Report

2022-05-20

Executive Summary

TBD

Introduction

In the bank industry many bankers have to decided whether or not they should issue a loan to a new coming applicant. In this report, we will use the data set called **German Credit data** which was given to us.

The German credit data set contains 1000 observations of past credit applicants, described by 30 variables. The applicants are described as **Good Credit** risk or **Bad Credit** risk: Therefore, the response variable, studied, is the credit rating.

Response variable: **RESPONSE** in the dataset:

- 0 : Bad credit. In case of bad credit, the banker would not want to issue loan to this person.
- 1 : Good credit. In case of good credit, the banker will want to issue loan to this applicant as it is more likely that the company will benefit from it.

All the other observations are features of the applicants that are going to be studied. It will allow us to perform several machine learning models and deploy a CRISP-DM model to come up with the best classifying model with the highest accuracy as possible. We want to determine whether the new applicant has a 'Good' credit risk, in which case the loan should be issued, or a 'Bad' credit risk, in which case it is not advisable to give him a loan.

The tasks required to perform our analysis is stated as follow.

1/ We first proceeded to some data cleaning, meaning that we sorted the dataset to make it ready for the analysis.

2/ Then we followed by an exploratory data analysis (EDA) where we studied the dataset and the different variables, one by one, and we made an principal component analysis.

3/ Next, came the models analysis, the steps are listed below:

a) Splitting the dataset
b) Balancing the data
c) Fitting the models
d) Accuracy study (scoring)
e) Variable selection and importance
f) Cross-validation / Bootstrap
g) Final Best model

Our very first steps once we received the **German Credit data** was to dig into it and get to know the observations and features we were going to work with.

Get to know the data

The title of the dataset is *German credit data* and the name of the file is **GermanCredit.cvs**.

As said in the introduction, the German Credit data has data on 1000 observations on past credit applicants and it is described by 30 attributes. Each applicant is rated as "Good" or "Bad" credit (encoded as 1 and 0 respectively in the **RESPONSE** variable).

We looked at the attribute Information :

```
## 'data.frame':    1000 obs. of  32 variables:
##  $ OBS.            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ CHK_ACCT        : int  0 1 3 0 0 3 3 1 3 1 ...
##  $ DURATION        : int  6 48 12 42 24 36 24 36 12 30 ...
##  $ HISTORY         : int  4 2 4 2 3 2 2 2 2 4 ...
##  $ NEW_CAR         : int  0 0 0 0 1 0 0 0 0 1 ...
##  $ USED_CAR        : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ FURNITURE       : int  0 0 0 1 0 0 1 0 0 0 ...
##  $ RADIO.TV        : int  1 1 0 0 0 0 0 0 1 0 ...
##  $ EDUCATION       : int  0 0 1 0 0 1 0 0 0 0 ...
##  $ RETRAINING      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ AMOUNT          : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
##  $ SAV_ACCT        : int  4 0 0 0 0 4 2 0 3 0 ...
##  $ EMPLOYMENT      : int  4 2 3 3 2 2 4 2 3 0 ...
##  $ INSTALL_RATE    : int  4 2 2 2 3 2 3 2 2 4 ...
##  $ MALE_DIV        : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ MALE_SINGLE     : int  1 0 1 1 1 1 1 1 0 0 ...
##  $ MALE_MAR_or_WID : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ CO.APPLICANT    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GUARANTOR       : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ PRESENT_RESIDENT: int  4 2 3 4 4 4 4 2 4 2 ...
##  $ REAL_ESTATE     : int  1 1 1 0 0 0 0 0 1 0 ...
##  $ PROP_UNKN_NONE  : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ AGE             : int  67 22 49 45 53 35 53 35 61 28 ...
##  $ OTHER_INSTALL   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ RENT            : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ OWN_RES         : int  1 1 1 0 0 0 1 0 1 1 ...
##  $ NUM_CREDITS     : int  2 1 1 1 2 1 1 1 1 2 ...
##  $ JOB             : int  2 2 1 2 2 1 2 3 1 3 ...
##  $ NUM_DEPENDENTS  : int  1 1 2 2 2 2 1 1 1 1 ...
##  $ TELEPHONE       : int  1 0 0 0 0 1 0 1 0 0 ...
##  $ FOREIGN         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ RESPONSE        : int  1 0 1 1 0 1 1 1 1 0 ...
```

```
##       OBS.           CHK_ACCT        DURATION       HISTORY
##  Min.   :   1.0   Min.   :0.000   Min.   : 4.0   Min.   :0.000
##  1st Qu.: 250.8   1st Qu.:0.000   1st Qu.:12.0   1st Qu.:2.000
##  Median : 500.5   Median :1.000   Median :18.0   Median :2.000
##  Mean   : 500.5   Mean   :1.577   Mean   :20.9   Mean   :2.545
##  3rd Qu.: 750.2   3rd Qu.:3.000   3rd Qu.:24.0   3rd Qu.:4.000
##  Max.   :1000.0   Max.   :3.000   Max.   :72.0   Max.   :4.000
##     NEW_CAR         USED_CAR       FURNITURE        RADIO.TV
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.00
##  Median :0.000   Median :0.000   Median :0.000   Median :0.00
##  Mean   :0.234   Mean   :0.103   Mean   :0.181   Mean   :0.28
```

```
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:1.00
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.00
##    EDUCATION        RETRAINING        AMOUNT         SAV_ACCT
##  Min.   :-1.000   Min.   :0.000   Min.   :  250   Min.   :0.000
##  1st Qu.: 0.000   1st Qu.:0.000   1st Qu.: 1366   1st Qu.:0.000
##  Median : 0.000   Median :0.000   Median : 2320   Median :0.000
##  Mean   : 0.048   Mean   :0.097   Mean   : 3271   Mean   :1.105
##  3rd Qu.: 0.000   3rd Qu.:0.000   3rd Qu.: 3972   3rd Qu.:2.000
##  Max.   : 1.000   Max.   :1.000   Max.   :18424   Max.   :4.000
##    EMPLOYMENT     INSTALL_RATE      MALE_DIV     MALE_SINGLE    MALE_MAR_or_WID
##  Min.   :0.000   Min.   :1.000   Min.   :0.00   Min.   :0.000   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:0.00   1st Qu.:0.000   1st Qu.:0.000
##  Median :2.000   Median :3.000   Median :0.00   Median :1.000   Median :0.000
##  Mean   :2.384   Mean   :2.973   Mean   :0.05   Mean   :0.548   Mean   :0.092
##  3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:0.00   3rd Qu.:1.000   3rd Qu.:0.000
##  Max.   :4.000   Max.   :4.000   Max.   :1.00   Max.   :1.000   Max.   :1.000
##   CO.APPLICANT     GUARANTOR     PRESENT_RESIDENT  REAL_ESTATE
##  Min.   :0.000   Min.   :0.000   Min.   :1.000    Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:2.000    1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :3.000    Median :0.000
##  Mean   :0.041   Mean   :0.053   Mean   :2.845    Mean   :0.282
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:4.000    3rd Qu.:1.000
##  Max.   :1.000   Max.   :2.000   Max.   :4.000    Max.   :1.000
##  PROP_UNKN_NONE       AGE        OTHER_INSTALL        RENT
##  Min.   :0.000   Min.   : 19.0   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.: 27.0   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median : 33.0   Median :0.000   Median :0.000
##  Mean   :0.154   Mean   : 35.6   Mean   :0.186   Mean   :0.179
##  3rd Qu.:0.000   3rd Qu.: 42.0   3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :125.0   Max.   :1.000   Max.   :1.000
##     OWN_RES       NUM_CREDITS         JOB        NUM_DEPENDENTS
##  Min.   :0.000   Min.   :1.000   Min.   :0.000   Min.   :1.000
##  1st Qu.:0.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:1.000
##  Median :1.000   Median :1.000   Median :2.000   Median :1.000
##  Mean   :0.713   Mean   :1.407   Mean   :1.904   Mean   :1.155
##  3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :4.000   Max.   :3.000   Max.   :2.000
##    TELEPHONE        FOREIGN         RESPONSE
##  Min.   :0.000   Min.   :0.000   Min.   :0.0
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.0
##  Median :0.000   Median :0.000   Median :1.0
##  Mean   :0.404   Mean   :0.037   Mean   :0.7
##  3rd Qu.:1.000   3rd Qu.:0.000   3rd Qu.:1.0
##  Max.   :1.000   Max.   :1.000   Max.   :1.0
```

We noticed that the variable **EDUCATION** has a minimum value of '-1' but it should be '0' since there are only 2 levels (0 and 1). Indeed, the observation 37 indicate a value of '-1' for **EDUCATION**. We notice another strange value, in the variable **GUARANTOR**, the maximum value is of '2' while it does not mean anything in our data set.

After discussion with the Banker, he gave us the correct values to these 2 mistakes. Observation 37 of **EDUCATION** and observation 234 of **GUARANTOR** should be equal to 1. We corrected these two values.

We also saw that the variable **AGE** has a maximum of 125. This is strange because it is very unlikely that someone lives to the age of 125. We talked to the banker again and he confirmed our doubts by telling us that a mistake has been made. At the observation 537, the correct age of the client is 75 years old. He asked us to correct this value in our data set.

After looking at the different attributes, we concluded that there were no missing values.

The response variable is identified as being the column named '**Response**' and it apprears to be the last column on the data.

It is a dummy variable with 0/1.

1. 0 : No, the credit rating is bad.
2. 1 : Yes, the credit rating is good.

We had to make sure that the class of the variables are correct. As described above, all the variables are defined as *integer* but we know that we should have numerical and categorical variables in our dataset. Therefore, we have to transform the class of some of them.

```
## 'data.frame':    1000 obs. of  32 variables:
##  $ OBS.            : Factor w/ 1000 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ CHK_ACCT        : Factor w/ 4 levels "0","1","2","3": 1 2 4 1 1 4 4 2 4 2 ...
##  $ DURATION        : num  6 48 12 42 24 36 24 36 12 30 ...
##  $ HISTORY         : Factor w/ 5 levels "0","1","2","3",..: 5 3 5 3 4 3 3 3 3 5 ...
##  $ NEW_CAR         : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 2 ...
##  $ USED_CAR        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ FURNITURE       : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 2 1 1 1 ...
##  $ RADIO.TV        : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 2 1 ...
##  $ EDUCATION       : Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 1 1 1 ...
##  $ RETRAINING      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AMOUNT          : num  1169 5951 2096 7882 4870 ...
##  $ SAV_ACCT        : Factor w/ 5 levels "0","1","2","3",..: 5 1 1 1 1 5 3 1 4 1 ...
##  $ EMPLOYMENT      : Factor w/ 5 levels "0","1","2","3",..: 5 3 4 4 3 3 5 3 4 1 ...
##  $ INSTALL_RATE    : num  4 2 2 2 3 2 3 2 2 4 ...
##  $ MALE_DIV        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ MALE_SINGLE     : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2 1 1 ...
##  $ MALE_MAR_or_WID : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ CO.APPLICANT    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ GUARANTOR       : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ PRESENT_RESIDENT: Factor w/ 4 levels "1","2","3","4": 4 2 3 4 4 4 4 2 4 2 ...
##  $ REAL_ESTATE     : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 1 2 1 ...
##  $ PROP_UNKN_NONE  : Factor w/ 2 levels "0","1": 1 1 1 1 2 2 1 1 1 1 ...
##  $ AGE             : num  67 22 49 45 53 35 53 35 61 28 ...
##  $ OTHER_INSTALL   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ RENT            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ OWN_RES         : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 2 1 2 2 ...
##  $ NUM_CREDITS     : num  2 1 1 1 2 1 1 1 1 2 ...
##  $ JOB             : Factor w/ 4 levels "0","1","2","3": 3 3 2 3 3 2 3 4 2 4 ...
##  $ NUM_DEPENDENTS  : num  1 1 2 2 2 2 1 1 1 1 ...
##  $ TELEPHONE       : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 1 2 1 1 ...
##  $ FOREIGN         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ RESPONSE        : Factor w/ 2 levels "0","1": 2 1 2 2 1 2 2 2 2 1 ...
```

The binomial data are set as factors and the others as numerical.

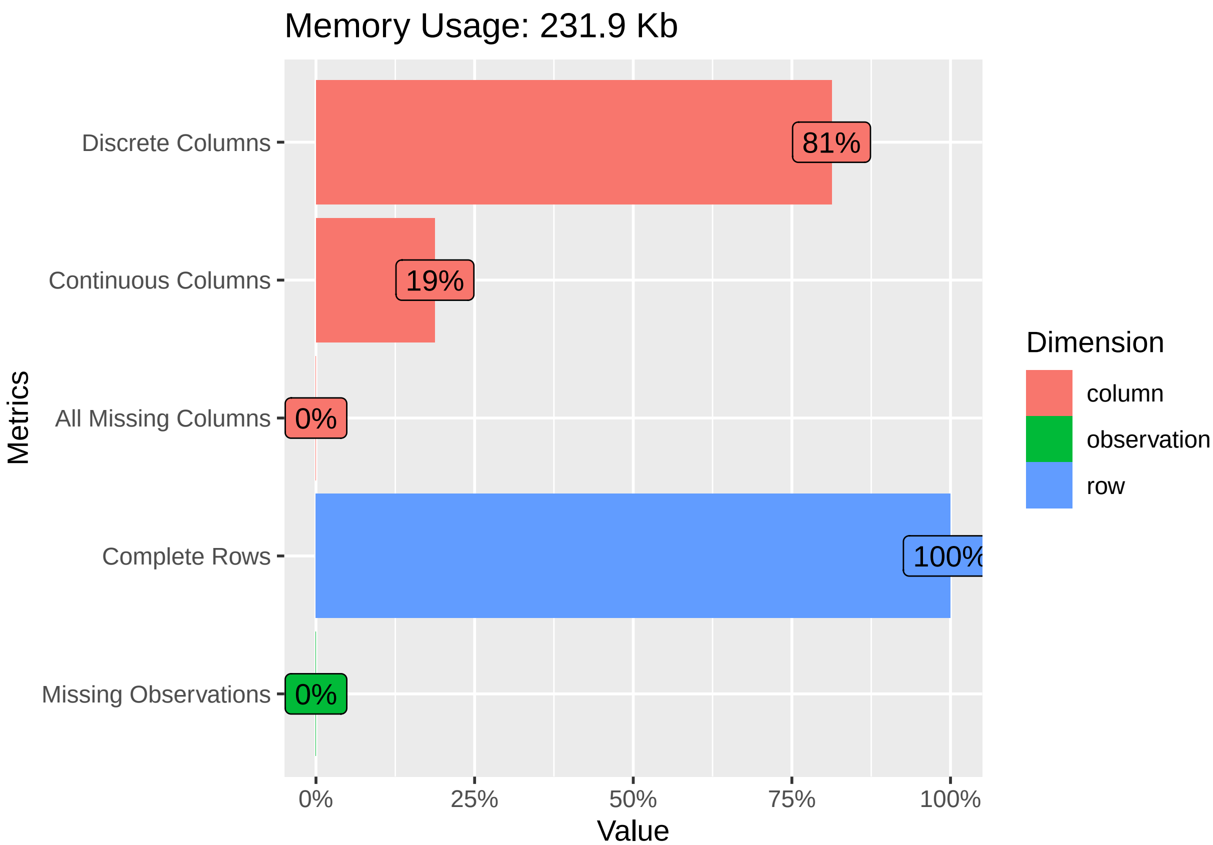
We then described the variables one more time and we should get better results.

```
##                   vars    n    mean      sd median trimmed     mad min   max
## OBS.*                1 1000  500.50  288.82  500.5  500.50  370.65   1  1000
```

```
## CHK_ACCT*            2 1000    2.58    1.26    2.0    2.60    1.48   1     4
## DURATION             3 1000   20.90   12.06   18.0   19.47    8.90   4    72
## HISTORY*             4 1000    3.54    1.08    3.0    3.59    0.00   1     5
## NEW_CAR*             5 1000    1.23    0.42    1.0    1.17    0.00   1     2
## USED_CAR*            6 1000    1.10    0.30    1.0    1.00    0.00   1     2
## FURNITURE*           7 1000    1.18    0.39    1.0    1.10    0.00   1     2
## RADIO.TV*            8 1000    1.28    0.45    1.0    1.23    0.00   1     2
## EDUCATION*           9 1000    1.05    0.22    1.0    1.00    0.00   1     2
## RETRAINING*         10 1000    1.10    0.30    1.0    1.00    0.00   1     2
## AMOUNT              11 1000 3271.26 2822.74 2319.5 2754.57 1627.15 250 18424
## SAV_ACCT*           12 1000    2.10    1.58    1.0    1.88    0.00   1     5
## EMPLOYMENT*         13 1000    3.38    1.21    3.0    3.43    1.48   1     5
## INSTALL_RATE        14 1000    2.97    1.12    3.0    3.09    1.48   1     4
## MALE_DIV*           15 1000    1.05    0.22    1.0    1.00    0.00   1     2
## MALE_SINGLE*        16 1000    1.55    0.50    2.0    1.56    0.00   1     2
## MALE_MAR_or_WID*    17 1000    1.09    0.29    1.0    1.00    0.00   1     2
## CO.APPLICANT*       18 1000    1.04    0.20    1.0    1.00    0.00   1     2
## GUARANTOR*          19 1000    1.05    0.22    1.0    1.00    0.00   1     2
## PRESENT_RESIDENT*   20 1000    2.85    1.10    3.0    2.93    1.48   1     4
## REAL_ESTATE*        21 1000    1.28    0.45    1.0    1.23    0.00   1     2
## PROP_UNKN_NONE*     22 1000    1.15    0.36    1.0    1.07    0.00   1     2
## AGE                 23 1000   35.55   11.38   33.0   34.17   10.38  19    75
## OTHER_INSTALL*      24 1000    1.19    0.39    1.0    1.11    0.00   1     2
## RENT*               25 1000    1.18    0.38    1.0    1.10    0.00   1     2
## OWN_RES*            26 1000    1.71    0.45    2.0    1.77    0.00   1     2
## NUM_CREDITS         27 1000    1.41    0.58    1.0    1.33    0.00   1     4
## JOB*                28 1000    2.90    0.65    3.0    2.91    0.00   1     4
## NUM_DEPENDENTS      29 1000    1.16    0.36    1.0    1.07    0.00   1     2
## TELEPHONE*          30 1000    1.40    0.49    1.0    1.38    0.00   1     2
## FOREIGN*            31 1000    1.04    0.19    1.0    1.00    0.00   1     2
## RESPONSE*           32 1000    1.70    0.46    2.0    1.75    0.00   1     2
##                   range  skew kurtosis    se
## OBS.*               999  0.00    -1.20  9.13
## CHK_ACCT*             3  0.01    -1.66  0.04
## DURATION             68  1.09     0.90  0.38
## HISTORY*              4 -0.01    -0.59  0.03
## NEW_CAR*              1  1.25    -0.43  0.01
## USED_CAR*             1  2.61     4.81  0.01
## FURNITURE*            1  1.65     0.74  0.01
## RADIO.TV*             1  0.98    -1.04  0.01
## EDUCATION*            1  4.12    15.02  0.01
## RETRAINING*           1  2.72     5.40  0.01
## AMOUNT            18174  1.94     4.25 89.26
## SAV_ACCT*             4  1.01    -0.69  0.05
## EMPLOYMENT*           4 -0.12    -0.94  0.04
## INSTALL_RATE          3 -0.53    -1.21  0.04
## MALE_DIV*             1  4.12    15.02  0.01
## MALE_SINGLE*          1 -0.19    -1.96  0.02
## MALE_MAR_or_WID*      1  2.82     5.95  0.01
## CO.APPLICANT*         1  4.62    19.39  0.01
## GUARANTOR*            1  4.03    14.25  0.01
## PRESENT_RESIDENT*     3 -0.27    -1.38  0.03
## REAL_ESTATE*          1  0.97    -1.07  0.01
## PROP_UNKN_NONE*       1  1.91     1.67  0.01
```

```
## AGE                   56  1.02     0.58  0.36
## OTHER_INSTALL*         1  1.61     0.60  0.01
## RENT*                  1  1.67     0.80  0.01
## OWN_RES*               1 -0.94    -1.12  0.01
## NUM_CREDITS            3  1.27     1.58  0.02
## JOB*                   3 -0.37     0.49  0.02
## NUM_DEPENDENTS         1  1.90     1.63  0.01
## TELEPHONE*             1  0.39    -1.85  0.02
## FOREIGN*               1  4.90    22.02  0.01
## RESPONSE*              1 -0.87    -1.24  0.01
```

