Coursework

EDHEC Business School

Master in Management Financial Economics

Introduction to Machine Learning through Classification

Authors:
Matteo Mario Di Venti; Lorenzo Perlini (ID: 80389; 80357)

Date: May 29, 2022

1 Exercise 1: General questions

1.1 Question 1

What are the advantages and disadvantages of a low-dimensional simple model over a very flexible (complex) model?

The advantages of a low-dimensional simple model over a very flexible (complex) model can be summarised as:

- Easier to understand and thus to implement without errors
- Less overfitting as the variance of residuals is lower
- It performs better than complex models Out-of-sample
- Finally, according to Occam's razor argument: simpler models should be chosen over complex models as they avoid redundant elements

While, the disadvantages of a low-dimensional simple model over a very flexible (complex) model can be summarised as:

- Higher bias as it has been fitted loosely
- captures data less well
- lower flexibility
- It performs worse than complex models In-sample

1.2 Question 2

A student wants to automatically identify which of her photos were taken indoors, outdoors, day or night. Should you recommend to use four binary classifiers, two binary classifiers or a multi-class classifier? Justify your answer.

To recognize between photos taken indoors, outdoors, day or night we would recommend two binary classifiers. As indoor is mutually exclusive with outdoor and day is mutually exclusive with night the two pairs can be correctly modeled by two binary classifiers:

$$environment = \begin{cases} 1 & if\ indoor \\ 0 & if\ outdoor \end{cases}$$

$$time = \begin{cases} 1 & if\ day \\ 0 & if\ night \end{cases}$$

Using 4 binary classifiers would be redundant because we have two mutually exclusive pairs.
Using 1 4-class classifiers would incorrectly model the problem as it will not be able to correctly classify for example photos that are taken in a outdoor environment in day time as they will either be classified as outdoor or day and not both.

2 Exercise 2: Precision, Recall and Accuracy

2.1 Question 1

What is the main difference between the accuracy and the precision of a model?

The **accuracy** of a model is defined as the percentage of outputs correctly labeled.
Denoting TP and TN respectively the number of correctly labeled positive and negative outputs.

$$Accuracy = \frac{TP + TN}{n}$$

where n is the number of outputs.

The **precision** of a model (also called positive prediction value) is defined as the proportion of positives among positive predictions.
Denoting FP as the number of false positive outputs.

$$Precision = \frac{TP}{TP + FP}$$

Therefore the main difference is that accuracy takes into account the correctly labeled between all outputs, while precision takes into account the correctly labeled between all the similarly predicted.

2.2 Question 2

We are interested in a ”random” binary classification algorithm, that is to say that predicts ”negative” or ”positive” with an occurrence of 0.5 for each class. Suppose that the training set contains 80% of ”positive” labels and therefore 20% of ”negative” labels. Determine the accuracy, recall, and precision. Interpret.

Taking into consideration the ”random” binary classification we construct the following Confusion Matrix table for 100 cases.

		Actual value		
		p	**n**	**total**
Prediction outcome	**p′**	True Positive =40	False Positive =10	P′ = 50
	n′	False negative =40	True Negative =10	N′ = 50
	total	P=80	N=20	TOT=100

Therefore the accuracy, precision and recall are given by

$$Accuracy = \frac{TP + TN}{n} = \frac{40 + 10}{100} = 50\%$$
$$Precision = \frac{TP}{TP + FP} = \frac{40}{50} = 80\%$$
$$Recall = \frac{TP}{TP + FN} = \frac{40}{40 + 40} = 50\%$$

The interpretation is that the ”random” binary classification is a simple benchmark for all other models given its accuracy and recall of 50%. The precision is high just because the dataset is unbalanced towards positives. If we calculate precision for negatives the value would be 20%.

2.3 Question 3

We are interested in a binary classification algorithm by majority rule, that is to say that predicts (here) only "positives". Suppose that the training set contains 78% of "positive" labels and therefore 22% of "negative" labels. Determine the accuracy, recall, and precision. Interpret.

Taking into consideration the majority rule classification we construct the following Confusion Matrix table for 100 cases.

		Actual value		
		p	**n**	**total**
Prediction outcome	**p′**	True Positive =78	False Positive =22	P′ = 100
	n′	False Negative =0	True Negative =0	N′ = 0
	total	P=78	N=22	TOT=100

Therefore the accuracy, precision and recall are given by

$$Accuracy = \frac{TP + TN}{n} = \frac{78 + 0}{100} = 78\%$$

$$Precision = \frac{TP}{TP + FP} = \frac{78}{100} = 78\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{78}{78} = 100\%$$

The interpretation is that the majority rule works very well when the dataset is strongly unbalanced. As we can see, the accuracy and precision are simply given by the prevalence of the positive over the negatives in the dataset. The recall is high for the same reason. The majority rule expresses another benchmark when evaluating a model.

3 Exercise 3: Roc curve and AUC

Note: As encourage and approved by the professor in class, we decided to solve the exercises in R. The relevant R file is attached to the report under the name "Exercise_3_Roc_curve_and_AUC.R". The input file is an Excel file containing the above table. The input file attached as "coursework_exercise_3.xlsx".

3.1 Question 1

Construct and plot the ROC curve with $\alpha \in [0;1]$ and an increment of 0.10

We construct the Roc curve by calculating the TPR and FPR for every threshold α between 0 and 1 in steps 0.1. Plotting we start from $\alpha = 1$ for convention.

TP	TN	FP	FN	TPR	FPR	threshold	Trapeze	AUC
0	12	0	8	0	0	1	0	0
2	11	1	6	0.25	0.083333333	0.9	0.010416667	0.010416667
3	10	2	5	0.375	0.166666667	0.8	0.026041667	0.036458333
4	9	3	4	0.5	0.25	0.7	0.036458333	0.072916667
5	8	4	3	0.625	0.333333333	0.6	0.046875	0.119791667
5	7	5	3	0.625	0.416666667	0.5	0.052083333	0.171875
6	5	7	2	0.75	0.583333333	0.4	0.114583333	0.286458333
6	3	9	2	0.75	0.75	0.3	0.125	0.411458333
8	3	9	0	1	0.75	0.2	0	0.411458333
8	2	10	0	1	0.833333333	0.1	0.083333333	0.494791667
8	0	12	0	1	1	0	0.166666667	0.661458333

Here is the plotted result.

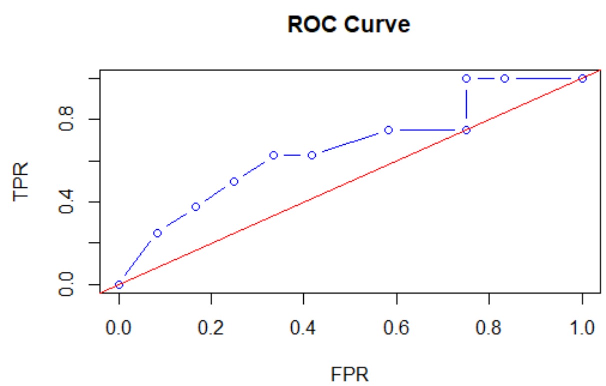


Figure 1: Roc curve for exercise 3

3.2 Question 2

Is the model considered better than a random classification? Explain carefully.

The model is considered better than a random classification as the Roc curve is above the bisecting line x=y which denotes the roc curve of a random classifier.

3.3 Question 3

What is the AUC?

The **Area Under the Curve** also called **AUC** is the numeric value indicating extension of the area that is comprised between the Roc curve, the x axis (denoting antispecificity) and the line x=1. Hastie [2017] Gareth [2021]
From the R calculations the AUC is **0.66145833**

4 Exercise 4: Application (R code) Pelgrin [2022]

Please see the attached R markdown file

References

Robert; Friedman Jerome Hastie, Trevor; Tibshirani. **The Elements of statistical learning**. Springer, 2017. pages 6

Daniela;Hastie Trevor; Tibshirani Robert Gareth, James; Witten. **An introduction to Statistical Learning**. Springer, 2021. pages 6

Florian Pelgrin. Introduction to machine learning through classification. 2022. pages 6

ML Coursework

Matteo Mario Di Venti, Lorenzo Perlini

29/05/2022

Preamble

Please note that the dataset included (german.data) is the raw subset not yet encoded. We decided to automate the encoding in the first section of the R markdown that follows.

```
library(ggplot2)
library(tidyverse)
library(rpart)
library(rpart.plot)
library(plotROC)
library(ROCR)
library(pROC)
library(caret)
library(ggpubr)
library(InformationValue)
library(tidyverse)
library(leaps)
library(glmnet)
library(mlbench)
#library(doMC)
library(caret)
library(class)
library(reshape2)
library(dplyr)
```

Question 1

We start by downloading the German credit data

```
####Please place your german.data file in the same location as the code###
getwd()

## [1] "C:/Users/mmd2218/OneDrive - EDHEC/Desktop"

data <- read.table("german.data", sep = "")

margin.table(prop.table(table(data$V1, data$V3, data$V4, data$V6, data$V7,
                              data$V9, data$V14)),6)
```

```
##
##   A91   A92   A93   A94
## 0.050 0.310 0.548 0.092

##We compute margins so to see the distribution of each category
## and to choose how to divide them
```

