Analysis

Elodie Kwan and Katia Voltz

2022-04-26

Import libraries and data

```r
German_credit_cleaned <- read.csv("./../Data_DA/GermanCredit_cleaned.csv", header = TRUE)

German_credit_cleaned$DURATION <- as.numeric(German_credit_cleaned$DURATION)
German_credit_cleaned$AMOUNT <- as.numeric(German_credit_cleaned$AMOUNT)
German_credit_cleaned$INSTALL_RATE <- as.numeric(German_credit_cleaned$INSTALL_RATE)
German_credit_cleaned$AGE <- as.numeric(German_credit_cleaned$AGE)
German_credit_cleaned$NUM_CREDITS <- as.numeric(German_credit_cleaned$NUM_CREDITS)
German_credit_cleaned$NUM_DEPENDENTS <- as.numeric(German_credit_cleaned$NUM_DEPENDENTS)

for (i in 1:ncol(German_credit_cleaned)){
  if (class(German_credit_cleaned[,i])=="integer"){
    German_credit_cleaned[,i] <- factor(German_credit_cleaned[,i])
    }
}

str(German_credit_cleaned)
```

```
## '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 ...
```

Traning set and Test set

In order to fit some model, we are going to split our dataset into 2 sets. The **training set** and the **test set**. We do not forget to take the first variable **OBS.** out as it represents the index number for each observation.

```
set.seed(1234)
# indexes of the training set observations
index.tr <- sample(x = 1:nrow(German_credit_cleaned),
                   size= 3/4*nrow(German_credit_cleaned), replace=FALSE)
# training set
German_credit.tr <- German_credit_cleaned[index.tr,-1]
# test set
German_credit.te <- German_credit_cleaned[-index.tr,-1]
```

Balancing the dataset

In this part, we apply the balancing data technique in order to improve the predictions of "Good Credit" and "Bad Credit", since we have more observations on the "Bad Credit".

The table below reveals the unbalanced problem.

```
## Statistics on the training set
table(German_credit.tr$RESPONSE)
```

```
##
##   0   1
## 223 527
```

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").

```
n.Bad_Credit <- min(table(German_credit.tr$RESPONSE)) ## 32

## the "0" cases
German_credit.tr.Bad_Credit <- filter(German_credit.tr, RESPONSE == 0)

## The "1" cases
German_credit.tr.Good_Credit <- filter(German_credit.tr, RESPONSE == 1)

## sub-sample 32 instances from the "Good Credit" by drawing indexes
index.no <- sample(size=n.Bad_Credit, x=1:nrow(German_credit.tr.Good_Credit), replace=FALSE)
```

```r
## Bind all the "Bad" and the sub-sampled "Good"
German_Credit.tr.subs <- data.frame(rbind(German_credit.tr.Bad_Credit,
                                          German_credit.tr.Good_Credit[index.no,]))

## The cases are balanced
table(German_Credit.tr.subs$RESPONSE)
```

```
##
##   0   1
## 223 223
```

The dataset is now balanced.

Fitting a model : Classification Tree (Decision Tree)