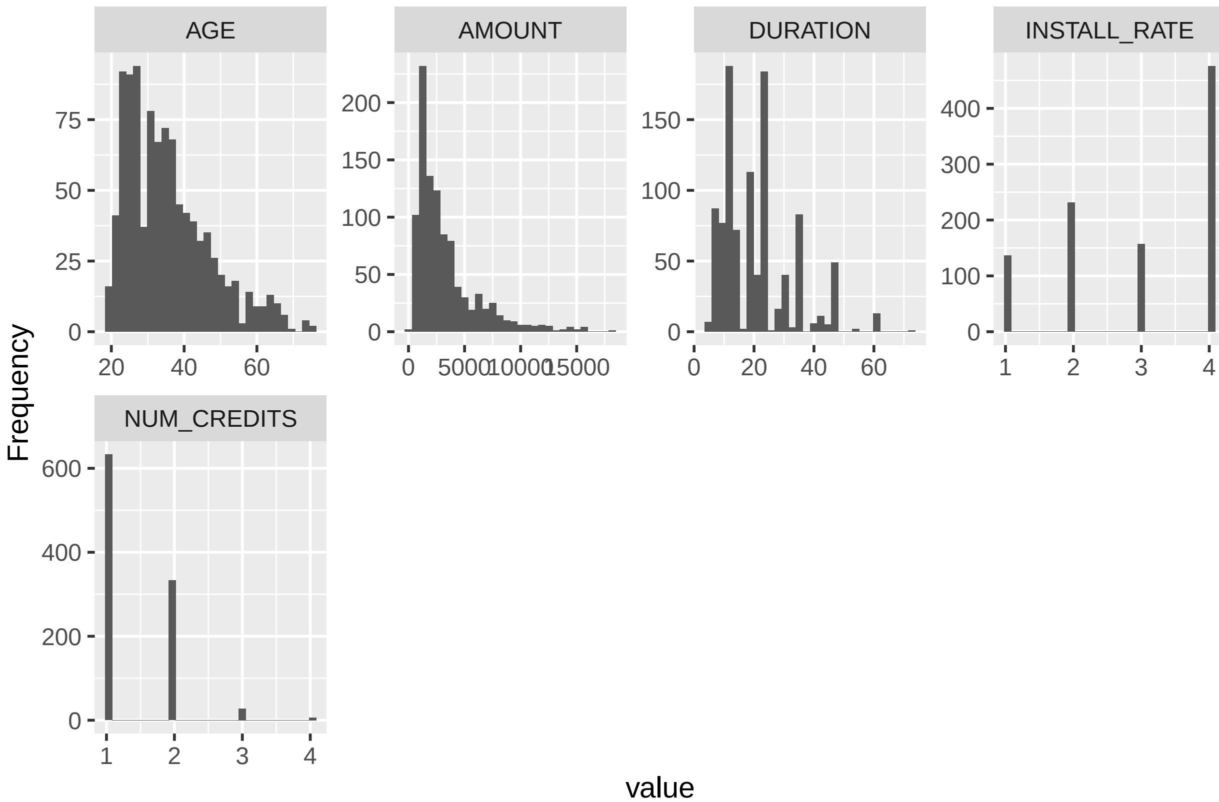
```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 1000      32               26                  6                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    0          1000              32000       237424
```

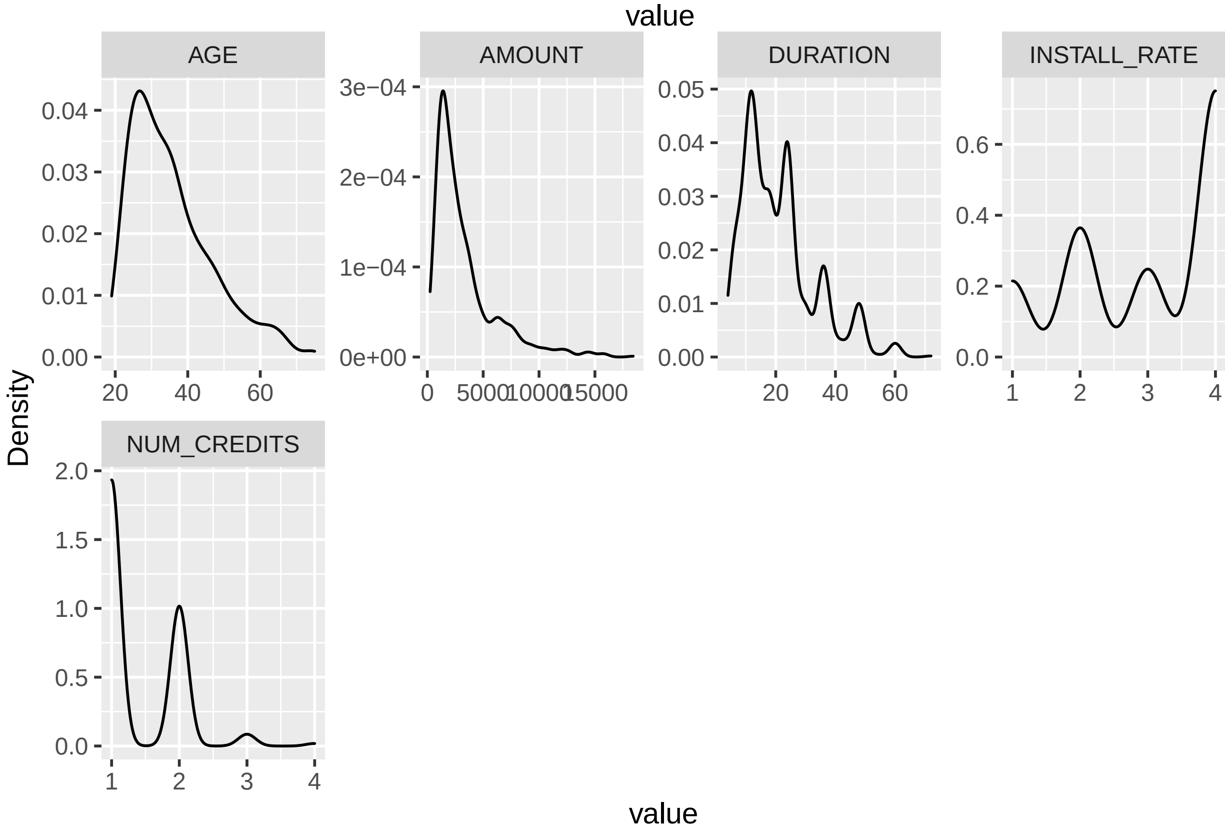


The plot helps us to see the percentage of continuous variable, the percentage of discrete variables and whether or not some observations are missing.

Visualization of the data

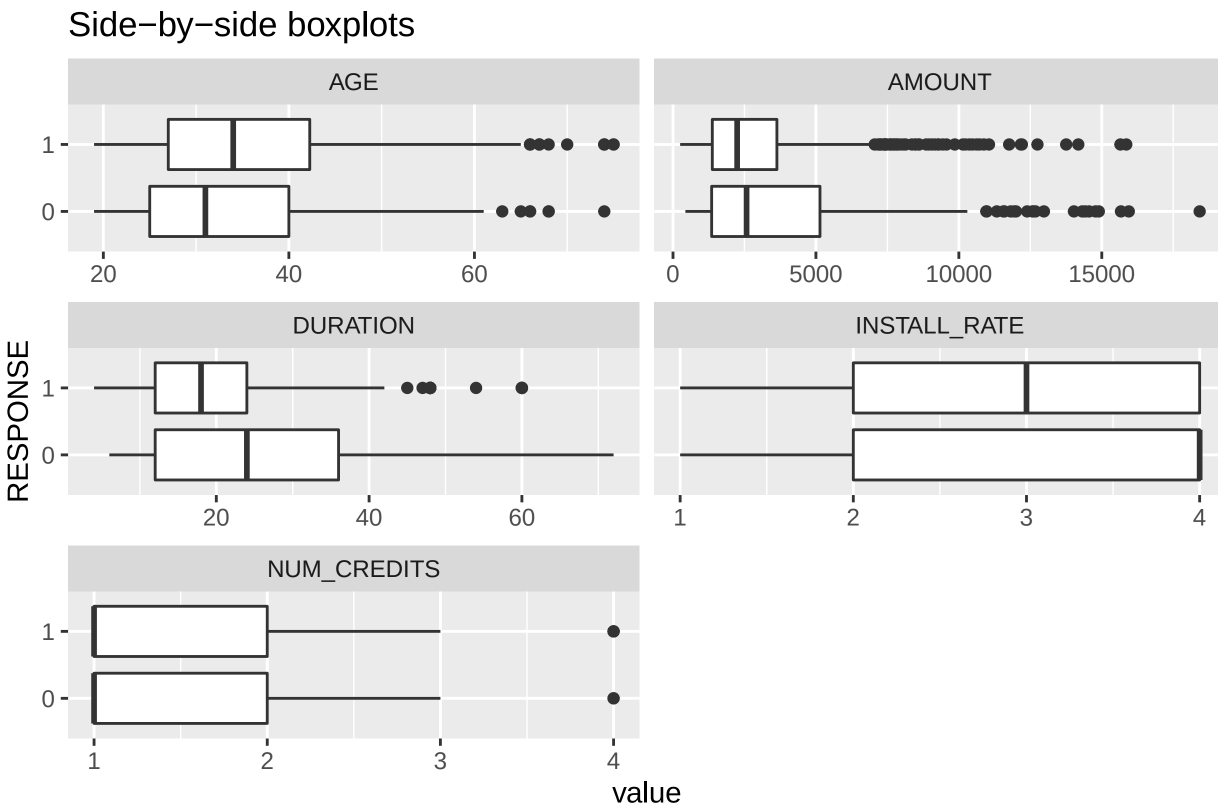
First, we plot all the continuous variables into histograms and their corresponding density plots.





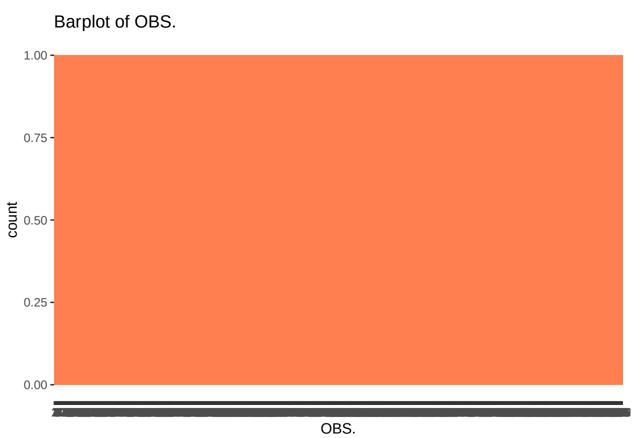
Our first notice is that the data are not really normally distributed. Some of them are right-tailed.

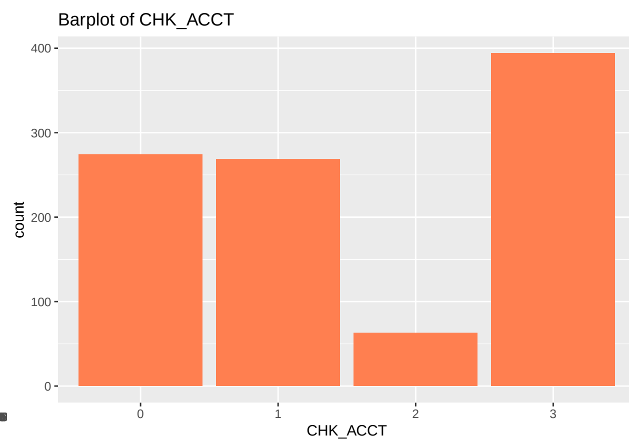
We can look at the tails and outliers more carefully through boxplots.

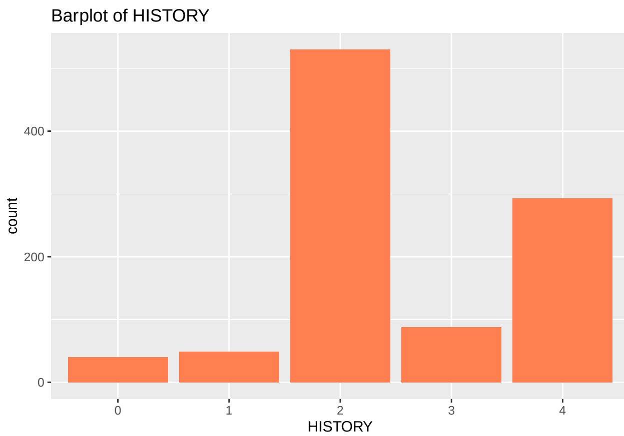


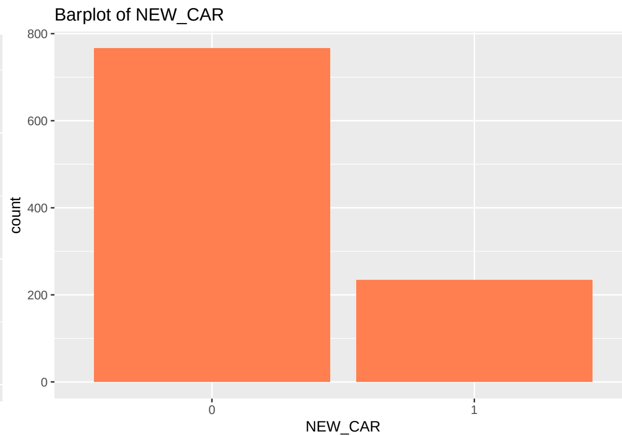
This seems not to be disturbing. It makes sense that only a few people has a big credit amount. However it seems that the 'bad' clients tends to ask for bigger credit amount than 'good' clients.

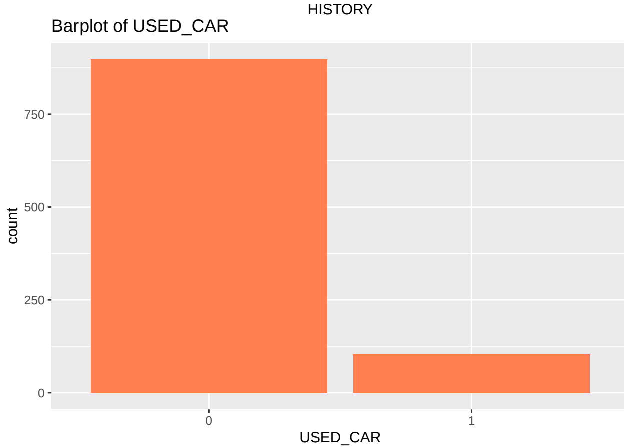
Now, we can make some barplots of the categorical variables.