We prooceed by encoding all the explanatory variables wheter quantitative or qualitative

```
#1 Credit balance
Balance <- c()
for (i in 1:1000){
  if (data$V1[i] == "A12") {
    Balance[i]= 1
  } else if ( data$V1[i]=="A13") {
    Balance[i]= 1} else {
    Balance[i]= 0}
}

#That way, I have created a vector in which users with no account or zero
#balance are assgned to 0, users with some balance are assigned 1. The
#Distinction make sense, as the 2 groups have similar magnitude.

#2 Duration in months : it is nuymerical
Duration_months <- data$V2

#3 Credit history
History <- c()
for (i in 1:1000){
  if (data$V3[i] == "A30") {
    History[i]= 1
  } else if ( data$V3[i]=="A31") {
    History[i]= 1} else if ( data$V3[i]=="A32") {
      History[i]= 1} else {
      History[i]= 0}
}
#Here I have assigned 1 to users with good credit history, 0 to others

#4 Purpose
#For this group I will select 3 kinds of purspose : buying a car, buying
#something related to house and others
Purpose1 <- c()
for (i in 1:1000){
  if (data$V4[i] == "A40") {
    Purpose1[i]= 1
  } else if ( data$V4[i]=="A41") {
    Purpose1[i]= 1} else {
        Purpose1[i]= 0}
}
```

```r
Purpose2 <- c()
for (i in 1:1000){
  if (data$V4[i] == "A42") {
    Purpose2[i]= 1
  } else if (data$V4[i]=="A43") {
    Purpose2[i]= 1} else if ( data$V4[i]=="A44") {
      Purpose2[i]= 1} else if (data$V5[i]== "A45"){
        Purpose2[i] = 1
      } else {
        Purpose2[i]= 0}
}

#So, Purpose1 will tell me if the person has used the money to buy a car,
#Purpose2 will tell me if she used it for her house, the control group
#(all zeros) is composed by people using it for other purposes

#5 Credit ammount (numerical)
Credit_ammount <- data$V5

#6 Savings account
#Here I will create 4 groups with each value being the inferior boundary of
the set
Savings1 <- c()
for (i in 1:1000){
  if (data$V6[i] == "A61") {
    Savings1[i]= 0
  } else if ( data$V6[i]=="A62") {
    Savings1[i]= 100} else if (data$V6[i] == "A63"){
      Savings1[i]= 500}else if (data$V6[i] == "A64"){
        Savings1[i]= 1000} else {
      Savings1[i]= 0}
}

#7 Employment length
#Here I will create 3 groups : employed for less than 1 year (including
#unemployed), employed for 1 to 4 years, employed for more than 4 years

Employment_length1<- c()
for (i in 1:1000){
  if (data$V7[i] == "A71") {
    Employment_length1[i]= 1
  } else if ( data$V7[i]=="A74") {
    Employment_length1[i]= 1} else {
        Employment_length1[i]= 0}
}

Employment_length2<- c()
for (i in 1:1000){
  if (data$V7[i] == "A75") {
```

```
    Employment_length2[i]= 1
  } else {
      Employment_length2[i]= 0}
}

#So, Employment_length1 displays the users with 1 to 4 years of work,
#Employment_length2 the users with more than 4 years and the control group
#are the ones with less than 1 year + unemployed

#8 Installment rate in percentage of disposable income (numerical)
Installment_rate <- data$V8

#9 sex and marital status
#Here I divide by sex. However,
#there seems to be some kind of mistake in the explanation of data,
#as A95 value (single female) seems to be missing.
#I will proceed by considering A91 as male divorced/seprated,
#A92 as female divorced/separated/married, A93 as male single
#and A94 as male married/widowed.

Male <-c()
for (i in 1:1000){
  if (data$V9[i] != "A92") {
    Male[i]= 1
  } else {
    Male[i]= 0}
}

#10 Guarantor
Guarantor<- c()
for (i in 1:1000){
  if (data$V10[i] != "A101") {
    Guarantor[i]= 1
  } else {
    Guarantor[i]= 0}
}

#Here, the vector displays 1 if the user has a guarantor, 0 if not

#11 Present redisence since (numerical)
Residence_since <- data$V11

#12
House <- c()
for (i in 1:1000){
  if (data$V12[i] == "A121") {
    House[i]= 1
  } else {
      House[i]= 0}
```

```
}

Insurance <- c()
for (i in 1:1000){
  if (data$V12[i] == "A122") {
    Insurance[i]= 1
  } else {
    Insurance[i]= 0}
}

Car <-c()
for (i in 1:1000){
  if (data$V12[i] == "A123") {
    Car[i]= 1
  } else {
    Car[i]= 0}
}

#The vector house displays 1 if the users owns a house; if not, the
#vector Insurance displays 1 if she owns an insurance. If not, the vector
#car displays 1 if she owns a car

#13 Age (numerical)
Age<-data$V13

#14 Other installment plans
Other_plans<- c()
for (i in 1:1000){
  if (data$V14[i] != "A143") {
    Other_plans[i]= 1
  } else {
    Other_plans[i]= 0}
}
#Here the vector displays 1 if the user has a concurrent creditor, 0 if not

#15 Housing
FreeHousing <-c()
for (i in 1:1000){
  if (data$V15[i] == "A153") {
    FreeHousing[i]= 1
  } else {
    FreeHousing[i]= 0}
}

OwnHouse <- c()
for (i in 1:1000){
  if (data$V15[i] == "A152") {
    OwnHouse[i]= 1
  } else {
```

```
    OwnHouse[i]= 0}
}

# Here the vector Free housing displays the users with a free house, the
# vetor OwnHouse users who own their house and the ones renting have
# zeros in both

#16 Number of existing credits at this bank (numerical)
Number_credits <- data$V16

#17 Job
Skilled <- c()
for (i in 1:1000){
  if (data$V17[i] == "A173") {
    Skilled[i]= 1
  } else {
    Skilled[i]= 0}
}

Highly_Qualified <- c()
for (i in 1:1000){
  if (data$V17[i] == "A174") {
    Highly_Qualified[i]= 1
  } else {
    Highly_Qualified[i]= 0}
}

#Skilled displays 1 if the users is a skilled worker, Highly_Qualified
#displays 1 if she is highly qualified ; if both display zeros, the user
#is unemployed/unskilled

#18 Number of people being liable to provide maintenance for (numerical)
People_to_mantain <- data$V18

#19 Telephone
Telephone <- c()
for (i in 1:1000){
  if (data$V19[i] == "A192") {
    Telephone[i]= 1
  } else {
    Telephone[i]= 0}
}

#20 foreign worker
Foreign_worker <- c()
for (i in 1:1000){
  if (data$V20[i] == "A201") {
    Foreign_worker[i]= 1
  } else {
```

```
    Foreign_worker[i]= 0}
}

#21 Output
Output <- data$V21-1
```

Finally we create the dataframe

```
German <- cbind(Balance,Duration_months,History,Purpose1, Purpose2,
                Credit_ammount,Savings1,Employment_length1,
                Employment_length2,Installment_rate,Male,Guarantor,
                Residence_since,House,Insurance,Car,Age,Other_plans,
                FreeHousing,OwnHouse,Number_credits,Skilled,
                Highly_Qualified,People_to_mantain,Telephone,
                Foreign_worker, Output)
```

After having encode our dataset and having described each of its components, we want to have a closer look at its summary statistics:

```
summary(German)

##     Balance      Duration_months    History          Purpose1    
##  Min.   :0.000   Min.   : 4.0    Min.   :0.000   Min.   :0.000  
##  1st Qu.:0.000   1st Qu.:12.0    1st Qu.:0.000   1st Qu.:0.000  
##  Median :0.000   Median :18.0    Median :1.000   Median :0.000  
##  Mean   :0.332   Mean   :20.9    Mean   :0.619   Mean   :0.337  
##  3rd Qu.:1.000   3rd Qu.:24.0    3rd Qu.:1.000   3rd Qu.:1.000  
##  Max.   :1.000   Max.   :72.0    Max.   :1.000   Max.   :1.000  
##     Purpose2     Credit_ammount     Savings1       Employment_length1
##  Min.   :0.000   Min.   :  250   Min.   :   0.0   Min.   :0.000    
##  1st Qu.:0.000   1st Qu.: 1366   1st Qu.:   0.0   1st Qu.:0.000    
##  Median :0.000   Median : 2320   Median :   0.0   Median :0.000    
##  Mean   :0.473   Mean   : 3271   Mean   :  89.8   Mean   :0.236    
##  3rd Qu.:1.000   3rd Qu.: 3972   3rd Qu.:   0.0   3rd Qu.:0.000    
##  Max.   :1.000   Max.   :18424   Max.   :1000.0   Max.   :1.000    
##  Employment_length2 Installment_rate      Male         Guarantor    
##  Min.   :0.000      Min.   :1.000    Min.   :0.00   Min.   :0.000  
##  1st Qu.:0.000      1st Qu.:2.000    1st Qu.:0.00   1st Qu.:0.000  
##  Median :0.000      Median :3.000    Median :1.00   Median :0.000  
##  Mean   :0.253      Mean   :2.973    Mean   :0.69   Mean   :0.093  
##  3rd Qu.:1.000      3rd Qu.:4.000    3rd Qu.:1.00   3rd Qu.:0.000  
##  Max.   :1.000      Max.   :4.000    Max.   :1.00   Max.   :1.000  
##  Residence_since     House          Insurance          Car       
##  Min.   :1.000   Min.   :0.000   Min.   :0.000   Min.   :0.000  
##  1st Qu.:2.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000  
##  Median :3.000   Median :0.000   Median :0.000   Median :0.000  
##  Mean   :2.845   Mean   :0.282   Mean   :0.232   Mean   :0.332  
##  3rd Qu.:4.000   3rd Qu.:1.000   3rd Qu.:0.000   3rd Qu.:1.000  
##  Max.   :4.000   Max.   :1.000   Max.   :1.000   Max.   :1.000  
##       Age         Other_plans      FreeHousing       OwnHouse    
##  Min.   :19.00   Min.   :0.000   Min.   :0.000   Min.   :0.000  
```

```
##  1st Qu.:27.00   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000  
##  Median :33.00   Median :0.000   Median :0.000   Median :1.000  
##  Mean   :35.55   Mean   :0.186   Mean   :0.108   Mean   :0.713  
##  3rd Qu.:42.00   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:1.000  
##  Max.   :75.00   Max.   :1.000   Max.   :1.000   Max.   :1.000  
##  Number_credits     Skilled     Highly_Qualified People_to_mantain
##  Min.   :1.000   Min.   :0.00   Min.   :0.000    Min.   :1.000    
##  1st Qu.:1.000   1st Qu.:0.00   1st Qu.:0.000    1st Qu.:1.000    
##  Median :1.000   Median :1.00   Median :0.000    Median :1.000    
##  Mean   :1.407   Mean   :0.63   Mean   :0.148    Mean   :1.155    
##  3rd Qu.:2.000   3rd Qu.:1.00   3rd Qu.:0.000    3rd Qu.:1.000    
##  Max.   :4.000   Max.   :1.00   Max.   :1.000    Max.   :2.000    
##    Telephone     Foreign_worker      Output   
##  Min.   :0.000   Min.   :0.000   Min.   :0.0  
##  1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.0  
##  Median :0.000   Median :1.000   Median :0.0  
##  Mean   :0.404   Mean   :0.963   Mean   :0.3  
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.0  
##  Max.   :1.000   Max.   :1.000   Max.   :1.0
```

For the majority of our variables (namely, the qualitative ones) the only use of descriptive statistics is to see by how much the condition is more/less respected than not. In other words, how many are the ones relatively to the zeros. For example, if mean of Balance is 0.332, it means that we have 32.2% of ones. To better visualize this issue, we could use a table showing the percentages of the dummy variables

```
margin.table(prop.table(table(Balance, History, Purpose1, Purpose2,
                              Savings1, Employment_length1,Employment_length2
,
                              Male, Guarantor,Residence_since,House,Insurance
,
                              Car, Other_plans, FreeHousing, OwnHouse, Skille
d,
                              Highly_Qualified, Telephone, Foreign_worker)),6
)
```

```
## Employment_length1
##     0     1 
## 0.764 0.236
```

And by choosing the item we are interested in, we can see how many its proportions. In particular, we have that History,Male, OwnHouse, Skilled, ForeignWorker are the only items with more ones than zeros. As far as quantitative variables are concerned :

```
summary(cbind(Duration_months, Credit_ammount, Installment_rate, Residence_si
nce,
        Number_credits,People_to_mantain, Age))
```

```
##  Duration_months Credit_ammount  Installment_rate Residence_since
##  Min.   : 4.0    Min.   :  250   Min.   :1.000    Min.   :1.000  
##  1st Qu.:12.0    1st Qu.: 1366   1st Qu.:2.000    1st Qu.:2.000
```

```
##  Median :18.0    Median : 2320   Median :3.000    Median :3.000
##  Mean   :20.9    Mean   : 3271   Mean   :2.973    Mean   :2.845
##  3rd Qu.:24.0    3rd Qu.: 3972   3rd Qu.:4.000    3rd Qu.:4.000
##  Max.   :72.0    Max.   :18424   Max.   :4.000    Max.   :4.000
##  Number_credits  People_to_mantain      Age
##  Min.   :1.000   Min.   :1.000     Min.   :19.00
##  1st Qu.:1.000   1st Qu.:1.000     1st Qu.:27.00
##  Median :1.000   Median :1.000     Median :33.00
##  Mean   :1.407   Mean   :1.155     Mean   :35.55
##  3rd Qu.:2.000   3rd Qu.:1.000     3rd Qu.:42.00
##  Max.   :4.000   Max.   :2.000     Max.   :75.00
```

We have that on average the duration is 20.9 months, and that duration is between 4 and 72 months. The user with the lowest credit amount has 250DM, the one with the highest has 18424 DM, with an average of 3271. Given that the 3rd quarter is not far from the mean (3972), we can say that the distance between that value and the max is very high.

On average, users have owned a residence for 2.845 years and they have 1.407 existing credits at this bank. They have on average 1.155 people to maintain (and in no case more than 2). The average age of the users is 35.55 years, however it seems that the distribution displays similarities with respect to the amount of credits, as the maximum age is much larger than the mean and the 3rd quarter. We expect to have a skewed distribution.

