sample <- sample(1000, 900)  
# Constructing traing and test sets  
credit\_train <- credit[train\_sample, ]  
credit\_test <- credit[-train\_sample, ]  
  
# Checking the class proportions  
prop.table(table(credit\_train$default))

##   
## no yes   
## 0.7033333 0.2966667

prop.table(table(credit\_test$default))

##   
## no yes   
## 0.67 0.33

STEP 3: Training a model on the data The next step is to build an actual Decision Tree. To be able to do that, we need to install and load the C5.0 package. After the training the model, we can take a look at the Tree that we grew. It has 57 decision nodes.

library(C50)  
credit\_model <- C5.0(credit\_train[-17], credit\_train$default)  
summary(credit\_model)

##   
## Call:  
## C5.0.default(x = credit\_train[-17], y = credit\_train$default)  
##   
##   
## C5.0 [Release 2.07 GPL Edition] Tue Apr 25 19:08:55 2017  
## -------------------------------  
##   
## Class specified by attribute `outcome'  
##   
## Read 900 cases (17 attributes) from undefined.data  
##   
## Decision tree:  
##   
## checking\_balance in {> 200 DM,unknown}: no (412/50)  
## checking\_balance in {< 0 DM,1 - 200 DM}:  
## :...credit\_history in {perfect,very good}: yes (59/18)  
## credit\_history in {critical,good,poor}:  
## :...months\_loan\_duration <= 22:  
## :...credit\_history = critical: no (72/14)  
## : credit\_history = poor:  
## : :...dependents > 1: no (5)  
## : : dependents <= 1:  
## : : :...years\_at\_residence <= 3: yes (4/1)  
## : : years\_at\_residence > 3: no (5/1)  
## : credit\_history = good:  
## : :...savings\_balance in {> 1000 DM,500 - 1000 DM}: no (15/1)  
## : savings\_balance = 100 - 500 DM:  
## : :...other\_credit = bank: yes (3)  
## : : other\_credit in {none,store}: no (9/2)  
## : savings\_balance = unknown:  
## : :...other\_credit = bank: yes (1)  
## : : other\_credit in {none,store}: no (21/8)  
## : savings\_balance = < 100 DM:  
## : :...purpose in {business,car0,renovations}: no (8/2)  
## : purpose = education:  
## : :...checking\_balance = < 0 DM: yes (4)  
## : : checking\_balance = 1 - 200 DM: no (1)  
## : purpose = car:  
## : :...employment\_duration = > 7 years: yes (5)  
## : : employment\_duration = unemployed: no (4/1)  
## : : employment\_duration = < 1 year:  
## : : :...years\_at\_residence <= 2: yes (5)  
## : : : years\_at\_residence > 2: no (3/1)  
## : : employment\_duration = 1 - 4 years:  
## : : :...years\_at\_residence <= 2: yes (2)  
## : : : years\_at\_residence > 2: no (6/1)  
## : : employment\_duration = 4 - 7 years:  
## : : :...amount <= 1680: yes (2)  
## : : amount > 1680: no (3)  
## : purpose = furniture/appliances:  
## : :...job in {management,unskilled}: no (23/3)  
## : job = unemployed: yes (1)  
## : job = skilled:  
## : :...months\_loan\_duration > 13: [S1]  
## : months\_loan\_duration <= 13:  
## : :...housing in {other,own}: no (23/4)  
## : housing = rent:  
## : :...percent\_of\_income <= 3: yes (3)  
## : percent\_of\_income > 3: no (2)  
## months\_loan\_duration > 22:  
## :...savings\_balance = > 1000 DM: no (2)  
## savings\_balance = 500 - 1000 DM: yes (4/1)  
## savings\_balance = 100 - 500 DM:  
## :...credit\_history in {critical,poor}: no (14/3)  
## : credit\_history = good:  
## : :...other\_credit = bank: no (1)  
## : other\_credit in {none,store}: yes (12/2)  
## savings\_balance = unknown:  
## :...checking\_balance = 1 - 200 DM: no (17)  
## : checking\_balance = < 0 DM:  
## : :...credit\_history = critical: no (1)  
## : credit\_history in {good,poor}: yes (12/3)  
## savings\_balance = < 100 DM:  
## :...months\_loan\_duration > 47: yes (21/2)  
## months\_loan\_duration <= 47:  
## :...housing = other:  
## :...percent\_of\_income <= 2: no (6)  
## : percent\_of\_income > 2: yes (9/3)  
## housing = rent:  
## :...other\_credit = bank: no (1)  
## : other\_credit in {none,store}: yes (16/3)  
## housing = own:  
## :...employment\_duration = > 7 years: no (13/4)  
## employment\_duration = 4 - 7 years:  
## :...job in {management,skilled,  
## : : unemployed}: yes (9/1)  
## : job = unskilled: no (1)  
## employment\_duration = unemployed:  
## :...years\_at\_residence <= 2: yes (4)  
## : years\_at\_residence > 2: no (3)  
## employment\_duration = 1 - 4 years:  
## :...purpose in {business,car0,education}: yes (7/1)  
## : purpose in {furniture/appliances,  
## : : renovations}: no (7)  
## : purpose = car:  
## : :...years\_at\_residence <= 3: yes (3)  
## : years\_at\_residence > 3: no (3)  
## employment\_duration = < 1 year:  
## :...years\_at\_residence > 3: yes (5)  
## years\_at\_residence <= 3:  
## :...other\_credit = bank: no (0)  
## other\_credit = store: yes (1)  
## other\_credit = none:  
## :...checking\_balance = 1 - 200 DM: no (8/2)  
## checking\_balance = < 0 DM:  
## :...job in {management,skilled,  
## : unemployed}: yes (2)  
## job = unskilled: no (3/1)  
##   
## SubTree [S1]  
##   
## employment\_duration in {< 1 year,4 - 7 years}: no (4)  
## employment\_duration in {> 7 years,1 - 4 years,unemployed}: yes (10)  
##   
##   
## Evaluation on training data (900 cases):  
##   
## Decision Tree   
## ----------------   
## Size Errors   
##   
## 56 133(14.8%) <<  
##   
##   
## (a) (b) <-classified as  
## ---- ----  
## 598 35 (a): class no  
## 98 169 (b): class yes  
##   
##   
## Attribute usage:  
##   
## 100.00% checking\_balance  
## 54.22% credit\_history  
## 47.67% months\_loan\_duration  
## 38.11% savings\_balance  
## 14.33% purpose  
## 14.33% housing  
## 12.56% employment\_duration  
## 9.00% job  
## 8.67% other\_credit  
## 6.33% years\_at\_residence  
## 2.22% percent\_of\_income  
## 1.56% dependents  
## 0.56% amount  
##   
##   
## Time: 0.0 secs