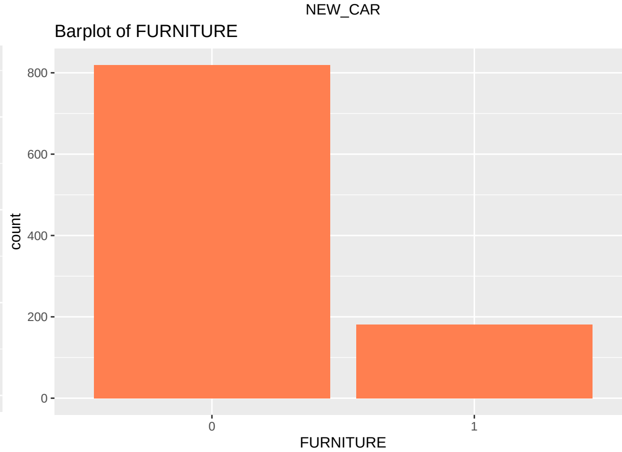


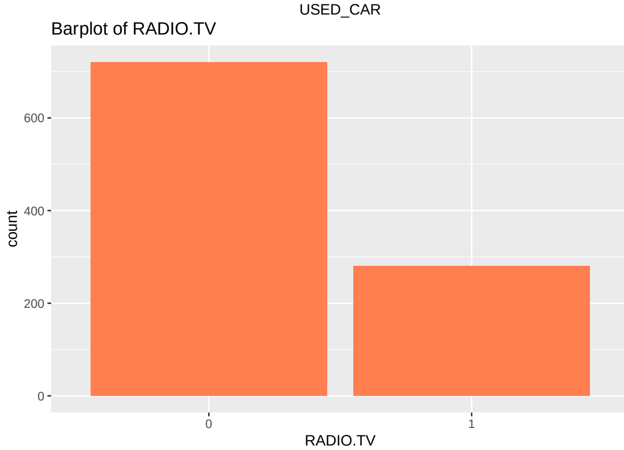


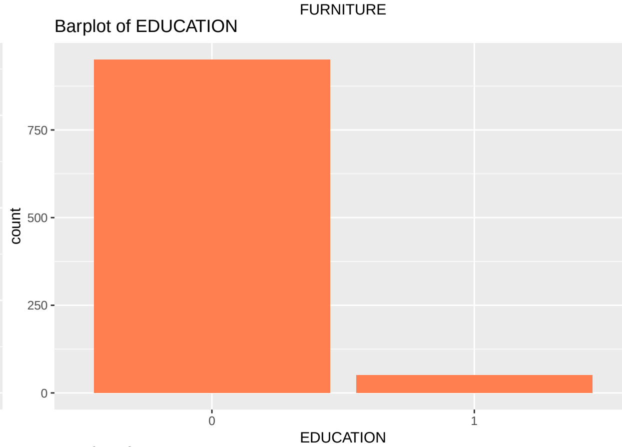

Barplot of HISTORY
count
0
200
400
0
1
2
3
4
HISTORY

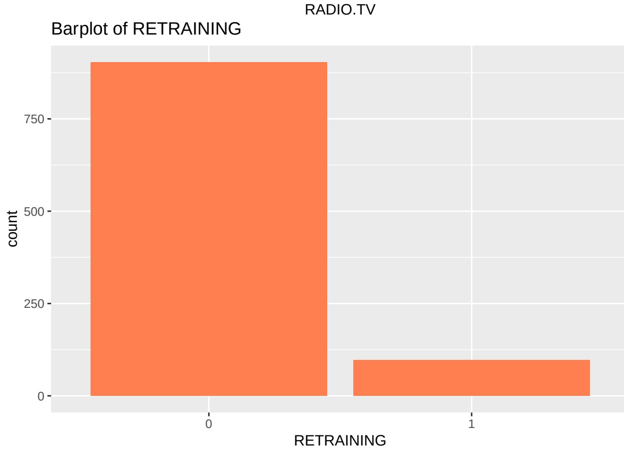

Barplot of NEW_CAR
count
0
200
400
600
800
0
1
NEW_CAR

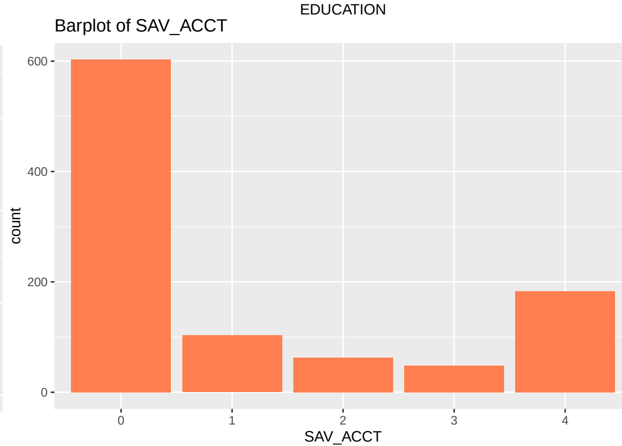

Barplot of USED_CAR
count
0
250
500
750
0
1
USED_CAR

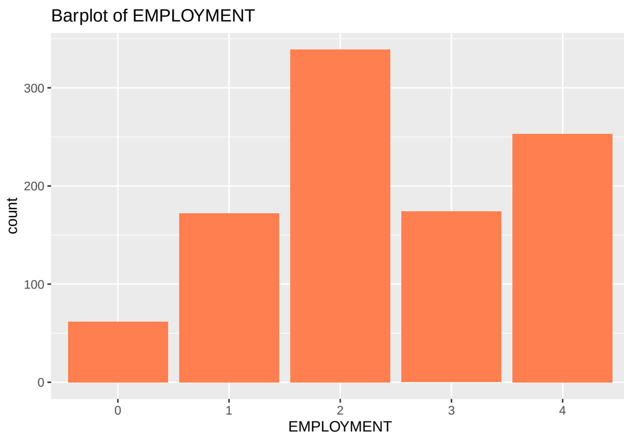

Barplot of FURNITURE
count
0
200
400
600
800
0
1
FURNITURE

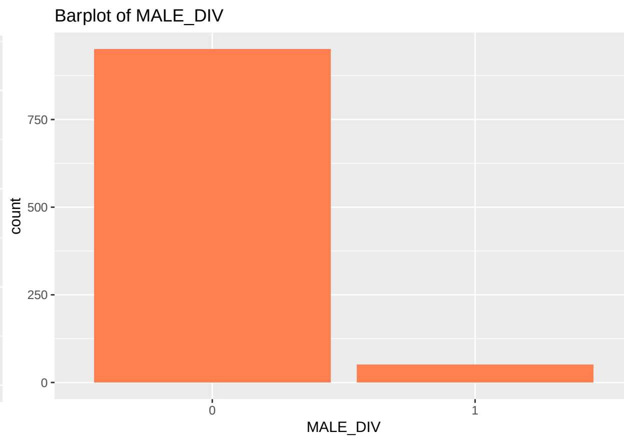

Barplot of RADIO.TV
count
0
200
400
600
0
1
RADIO.TV

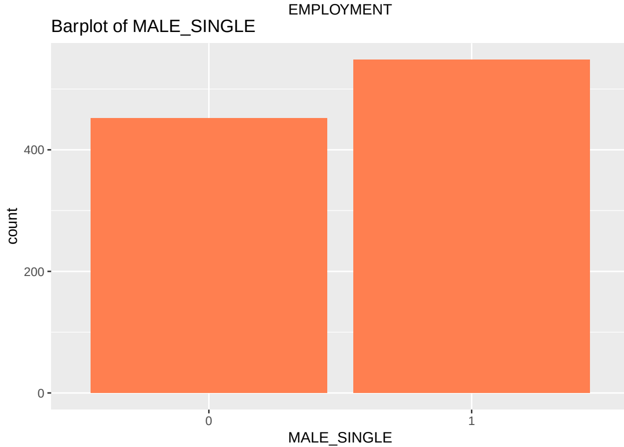

Barplot of EDUCATION
count
0
250
500
750
0
1
EDUCATION

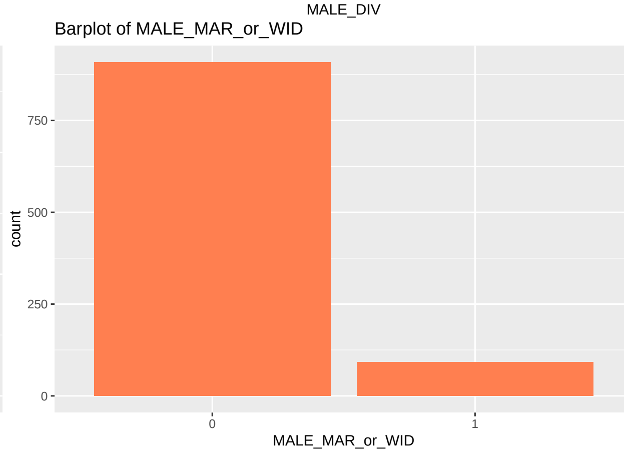

Barplot of RETRAINING
count
0
250
500
750
0
1
RETRAINING

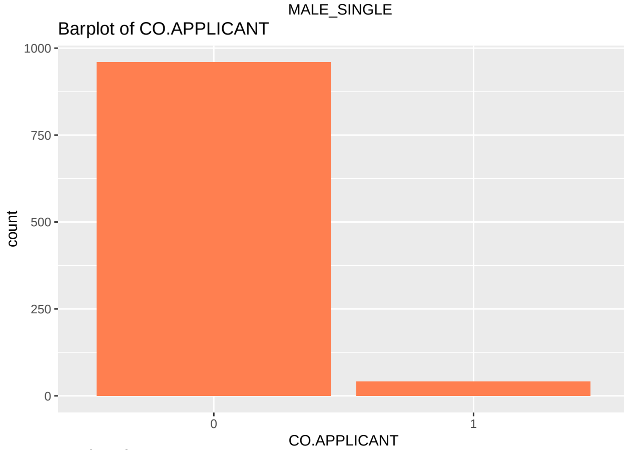

Barplot of SAV_ACCT
count
0
200
400
600
0
1
2
3
4
SAV_ACCT

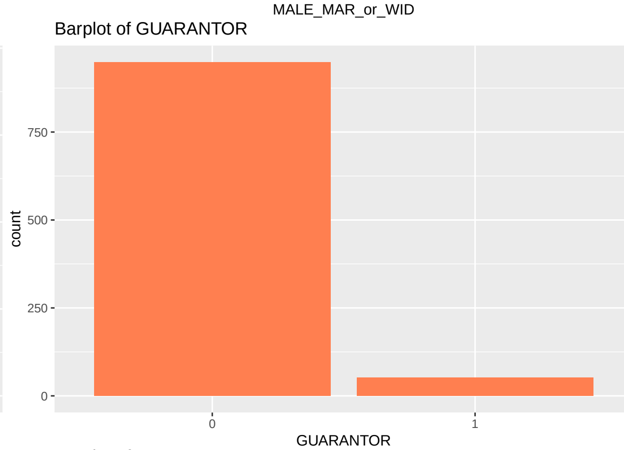

Barplot of EMPLOYMENT
count
300
200
100
0
0 1 2 3 4
EMPLOYMENT

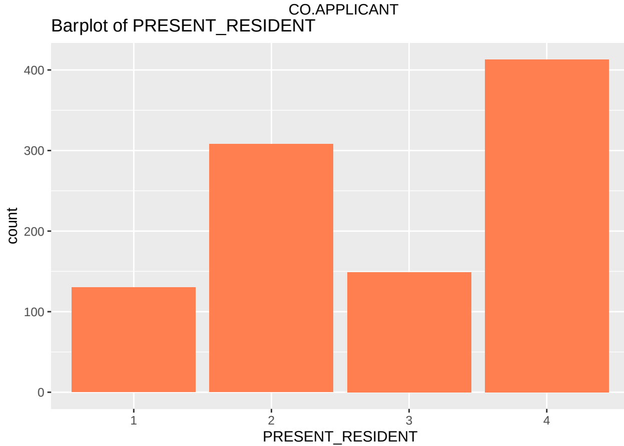

Barplot of MALE_DIV
count
750
500
250
0
0 1
MALE_DIV

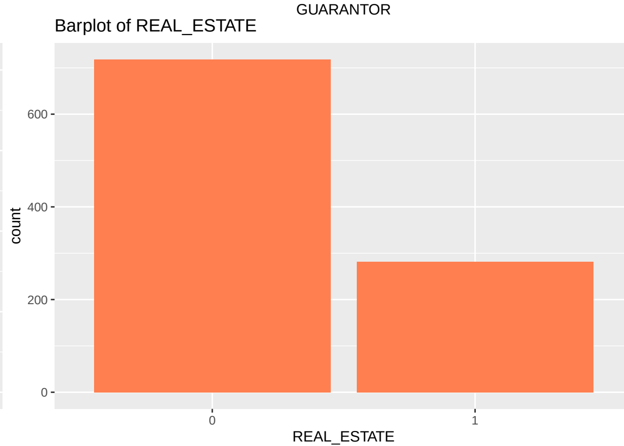

Barplot of MALE_SINGLE
count
400
200
0
0 1
MALE_SINGLE

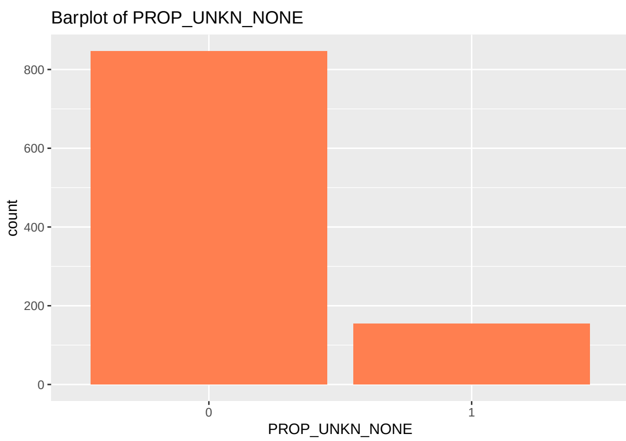

Barplot of MALE_MAR_or_WID
count
750
500
250
0
0 1
MALE_MAR_or_WID

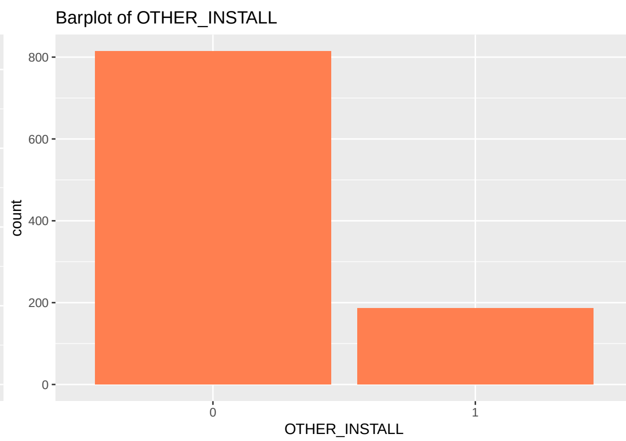

Barplot of CO.APPLICANT
count
1000
750
500
250
0
0 1
CO.APPLICANT


Barplot of GUARANTOR
count
750
500
250
0
0 1
GUARANTOR

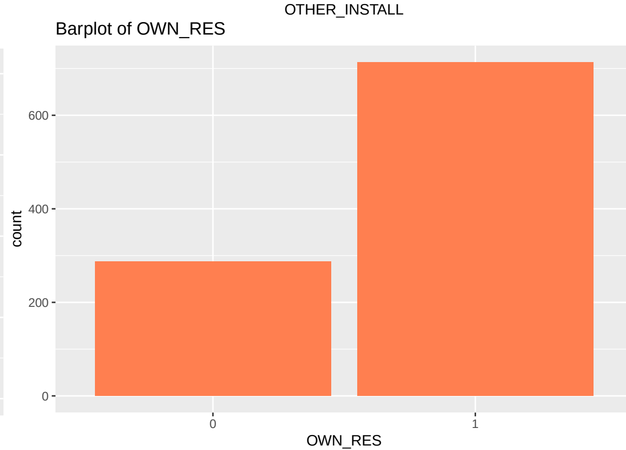

Barplot of PRESENT_RESIDENT
count
400
300
200
100
0
1 2 3 4
PRESENT_RESIDENT


Barplot of REAL_ESTATE
count
600
400
200
0
0 1
REAL_ESTATE

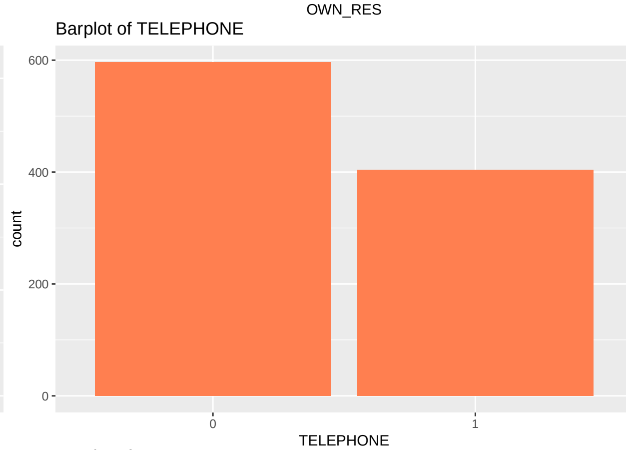

Barplot of PROP_UNKN_NONE
count
800
600
400
200
0
0
1
PROP_UNKN_NONE

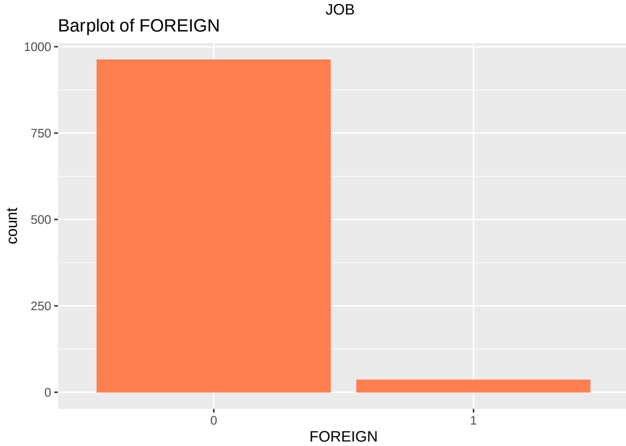

Barplot of OTHER_INSTALL
count
800
600
400
200
0
0
1
OTHER_INSTALL

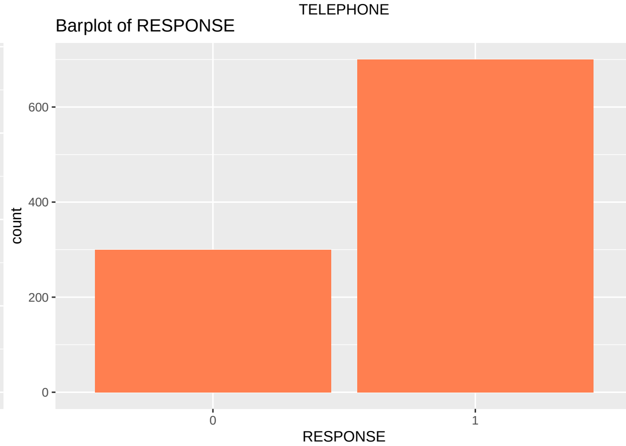

Barplot of RENT
count
800
600
400
200
0
0
1
RENT


Barplot of OWN_RES
count
600
400
200
0
0
1
OWN_RES


Barplot of JOB
count
600
400
200
0
0
1
2
3
JOB


Barplot of TELEPHONE
count
600
400
200
0
0
1
TELEPHONE


Barplot of FOREIGN
count
1000
750
500
250
0
0
1
FOREIGN


Barplot of RESPONSE
count
600
400
200
0
0
1
RESPONSE

From those barplots we can see:

- The majority of people do not check their account status. (CHK_ACCT)
- Most people have an average balance of less than < 100 DM in their saving account. (SAV_ACCT)
- Most of the applicants has its own residence. (OWN_RES)
- Almost none of the applicants is a foreign worker. (FOREIGN)
- We have more information on people that are 'good' applicants and less information on 'bad' applicants. It would be better to have more information on 'bad' applicants as well in order to make good predictions with models. We will have to take this into account later. (RESPONSE)