QUESTION 2

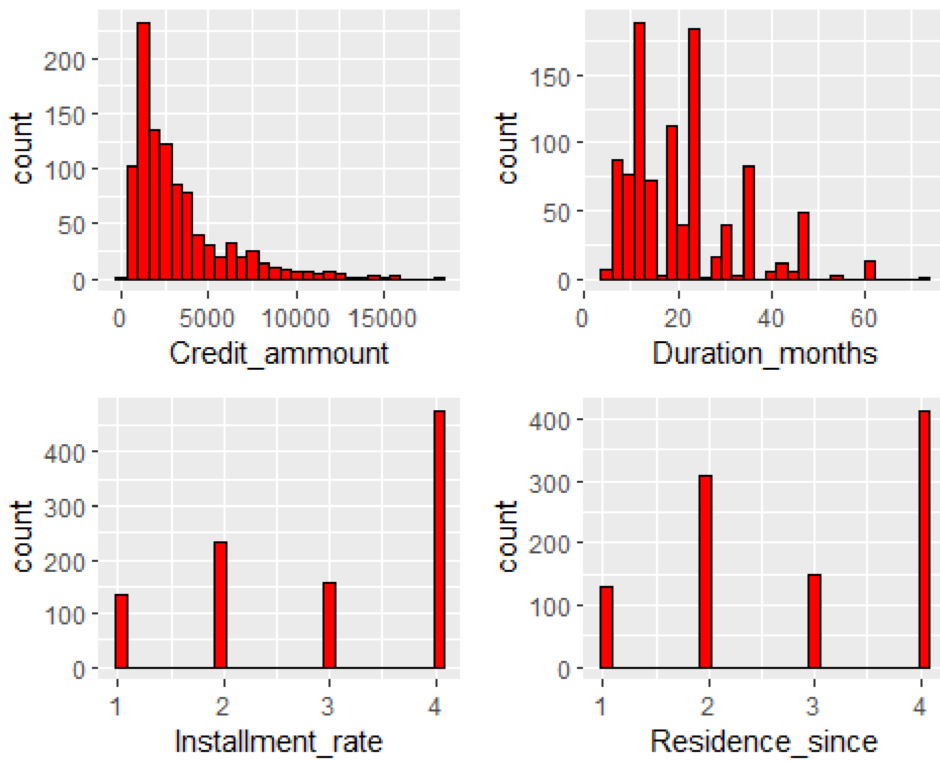
Now we want to investigate how much correlated are our variables to our outcome. We will do it only for our training sample, which we define as follows :

```
set.seed(5257)
random_vector <- sample(c(1:1000), replace = FALSE, prob = NULL)
German_rd <- c()
for (i in random_vector) {
  German_rd = rbind(German_rd, German[i,])
}
Training_Validation <- German_rd[1:750,]
Just_training<-German_rd[1:500,]
Just_validation<-German_rd[501:750,]
Testing <- German_rd[751:1000,]

German_data <- as.data.frame(German_rd)
Training_sample <- as.data.frame(Training_Validation)
Testing_sample <- as.data.frame(Testing)
dTRAIN<-as.data.frame(Just_training)
dcv<-as.data.frame(Just_validation)
```

We have randomly created the two groups, the training and the testing Now, just to get an idea, let's plot Outcome and the variables.

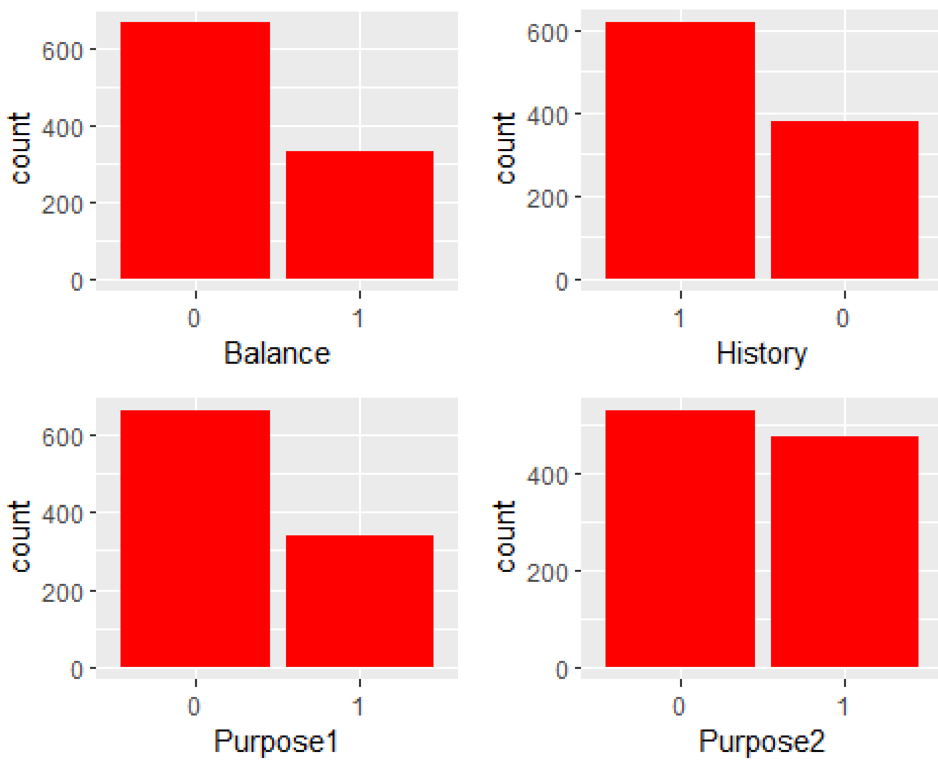
Here we have histograms for the quantitative variables ; the more interesting to comment are age and Credit_ammount, that seems skewned to the left. This proves our suspects of having high values in one of the extremes. Then, we provided barplots for the qualitative parameters. This plots simply show us graphically, for each parameters, the relative frequency of each attribute.

```
ggarrange(
ggplot(German_data, aes(x=Credit_ammount)) + geom_histogram(color="black", fi
ll="red"),
ggplot(German_data, aes(x=Duration_months)) + geom_histogram(color="black", f
ill="red"),
ggplot(German_data, aes(x=Installment_rate)) + geom_histogram(color="black",
fill="red"),
ggplot(German_data, aes(x=Residence_since)) + geom_histogram(color="black", f
ill="red"))

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



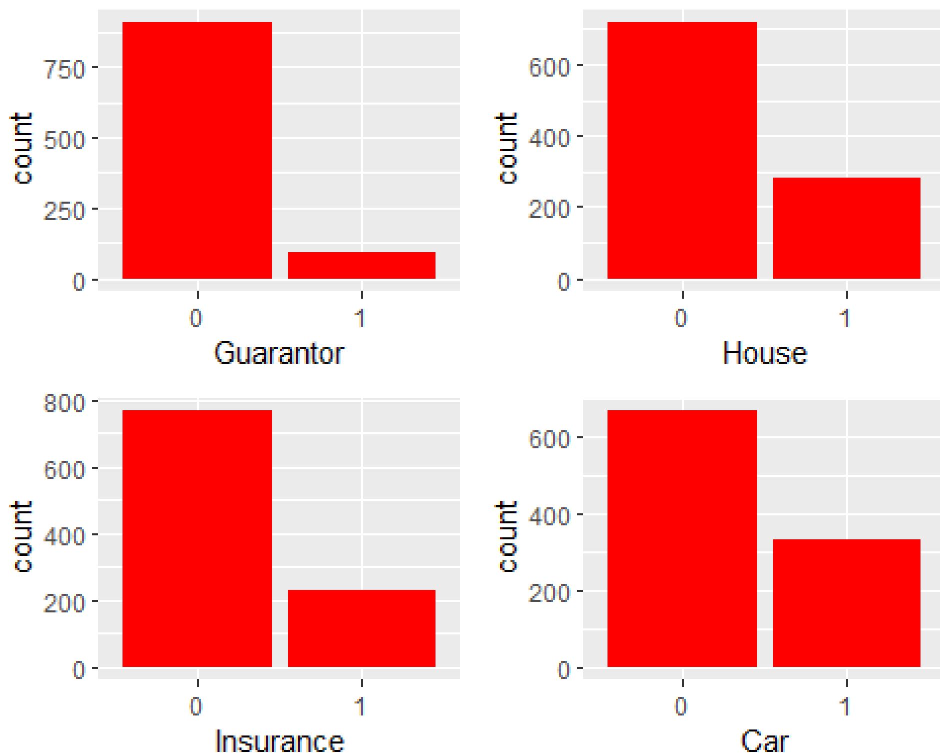

```
ggarrange(
  ggplot(German_data, aes(x=Number_credits)) + geom_histogram(color="black",
fill="red"),
  ggplot(German_data, aes(x=People_to_mantain)) + geom_histogram(color="black
", fill="red"),
  ggplot(German_data, aes(x=Age)) + geom_histogram(color="black", fill="red")
)

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


```
ggarrange(
ggplot(German_data, aes(x=reorder(Balance, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='Balance'),
ggplot(German_data, aes(x=reorder(History, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='History'),
ggplot(German_data, aes(x=reorder(Purpose1, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Purpose1'),
ggplot(German_data, aes(x=reorder(Purpose2, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Purpose2'))
```

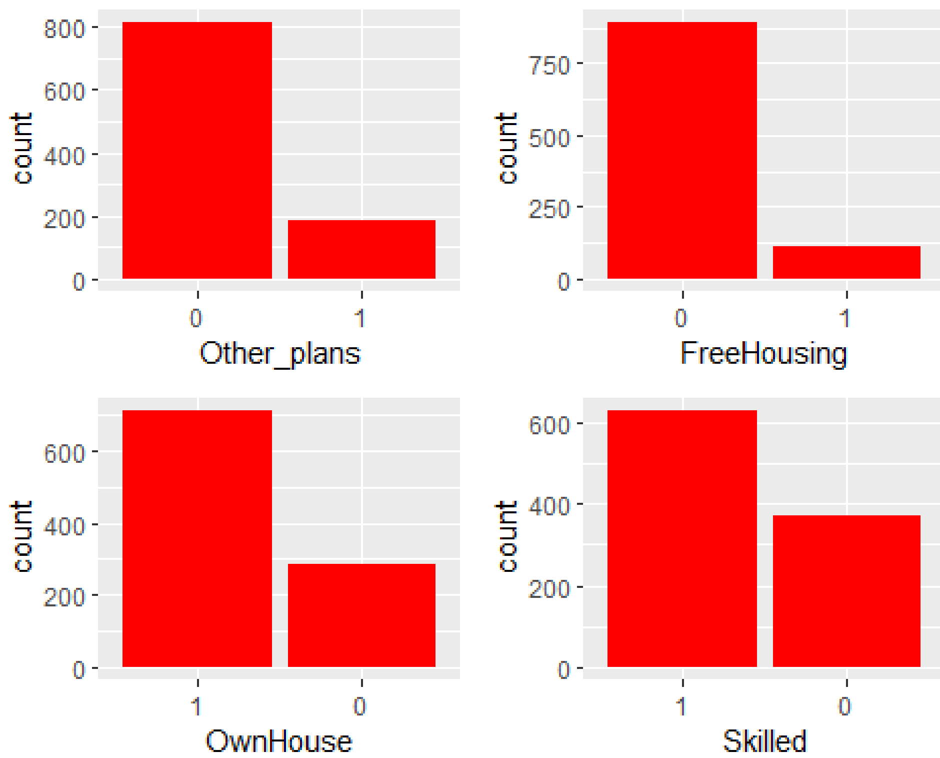



```
ggarrange(
ggplot(German_data, aes(x=reorder(Savings1, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Savings1'),
ggplot(German_data, aes(x=reorder(Employment_length1, Output, function(x)-len
gth(x)))) +
  geom_bar(fill='red') +  labs(x='Employment:length1'),
ggplot(German_data, aes(x=reorder(Employment_length2, Output, function(x)-len
gth(x)))) +
  geom_bar(fill='red') +  labs(x='Employment_length2'),
ggplot(German_data, aes(x=reorder(Male, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='Male'))
```