Let's try a classification tree

```r
german.ct <- rpart(RESPONSE ~ ., method = "class", data = German_Credit.tr.subs )
summary(german.ct)
```

```
## Call:
## rpart(formula = RESPONSE ~ ., data = German_Credit.tr.subs, method = "class")
##   n= 446
##
##           CP nsplit rel error    xerror       xstd
## 1 0.39910314      0 1.0000000 1.0852018 0.04717919
## 2 0.02242152      1 0.6008969 0.6367713 0.04411727
## 3 0.01457399      2 0.5784753 0.6681614 0.04466826
## 4 0.01195815      7 0.4843049 0.6905830 0.04502767
## 5 0.01121076     10 0.4484305 0.6726457 0.04474239
## 6 0.01000000     12 0.4260090 0.6726457 0.04474239
##
## Variable importance
##        CHK_ACCT          AMOUNT         HISTORY        DURATION       GUARANTOR
##              34              15              14              10               6
##      EMPLOYMENT        SAV_ACCT             AGE     MALE_SINGLE     NUM_CREDITS
##               6               4               2               2               2
##    INSTALL_RATE     REAL_ESTATE MALE_MAR_or_WID
##               1               1               1
##
## Node number 1: 446 observations,    complexity param=0.3991031
##   predicted class=0  expected loss=0.5  P(node) =1
##     class counts:   223   223
##    probabilities: 0.500 0.500
##   left son=2 (269 obs) right son=3 (177 obs)
##   Primary splits:
##       CHK_ACCT splits as  LLRR,       improve=37.098750, (0 missing)
##       HISTORY  splits as  LLLLR,      improve=12.867960, (0 missing)
##       AMOUNT   < 8962.5 to the right, improve=12.636380, (0 missing)
##       DURATION < 34.5   to the right, improve=11.073260, (0 missing)
##       SAV_ACCT splits as  LLRRR,      improve= 9.538067, (0 missing)
##   Surrogate splits:
##       SAV_ACCT    splits as  LLRRL,      agree=0.648, adj=0.113, (0 split)
##       DURATION    < 10.5   to the right, agree=0.621, adj=0.045, (0 split)
##       HISTORY     splits as  LLLLR,      agree=0.617, adj=0.034, (0 split)
```

```
##       AMOUNT      < 440.5  to the right, agree=0.610, adj=0.017, (0 split)
##       REAL_ESTATE splits as  LR,         agree=0.608, adj=0.011, (0 split)
##
## Node number 2: 269 observations,    complexity param=0.01457399
##   predicted class=0  expected loss=0.3345725  P(node) =0.603139
##     class counts:   179    90
##    probabilities: 0.665 0.335
##   left son=4 (22 obs) right son=5 (247 obs)
##   Primary splits:
##       AMOUNT         < 9340.5 to the right, improve=5.363996, (0 missing)
##       DURATION       < 27.5   to the right, improve=5.072190, (0 missing)
##       HISTORY        splits as  LLLRR,      improve=5.002720, (0 missing)
##       GUARANTOR      splits as  LR,         improve=3.606846, (0 missing)
##       PROP_UNKN_NONE splits as  RL,         improve=3.532773, (0 missing)
##
## Node number 3: 177 observations,    complexity param=0.02242152
##   predicted class=1  expected loss=0.2485876  P(node) =0.396861
##     class counts:    44   133
##    probabilities: 0.249 0.751
##   left son=6 (17 obs) right son=7 (160 obs)
##   Primary splits:
##       HISTORY       splits as  LRRLR,      improve=5.972088, (0 missing)
##       AMOUNT        < 7839.5 to the right, improve=5.310802, (0 missing)
##       EMPLOYMENT    splits as  LLLRR,      improve=4.645266, (0 missing)
##       RETRAINING    splits as  RL,         improve=3.509915, (0 missing)
##       OTHER_INSTALL splits as  RL,         improve=3.216630, (0 missing)
##
## Node number 4: 22 observations
##   predicted class=0  expected loss=0  P(node) =0.04932735
##     class counts:    22     0
##    probabilities: 1.000 0.000
##
## Node number 5: 247 observations,    complexity param=0.01457399
##   predicted class=0  expected loss=0.3643725  P(node) =0.5538117
##     class counts:   