Homework 5

Marc Mendez & Joel Cantero

5th May, 2019

First of all, we are going to install and load all the libraries we need for this exercise.

```r
packages <- c("rpart","ROCR", "readxl", "randomForest", "mice")
for (package in packages) {
    if(!require(package,  character.only=TRUE)){
        install.packages(package, repos="http://cran.rstudio.com")
        library(package,  character.only=TRUE)
    }
}
```

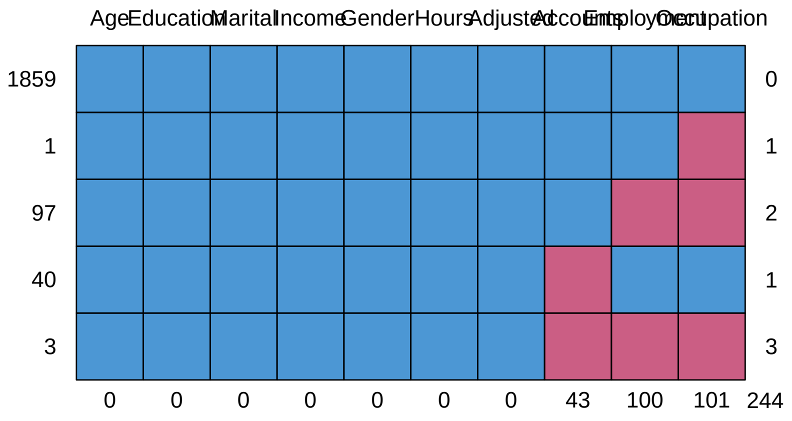
1. Read the Audit.xlsx file and convert it to the csv extension.

Once we have loaded and installed all the packages, we are going to load the excel file. Also, we are going to convert Employment. Education, Marital, Occupation, Gender, Accounts and Adjusted attributes to factors. Then, we will convert it to CSV file thanks to write.csv2 function.

```r
audit <- read_excel("audit.xlsx", na = "NA")
factors <- c("Employment", "Education", "Marital", "Occupation", "Gender",
             "Accounts", "Adjusted")
for (fac in factors) {
  audit[fac] <- lapply(audit[fac], factor)
}
write.csv2(audit, "audit.csv")
```

2. The goal is to use a decision tree to predict the binary "Adjusted" variable, whether the individuals had made a correct financial statement or not. Decide which predictors you would use and eventually preprocess these variables.

First of all, we have to remove these attributes that will not help us for splitting the tree. We can observe that "ID", "Deductions" and "Adjusted" are related to statement made and we do not need them. After that, we will imput missing values with MICE function. We have found 244 missing values and we have decided to use MICE because it is a small percentatge of all our instances.



3. Select the 1/3 of the last observations as test data.

We have to select the last 33% of data instances as test, and the first 66% instances as training data.

```
test <- imputedAudit[seq(0.66*nrow(imputedAudit), nrow(imputedAudit)), ]
training <- imputedAudit[-seq(0.66*nrow(imputedAudit), nrow(imputedAudit)), ]
```

4. Obtain the decision tree to predict whether the variable "Adjusted" on the training data. Decide the cutoff value for taking the decision.

```
##
## Classification tree:
## rpart(formula = Adjusted ~ ., data = training, method = "class",
##     control = rpart.control(xval = 10, cp = 0.001))
##
## Variables actually used in tree construction:
## [1] Accounts   Age        Education  Employment Gender     Hours
## [7] Income     Marital    Occupation
##
## Root node error: 317/1319 = 0.24033
##
## n= 1319
##
##           CP nsplit rel error  xerror     xstd
## 1  0.1451104      0   1.00000 1.00000 0.048953
## 2  0.0362776      2   0.70978 0.73186 0.043619
## 3  0.0189274      4   0.63722 0.71293 0.043170
## 4  0.0084122      7   0.58044 0.69085 0.042632
## 5  0.0063091     10   0.55521 0.69401 0.042710
## 6  0.0052576     12   0.54259 0.71293 0.043170
## 7  0.0047319     15   0.52681 0.73186 0.043619
## 8  0.0037855     19   0.50789 0.74448 0.043913
## 9  0.0031546     24   0.48896 0.75394 0.044130
## 10 0.0021030     26   0.48265 0.75394 0.044130
## 11 0.0015773     29   0.47634 0.75394 0.044130
## 12 0.0010000     33   0.47003 0.76341 0.044344
```

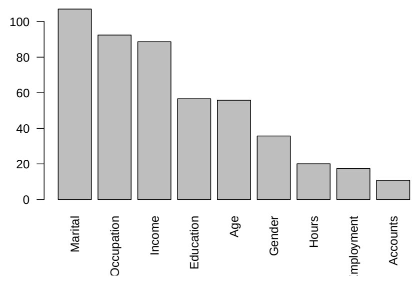
If we calculate the cutoff and with the cutoff we look at the table we obtain the following:

```
## Cutoff idx: 4  With min value: 0.7334838
```

```
## CP: 0.008412198  nsplit: 7 rel error: 0.5804416 xerror: 0.6908517 xstd: 0.0426321
```

In our case the cutoff will be on 5 splits.