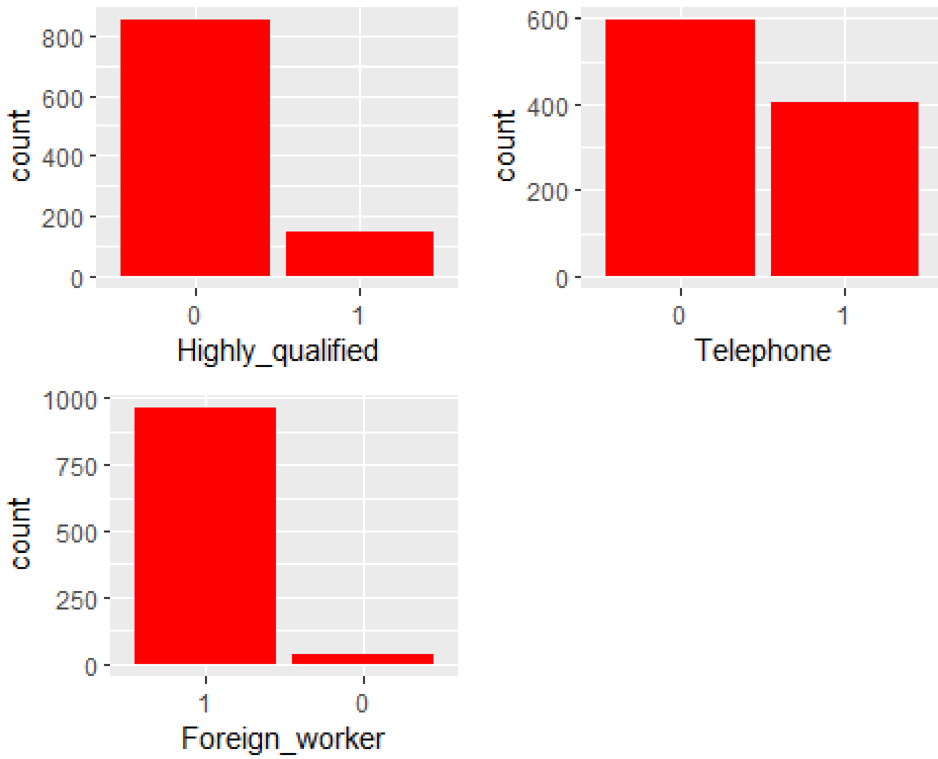

```
ggarrange(
ggplot(German_data, aes(x=reorder(Guarantor, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Guarantor'),
ggplot(German_data, aes(x=reorder(House, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='House'),
ggplot(German_data, aes(x=reorder(Insurance, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Insurance'),
ggplot(German_data, aes(x=reorder(Car, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='Car'))
```




```
ggarrange(
ggplot(German_data, aes(x=reorder(Other_plans, Output, function(x)-length(x))
)) +
  geom_bar(fill='red') +  labs(x='Other_plans'),
ggplot(German_data, aes(x=reorder(FreeHousing, Output, function(x)-length(x))
)) +
  geom_bar(fill='red') +  labs(x='FreeHousing'),
ggplot(German_data, aes(x=reorder(OwnHouse, Output, function(x)-length(x)))) 
+
  geom_bar(fill='red') +  labs(x='OwnHouse'),
ggplot(German_data, aes(x=reorder(Skilled, Output, function(x)-length(x)))) +
  geom_bar(fill='red') +  labs(x='Skilled'))
```




```
ggarrange(
ggplot(German_data, aes(x=reorder(Highly_Qualified, Output, function(x)-lengt
h(x)))) +
  geom_bar(fill='red') +  labs(x='Highly_qualified'),
ggplot(German_data, aes(x=reorder(Telephone, Output, function(x)-length(x))))
+
  geom_bar(fill='red') +  labs(x='Telephone'),
ggplot(German_data, aes(x=reorder(Foreign_worker, Output, function(x)-length(
x)))) +
  geom_bar(fill='red') +  labs(x='Foreign_worker'))
```


count
800
600
400
200
0
0
1
Highly_qualified
count
600
400
200
0
0
1
Telephone
count
1000
750
500
250
0
1
0
Foreign_worker

Question 3

First we randomly split the dataset into two parts: the training and the testing

```
Y <- German_data[1:1000,27]
donnees <- German_data
indapp <- 1:750
dapp <- Training_sample
dtest <-Testing_sample
loss = rbind(c(0,1), c(5,0))
dapp.X<-model.matrix(Output~.,data=dapp)
dtest.X<-model.matrix(Output~.,data=dtest)
```

We have many explanatory variables so we have to perform variable selection for some models.

Linear Probability Model

Let's first have a look at a linear regression on all the data

The problem is to explain the `Output` (column 27) by the other variables. We first consider the linear model. For linear models we will have to first fit the betas on the training sample (dTRAIN) and then use the cross-validation sample (dcv) to select the hyperparameter which in this case is the number of explanatory variables

```
#recall that dcv is our cross validation set and
linear.model <- lm(Output~.,data=dTRAIN)
summary(linear.model)

##
## Call:
## lm(formula = Output ~ ., data = dTRAIN)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6941 -0.3046 -0.1574  0.4181  0.9716
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -4.804e-02  2.026e-01  -0.237  0.81270
## Balance              5.177e-02  4.192e-02   1.235  0.21744
## Duration_months      5.333e-03  2.332e-03   2.287  0.02265 *
## History              1.343e-01  4.848e-02   2.771  0.00581 **
## Purpose1             9.958e-02  5.543e-02   1.796  0.07308 .
## Purpose2             1.569e-02  5.396e-02   0.291  0.77140
## Credit_ammount       7.064e-06  1.138e-05   0.621  0.53518
## Savings1            -1.428e-04  8.528e-05  -1.675  0.09469 .
## Employment_length1 -1.254e-01  5.016e-02  -2.499  0.01279 *
## Employment_length2 -1.223e-01  5.468e-02  -2.236  0.02578 *
```

```
## Installment_rate    3.073e-02  1.973e-02   1.557  0.12002
## Male               -1.393e-02  4.522e-02  -0.308  0.75813
## Guarantor          -1.683e-02  6.442e-02  -0.261  0.79403
## Residence_since     1.515e-02  1.906e-02   0.795  0.42711
## House              -2.088e-01  9.535e-02  -2.190  0.02901 *
## Insurance          -1.196e-01  9.417e-02  -1.270  0.20456
## Car                -1.600e-01  9.284e-02  -1.723  0.08554 .
## Age                -3.103e-03  1.894e-03  -1.639  0.10193
## Other_plans         6.774e-02  5.226e-02   1.296  0.19551
## FreeHousing        -1.362e-01  1.116e-01  -1.220  0.22316
## OwnHouse           -1.372e-01  5.704e-02  -2.405  0.01657 *
## Number_credits      7.212e-02  3.982e-02   1.811  0.07074 .
## Skilled            -3.406e-03  5.064e-02  -0.067  0.94641
## Highly_Qualified    2.460e-02  7.734e-02   0.318  0.75058
## People_to_mantain   2.405e-02  5.893e-02   0.408  0.68343
## Telephone          -4.657e-02  4.464e-02  -1.043  0.29734
## Foreign_worker      2.608e-01  1.176e-01   2.217  0.02708 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4248 on 473 degrees of freedom
## Multiple R-squared:  0.1531, Adjusted R-squared:  0.1066
## F-statistic: 3.289 on 26 and 473 DF,  p-value: 1.719e-07
```

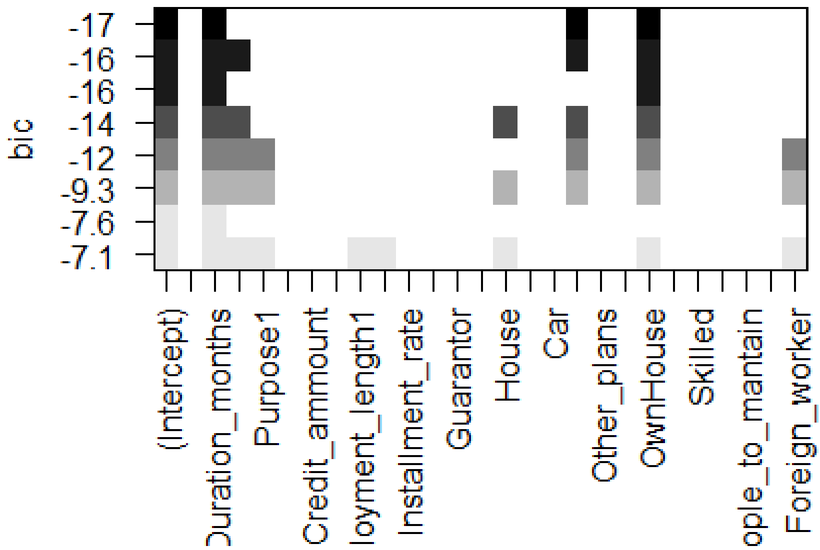
Some variables seem not to be useful so we would have to proceed with some selection tecniques

We can try with subset selection:

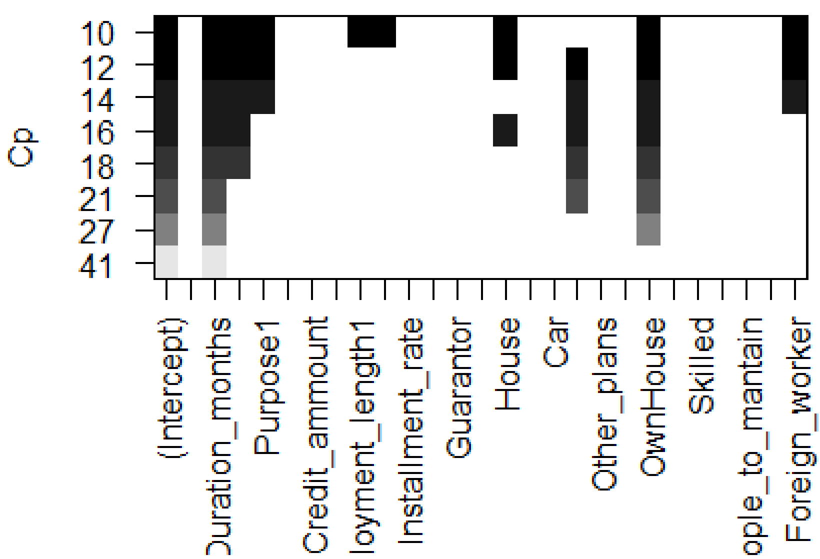
```
# either best subset
mod.sel <- regsubsets(Output~.,data=dTRAIN)
#or backward stepwise selection
m.back1 <- regsubsets(Output~.,data=dTRAIN,method="backward")
m.for1 <- regsubsets(Output~.,data=dTRAIN, method="forward")
```

We can select the best models according to BIC and Cp

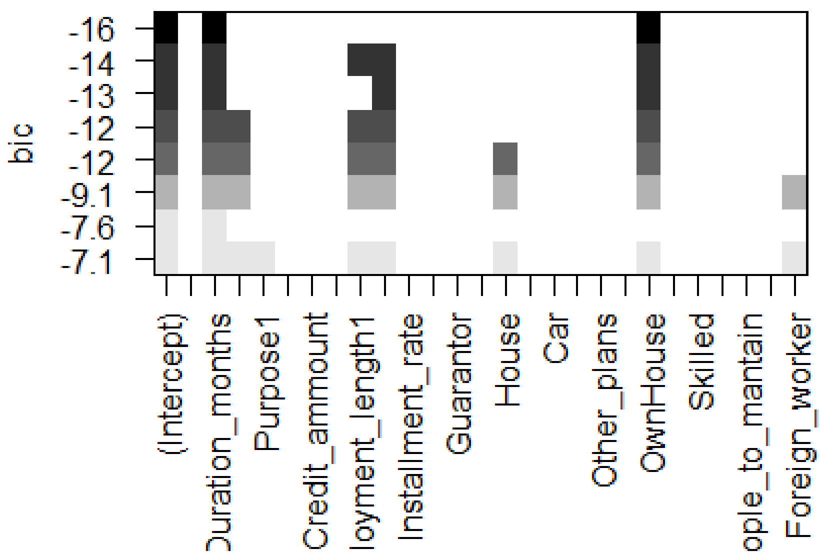
```
#BIC
plot(mod.sel,scale="bic")
```



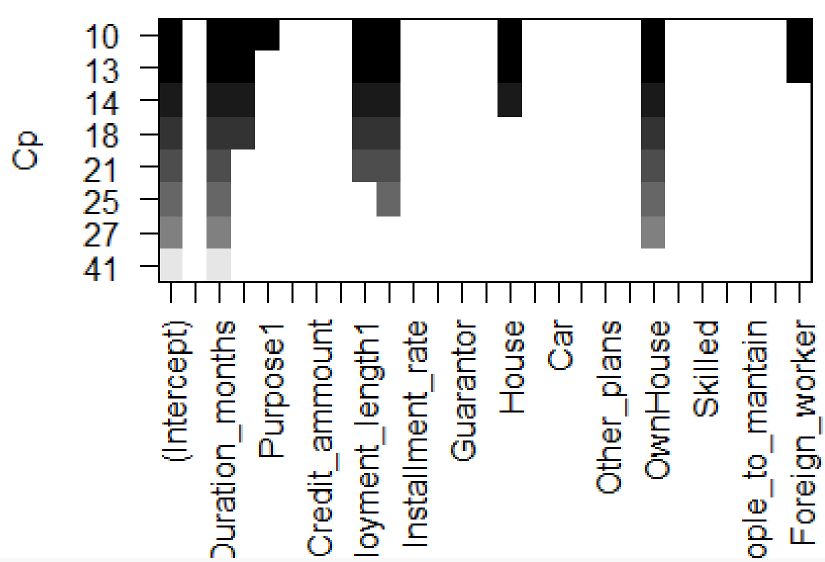

```r
#Mallow's Cp
plot(mod.sel,scale="Cp")
```