157    90
##    probabilities: 0.636 0.364
##   left son=10 (38 obs) right son=11 (209 obs)
##   Primary splits:
##       HISTORY   splits as  LLRRR,      improve=4.867501, (0 missing)
##       SAV_ACCT  splits as  LLLRR,      improve=3.593650, (0 missing)
##       DURATION  < 27.5   to the right, improve=3.361335, (0 missing)
##       GUARANTOR splits as  LR,         improve=2.939271, (0 missing)
##       USED_CAR  splits as  LR,         improve=2.918703, (0 missing)
##
## Node number 6: 17 observations
##   predicted class=0  expected loss=0.3529412  P(node) =0.03811659
##     class counts:    11     6
##    probabilities: 0.647 0.353
##
## Node number 7: 160 observations,    complexity param=0.01121076
##   predicted class=1  expected loss=0.20625  P(node) =0.3587444
##     class counts:    33   127
##    probabilities: 0.206 0.794
##   left son=14 (82 obs) right son=15 (78 obs)
```

```
##   Primary splits:
##       EMPLOYMENT    splits as  LLLRR,      improve=3.272428, (0 missing)
##       OTHER_INSTALL splits as  RL,         improve=3.098970, (0 missing)
##       AMOUNT        < 7447   to the right, improve=2.953289, (0 missing)
##       CHK_ACCT      splits as  --LR,       improve=1.749795, (0 missing)
##       DURATION      < 34.5   to the right, improve=1.608088, (0 missing)
##   Surrogate splits:
##       MALE_SINGLE    splits as  LR,         agree=0.650, adj=0.282, (0 split)
##       REAL_ESTATE    splits as  RL,         agree=0.613, adj=0.205, (0 split)
##       AGE            < 33.5   to the left,  agree=0.600, adj=0.179, (0 split)
##       NUM_DEPENDENTS < 1.5    to the left,  agree=0.581, adj=0.141, (0 split)
##       INSTALL_RATE   < 3.5    to the left,  agree=0.575, adj=0.128, (0 split)
##
## Node number 10: 38 observations
##   predicted class=0  expected loss=0.1315789  P(node) =0.08520179
##     class counts:    33     5
##    probabilities: 0.868 0.132
##
## Node number 11: 209 observations,    complexity param=0.01457399
##   predicted class=0  expected loss=0.4066986  P(node) =0.4686099
##     class counts:   124    85
##    probabilities: 0.593 0.407
##   left son=22 (60 obs) right son=23 (149 obs)
##   Primary splits:
##       DURATION   < 27.5   to the right, improve=6.078470, (0 missing)
##       SAV_ACCT   splits as  LLLRR,      improve=3.141704, (0 missing)
##       USED_CAR   splits as  LR,         improve=3.000838, (0 missing)
##       EMPLOYMENT splits as  LLRRR,      improve=2.828791, (0 missing)
##       AGE        < 25.5   to the left,  improve=2.823140, (0 missing)
##   Surrogate splits:
##       AMOUNT         < 4201   to the right, agree=0.804, adj=0.317, (0 split)
##       PROP_UNKN_NONE splits as  RL,         agree=0.732, adj=0.067, (0 split)
##       EDUCATION      splits as  RL,         agree=0.718, adj=0.017, (0 split)
##
## Node number 14: 82 observations,    complexity param=0.01121076
##   predicted class=1  expected loss=0.304878  P(node) =0.1838565
##     class counts:    25    57
##    probabilities: 0.305 0.695
##   left son=28 (7 obs) right son=29 (75 obs)
##   Primary splits:
##       AMOUNT        < 6827   to the right, improve=4.668479, (0 missing)
##       DURATION      < 25.5   to the right, improve=4.325542, (0 missing)
##       JOB           splits as  LRRL,       improve=3.679907, (0 missing)
##       OTHER_INSTALL splits as  RL,         improve=2.398955, (0 missing)
##       AGE           < 26.5   to the left,  improve=1.921904, (0 missing)
##   Surrogate splits:
##       GUARANTOR splits as  RL, agree=0.927, adj=0.143, (0 split)
##
## Node number 15: 78 observations
##   predicted class=1  expected loss=0.1025641  P(node) =0.1748879
##     class counts:     8    70
##    probabilities: 0.103 0.897
##
## Node number 22: 60 observations
```