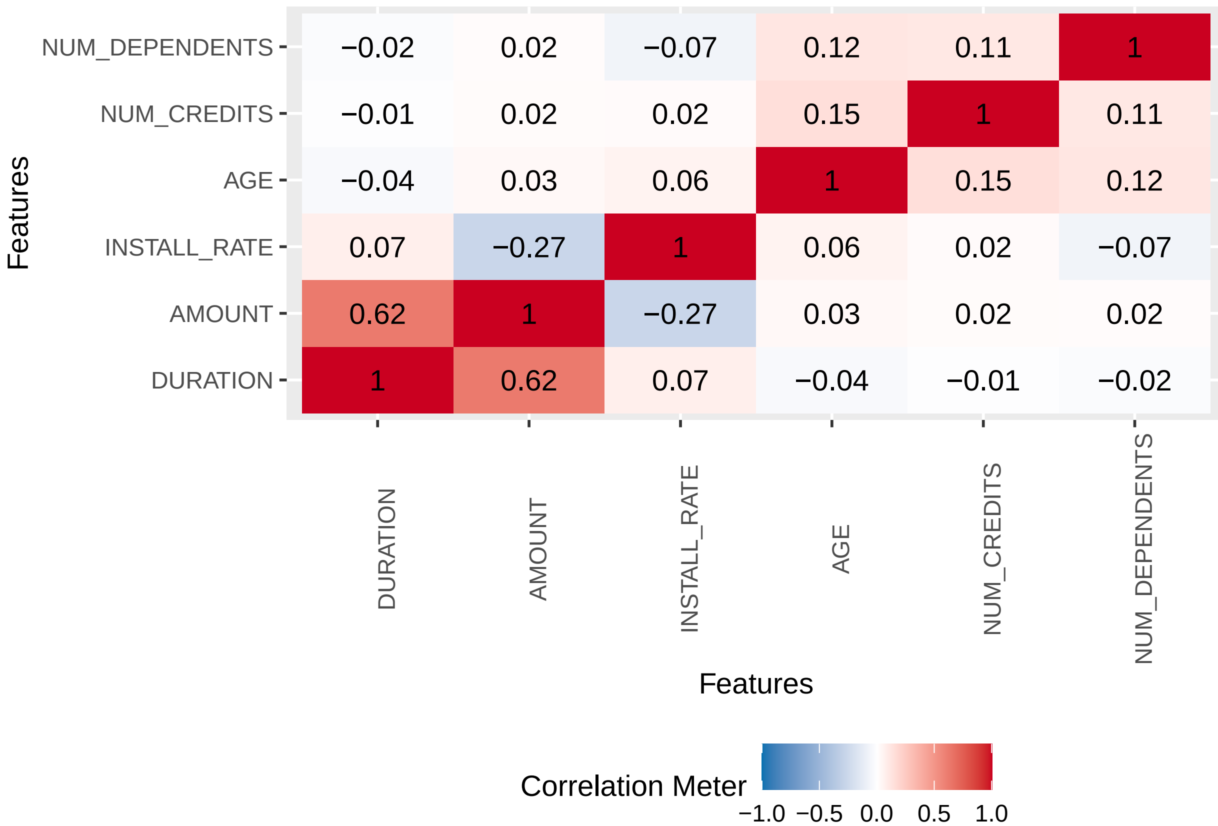
A general summary can be done.

```
## Data Frame Summary
## Dimensions: 1000 x 32
## Duplicates: 0
##
## +----+------------------+------------------------------+---------------------+-----------------------
## | No | Variable         | Stats / Values               | Freqs (% of Valid)  | Graph
## +====+==================+==============================+=====================+=======================
## | 1  | OBS.             | 1. 1                         |   1 ( 0.1%)         |
## |    | [factor]         | 2. 2                         |   1 ( 0.1%)         |
## |    |                  | 3. 3                         |   1 ( 0.1%)         |
## |    |                  | 4. 4                         |   1 ( 0.1%)         |
## |    |                  | 5. 5                         |   1 ( 0.1%)         |
## |    |                  | 6. 6                         |   1 ( 0.1%)         |
## |    |                  | 7. 7                         |   1 ( 0.1%)         |
## |    |                  | 8. 8                         |   1 ( 0.1%)         |
## |    |                  | 9. 9                         |   1 ( 0.1%)         |
## |    |                  | 10. 10                       |   1 ( 0.1%)         |
## |    |                  | [ 990 others ]               | 990 (99.0%)         | IIIIIIIIIIIIIIIIIII
## +----+------------------+------------------------------+---------------------+-----------------------
## | 2  | CHK_ACCT         | 1. 0                         | 274 (27.4%)         | IIIII
## |    | [factor]         | 2. 1                         | 269 (26.9%)         | IIIII
## |    |                  | 3. 2                         |  63 ( 6.3%)         | I
## |    |                  | 4. 3                         | 394 (39.4%)         | IIIIIII
## +----+------------------+------------------------------+---------------------+-----------------------
## | 3  | DURATION         | Mean (sd) : 20.9 (12.1)      | 33 distinct values  |     :
## |    | [numeric]        | min < med < max:             |                     |   : :
## |    |                  | 4 < 18 < 72                  |                     | . : :
## |    |                  | IQR (CV) : 12 (0.6)          |                     | : : :   .
## |    |                  |                              |                     | : : : : : . :
## +----+------------------+------------------------------+---------------------+-----------------------
## | 4  | HISTORY          | 1. 0                         |  40 ( 4.0%)         |
## |    | [factor]         | 2. 1                         |  49 ( 4.9%)         |
## |    |                  | 3. 2                         | 530 (53.0%)         | IIIIIIIIII
## |    |                  | 4. 3                         |  88 ( 8.8%)         | I
## |    |                  | 5. 4                         | 293 (29.3%)         | IIIII
## +----+------------------+------------------------------+---------------------+-----------------------
## | 5  | NEW_CAR          | 1. 0                         | 766 (76.6%)         | IIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 234 (23.4%)         | IIII
## +----+------------------+------------------------------+---------------------+-----------------------
## | 6  | USED_CAR         | 1. 0                         | 897 (89.7%)         | IIIIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 103 (10.3%)         | II
## +----+------------------+------------------------------+---------------------+-----------------------
## | 7  | FURNITURE        | 1. 0                         | 819 (81.9%)         | IIIIIIIIIIIIIIII
```

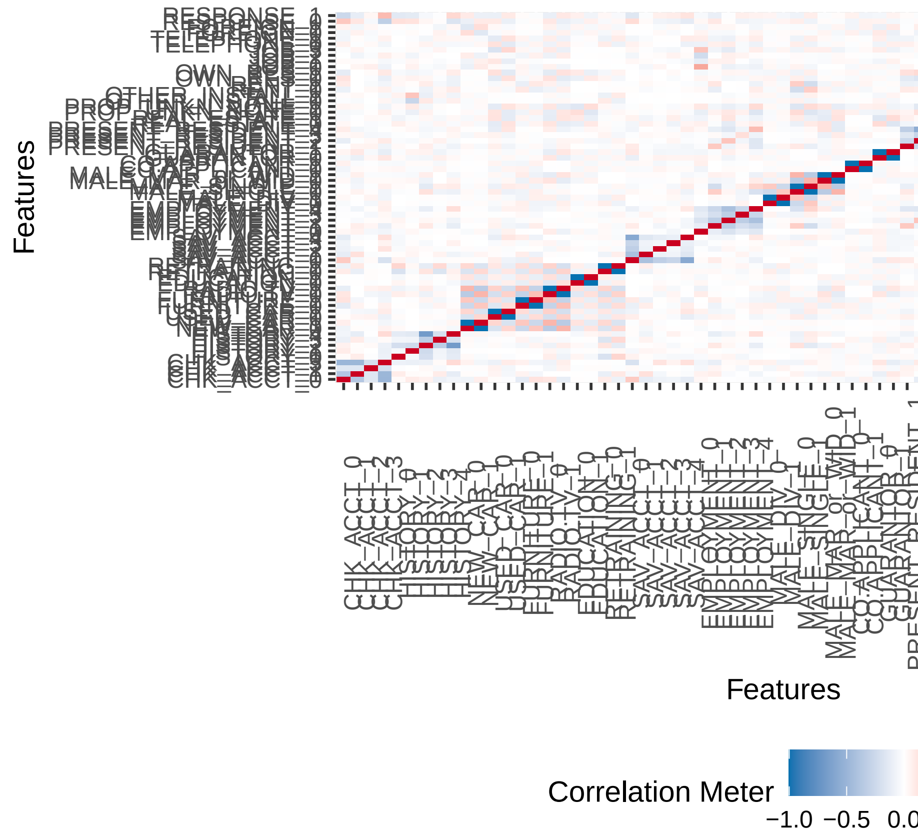
```
## |    | [factor]        | 2. 1                         | 181 (18.1%)         | III
## +----+-----------------+------------------------------+---------------------+----------------------
## | 8  | RADIO.TV        | 1. 0                         | 720 (72.0%)         | IIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         | 280 (28.0%)         | IIIII
## +----+-----------------+------------------------------+---------------------+----------------------
## | 9  | EDUCATION       | 1. 0                         | 950 (95.0%)         | IIIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  50 ( 5.0%)         | I
## +----+-----------------+------------------------------+---------------------+----------------------
## | 10 | RETRAINING      | 1. 0                         | 903 (90.3%)         | IIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  97 ( 9.7%)         | I
## +----+-----------------+------------------------------+---------------------+----------------------
## | 11 | AMOUNT          | Mean (sd) : 3271.3 (2822.7) | 921 distinct values | :
## |    | [numeric]       | min < med < max:             |                     | : .
## |    |                 | 250 < 2319.5 < 18424         |                     | : :
## |    |                 | IQR (CV) : 2606.8 (0.9)      |                     | : :
## |    |                 |                              |                     | : : : : .
## +----+-----------------+------------------------------+---------------------+----------------------
## | 12 | SAV_ACCT        | 1. 0                         | 603 (60.3%)         | IIIIIIIIIIII
## |    | [factor]        | 2. 1                         | 103 (10.3%)         | II
## |    |                 | 3. 2                         |  63 ( 6.3%)         | I
## |    |                 | 4. 3                         |  48 ( 4.8%)         |
## |    |                 | 5. 4                         | 183 (18.3%)         | III
## +----+-----------------+------------------------------+---------------------+----------------------
## | 13 | EMPLOYMENT      | 1. 0                         |  62 ( 6.2%)         | I
## |    | [factor]        | 2. 1                         | 172 (17.2%)         | III
## |    |                 | 3. 2                         | 339 (33.9%)         | IIIIII
## |    |                 | 4. 3                         | 174 (17.4%)         | III
## |    |                 | 5. 4                         | 253 (25.3%)         | IIIII
## +----+-----------------+------------------------------+---------------------+----------------------
## | 14 | INSTALL_RATE    | Mean (sd) : 3 (1.1)          | 1 : 136 (13.6%)     | II
## |    | [numeric]       | min < med < max:             | 2 : 231 (23.1%)     | IIII
## |    |                 | 1 < 3 < 4                    | 3 : 157 (15.7%)     | III
## |    |                 | IQR (CV) : 2 (0.4)           | 4 : 476 (47.6%)     | IIIIIIIII
## +----+-----------------+------------------------------+---------------------+----------------------
## | 15 | MALE_DIV        | 1. 0                         | 950 (95.0%)         | IIIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  50 ( 5.0%)         | I
## +----+-----------------+------------------------------+---------------------+----------------------
## | 16 | MALE_SINGLE     | 1. 0                         | 452 (45.2%)         | IIIIIIIII
## |    | [factor]        | 2. 1                         | 548 (54.8%)         | IIIIIIIIII
## +----+-----------------+------------------------------+---------------------+----------------------
## | 17 | MALE_MAR_or_WID | 1. 0                         | 908 (90.8%)         | IIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  92 ( 9.2%)         | I
## +----+-----------------+------------------------------+---------------------+----------------------
## | 18 | CO.APPLICANT    | 1. 0                         | 959 (95.9%)         | IIIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  41 ( 4.1%)         |
## +----+-----------------+------------------------------+---------------------+----------------------
## | 19 | GUARANTOR       | 1. 0                         | 948 (94.8%)         | IIIIIIIIIIIIIIIIII
## |    | [factor]        | 2. 1                         |  52 ( 5.2%)         | I
## +----+-----------------+------------------------------+---------------------+----------------------
## | 20 | PRESENT_RESIDENT | 1. 1                        | 130 (13.0%)         | II
## |    | [factor]        | 2. 2                         | 308 (30.8%)         | IIIIII
## |    |                 | 3. 3                         | 149 (14.9%)         | II
## |    |                 | 4. 4                         | 413 (41.3%)         | IIIIIIII
## +----+-----------------+------------------------------+---------------------+----------------------
```

```
## | 21 | REAL_ESTATE      | 1. 0                         | 718 (71.8%)        | IIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 282 (28.2%)        | IIIII
## +----+------------------+------------------------------+--------------------+----------------------
## | 22 | PROP_UNKN_NONE   | 1. 0                         | 846 (84.6%)        | IIIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 154 (15.4%)        | III
## +----+------------------+------------------------------+--------------------+----------------------
## | 23 | AGE              | Mean (sd) : 35.5 (11.4)      | 53 distinct values |   :
## |    | [numeric]        | min < med < max:             |                    |   : .
## |    |                  | 19 < 33 < 75                 |                    | : : : :
## |    |                  | IQR (CV) : 15 (0.3)          |                    | : : : : :
## |    |                  |                              |                    | : : : : : : : . .
## +----+------------------+------------------------------+--------------------+----------------------
## | 24 | OTHER_INSTALL    | 1. 0                         | 814 (81.4%)        | IIIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 186 (18.6%)        | III
## +----+------------------+------------------------------+--------------------+----------------------
## | 25 | RENT             | 1. 0                         | 821 (82.1%)        | IIIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         | 179 (17.9%)        | III
## +----+------------------+------------------------------+--------------------+----------------------
## | 26 | OWN_RES          | 1. 0                         | 287 (28.7%)        | IIIII
## |    | [factor]         | 2. 1                         | 713 (71.3%)        | IIIIIIIIIIIIII
## +----+------------------+------------------------------+--------------------+----------------------
## | 27 | NUM_CREDITS      | Mean (sd) : 1.4 (0.6)        | 1 : 633 (63.3%)    | IIIIIIIIIIII
## |    | [numeric]        | min < med < max:             | 2 : 333 (33.3%)    | IIIIII
## |    |                  | 1 < 1 < 4                    | 3 :  28 ( 2.8%)    |
## |    |                  | IQR (CV) : 1 (0.4)           | 4 :   6 ( 0.6%)    |
## +----+------------------+------------------------------+--------------------+----------------------
## | 28 | JOB              | 1. 0                         |  22 ( 2.2%)        |
## |    | [factor]         | 2. 1                         | 200 (20.0%)        | IIII
## |    |                  | 3. 2                         | 630 (63.0%)        | IIIIIIIIIIII
## |    |                  | 4. 3                         | 148 (14.8%)        | II
## +----+------------------+------------------------------+--------------------+----------------------
## | 29 | NUM_DEPENDENTS   | Min  : 1                     | 1 : 845 (84.5%)    | IIIIIIIIIIIIIIII
## |    | [numeric]        | Mean : 1.2                   | 2 : 155 (15.5%)    | III
## |    |                  | Max  : 2                     |                    |
## +----+------------------+------------------------------+--------------------+----------------------
## | 30 | TELEPHONE        | 1. 0                         | 596 (59.6%)        | IIIIIIIIIII
## |    | [factor]         | 2. 1                         | 404 (40.4%)        | IIIIIIII
## +----+------------------+------------------------------+--------------------+----------------------
## | 31 | FOREIGN          | 1. 0                         | 963 (96.3%)        | IIIIIIIIIIIIIIIIIII
## |    | [factor]         | 2. 1                         |  37 ( 3.7%)        |
## +----+------------------+------------------------------+--------------------+----------------------
## | 32 | RESPONSE         | 1. 0                         | 300 (30.0%)        | IIIIII
## |    | [factor]         | 2. 1                         | 700 (70.0%)        | IIIIIIIIIIIIII
## +----+------------------+------------------------------+--------------------+----------------------
```

Correlation Analysis : Correlation plot between continuous variables :



There are little correlation between the continuous variables. We can notice that there is a correlation of 62% between the variable **DURATION** and **AMOOUNT**. This makes sense and is known by the bankers that the bigger the amount of credit, the longer the duration of the credit in months will last.



Correlation plot between categorical variables :

It is difficult to look at the correlation since there are a lot of variables on the graph. We can still see higher correlation between **RESPONSE 1**:

- and people that do not check their account (CHK_ACCT_3)
- and people that have a critical historical account (HISTORY_4)
- and the variable *REAL_ESTATE* (REAL_ESTATE)
- and applicant that does not have their own property (PROP_UNKN_NONE_0)
- and applicant that have their own residence (OWN_RES_1)

We can also see some correlation between **RESPONSE 0**:

- and people that have a checking account status < 0 DM (CHK_ACCT_0)
- and people that have an average balance in savings account < 100 DM (SAV_ACCT_0)
- and the variable *REAL_ESTATE* (REAL_ESTATE)

Principal Component Analysis Exploration: It is good to perform a PCA Exploration in order to reduce the dimensions or/and see which variables to select.

We start by selecting the numerical values because the PCA only works on numerical variables.