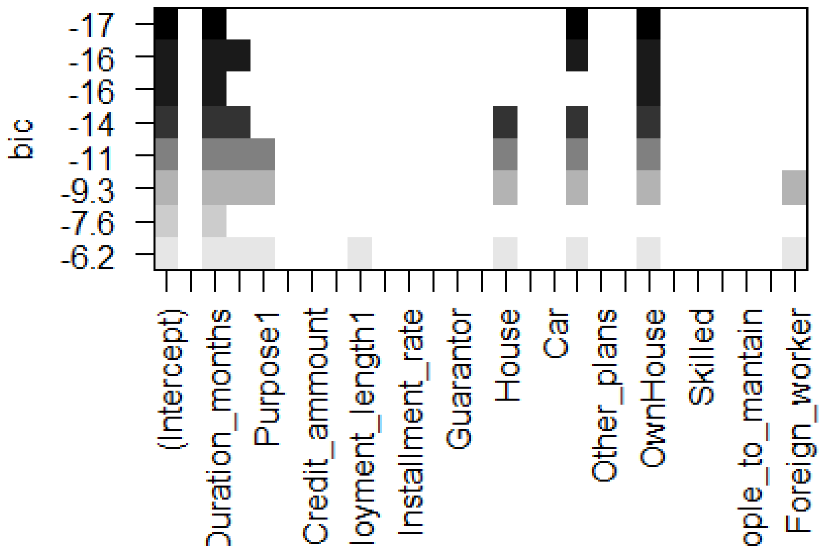
```
#backward BIC
plot(m.back1,scale="bic")
```



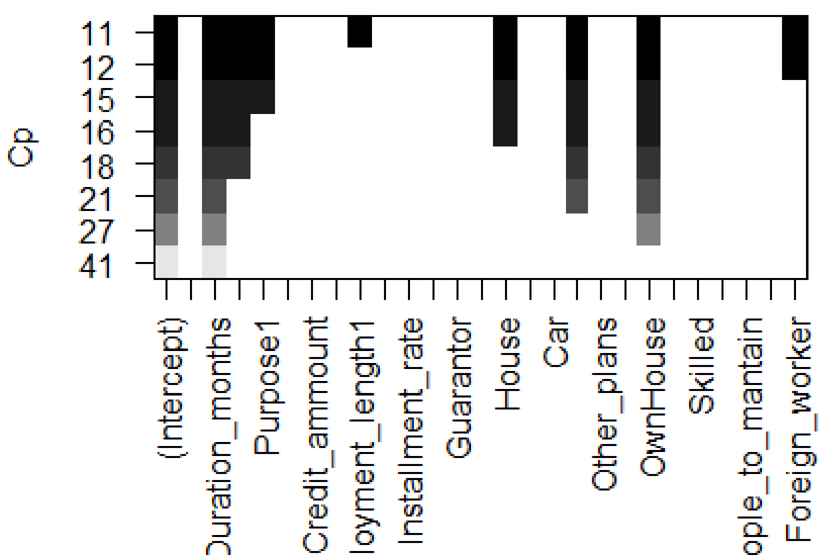
```
#backward Cp
plot(m.back1,scale="Cp")
```




```
#forward Cp
plot(m.for1,scale="bic")
```




```
#forward Cp
plot(m.for1,scale="Cp")
```



```r
a <- summary(mod.sel)
number <- order(a$bic)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.BIC <- lm(form,data=dTRAIN)

a <- summary(mod.sel)
number <- order(a$cp)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.CP <- lm(form,data=dTRAIN)

a <- summary(m.back1)
number <- order(a$bic)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form.back <- formula(paste("Output~",var.sel1,sep=""))
mod.BIC.back <- lm(form.back,data=dTRAIN)

a <- summary(m.back1)
number <- order(a$cp)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.CP.back <- lm(form,data=dTRAIN)

a <- summary(m.for1)
number <- order(a$bic)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.BIC.for <- lm(form,data=dTRAIN)

a <- summary(m.for1)
number <- order(a$cp)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.CP.for <- lm(form,data=dTRAIN)
```

We consider the quadratic risk for the models:

$$E\left[\left(Y - \widehat{m}(X)\right)^2\right].$$

This risk is estimated with the test set according to

$$\frac{1}{n_{test}} \sum_{i \in test} \left(Y_i - \widehat{m}(X_i)\right)^2.$$

Compute the estimated risks for the three linear models:

```
prev <- data.frame(Y=dtest$Output,lin=predict(linear.model,newdata=dcv),BIC=p
redict(mod.BIC,newdata=dcv),CP=predict(mod.CP,newdata=dcv),BIC.back=predict(m
od.BIC.back,newdata=dcv),CP.back=predict(mod.CP.back,newdata=dcv),BIC.for=pre
dict(mod.BIC.for,newdata=dcv),CP.for=predict(mod.CP.for,newdata=dcv))

prev %>% summarize(Err_lin=mean((Y-lin)^2),Err_BIC=mean((Y-BIC)^2),Err_CP=mea
n((Y-CP)^2),Err_BIC_back=mean((Y-BIC.back)^2),Err_CP_back=mean((Y-CP.back)^2)
,Err_BIC_for=mean((Y-BIC.for)^2),Err_CP_for=mean((Y-CP.for)^2))

##     Err_lin   Err_BIC    Err_CP Err_BIC_back Err_CP_back Err_BIC_for Err_C
P_for
## 1 0.2586208 0.2319694 0.2479865    0.2313648   0.2479865   0.2319694  0.24
69547
```

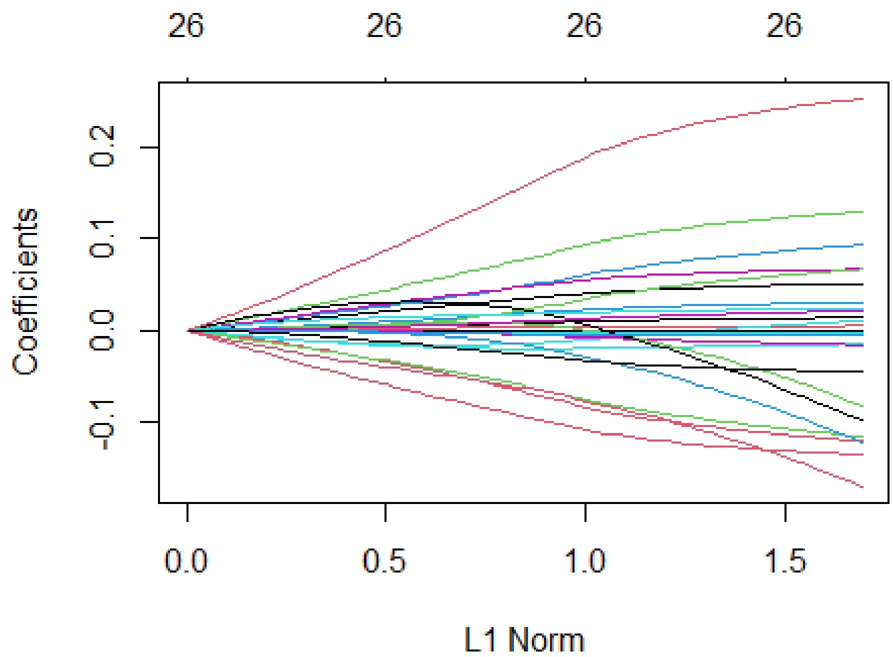
The first variable selection procedure is obtained through a backward selection approach. The statistical information criteria is BIC.

Let us run also a RIDGE and LASSO regressions

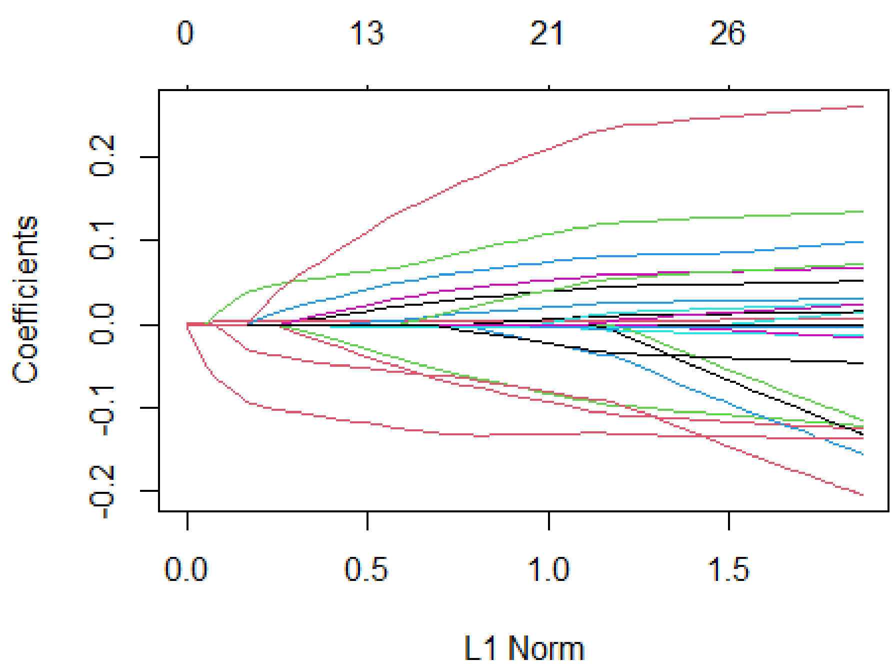
```
dTRAIN.X <- model.matrix(Output~.,data=dTRAIN)
dcv.X <- model.matrix(Output~.,data=dcv)
```

We draw the coefficient paths for ridge and lasso.

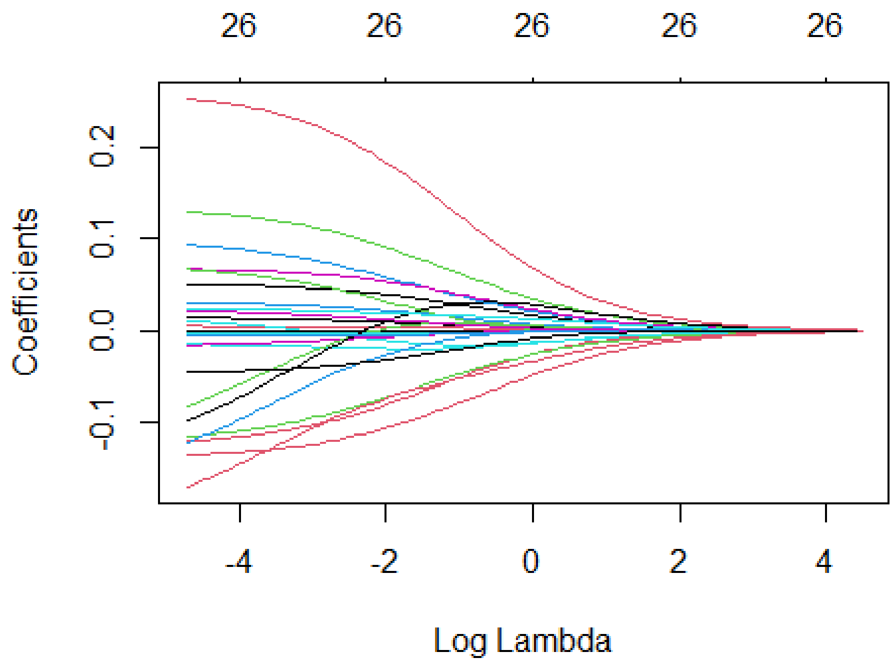
```
mod.R <- glmnet(dTRAIN.X,dTRAIN$Output,alpha=0)
mod.L <- glmnet(dTRAIN.X,dTRAIN$Output,alpha=1)

plot(mod.R)
```



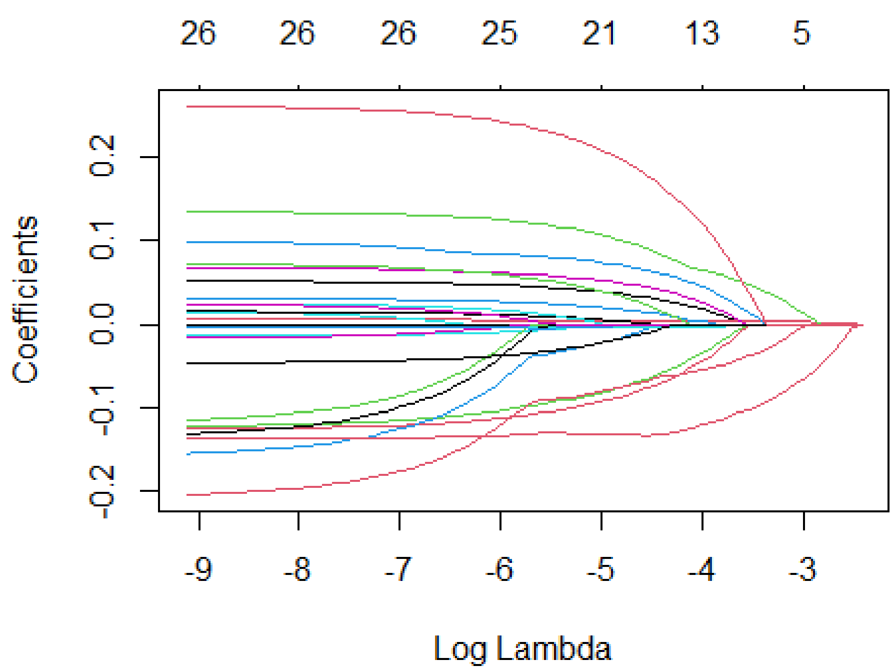

```
plot(mod.L)
```




```
plot(mod.R,xvar="lambda")
```




```
plot(mod.L,xvar="lambda")
```