STEP 4: Evaluating model performance Next step is to test a model and evalute the performance looking at the confuson matrix. The total Accuracy rate is 73%. Unfortunately, the biggest concern is the high number of False Negatives (19) - those are "Defaulted" cases predicted as "No Default". This is very critical for the model, since those errors are the most costly to the financial organizations.

credit\_pred <- predict(credit\_model, credit\_test)  
  
# Displaying the confusion matrix  
library(gmodels)  
CrossTable(credit\_test$default, credit\_pred,  
 prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,  
 dnn = c('actual default', 'predicted default'))

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 100   
##   
##   
## | predicted default   
## actual default | no | yes | Row Total |   
## ---------------|-----------|-----------|-----------|  
## no | 59 | 8 | 67 |   
## | 0.590 | 0.080 | |   
## ---------------|-----------|-----------|-----------|  
## yes | 19 | 14 | 33 |   
## | 0.190 | 0.140 | |   
## ---------------|-----------|-----------|-----------|  
## Column Total | 78 | 22 | 100 |   
## ---------------|-----------|-----------|-----------|  
##   
##

STEP 5: Improving model performance Boosting Adaptive Boosting can be applied to C5.0 algorithm to improve the performance. We can do this by adding trials parameter. 10 trials is a good number to use. Boosting helps to build a stronger classifier with iteratively learning weak classifiers.

# Adding trials parameter to the model  
credit\_boost10 <- C5.0(credit\_train[-17], credit\_train$default,  
 trials = 10)  
credit\_boost10

##   
## Call:  
## C5.0.default(x = credit\_train[-17], y = credit\_train$default, trials = 10)  
##   
## Classification Tree  
## Number of samples: 900   
## Number of predictors: 16   
##   
## Number of boosting iterations: 10   
## Average tree size: 47.5   
##   
## Non-standard options: attempt to group attributes

# Testing updated model  
credit\_boost\_pred10 <- predict(credit\_boost10, credit\_test)  
CrossTable(credit\_test$default, credit\_boost\_pred10,  
 prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,  
 dnn = c('actual default', 'predicted default'))

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 100   
##   
##   
## | predicted default   
## actual default | no | yes | Row Total |   
## ---------------|-----------|-----------|-----------|  
## no | 62 | 5 | 67 |   
## | 0.620 | 0.050 | |   
## ---------------|-----------|-----------|-----------|  
## yes | 13 | 20 | 33 |   
## | 0.130 | 0.200 | |   
## ---------------|-----------|-----------|-----------|  
## Column Total | 75 | 25 | 100 |   
## ---------------|-----------|-----------|-----------|  
##   
##

The size of the new Tree is now smaller. The Accuracy rate of the new model is 88%. Boosting gave us a slight improvement of the accuracy. It also reduced the number of False Negatives to 13.

Another way to improve the model is to assing weights to the type of errors. As we discussed above, classifying the actual "Default" cases as negative (False Negatives) are the most costly. We can assign the highest (critical) weight of 4 to this type of error. We can also assign the weight of 1 to the False Positives error, since those cases are missed opportunities for the bank.

matrix\_dimensions <- list(c("no", "yes"), c("no", "yes"))  
names(matrix\_dimensions) <- c("actual", "predicted")  
error\_cost <- matrix(c(0, 4, 1, 0), nrow = 2, dimnames = matrix\_dimensions)  
error\_cost

## predicted  
## actual no yes  
## no 0 1  
## yes 4 0

# New model with cost parameter incorporated  
credit\_cost <- C5.0(credit\_train[-17], credit\_train$default,  
 costs = error\_cost)  
credit\_cost\_pred <- predict(credit\_cost, credit\_test)  
  
CrossTable(credit\_test$default, credit\_cost\_pred,  
 prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,  
 dnn = c('actual default', 'predicted default'))

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 100   
##   
##   
## | predicted default   
## actual default | no | yes | Row Total |   
## ---------------|-----------|-----------|-----------|  
## no | 66 | 1 | 67 |   
## | 0.660 | 0.010 | |   
## ---------------|-----------|-----------|-----------|  
## yes | 29 | 4 | 33 |   
## | 0.290 | 0.040 | |   
## ---------------|-----------|-----------|-----------|  
## Column Total | 95 | 5 | 100 |   
## ---------------|-----------|-----------|-----------|  
##   
##

Unfortunately, incorporating the cost factor did not improve the model performance. Most likely in the real world banks would be hesitant to adopt the model due to the low Accuracy rate and significantly high number of False Negatives in the model. One of the factors conributing to that could be the relatively small size of our training set.