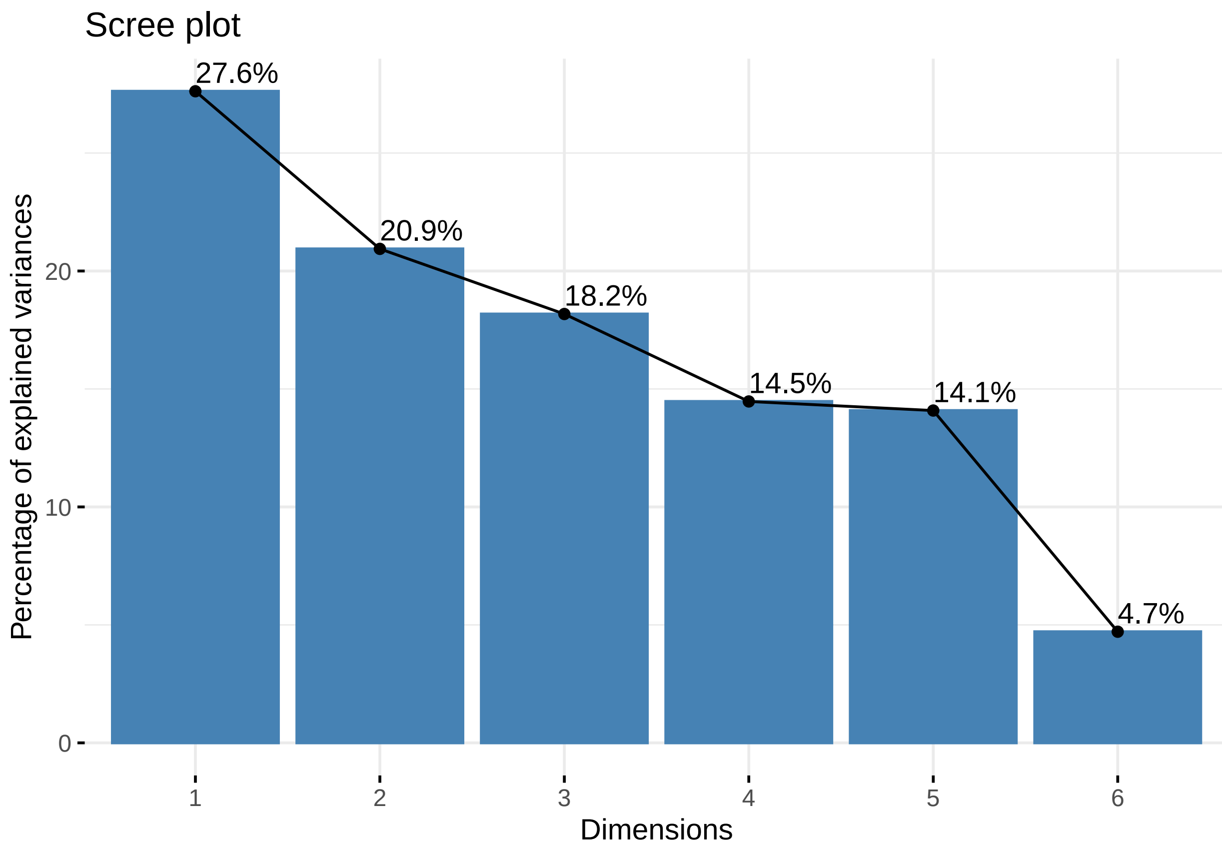
```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6
## Standard deviation     1.2873 1.1208 1.0443 0.9318 0.9193 0.53164
## Proportion of Variance 0.2762 0.2094 0.1818 0.1447 0.1409 0.04711
## Cumulative Proportion  0.2762 0.4856 0.6673 0.8120 0.9529 1.00000
```

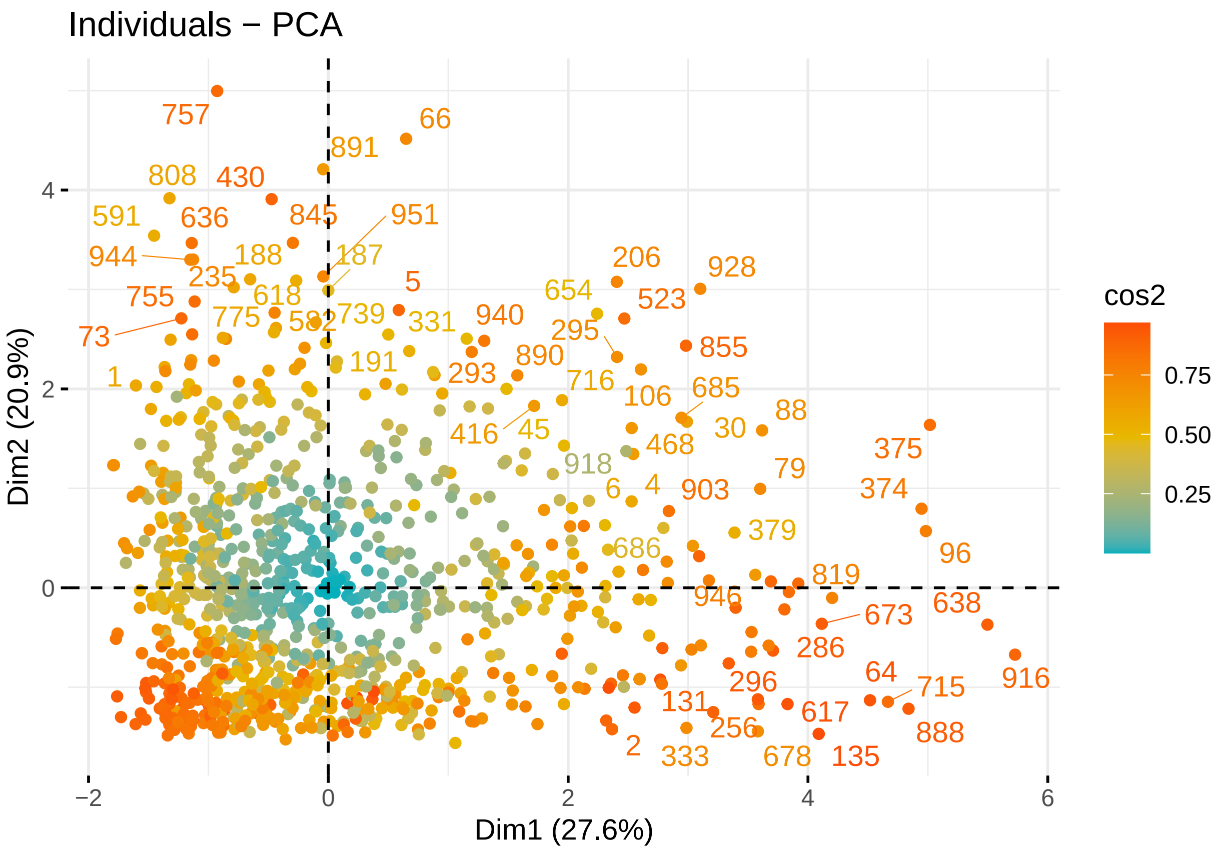
From the PCA summary we can see 4 principal components should be taken into account in order to explain approximately 81% of the variation of the data.

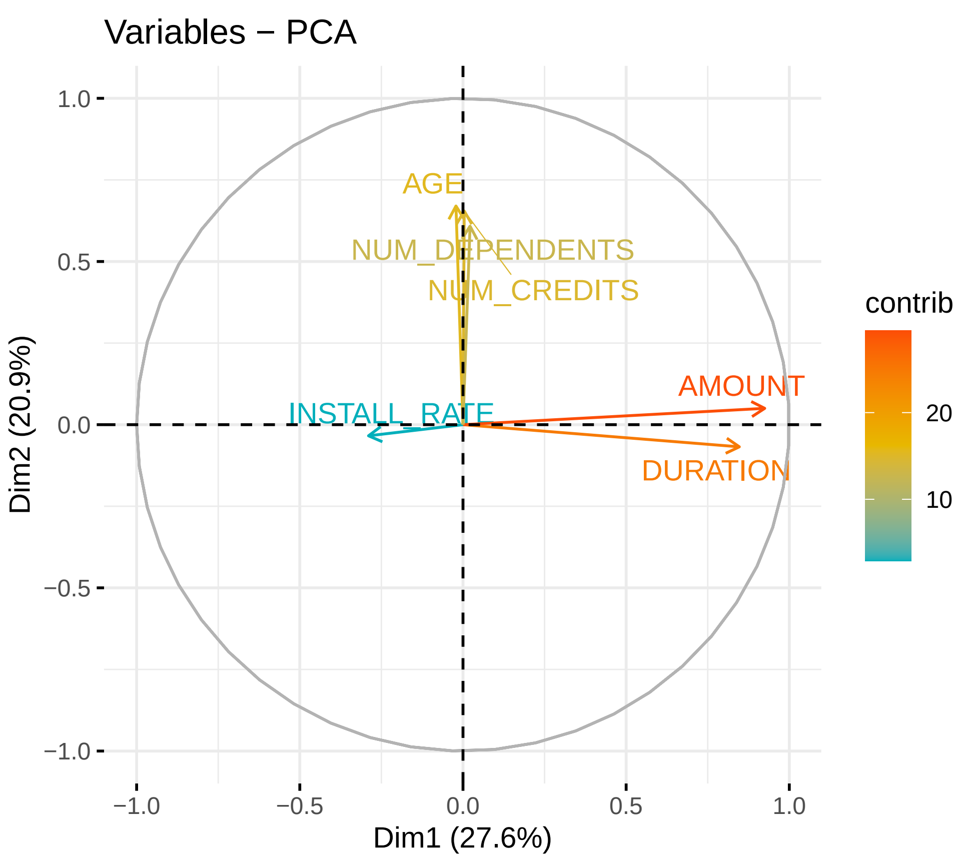
Eigenvalue analysis:

```
##        eigenvalue variance.percent cumulative.variance.percent
## Dim.1  1.6570953        27.618256                    27.61826
## Dim.2  1.2562810        20.938016                    48.55627
## Dim.3  1.0906419        18.177365                    66.73364
## Dim.4  0.8682109        14.470181                    81.20382
## Dim.5  0.8451277        14.085462                    95.28928
## Dim.6  0.2826431         4.710719                   100.00000
```

Then from this eigenvalues table, we know that we need to choose 3 dimensions because the first 3 dimensions are higher than the value 1.




Individuals − PCA
757
66
891
808
430
591
636
845
951
944
188
187
235
5
755
618
775
58
739
331
940
73
191
890
293
1
716
206
928
654
523
295
855
106
685
88
416
45
30
918
468
375
6
4
903
79
374
379
96
686
819
946
638
673
286
916
64
296
715
131
617
256
888
2
333
678
135
cos2
0.75
0.50
0.25
Dim2 (20.9%)
Dim1 (27.6%)
4
2
0
−2
0
2
4
6

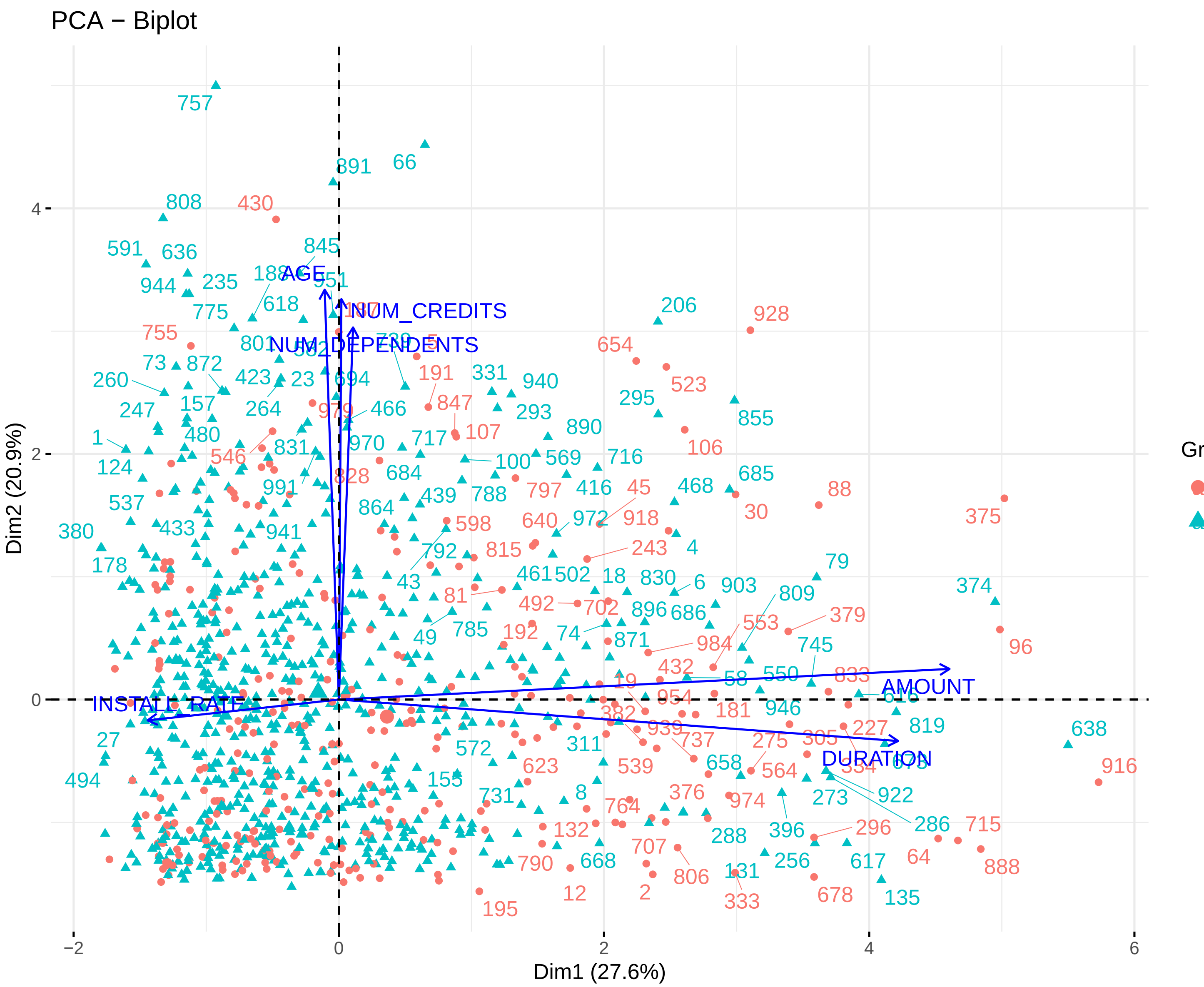

Variables − PCA
AGE
NUM_DEPENDENTS
NUM_CREDITS
AMOUNT
INSTALL_RATE
DURATION
contrib
20
10
Dim2 (20.9%)
Dim1 (27.6%)
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

From this circle of correlations, we see that :

- The first principal component PC1 is strongly positively correlated with the variables **AMOUNT** and **DURATION**. So the larger PC1, the larger these features. It is also a little bit negatively correlated with **INSTALL_RATE**.

- The second principal component PC2 is strongly positively correlated with **AGE**, **NUM_DEPENDENTS** and **NUM_CREDITS**.

From this biplot, we can see some characteristics of the observations.



Here, we can see the distribution of the response variables (0-1) according to the reduced dimension. What we can observe, is that the Bad credits: 0, look a little bit more positively correlated ot dimension 1. Therefore, more correlated to Amount and Duration. Compared to Good Credits, it looks positively correlated to dimesion 2; **AGE**, **NUM_CREDITS**, **NUM_DEPENDENTS**.

After having cleaned the dataset and looked at all the different features, we created a new dataset that contains our modifications in order to use it for our analysis.

Methodology

In this section we will talk about the methodology that has been used and the different models analysis that has been conducted.

Traning set and Test set

First of all we started by splitting our dataset into 2 sets: **training set** (`German_credit.tr`) and **test set** (`German_credit.te`). We do not forget to take the first variable **OBS.** out as it represents the index number for each observation. These two sets will allow us to train some models on the **training set** and then test the accuracy of the model fit on the **test set**.

Balancing the dataset

Then, we applied the balancing data technique in order to improve the predictions of **Good Credit** and **Bad Credit**, since we have more observations on the **Good Credit**.

The table below reveals the unbalanced problem.

Indeed, we observe that the "Good Credit" (1) response appears **527** times in the training set and "Bad Credit" (0) **223**, two times less. Since there are many more "Good Credit" than "Bad Credit", any model favors the prediction of the "Good Credit". It results a good accuracy but the specificity is low, as well as the balanced accuracy.

Sub-sampling Balancing using sub-sampling consists of taking all the cases in the smallest class (here "Bad Credit") and extract at random the same amount of cases in the largest category (here "Good").

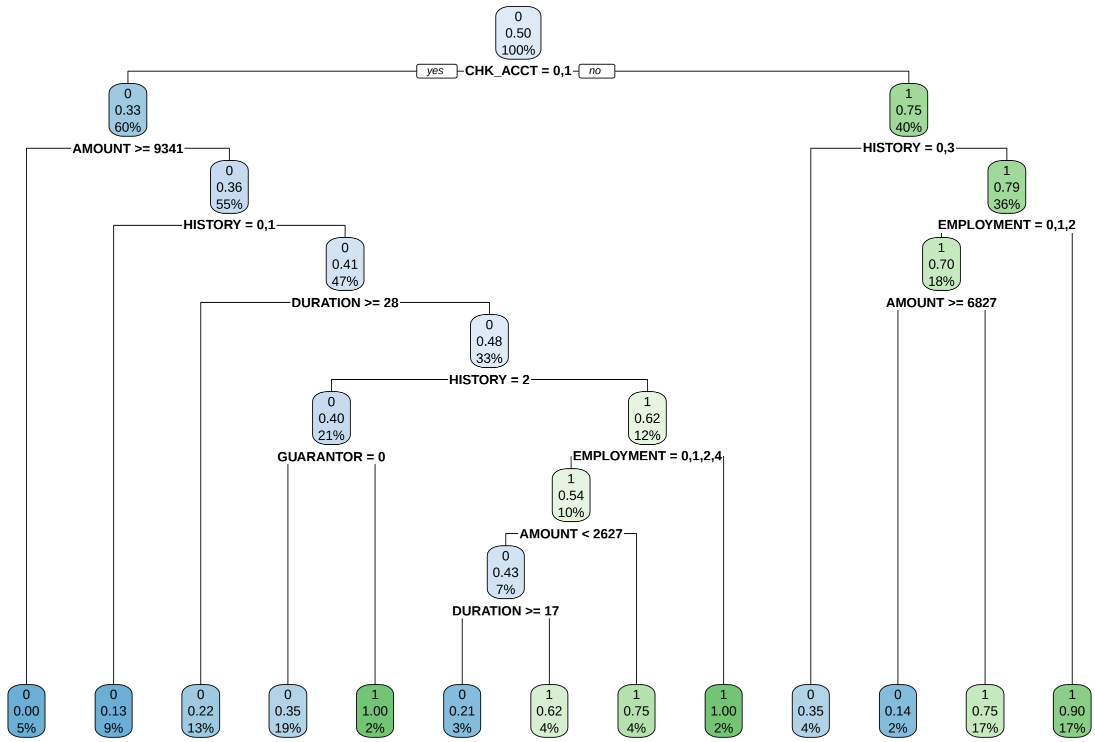
The **training set** is now balanced, we have 223 observations for both "Good Credit" (1) and "Bad Credit" (0). The new balanced training set is called `German_Credit.tr.subs`.

Models Fitting

Once we had our training set and test set, we could fit some models and compare them with together to choose the best model.

1. Classification Tree (Decision Tree) We first started with a decision tree and more specifically we chose the classification tree as we want to classify the applicants. The model was build on our previously balanced training set `German_Credit.tr.subs`. We used the R function `rpart`.

We obtained the following large tree.