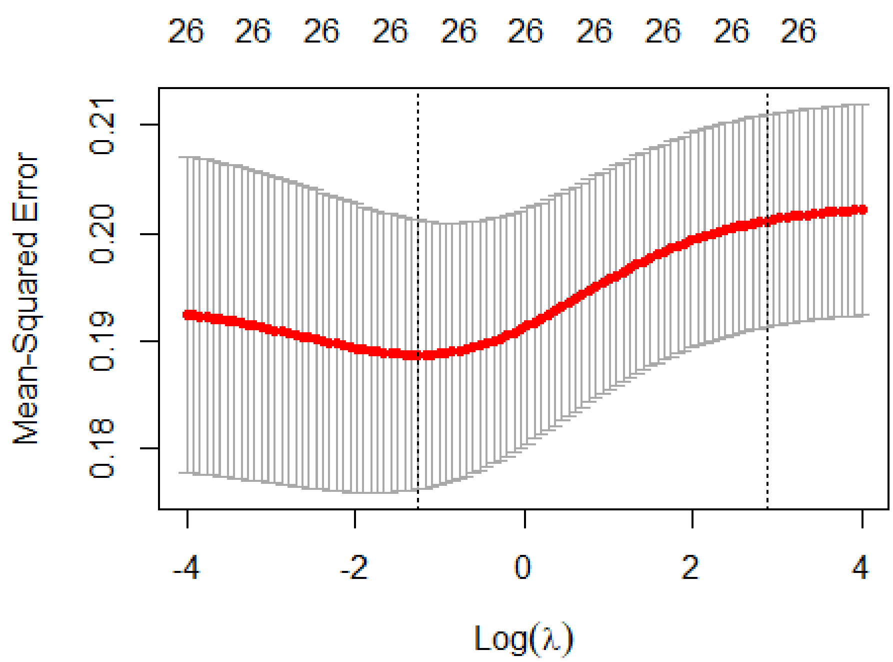


We select the shrinkage parameter for lasso regression with **cv.glmnet**.

```
lassoCV <- cv.glmnet(dTRAIN.X,dTRAIN$Output,alpha=1)
lassoCV$lambda.min

## [1] 0.006569549

lasso.sel <- glmnet(dTRAIN.X,dTRAIN$Output,alpha=1,lambda=lassoCV$lambda.min)
```

Now fit the selected ridge model.

```
ridgeCV <- cv.glmnet(dTRAIN.X,dTRAIN$Output,alpha=0,lambda=exp(seq(-4,4,lengt
h=100)))
plot(ridgeCV)
```




```
ridge.sel <- glmnet(dTRAIN.X,dTRAIN$Output,alpha=0,lambda=ridgeCV$lambda.min)
```

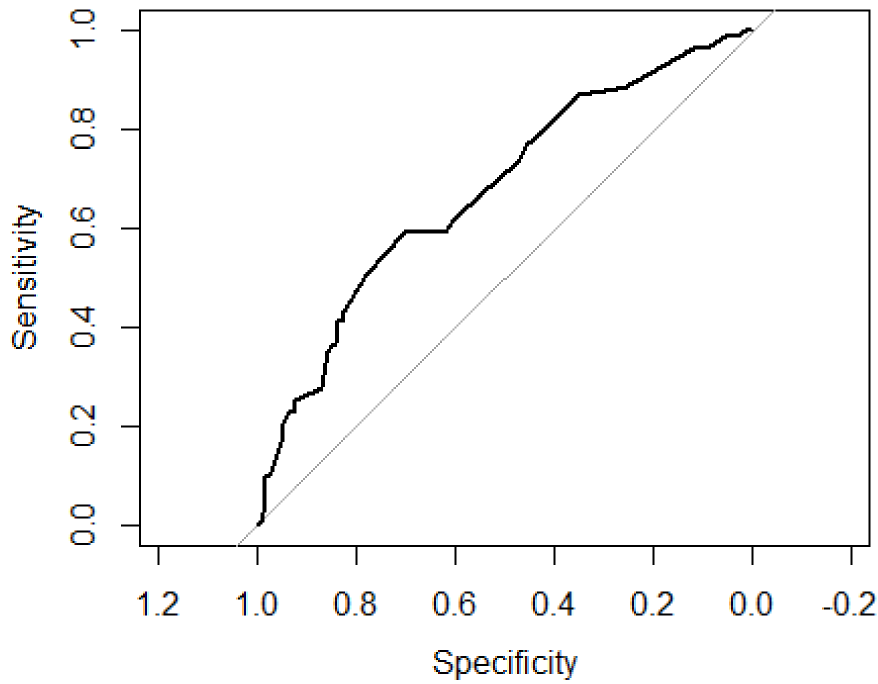
Estimate the quadratic error for the selected ridge and lasso models.

```
prev1 <- prev %>% mutate(ridge=as.vector(predict(ridge.sel,newx=dcv.X)),lasso
=as.vector(predict(lasso.sel,newx=dcv.X)))
prev1 %>% summarize(Err_lin=mean((Y-lin)^2),Err_BIC=mean((Y-BIC)^2),Err_CP=me
an((Y-CP)^2),Err_BIC_back=mean((Y-BIC.back)^2),Err_CP_back=mean((Y-CP.back)^2
),Err_BIC_for=mean((Y-BIC.for)^2),Err_CP_for=mean((Y-CP.for)^2),Err_ridge=mea
n((Y-ridge)^2),Err_lasso=mean((Y-lasso)^2))

##     Err_lin   Err_BIC    Err_CP Err_BIC_back Err_CP_back Err_BIC_for Err_C
P_for
## 1 0.2586208 0.2319694 0.2479865    0.2313648    0.2479865   0.2319694  0.24
69547
```

```
##   Err_ridge Err_lasso
## 1 0.2365286  0.245584
```

Conclusion: we select the best explanatory variables subset which is the one given by the model called Err_BIC_back. Therefore can rerun a regression on the training+cross-validation samples and do our final predictions with the test set

```
#rerun
a <- summary(m.back1)
number <- order(a$bic)[1]
var.sel <- a$which[number,][-1]
var.sel1 <- names(var.sel)[var.sel] %>% paste(collapse="+")
form <- formula(paste("Output~",var.sel1,sep=""))
mod.BIC.back <- lm(form,data=dapp)

#final prediction
prevlm<- predict(mod.BIC.back,newdata=dtest)

#MSE
MSE_lm<- mean(round(prevlm)!=dtest$Output)
MSE_lm

## [1] 0.316

#Roc and AUC
plot(roc(dtest$Output,prevlm))

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```




```
AUC_lm<-auc(roc(dtest$Output,prevlm))

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

AUC_lm

## Area under the curve: 0.6759

#confusionMatrix(data=prevlog,reference =dtest$Output)

#Misclassification error
optimal_lm <- optimalCutoff(dtest$Output, prevlm)[1]
conf_lm<-confusionMatrix(dtest$Output, prevlm)
misclass_lm <- (conf_lm[1,2]*loss[1,2]+conf_lm[2,1]*loss[2,1])/nrow(dapp)/mea
n(dapp$Output==1)
misclass_lm

## [1] 0.437788
```

Logistic model

```
full.logit <- glm(Output~.,data=dTRAIN,family="binomial")
full.logit
```

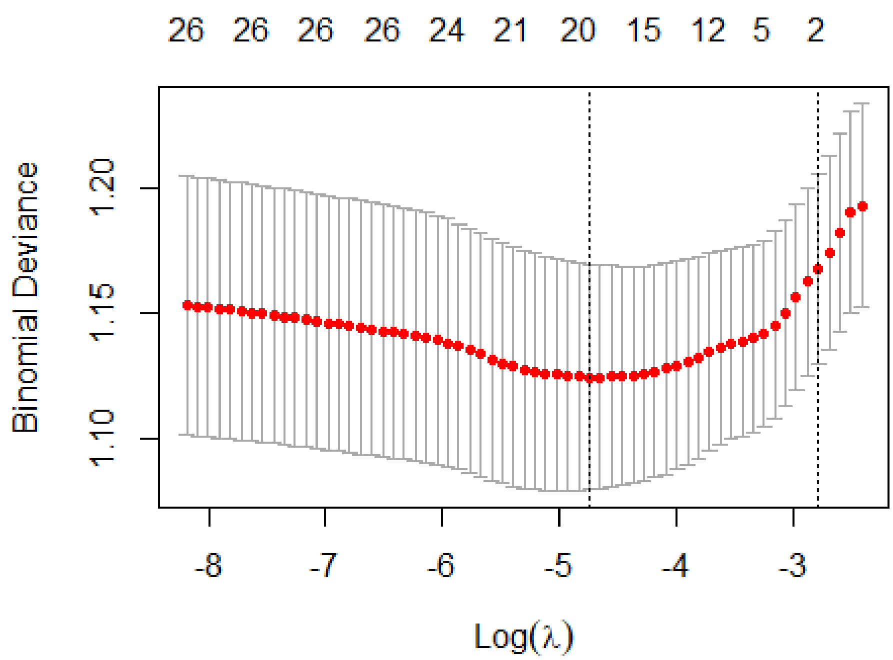
```
##
## Call:  glm(formula = Output ~ ., family = "binomial", data = dTRAIN)
##
## Coefficients:
##         (Intercept)              Balance      Duration_months             His
tory
##          -3.788e+00            3.262e-01            2.844e-02            7.737
e-01
##            Purpose1             Purpose2      Credit_ammount             Savi
ngs1
##           6.292e-01            1.201e-01            3.912e-05           -9.856
e-04
## Employment_length1  Employment_length2    Installment_rate
Male
##          -7.310e-01           -7.178e-01            2.080e-01           -8.431
e-02
##           Guarantor      Residence_since               House            Insur
ance
##          -6.238e-02            6.325e-02           -1.262e+00           -6.414
e-01
##                 Car                  Age          Other_plans         FreeHou
sing
##          -8.884e-01           -2.184e-02            4.608e-01           -7.402
e-01
##            OwnHouse       Number_credits              Skilled     Highly_Quali
fied
##          -7.581e-01            4.384e-01           -7.269e-03            1.895
e-01
##   People_to_mantain            Telephone      Foreign_worker
##           1.540e-01           -3.038e-01            2.314e+00
##
## Degrees of Freedom: 499 Total (i.e. Null);  473 Residual
## Null Deviance:       593
## Residual Deviance: 508.7     AIC: 562.7
```

We implement a variable selection procedure with a backward selection approach using BIC criterion. You just have to use the step function with the direction="backward" and k=log(nrow(train)) options. We call it mod.back

```
mod.back <- step(full.logit,direction="backward",k=log(nrow(dTRAIN)),trace=0)
```

we Fit a logistic lasso model on the training data (select the shrinkage parameter with **cv.glmnet**).

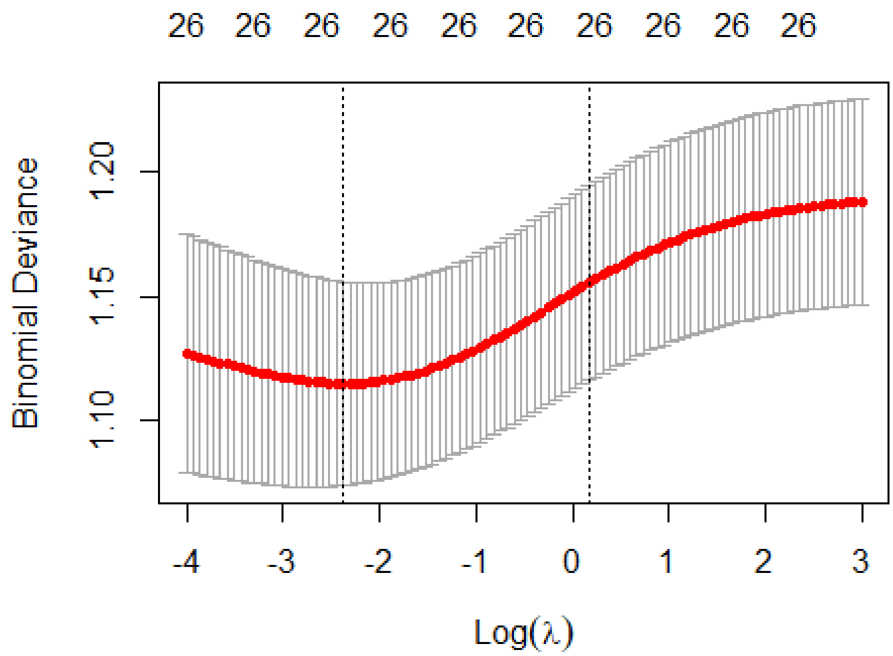
```
set.seed(1234)
cv.lasso <- cv.glmnet(dTRAIN.X,dTRAIN[,27],family="binomial",alpha=1)
plot(cv.lasso)
```




```
lambda.lasso <- cv.lasso$lambda.min
mod.lasso <- glmnet(dTRAIN.X,dTRAIN[,27],family="binomial",lambda=lambda.lass
o,alpha=1)
```

The fit a logistic ridge model on the training data (select the shrinkage parameter with **cv.glmnet**).