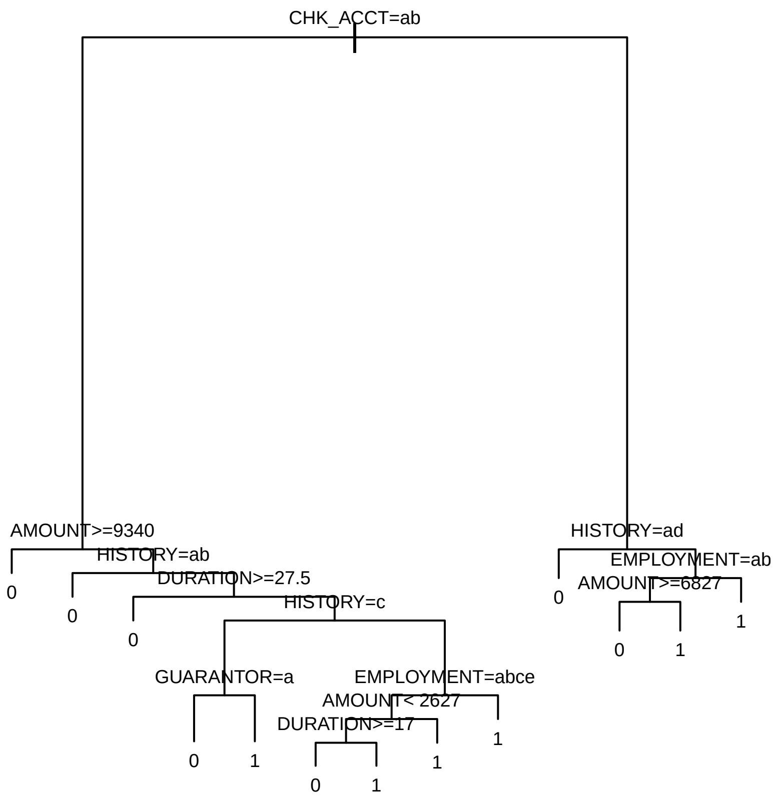
```
##   predicted class=0  expected loss=0.2166667  P(node) =0.1345291
##     class counts:    47    13
##    probabilities: 0.783 0.217
##
## Node number 23: 149 observations,    complexity param=0.01457399
##   predicted class=0  expected loss=0.4832215  P(node) =0.3340807
##     class counts:    77    72
##    probabilities: 0.517 0.483
##   left son=46 (94 obs) right son=47 (55 obs)
##   Primary splits:
##       HISTORY     splits as  --LRR, improve=3.175875, (0 missing)
##       EMPLOYMENT  splits as  LLRRR, improve=2.996737, (0 missing)
##       MALE_SINGLE splits as  LR,    improve=2.747980, (0 missing)
##       GUARANTOR   splits as  LR,    improve=2.665119, (0 missing)
##       USED_CAR    splits as  LR,    improve=2.595185, (0 missing)
##   Surrogate splits:
##       NUM_CREDITS < 1.5    to the left,  agree=0.846, adj=0.582, (0 split)
##       AGE         < 34.5   to the left,  agree=0.698, adj=0.182, (0 split)
##       AMOUNT      < 4212.5 to the left,  agree=0.664, adj=0.091, (0 split)
##       SAV_ACCT    splits as  LLLRL,      agree=0.658, adj=0.073, (0 split)
##       DURATION    < 6.5    to the right, agree=0.651, adj=0.055, (0 split)
##
## Node number 28: 7 observations
##   predicted class=0  expected loss=0.1428571  P(node) =0.01569507
##     class counts:     6     1
##    probabilities: 0.857 0.143
##
## Node number 29: 75 observations
##   predicted class=1  expected loss=0.2533333  P(node) =0.1681614
##     class counts:    19    56
##    probabilities: 0.253 0.747
##
## Node number 46: 94 observations,    complexity param=0.01457399
##   predicted class=0  expected loss=0.4042553  P(node) =0.2107623
##     class counts:    56    38
##    probabilities: 0.596 0.404
##   left son=92 (86 obs) right son=93 (8 obs)
##   Primary splits:
##       GUARANTOR   splits as  LR,         improve=6.206828, (0 missing)
##       NEW_CAR     splits as  RL,         improve=3.635910, (0 missing)
##       RADIO.TV    splits as  LR,         improve=2.878234, (0 missing)
##       MALE_SINGLE splits as  LR,         improve=2.142455, (0 missing)
##       AMOUNT      < 1491   to the left,  improve=1.798335, (0 missing)
##
## Node number 47: 55 observations,    complexity param=0.01195815
##   predicted class=1  expected loss=0.3818182  P(node) =0.1233184
##     class counts:    21    34
##    probabilities: 0.382 0.618
##   left son=94 (46 obs) right son=95 (9 obs)
##   Primary splits:
##       EMPLOYMENT    splits as  LLLRL,      improve=3.137549, (0 missing)
##       TELEPHONE     splits as  LR,         improve=1.982155, (0 missing)
##       SAV_ACCT      splits as  LRLLR,      improve=1.941414, (0 missing)
##       OTHER_INSTALL splits as  RL,         improve=1.857409, (0 missing)
```

```