We could see that among the 31 explanatory variables, this model uses 6 of them: **CHK_ACCT**, **AMOUNT**, **HISTORY**, **DURATION**, **GUARANTOR** and **EMPLOYMENT**.

```
##           Reference
## Prediction   0   1
##          0  58  70
##          1  19 103
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.7532468            0.5953757            0.4531250
##       Neg Pred Value            Precision               Recall
##            0.8442623            0.4531250            0.7532468
##                   F1           Prevalence       Detection Rate
##            0.5658537            0.3080000            0.2320000
## Detection Prevalence    Balanced Accuracy
##            0.5120000            0.6743112
```

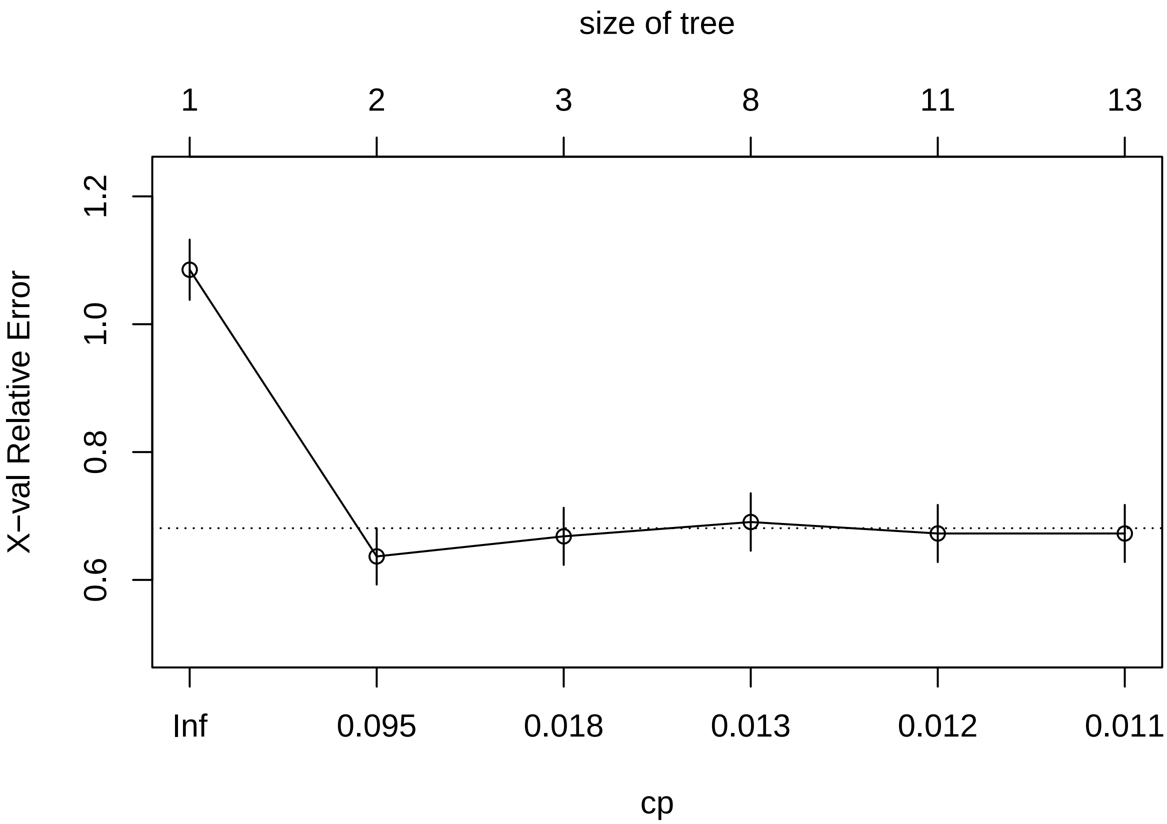
We first have an insight on how well it predict the test set (`German_credit.te`). We recall that 0 means a "Bad Credit" risk and 1 means a "Good Credit" risk. It seems that it has difficulty to predict the "Bad Credit" risk applicants.

As the tree is quite big and we want to know if we can prune it. To do so, we decided to use the `printcp` and `plotcp` commands and choose the best **cp** (complexity parameter) value to prune our tree.

Pruning the tree

```
##
## Classification tree:
## rpart(formula = RESPONSE ~ ., data = German_Credit.tr.subs, method = "class")
##
## Variables actually used in tree construction:
## [1] AMOUNT     CHK_ACCT   DURATION   EMPLOYMENT GUARANTOR  HISTORY
##
```

```
## Root node error: 223/446 = 0.5
##
## n= 446
##
##         CP nsplit rel error  xerror     xstd
## 1 0.399103      0   1.00000 1.08520 0.047179
## 2 0.022422      1   0.60090 0.63677 0.044117
## 3 0.014574      2   0.57848 0.66816 0.044668
## 4 0.011958      7   0.48430 0.69058 0.045028
## 5 0.011211     10   0.44843 0.67265 0.044742
## 6 0.010000     12   0.42601 0.67265 0.044742
```

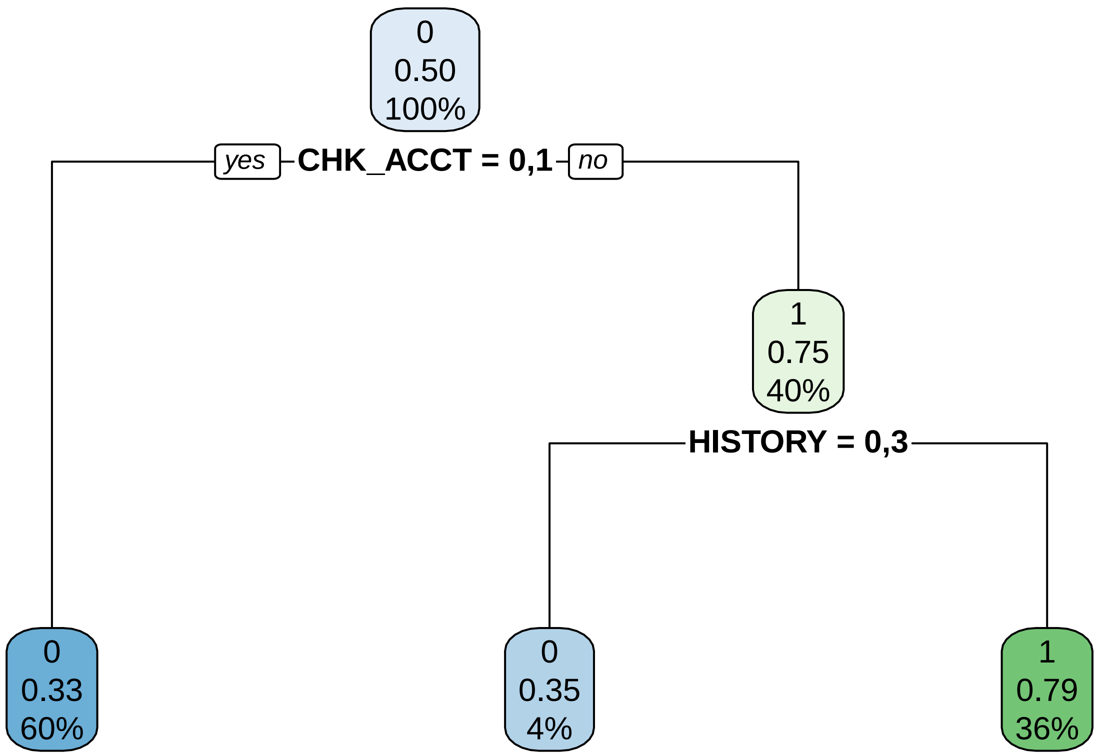


From the list of **cp** (complexity parameter), we would choose the line that has the lowest cross validation error. This can be seen on the column **xerror**. So the best cp would be 0.022422 with one split.

From the graph, we can identify that, according to the 1-SE rule, the tree with 2 and 3 are equivalent. The tree with 3 nodes should be preferred. It appears below the dotted-line.

The value of cp can be chosen arbitrarily between 0.018 and 0.095. So we decided to go with the suggested cp of 0.022 from the summary.

With these value, we obtain a very small tree.



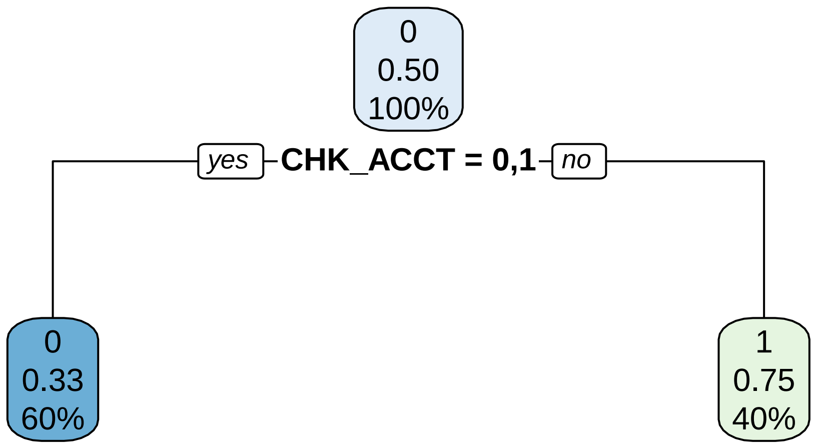
This pruned decision tree with a cp of 0.022 uses the variables **CHK_ACCT** and **HISTORY**.

Using this pruned tree, we can computed the prediction and build a confusion matrix to see the performance of the model.

```
##           Reference
## Prediction  0  1
##          0 63 95
##          1 14 78
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.8181818            0.4508671            0.3987342
##       Neg Pred Value            Precision               Recall
##            0.8478261            0.3987342            0.8181818
##                   F1           Prevalence       Detection Rate
##            0.5361702            0.3080000            0.2520000
## Detection Prevalence    Balanced Accuracy
##            0.6320000            0.6345244
```

We also decided to look at what would an automatically pruned using 1-SE rule would give us and whether or not it is better than the pruned tree we made by looking at the cp.



Here, only the variable **CHK_ACCT** is used. As we prune the tree more information are lost.

```
##             Reference
## Prediction   0  1
##            0 61 88
##            1 16 85
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.7922078            0.4913295            0.4093960
##       Neg Pred Value            Precision               Recall
##            0.8415842            0.4093960            0.7922078
##                   F1           Prevalence       Detection Rate
##            0.5398230            0.3080000            0.2440000
## Detection Prevalence    Balanced Accuracy
##            0.5960000            0.6417686
```

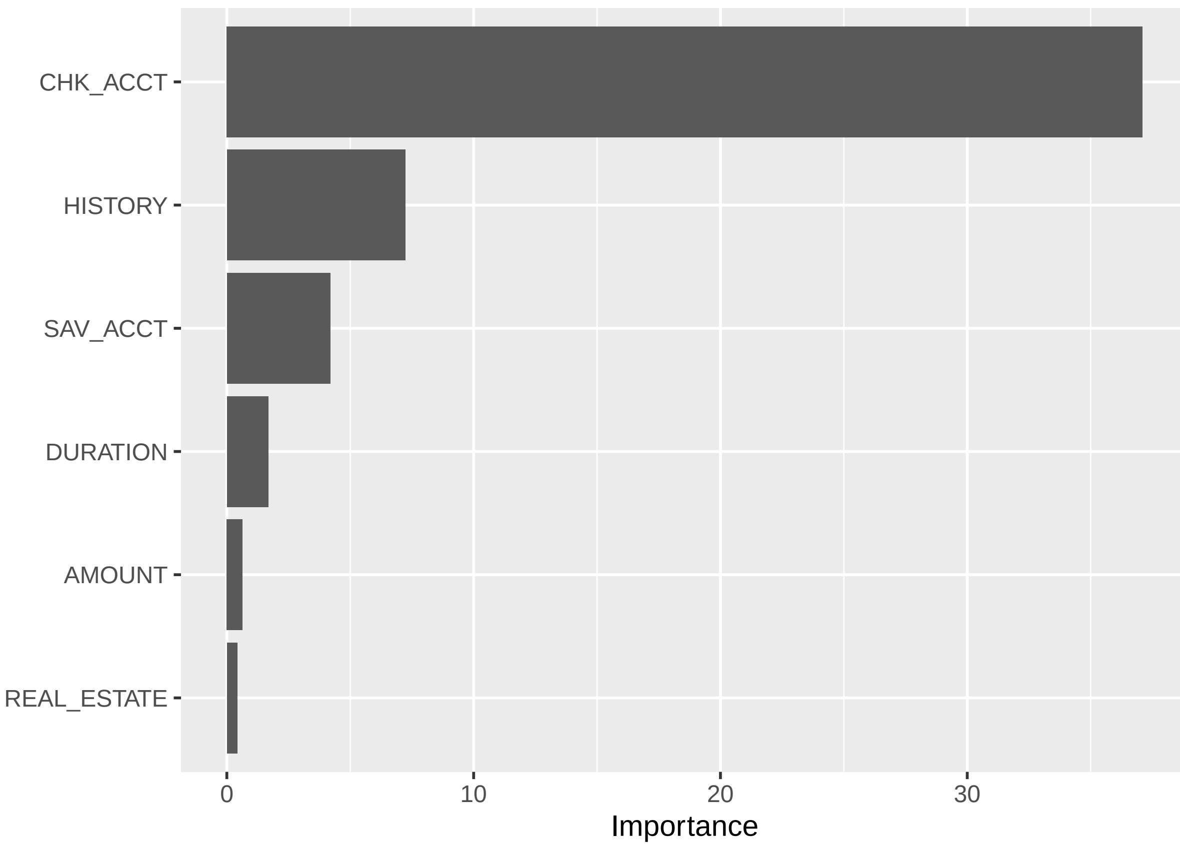
```
varImp(german.ct.prune)
```

Variable importance of the classification tree

```
##                  Overall
## AMOUNT         17.947179
## CHK_ACCT       37.098755
## DURATION       11.073258
## EMPLOYMENT      4.645266
## HISTORY        18.840050
## OTHER_INSTALL   3.216630
## RETRAINING      3.509915
## SAV_ACCT        9.538067
## NEW_CAR         0.000000
## USED_CAR        0.000000
## FURNITURE       0.000000
## RADIO.TV        0.000000
## EDUCATION       0.000000
## INSTALL_RATE    0.000000
## MALE_DIV        0.000000
## MALE_SINGLE     0.000000
## MALE_MAR_or_WID 0.000000
## CO.APPLICANT    0.000000
## GUARANTOR       0.000000
```

```
## PRESENT_RESIDENT  0.000000
## REAL_ESTATE       0.000000
## PROP_UNKN_NONE    0.000000
## AGE               0.000000
## RENT              0.000000
## OWN_RES           0.000000
## NUM_CREDITS       0.000000
## JOB               0.000000
## NUM_DEPENDENTS    0.000000
## TELEPHONE         0.000000
## FOREIGN           0.000000
```

```
vip(german.ct.prune)
```



From this plot, we see that the variables that influences the most are **CHK_ACCT**, **HISTORY**, **SAV_ACCT**, **DURATION**, **AMOUNT** and **REAL_ESTATE**. They are not exactly the same as the one we saw above.

The variable **CHK_ACCT** and **HISTORY** were noticed though.

2. Logistic Regression The next model we performed is a logistic regression.

```
##
## Call:
## glm(formula = RESPONSE ~ ., family = binomial, data = German_Credit.tr.subs)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.34578  -0.68043   0.00049   0.65178   2.74937
```

```
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.1911402  1.7958756   0.663 0.507161
## CHK_ACCT1           0.5692882  0.3363406   1.693 0.090533 .
## CHK_ACCT2           0.8404451  0.5339512   1.574 0.115485
## CHK_ACCT3           2.4337691  0.3770606   6.455 1.09e-10 ***
## DURATION           -0.0123731  0.0142153  -0.870 0.384078
## HISTORY1           -1.0734853  0.8514386  -1.261 0.207384
## HISTORY2            0.0865599  0.6747882   0.128 0.897930
## HISTORY3           -0.0598560  0.7410028  -0.081 0.935619
## HISTORY4            1.1072483  0.6576414   1.684 0.092246 .
## NEW_CAR1           -0.4538649  0.5853211  -0.775 0.438096
## USED_CAR1           1.6322817  0.7540134   2.165 0.030404 *
## FURNITURE1          0.0509645  0.6182782   0.082 0.934305
## RADIO.TV1           0.5261147  0.5896893   0.892 0.372291
## EDUCATION1          0.5441469  0.7499724   0.726 0.468111
## RETRAINING1        -0.4293160  0.6787931  -0.632 0.527080
## AMOUNT             -0.0002155  0.0000739  -2.916 0.003550 **
## SAV_ACCT1           0.6181742  0.4475399   1.381 0.167195
## SAV_ACCT2          -0.2531524  0.5541205  -0.457 0.647776
## SAV_ACCT3           0.7292579  0.6813687   1.070 0.284492
## SAV_ACCT4           1.4221687  0.4243610   3.351 0.000804 ***
## EMPLOYMENT1         0.7574673  0.7956778   0.952 0.341108
## EMPLOYMENT2         1.4785839  0.7640267   1.935 0.052959 .
## EMPLOYMENT3         1.9691166  0.7947873   2.478 0.013229 *
## EMPLOYMENT4         1.8560330  0.7511387   2.471 0.013475 *
## INSTALL_RATE       -0.3367533  0.1411404  -2.386 0.017035 *
## MALE_DIV1          -0.5653453  0.5705857  -0.991 0.321775
## MALE_SINGLE1        0.1618525  0.3327207   0.486 0.626647
## MALE_MAR_or_WID1   -0.5551862  0.5312986  -1.045 0.296041
## CO.APPLICANT1      -0.6994379  0.6920599  -1.011 0.312179
## GUARANTOR1          1.7126786  0.6556150   2.612 0.008993 **
## PRESENT_RESIDENT2 -1.1195205  0.4773294  -2.345 0.019008 *
## PRESENT_RESIDENT3 -0.2590309  0.5313455  -0.487 0.625904
## PRESENT_RESIDENT4 -0.9082582  0.4793144  -1.895 0.058104 .
## REAL_ESTATE1       -0.0137202  0.3384983  -0.041 0.967669
## PROP_UNKN_NONE1    -1.4578770  0.6505748  -2.241 0.025032 *
## AGE                 0.0167050  0.0141041   1.184 0.236255
## OTHER_INSTALL1     -0.6758552  0.3404321  -1.985 0.047113 *
## RENT1              -1.2066453  0.8244600  -1.464 0.143315
## OWN_RES1           -0.4707135  0.7665544  -0.614 0.539173
## NUM_CREDITS        -0.3634820  0.3011721  -1.207 0.227474
## JOB1               -0.7402802  1.1619781  -0.637 0.524069
## JOB2               -1.2142377  1.1317833  -1.073 0.283337
## JOB3               -1.4358446  1.1604352  -1.237 0.215964
## NUM_DEPENDENTS      0.1270172  0.3832474   0.331 0.740325
## TELEPHONE1          0.6259633  0.3143236   1.991 0.046430 *
## FOREIGN1            1.2496315  0.8543880   1.463 0.143576
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 618.29  on 445  degrees of freedom
## Residual deviance: 390.97  on 400  degrees of freedom
## AIC: 482.97
##
## Number of Fisher Scoring iterations: 5
```

We see that a lot of variables are not statistically significant for the model so we can think of a model reduction.