```
set.seed(1234)
cv.ridge <- cv.glmnet(dTRAIN.X,dTRAIN[,27],family="binomial",alpha=0,lambda=e
xp(seq(-4,3,length=100)))
plot(cv.ridge)
```




```
lambda.ridge<-cv.ridge$lambda.min
mod.ridge <- glmnet(dTRAIN.X,dTRAIN[,27],family="binomial",lambda=lambda.ridg
e,alpha=0)
```

Make a comparison of the methods with the error probability (estimated on the test dataset).

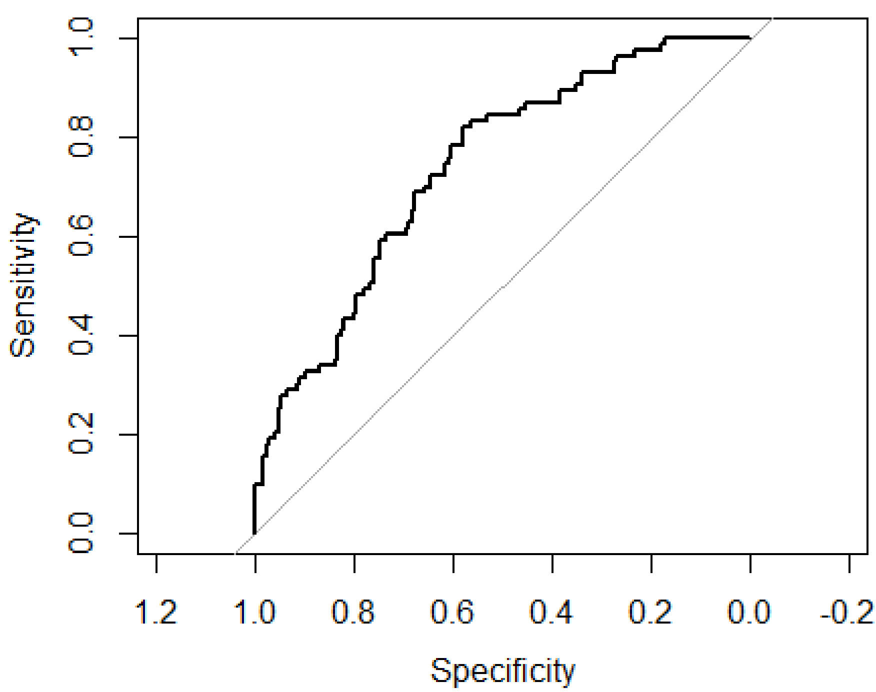
```
prev.full <- predict(full.logit,newdata=dcv,type="response") %>% round() %>%
as.factor()
prev.back <- predict(mod.back,newdata=dcv,type="response") %>% round() %>% as
.factor()

prev.lasso <- predict(mod.lasso,newx=dcv.X,type="class")
prev.ridge <- predict(mod.ridge,newx=dcv.X,type="class")
prev <- data.frame(full=prev.full,back=prev.back,lasso=as.vector(prev.lasso),
ridge=as.vector(prev.ridge))
prev %>% summarise_at(vars(1:4),~(mean((.!=Y)^2)))

##    full  back lasso ridge
## 1 0.344 0.316 0.324  0.31
```

Conclusion best model is ridge

Therefore can rerun a regression on the training+cross-validation samples and do our final predictions with the test set

```
#rerun
mod.ridge <- glmnet(dapp.X,dapp[,27],family="binomial",lambda=lambda.ridge,al
```

```
pha=0)
#final prediction
prevlog<- predict(mod.ridge,newx=dtest.X,type="response")
#prevlog

#MSE
MSE_log<- mean(round(prevlog)!=dtest$Output)
MSE_log

## [1] 0.296

#Roc and AUC
plot(roc(dtest$Output,prevlog))

## Setting levels: control = 0, case = 1

## Warning in roc.default(dtest$Output, prevlog): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric vecto
r.

## Setting direction: controls < cases
```



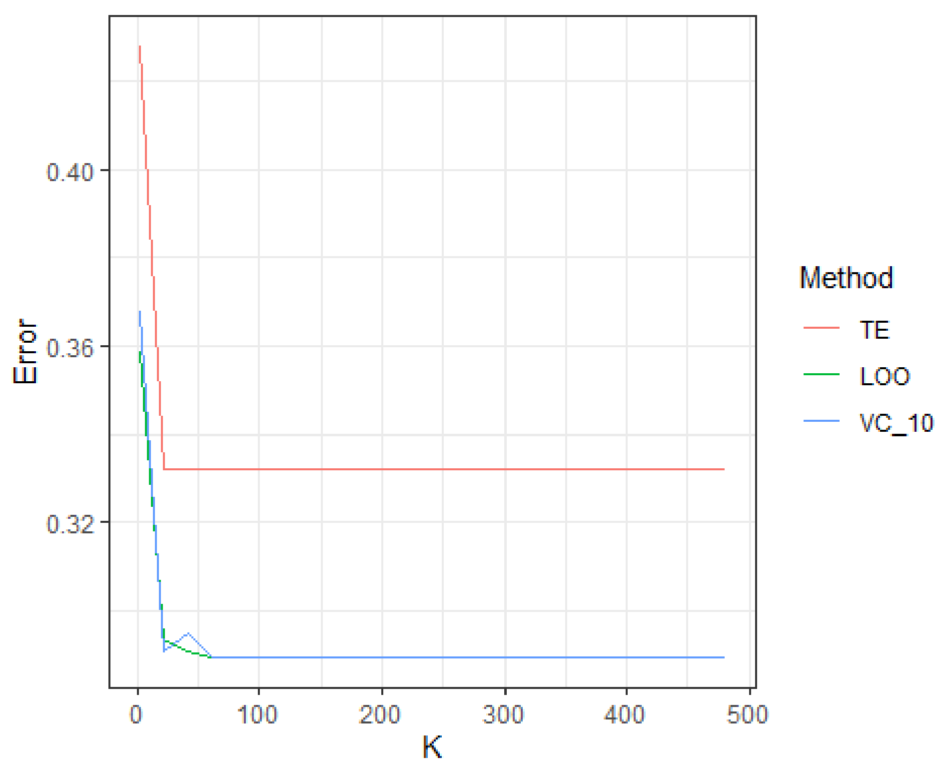

```
AUC_log<-auc(roc(dtest$Output,prevlog))

## Setting levels: control = 0, case = 1
```

```
## Warning in roc.default(dtest$Output, prevlog): Deprecated use a matrix as
## predictor. Unexpected results may be produced, please pass a numeric vecto
r.

## Setting direction: controls < cases

AUC_log

## Area under the curve: 0.737

#confusionMatrix(data=prevlog,reference =dtest$Output)

#Misclassification error
optimal_log <- optimalCutoff(dtest$Output, prevlog)[1]
conf_log<-confusionMatrix(dtest$Output, prevlog)
misclass_log <- (conf_log[1,2]*loss[1,2]+conf_log[2,1]*loss[2,1])/nrow(dapp)/
mean(dapp$Output==1)
misclass_log

## [1] 0.4884793
```

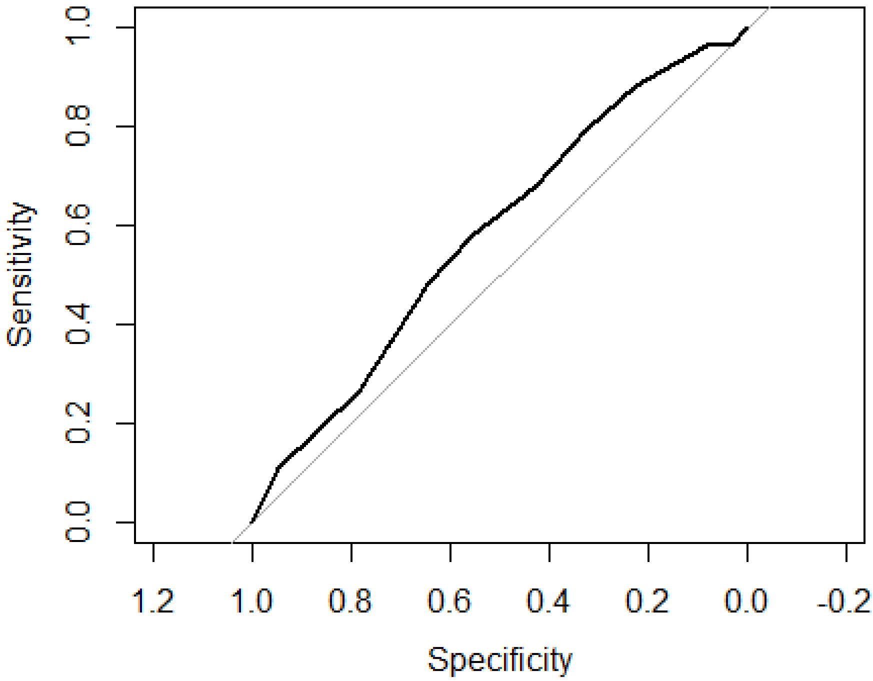
KNN Model

```
#cross-validation

library(class)
regle_ppv <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=81)

#function 1
K_cand <- seq(1,500,by=20)
err1 <- rep(0,length(K_cand))
for (i in 1:length(K_cand)){
  err1[i] <- mean(knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=K_cand[i])!=dte
st$Output)
}
K_cand[which.min(err1)]

## [1] 21

#Function 2 : Leave-one-out cross-validation

err2 <- rep(0,length(K_cand))
for (i in 1:length(K_cand)){
  prev_cv <- knn.cv(dapp[,-27],cl=dapp$Output,k=K_cand[i])
  err2[i] <- mean(prev_cv!=dapp$Output)
}
K_cand[which.min(err2)]

## [1] 61
```

```r
#Function 3 : M-fold cross-validation method

err3 <- rep(0,length(K_cand))
M <- 10
prev <- rep(0,nrow(dapp))
n_CV <- nrow(dapp)/M
for (i in 1:length(K_cand)){
  for (j in 1:M){
    ind_testj <- ((j-1)*n_CV+1):(j*n_CV)
    prev[ind_testj] <- knn(dapp[-ind_testj,-27],dapp[ind_testj,-27],cl=dapp$O
utput[-ind_testj],k=K_cand[i])
  }
  err3[i] <- mean((prev-1)!=dapp$Output)
}
K_cand[which.min(err3)]

## [1] 61

# Visual inspection
a <- data.frame(K_cand,err1,err2,err3)
names(a) <- c("K","TE","LOO","VC_10")
library(reshape2)
aa <- melt(a,id="K")
names(aa) <- c("K","Method","Error")
ggplot(aa)+aes(x=K,y=Error,color=Method)+geom_line()+theme_bw()
```



```
pred_21 <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=21)
pred_61 <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=61)
pred_81 <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=81)
MSE_f1<-mean(pred_21!=dtest$Output)
MSE_f2<-mean(pred_61!=dtest$Output)
MSE_f3<-mean(pred_81!=dtest$Output)
MSE_f1