5. Plot the importance of variables in the prediction.



As we can see in this plot, the three most important variables are: marital, occupation and income.

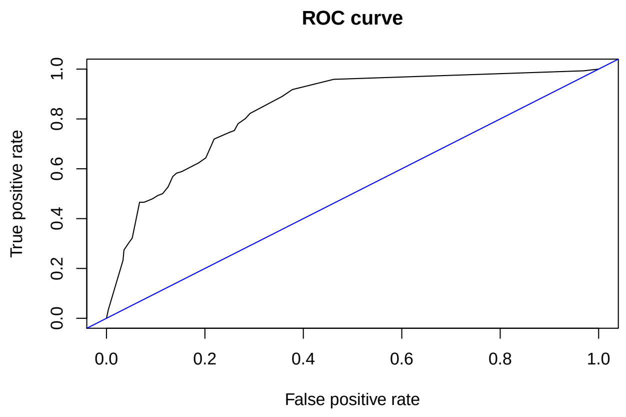
6. Compute the accuracy, precision, recall and AUC on the test individuals.

Before calculating the accuracy, precision, recall and AUC, we need to calculate the confusion matrix. After it we can start calculating the variables said earlier.

```
prediction <- predict(rt, test, type="class")
(results <- table(test$Adjusted, prediction))
```

```
##    prediction
##       0   1
##   0 479  56
##   1  74  72
```

```
# Accuracy
error <- (results[1,2] + results[2,1])/nrow(test)
accuracy <- 1 -  error
# Precision
precision_p <- results[1,1]/(results[1,1] + results[2,1])
precision_n <- results[2,2]/(results[1,2] + results[2,2])
precision <- (precision_n + precision_p)/2
# Recall
recall <- results[1,1]/(results[1,1] + results[1,2])
```



We obtain a very high accuracy. If we look at the amount of results predicted we see that our precision is also high, and last but not least, the recall value which is also very high.

```
## Accuracy:  0.8091043
```

```
## Precision:  0.7143422
## Recall:  0.8953271
```

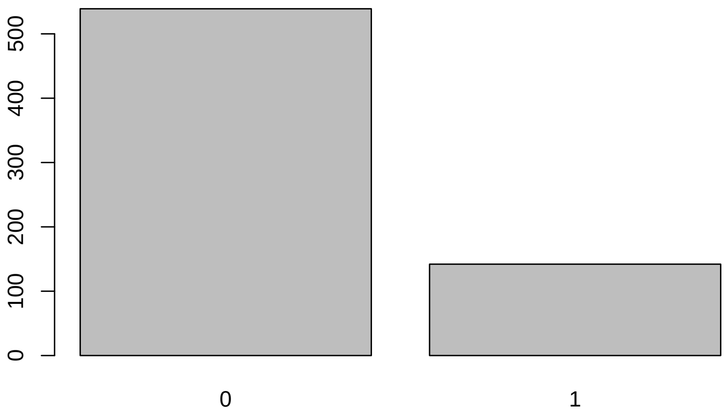
```
## AUC:  0.8359237
```

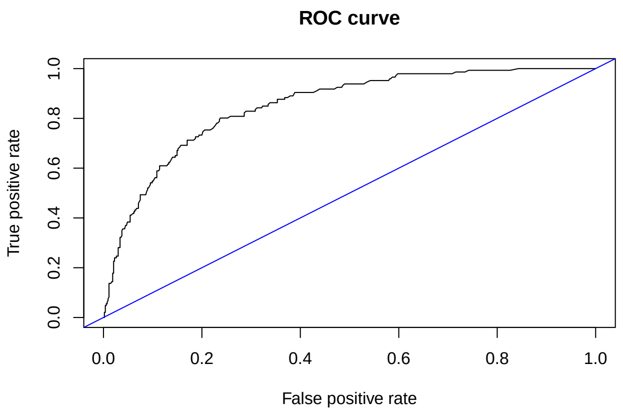
7. Perform a Random Forest on the same data.

As we did previously, we have to build the model and then test the results with the test data (the same split as we have used before, if we want to compare it).

The goal of this exercise is to compare the previous results with a random forest. For this reason, we will calculate again the accuracy, precision, recall and AUC.

```
##
## Call:
##  randomForest(formula = Adjusted ~ ., data = training, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 16.38%
## Confusion matrix:
##     0   1 class.error
## 0 908  94  0.09381238
## 1 122 195  0.38485804
```





As we can see the values here are all better than the others except for recall variable which is slightly lower.

```
## Accuracy:  0.8237885
```

```
## Precision:  0.7382607
## Recall:  0.8915888
```

```
## AUC:  0.8496543
```

Conclusions

To conclude, we can say that all the metrics have been improved using a random forest just using 500 trees (we can see that if we print randomForest variable). If we just use one decision tree (the previous exercise) against a random forest, we can observe that the results are not good as random forest ones.