Before doing a reduction of the model, we fitted the model and predicted on the test set.

```
##           Reference
## Prediction   0   1
##          0  49  46
##          1  28 127
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.6363636            0.7341040            0.5157895
##       Neg Pred Value            Precision               Recall
##            0.8193548            0.5157895            0.6363636
##                   F1           Prevalence       Detection Rate
##            0.5697674            0.3080000            0.1960000
## Detection Prevalence    Balanced Accuracy
##            0.3800000            0.6852338
```

Variable selection and interpretation with step method (AIC criteria) In order to reduce the logistic regression we used a stepwise variable selection. This has been done with the command `step`.

The final reduced model is as follow.

```
summary(mod.logreg.sel)
```

```
##
## Call:
## glm(formula = RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR +
##     RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE +
##     GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE + AGE + OTHER_INSTALL +
##     RENT + TELEPHONE + FOREIGN, family = binomial, data = German_Credit.tr.subs)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.39343  -0.68768  -0.02628   0.71315   2.60726
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -0.6339113  1.1570626  -0.548 0.583786
## CHK_ACCT1          0.5970566  0.3291383   1.814 0.069678 .
## CHK_ACCT2          1.1123874  0.5042812   2.206 0.027392 *
## CHK_ACCT3          2.4109175  0.3597629   6.701 2.06e-11 ***
## HISTORY1          -0.6393459  0.8007632  -0.798 0.424626
## HISTORY2           0.3153810  0.6295178   0.501 0.616379
## HISTORY3           0.0245812  0.7411154   0.033 0.973541
## HISTORY4           1.0638624  0.6476870   1.643 0.100475
## NEW_CAR1          -0.7159178  0.3141611  -2.279 0.022678 *
## USED_CAR1          1.3489217  0.5579072   2.418 0.015614 *
## RETRAINING1       -0.8489518  0.4619050  -1.838 0.066072 .
## AMOUNT            -0.0002631  0.0000595  -4.421 9.81e-06 ***
```

```
## SAV_ACCT1          0.5675624  0.4173990   1.360 0.173906
## SAV_ACCT2         -0.0693577  0.5399345  -0.128 0.897788
## SAV_ACCT3          0.5603771  0.6437202   0.871 0.384011
## SAV_ACCT4          1.3592948  0.4069584   3.340 0.000837 ***
## EMPLOYMENT1        0.5542570  0.6967423   0.795 0.426324
## EMPLOYMENT2        1.2338686  0.6524020   1.891 0.058588 .
## EMPLOYMENT3        1.7999683  0.6887566   2.613 0.008966 **
## EMPLOYMENT4        1.5521376  0.6518729   2.381 0.017264 *
## INSTALL_RATE      -0.3278020  0.1249721  -2.623 0.008716 **
## GUARANTOR1         1.6927223  0.6068573   2.789 0.005282 **
## PRESENT_RESIDENT2 -1.1117822  0.4641005  -2.396 0.016595 *
## PRESENT_RESIDENT3 -0.3408387  0.5041109  -0.676 0.498966
## PRESENT_RESIDENT4 -0.7613632  0.4531619  -1.680 0.092935 .
## PROP_UNKN_NONE1   -1.0532655  0.3848454  -2.737 0.006203 **
## AGE                0.0181856  0.0128738   1.413 0.157769
## OTHER_INSTALL1    -0.6281982  0.3256821  -1.929 0.053747 .
## RENT1             -0.8736712  0.3412119  -2.560 0.010452 *
## TELEPHONE1         0.5251823  0.2863172   1.834 0.066614 .
## FOREIGN1           1.2896516  0.8049248   1.602 0.109111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 618.29  on 445  degrees of freedom
## Residual deviance: 403.37  on 415  degrees of freedom
## AIC: 465.37
##
## Number of Fisher Scoring iterations: 5
```

The variables that have been removed are : **FURNITURE**, **RADIO.TV**, **EDUCATION**, **RETRAINING**, **MALE_DIV**, **MALE_SINGLE**, **MALE_MAR_or_WID**, **CO.APPLICANT**, **REAL_ESTATE**, **OWN_RES**, **NUM_CREDITS**, **JOB** and **NUM_DEPENDENTS**

In the end, we get the most significant model:

$$RESPONSE = -0.6339113 + 0.5970566 * CHK_{ACCT1} + 1.1123874 * CHK_{ACCT2} + 2.4109175 * CHK_{ACCT3} - 0.6393459 * HIST$$

$$p = (e^{RESPONSE})/(1 + e^{RESPONSE})$$

It means that :

- The predicted probability of being a good applicant for **CHCK_ACCT3** is higher than for **CHK_ACCT0** (and also higher than for **CHK_ACCT1** and **CHK_ACCT2**).
- The predicted probability of being a good applicant for **HISTORY1** is lower than for **HISTORY0**.
- The predicted probability of being a good applicant for **HISTORY4** is higher than for **HISTORY0** (and also higher than for **HISTORY2** and **HISTORY3**).
- The predicted probability of being a good applicant for **NEW_CAR1** is lower than for **NEW_CAR0**.
- The predicted probability of being a good applicant for **USED_CAR1** is higher than for **USED_CAR0**.
- The predicted probability of being a good applicant for **RETRAINING1** is lower than for **RETRAINING0**.
- **AMOUNT** is negatively associated with **RESPONSE**.

- The predicted probability of being a good applicant for **SAV_ACCT4** is higher than for **SAV_ACCT0** (and also higher than for **SAV_ACCT1** and **SAV_ACCT3**).
- The predicted probability of being a good applicant for **SAV_ACCT2** lower than for **SAV_ACCT0**.
- The predicted probability of being a good applicant for **EMPLOYMENT3** is higher than for **Employment0** (and also higher than for **EMPLOYMENT1**, **EMPLOYMENT2** and **EMPLOYMENT4**).
- **INSTALL_RATE** is negatively associated with **RESPONSE**.
- The predicted probability of being a good applicant for **GUARANTOR1** is higher than for **GUARANTOR0**.
- The predicted probability of being a good applicant for **PRESENT_RESIDENT2** is lower than for **PRESENT_RESIDENT0** (and also lower than **PRESENT_RESIDENT3** and **PRESENT_RESIDENT4**).
- The predicted probability of being a good applicant for **PROP_UNKN_NONE1** is lower than for **PROP_UNKN_NONE0**.
- **AGE** is positively associated with **RESPONSE**.
- The predicted probability of being a good applicant for **OTHER_INSTALL1** is lower than for **OTHER_INSTALL0**.
- The predicted probability of being a good applicant for **RENT1** is lower than for **RENT0**.
- The predicted probability of being a good applicant for **TELEPHONE1** is higher than for **TELEPHONE0**.
- The predicted probability of being a good applicant for **FOREIGN1** is higher than for **FOREIGN0**.

```
##           Reference
## Prediction   0   1
##          0  50  52
##          1  27 121
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.6493506            0.6994220            0.4901961
##       Neg Pred Value            Precision               Recall
##            0.8175676            0.4901961            0.6493506
##                   F1           Prevalence       Detection Rate
##            0.5586592            0.3080000            0.2000000
## Detection Prevalence    Balanced Accuracy
##            0.4080000            0.6743863
```

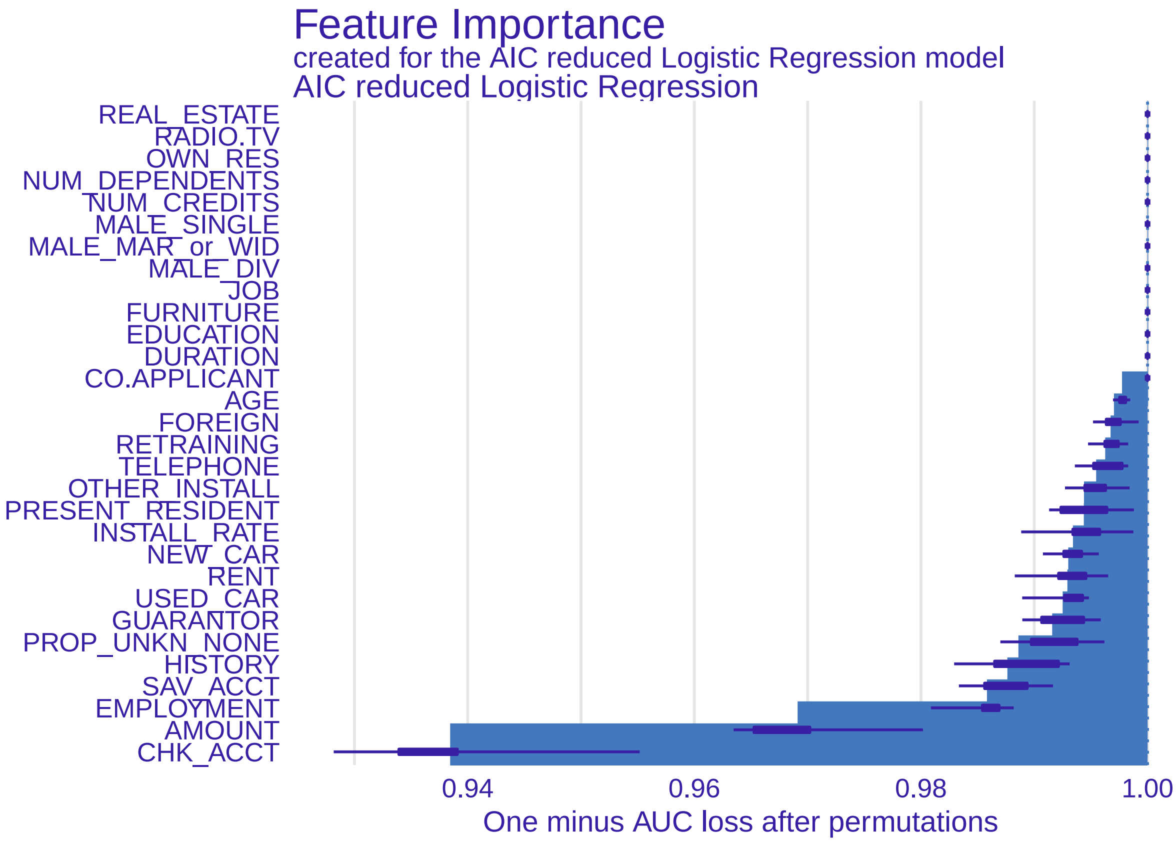
```
x_train <- select(German_Credit.tr.subs, -RESPONSE)
y_train <- pull(German_Credit.tr.subs, RESPONSE)

explainer_logreg <- explain(model = mod.logreg.sel,
                               data = x_train,
                               y = as.numeric(y_train),
                               label = "AIC reduced Logistic Regression")
```

Variable importance for logistic regression

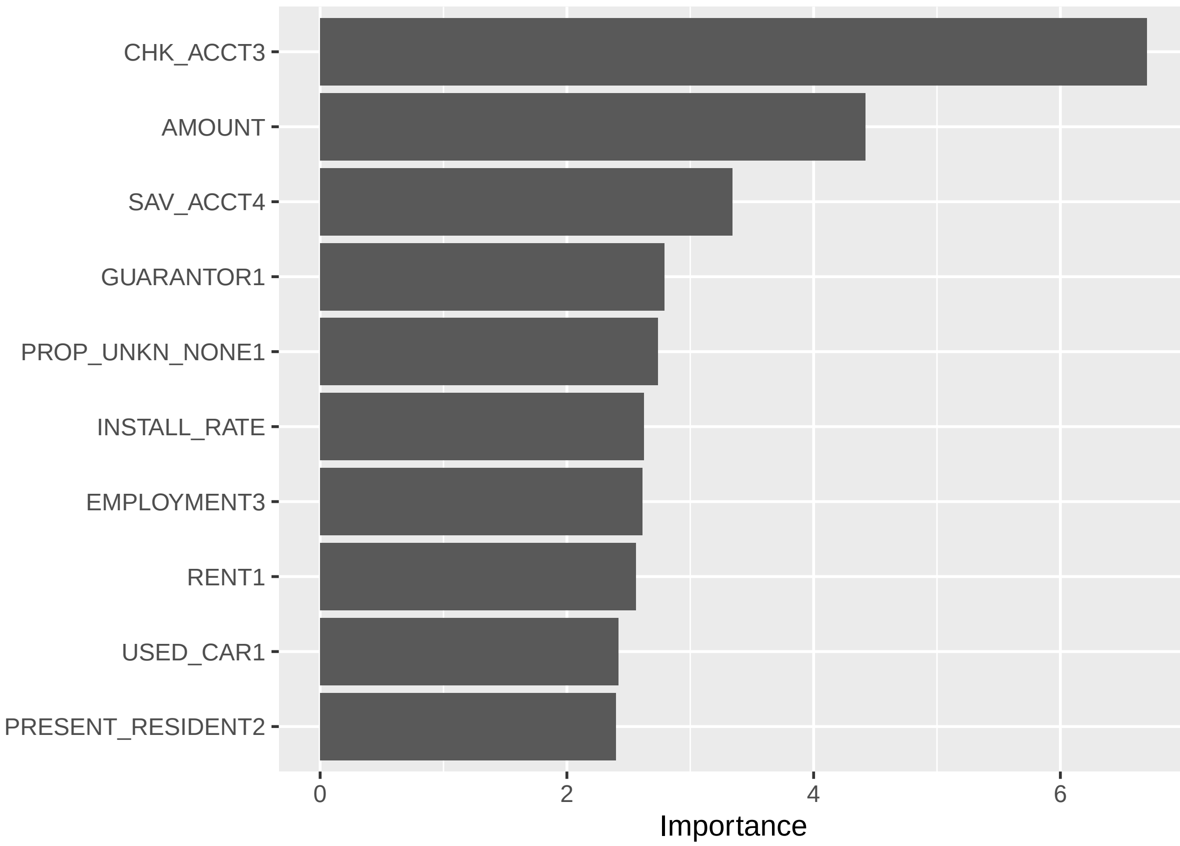
```
## Preparation of a new explainer is initiated
##   -> model label       :  AIC reduced Logistic Regression
##   -> data              :  446  rows  30  cols
##   -> target variable   :  446  values
##   -> predict function  :  yhat.glm  will be used (  default  )
##   -> predicted values  :  No value for predict function target column. (  default  )
##   -> model_info        :  package stats , ver. 4.1.3 , task classification (  default  )
##   -> predicted values  :  numerical, min =  0.007541882 , mean =  0.5 , max =  0.9975191
##   -> residual function :  difference between y and yhat (  default  )
##   -> residuals         :  numerical, min =  0.05702613 , mean =  1 , max =  1.96659
```

```
##   A new explainer has been created!
```

```
importance_logreg  <- calculate_importance(explainer_logreg)
plot(importance_logreg)
```




```
vip(mod.logreg.sel)
```



Listed above are the most important variables for the logarithmic regression we reduced.

3. K-Nearest Neighbor To perform a k-nearest neighbor method, we do not need to balance the data so we will use the unbalanced training set.

We first try to model it using a 2-NN (with Euclidean distance). Note that the model is fitting on the training set and the predictions are computed on the test set.

```
##           Reference
## Prediction   0   1
##          0  21  45
##          1  56 128
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.2727273            0.7398844            0.3181818
##       Neg Pred Value            Precision               Recall
##            0.6956522            0.3181818            0.2727273
##                   F1           Prevalence       Detection Rate
##            0.2937063            0.3080000            0.0840000
## Detection Prevalence    Balanced Accuracy
##            0.2640000            0.5063058
```

The table is read as follow :

- We predicted 21 Bad credits and there were indeed 21 observed Bad credits. But the prediction misjudges 45 good credits by predicting bad credits.
- We predicted 128 Good credits as it was in fact a Good credits but 56 where predicted as Good while it was in fact Bad.

The prediction is not perfect. We need to try to improve the prediction by changing K at that point. Therefore, we use K=3.

```
##           Reference
## Prediction   0   1
##          0  14  28
##          1  63 145
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.1818182            0.8381503            0.3333333
##       Neg Pred Value            Precision               Recall
##            0.6971154            0.3333333            0.1818182
##                   F1           Prevalence       Detection Rate
##            0.2352941            0.3080000            0.0560000
## Detection Prevalence    Balanced Accuracy
##            0.1680000            0.5099842
```

The table is read as follow :

- We predicted 14 Bad credits and they were indeed observed Bad credits. But the prediction misjudges 28 good credits by predicting bad credits.
- We predicted 145 Good credits as it was in fact a Good credits but 6 where predicted as Good while it was in fact Bad.