## [1] 0.332

MSE_f2

## [1] 0.332

MSE_f3

## [1] 0.332

pred_final <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=21)

#MSE
MSE_KNN<-mean(pred_final!=dtest$Output)
MSE_KNN

## [1] 0.332

#Roc and AUC
prev1 <- knn(dapp[,-27],dtest[,-27],cl=dapp$Output,k=21,prob=TRUE)
D <- data.frame(pred=attributes(prev1)$prob,obs=dtest$Output)
plot(roc(D$obs, D$pred))

## Setting levels: control = 0, case = 1

## Setting direction: controls > cases
```

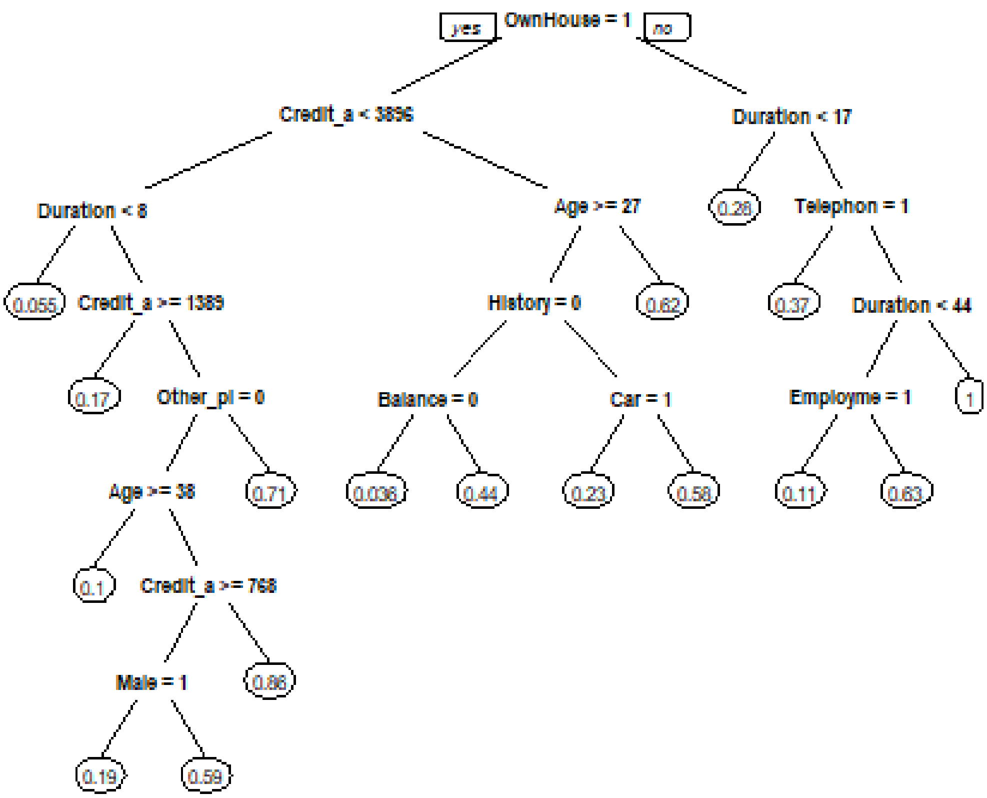



```
AUC_KNN<-auc(roc(D$obs, D$pred))

## Setting levels: control = 0, case = 1
## Setting direction: controls > cases

AUC_KNN

## Area under the curve: 0.5843

#Mispecification error

conf_KNN <- table(dtest$Output,pred_final)
conf_KNN

##    pred_final
##       0   1
##   0 160   7
##   1  76   7

misclass_KNN<-(conf_KNN[1,2]*loss[1,2]+conf_KNN[2,1]*loss[2,1])/nrow(dapp)/me
an(dapp$Output==1)
misclass_KNN

## [1] 1.78341
```

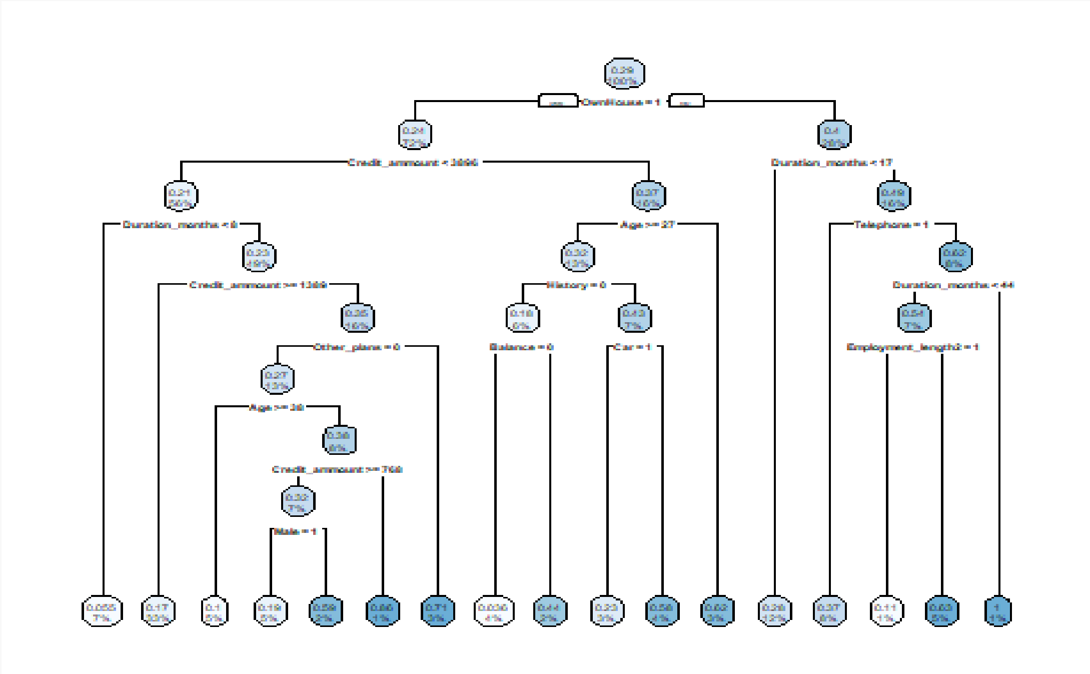
Decision trees

We start by inspecting the regression tree to have a "feeling" of the model.

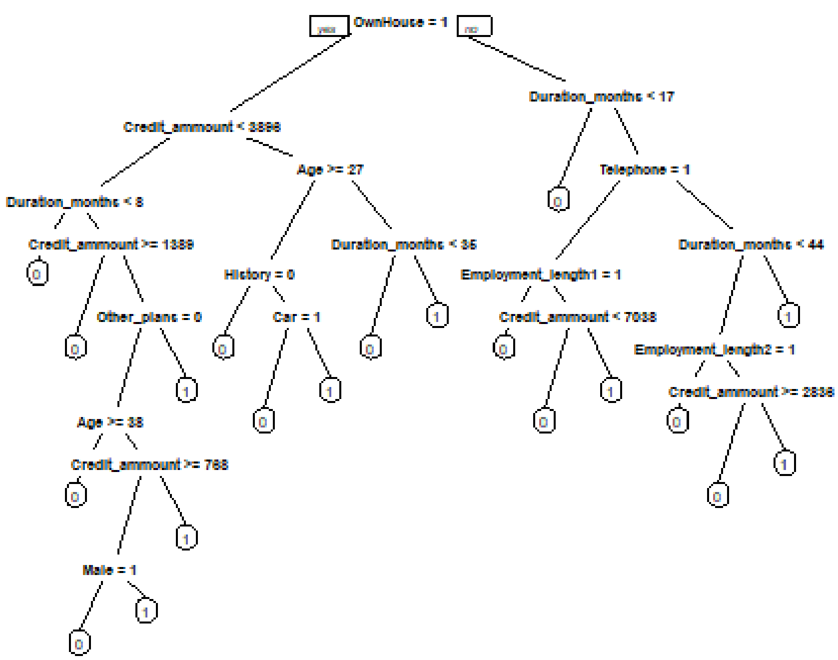
```
#regression tree
tree <- rpart(dapp$Output~.,data=dapp[,-27])
prp(tree)
```



```
rpart.plot(tree)
```



```
#decision tree
datafact<-as.factor(dapp$Output)
tree2<-rpart(datafact~.,data=dapp[,-27])
prp(tree2)
```



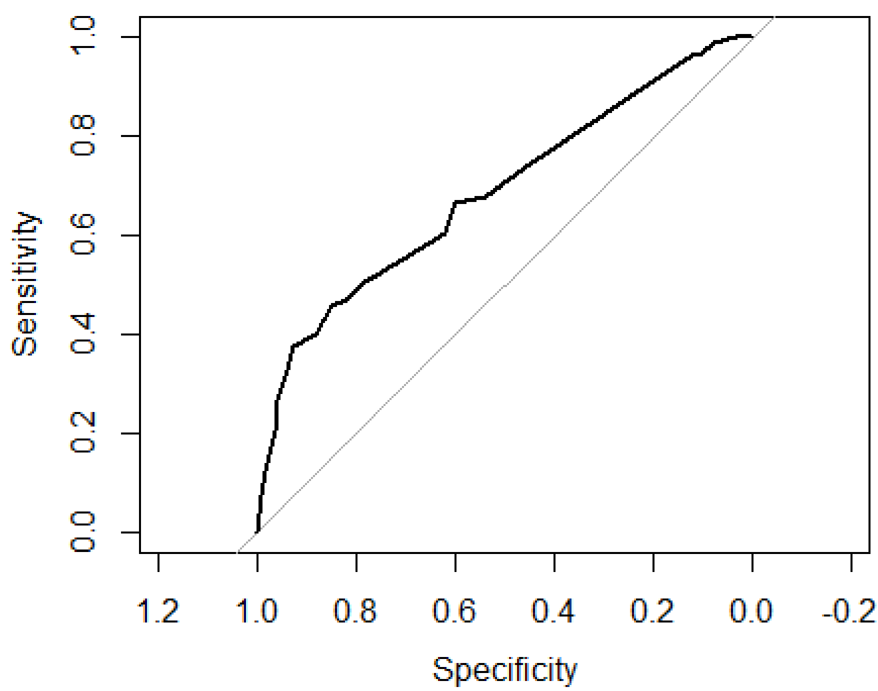
```r
predtree<-predict(tree2, newdata=dtest)
```

Nice interactive visualization that however has to be commented out for the markdown to knit properly

```r
#library(visNetwork)
#library('sparkline')
#visTree(tree2)
```

Finally we calculate our error and model selection values.

```r
#MSE
MSE_tree<-mean(predtree!=dtest$Output)
MSE_tree

## [1] 0.986

#Roc and AUC
plot(roc(dtest$Output,predtree[,2]))

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```




```r
AUC_tree<-auc(roc(dtest$Output,predtree[,2]))

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
AUC_tree

## Area under the curve: 0.6863

#table<-table(predtree[,2],dtest$Output)

#Misclassification error
predtree<-predict(tree2, newdata=dtest,type='class')

conf_tree <- table(dtest$Output,predtree)
conf_tree

##    predtree
##       0   1
##   0 142  25
##   1  45  38

misclass_tree<-(conf_tree[1,2]*loss[1,2]+conf_tree[2,1]*loss[2,1])/nrow(dapp)
/mean(dapp$Output==1)
misclass_tree

## [1] 1.152074
```

Question 4

Finally, we summarise our findings in the following table

```r
mytable <- data.frame(MSE=c(MSE_lm, MSE_log, MSE_KNN, MSE_tree),
    AUC=c(AUC_lm,AUC_log,AUC_KNN,AUC_tree),
    Misclassification_error=c(misclass_lm,misclass_log,misclass_KNN,misclass_
tree), row.names = c("linear probability","logistic", "KNN", "decision tree")
)
print(mytable)

##                      MSE       AUC Misclassification_error
## linear probability 0.316 0.6758892               0.4377880
## logistic           0.296 0.7369598               0.4884793
## KNN                0.332 0.5843013               1.7834101
## decision tree      0.986 0.6863141               1.1520737

View(mytable)
```

By looking at the table, we can see how the Logistic model seems more performing according to both MSE (as it displays the lowest value for the errors) and AUC (as it displays the largest area under the curve).

If we look at misclassification error, on the other hand, the linear probability model seems the most performing, as it displays the lowest value. Our recommendation would be to choose one of the two; in particular, the logistic seems the best one, as it outperformes the others according to 2 criteria out of 3.