4. Linear Support Vector Machine The next model is the linear Support Vector Machine.

```
##
## Call:
## svm(formula = RESPONSE ~ ., data = German_Credit.tr.subs, kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  246
```

```
##           Reference
## Prediction   0   1
##          0  50  50
##          1  27 123
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.6493506            0.7109827            0.5000000
##       Neg Pred Value            Precision               Recall
##            0.8200000            0.5000000            0.6493506
##                   F1           Prevalence       Detection Rate
##            0.5649718            0.3080000            0.2000000
## Detection Prevalence    Balanced Accuracy
##            0.4000000            0.6801667
```

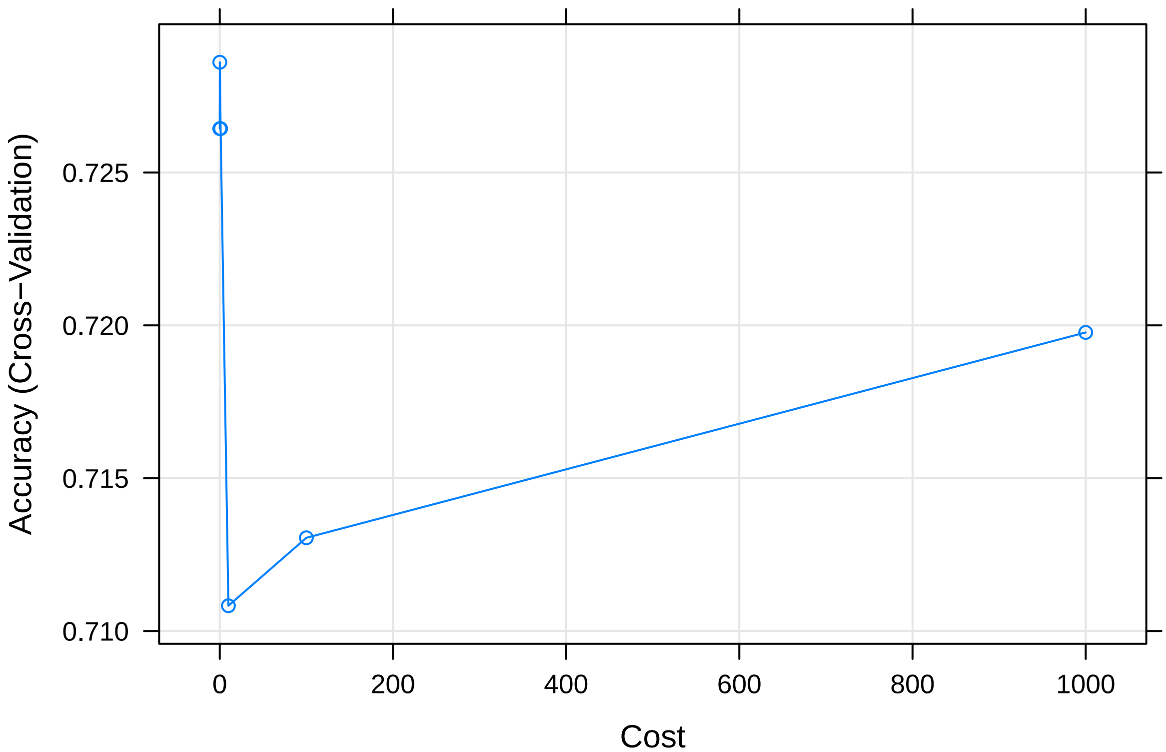
Tunning the hyperparameters of Linear SVM We want to select the good hyperparameters for our linear SVM.

```
## Support Vector Machines with Linear Kernel
##
## 446 samples
```

```
##  30 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 402, 400, 402, 401, 402, 402, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7264361  0.4530209
##
## Tuning parameter 'C' was held constant at a value of 1
```

We see that we have a good accuracy (0.72).

```
## Support Vector Machines with Linear Kernel
##
## 446 samples
##  30 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 402, 400, 402, 401, 402, 402, ...
## Resampling results across tuning parameters:
##
##   C      Accuracy   Kappa
##   1e-02  0.7264339  0.4532044
##   1e-01  0.7286056  0.4575791
##   1e+00  0.7264361  0.4530209
##   1e+01  0.7108278  0.4216992
##   1e+02  0.7130501  0.4261940
##   1e+03  0.7197672  0.4397558
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.1.
```




```
##            Reference
## Prediction   0   1
##          0  49  49
##          1  28 124
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.6363636            0.7167630            0.5000000
##       Neg Pred Value            Precision               Recall
##            0.8157895            0.5000000            0.6363636
##                   F1           Prevalence       Detection Rate
##            0.5600000            0.3080000            0.1960000
## Detection Prevalence    Balanced Accuracy
##            0.3920000            0.6765633
```

5. Radial Basis Support Vector Machine We try now with a radial basis kernel (the default).

```
German_credit.rbsvm <- svm(RESPONSE ~ ., data=German_Credit.tr.subs, kernel="radial")
German_credit.rbsvm
```

```
##
## Call:
## svm(formula = RESPONSE ~ ., data = German_Credit.tr.subs, kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  334
```

```r
German_credit.rbsvm.pred <- predict(German_credit.rbsvm,
                                    newdata = German_credit.te)

cm_rbsvm <- confusionMatrix(data=German_credit.rbsvm.pred,
                            reference = German_credit.te$RESPONSE )

cm_rbsvm$table
```

```
##           Reference
## Prediction   0   1
##          0  54  52
##          1  23 121
```

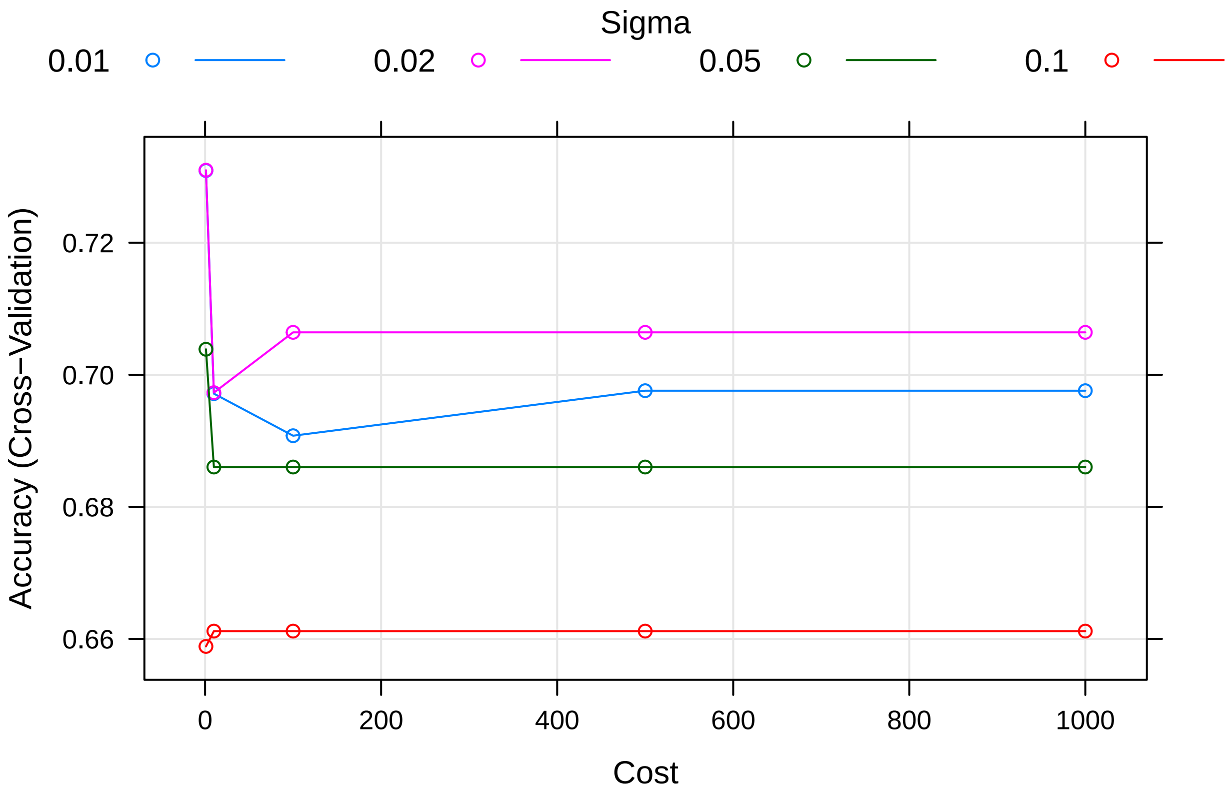
```r
cm_rbsvm$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.7012987            0.6994220            0.5094340
##       Neg Pred Value            Precision               Recall
##            0.8402778            0.5094340            0.7012987
##                   F1           Prevalence       Detection Rate
##            0.5901639            0.3080000            0.2160000
## Detection Prevalence    Balanced Accuracy
##            0.4240000            0.7003603
```

Tunning the hyperparameters of Radial basis SVM

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 446 samples
##  30 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 402, 400, 402, 401, 402, 402, ...
## Resampling results across tuning parameters:
##
##   sigma  C     Accuracy   Kappa
##   0.01      1  0.7309289  0.4618150
##   0.01     10  0.6971476  0.3942416
##   0.01    100  0.6907708  0.3814209
##   0.01    500  0.6975889  0.3950572
##   0.01   1000  0.6975889  0.3950572
##   0.02      1  0.7309816  0.4620420
##   0.02     10  0.6972925  0.3946754
##   0.02    100  0.7064273  0.4127482
##   0.02    500  0.7064273  0.4127482
##   0.02   1000  0.7064273  0.4127482
##   0.05      1  0.7038647  0.4085756
##   0.05     10  0.6860299  0.3726705
##   0.05    100  0.6860299  0.3726705
##   0.05    500  0.6860299  0.3726705
##   0.05   1000  0.6860299  0.3726705
##   0.10      1  0.6588603  0.3190546
##   0.10     10  0.6611792  0.3234506
```

```
##    0.10     100  0.6611792  0.3234506
##    0.10     500  0.6611792  0.3234506
##    0.10    1000  0.6611792  0.3234506
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.02 and C = 1.
```




```
##   sigma C
## 6  0.02 1
```

```
German_credit.rbsvm.tuned <- svm(RESPONSE ~ .,data = German_Credit.tr.subs,
                                  kernel = "radial",
                                  gamma = svm_Radial_Grid$bestTune$sigma,
                                  cost = svm_Radial_Grid$bestTune$C)

German_credit.rbsvm.tuned.pred <- predict(German_credit.rbsvm.tuned,
                                           newdata = German_credit.te)

cm_rbsvm_tuned <- confusionMatrix(data=German_credit.rbsvm.tuned.pred,
                                   reference = German_credit.te$RESPONSE)

cm_rbsvm_tuned$table
```

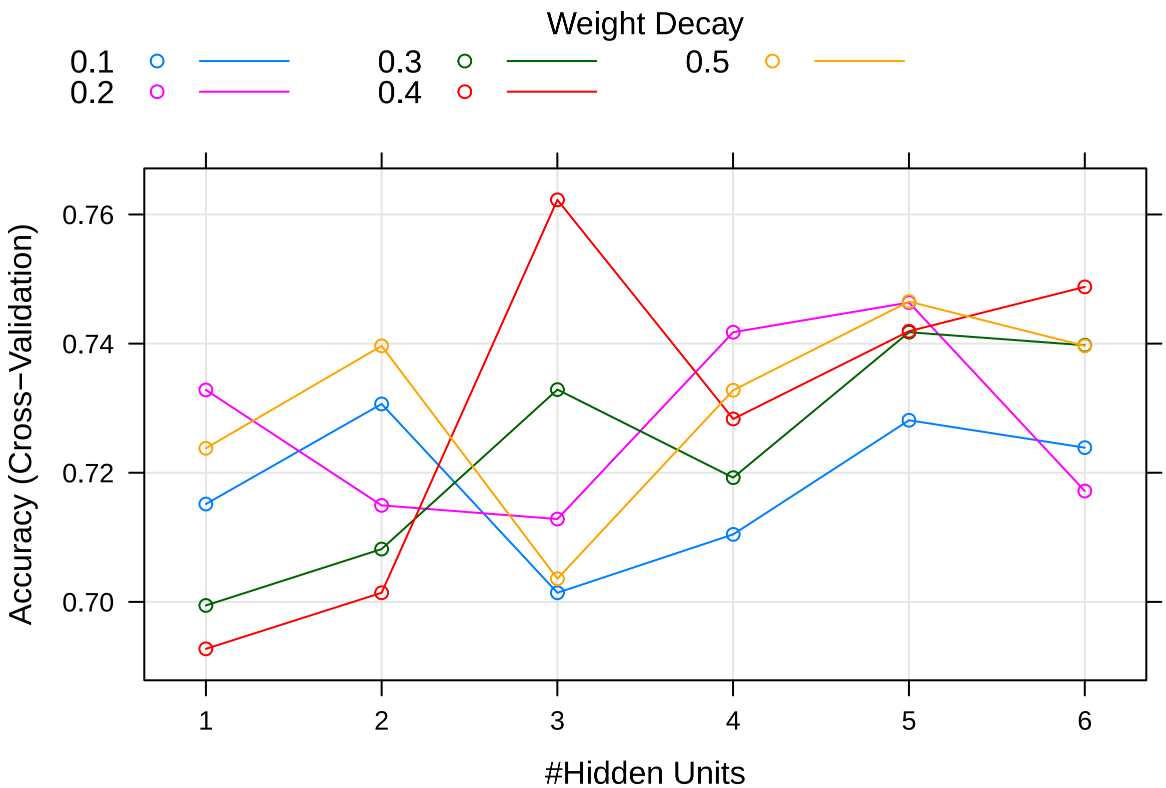
```
##           Reference
## Prediction   0   1
##          0  54  53
##          1  23 120
```

```
cm_rbsvm_tuned$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
```

```
##              0.7012987              0.6936416              0.5046729
##         Neg Pred Value              Precision                 Recall
##              0.8391608              0.5046729              0.7012987
##                     F1             Prevalence         Detection Rate
##              0.5869565              0.3080000              0.2160000
## Detection Prevalence      Balanced Accuracy
##              0.4280000              0.6974702
```

6. Neural Network - Simple hyperparameter tuning To select the good parameters, we build a search grid and fit the model with each possible value in the grid. This is brute force and time consuming. The best model is selected among all the possible choices.



The best Neural Networks parameters would be to choose 3 hidden layers, with a decay of 0.4.

```
##           Reference
## Prediction   0   1
##          0  52  53
##          1  25 120
```

```
##            Sensitivity            Specificity         Pos Pred Value
##              0.6753247              0.6936416              0.4952381
##         Neg Pred Value              Precision                 Recall
##              0.8275862              0.4952381              0.6753247
##                     F1             Prevalence         Detection Rate
##              0.5714286              0.3080000              0.2080000
## Detection Prevalence      Balanced Accuracy
##              0.4200000              0.6844831
```

7. Gradient Boosting The Gradient Boosting model accepts only numerical values so we have some transformation to do on our data in order to use it.

```
## ##### xgb.Booster
## raw: 31.2 Mb
## call:
##   xgb.train(params = xgb_params, data = xgb_train, nrounds = 5000,
##     verbose = 1)
## params (as set within xgb.train):
##   booster = "gbtree", eta = "0.01", max_depth = "8", gamma = "4", subsample = "0.75", colsample_bytr
## xgb.attributes:
##   niter
## callbacks:
##   cb.print.evaluation(period = print_every_n)
## # of features: 46
## niter: 5000
## nfeatures : 46
```

Here we have an accuracy of 68.4%. It is good but there is room for improvement.

```
##           Reference
## Prediction   0   1
##          0  57  59
##          1  20 114
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.7402597            0.6589595            0.4913793
##       Neg Pred Value            Precision               Recall
##            0.8507463            0.4913793            0.7402597
##                   F1           Prevalence       Detection Rate
##            0.5906736            0.3080000            0.2280000
## Detection Prevalence    Balanced Accuracy
##            0.4640000            0.6996096
```

Cross-validation with caret The 10-CV can be easily obtained from **caret**.

First, set up the splitting data method using the **trainControl** function.

```
## Generalized Linear Model with Stepwise Feature Selection
##
## 750 samples
##  30 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 674, 676, 675, 675, 675, 675, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7479834  0.3632119
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  35  23
##          1  42 150
##
##                Accuracy : 0.74
```

```
##                  95% CI : (0.681, 0.7932)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.05592
##
##                   Kappa : 0.3452
##
##  Mcnemar's Test P-Value : 0.02557
##
##             Sensitivity : 0.4545
##             Specificity : 0.8671
##          Pos Pred Value : 0.6034
##          Neg Pred Value : 0.7812
##              Prevalence : 0.3080
##          Detection Rate : 0.1400
##    Detection Prevalence : 0.2320
##       Balanced Accuracy : 0.6608
##
##        'Positive' Class : 0
##
```

Bootstrap with 10 replicates We now apply the bootstrap with 10 replicates. Like for CV, we use **caret**.

The approach is the same as before. We only need to change the method in the **trainControl** function. The corresponding method is "boot632".

100 replicates is veryyyy long to run... can do that on less sample ?? I put 10, takes 3 minutes for me

```
## Generalized Linear Model with Stepwise Feature Selection
##
## 750 samples
##  30 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (10 reps)
## Summary of sample sizes: 750, 750, 750, 750, 750, 750, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7544188  0.3893564
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  35  23
##          1  42 150
##
##                Accuracy : 0.74
##                  95% CI : (0.681, 0.7932)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.05592
##
##                   Kappa : 0.3452
##
##  Mcnemar's Test P-Value : 0.02557
```

Table 2: Scores of the models

	Big classification tree	Pruned classification tree	Autoprune classification tree	Logistic regression	AIC reduced Logistic regression	Linear support vector machine	Tuned linear support vector machine	Radial base support vector machine	Tuned radial base support vector machine	Hyperparameter tuned neural network 3 nodes	Gradient Boosting
Accuracy	0.6440	0.5640	0.5840	0.7040	0.6840	0.6920	0.6920	0.7000	0.6960	0.6880	0.6840
Kappa	0.2945	0.2083	0.2251	0.3479	0.3200	0.3328	0.3283	0.3628	0.3564	0.3352	0.3500
AccuracyLower	0.5812	0.5001	0.5202	0.6432	0.6224	0.6307	0.6307	0.6391	0.6349	0.6266	0.6224
AccuracyUpper	0.7033	0.6264	0.6458	0.7599	0.7411	0.7486	0.7486	0.7561	0.7524	0.7449	0.7411
AccuracyNull	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920	0.6920
AccuracyPValue	0.9552	1.0000	0.9999	0.3690	0.6369	0.5308	0.5308	0.4219	0.4762	0.5847	0.6369
McnemarPValue	0.0000	0.0000	0.0000	0.0481	0.0069	0.0122	0.0227	0.0012	0.0009	0.0022	0.0000

Table 3: Scores of the KNN models

	2-Nearest neighbor	3-Nearest neighbor
Accuracy	0.5960	0.6360
Kappa	0.0131	0.0229
AccuracyLower	0.5323	0.5730
AccuracyUpper	0.6574	0.6957
AccuracyNull	0.6920	0.6920
AccuracyPValue	0.9995	0.9752
McnemarPValue	0.3197	0.0004

```
##
##             Sensitivity : 0.4545
##             Specificity : 0.8671
##          Pos Pred Value : 0.6034
##          Neg Pred Value : 0.7812
##              Prevalence : 0.3080
##          Detection Rate : 0.1400
##    Detection Prevalence : 0.2320
##       Balanced Accuracy : 0.6608
##
##        'Positive' Class : 0
##
```

Review of statistics

Once all the models were modelized we have to compare them according to their scores and metrics. Below we summarized all their accuracy into one table.

Another table is done to compare the KNN because they were not performed on the balanced dataset.

Conclusion

Our recommendations/Suggestions

References

Annexes