##       AMOUNT        < 2627   to the left,  improve=1.056516, (0 missing)
##   Surrogate splits:
##       AMOUNT < 678.5  to the right, agree=0.855, adj=0.111, (0 split)
##
## Node number 92: 86 observations
##   predicted class=0  expected loss=0.3488372  P(node) =0.1928251
##     class counts:    56    30
##    probabilities: 0.651 0.349
##
## Node number 93: 8 observations
##   predicted class=1  expected loss=0  P(node) =0.01793722
##     class counts:     0     8
##    probabilities: 0.000 1.000
##
## Node number 94: 46 observations,    complexity param=0.01195815
##   predicted class=1  expected loss=0.4565217  P(node) =0.103139
##     class counts:    21    25
##    probabilities: 0.457 0.543
##   left son=188 (30 obs) right son=189 (16 obs)
##   Primary splits:
##       AMOUNT        < 2627   to the left,  improve=2.0927540, (0 missing)
##       SAV_ACCT      splits as  LRLLR,      improve=1.6246220, (0 missing)
##       TELEPHONE     splits as  LR,         improve=1.6139660, (0 missing)
##       OTHER_INSTALL splits as  RL,         improve=1.4339300, (0 missing)
##       EMPLOYMENT    splits as  LLR-R,      improve=0.9882491, (0 missing)
##   Surrogate splits:
##       INSTALL_RATE < 2.5    to the right, agree=0.826, adj=0.500, (0 split)
##       TELEPHONE    splits as  LR,         agree=0.739, adj=0.250, (0 split)
##       FURNITURE    splits as  LR,         agree=0.696, adj=0.125, (0 split)
##       USED_CAR     splits as  LR,         agree=0.674, adj=0.062, (0 split)
##       EDUCATION    splits as  LR,         agree=0.674, adj=0.062, (0 split)
##
## Node number 95: 9 observations
##   predicted class=1  expected loss=0  P(node) =0.02017937
##     class counts:     0     9
##    probabilities: 0.000 1.000
##
## Node number 188: 30 observations,    complexity param=0.01195815
##   predicted class=0  expected loss=0.4333333  P(node) =0.06726457
##     class counts:    17    13
##    probabilities: 0.567 0.433
##   left son=376 (14 obs) right son=377 (16 obs)
##   Primary splits:
##       DURATION < 17     to the right, improve=2.5190480, (0 missing)
##       AMOUNT   < 1911   to the right, improve=1.5407870, (0 missing)
##       SAV_ACCT splits as  LLRRR,      improve=1.4000000, (0 missing)
##       AGE      < 37     to the left,  improve=0.8960128, (0 missing)
##       JOB      splits as  RRLR,       improve=0.8333333, (0 missing)
##   Surrogate splits:
##       MALE_SINGLE      splits as  LR,         agree=0.733, adj=0.429, (0 split)
##       AGE              < 34     to the left,  agree=0.700, adj=0.357, (0 split)
##       MALE_MAR_or_WID  splits as  RL,         agree=0.667, adj=0.286, (0 split)
##       AMOUNT           < 1569.5 to the right, agree=0.633, adj=0.214, (0 split)
##       PRESENT_RESIDENT splits as  -LRR,       agree=0.633, adj=0.214, (0 split)
```

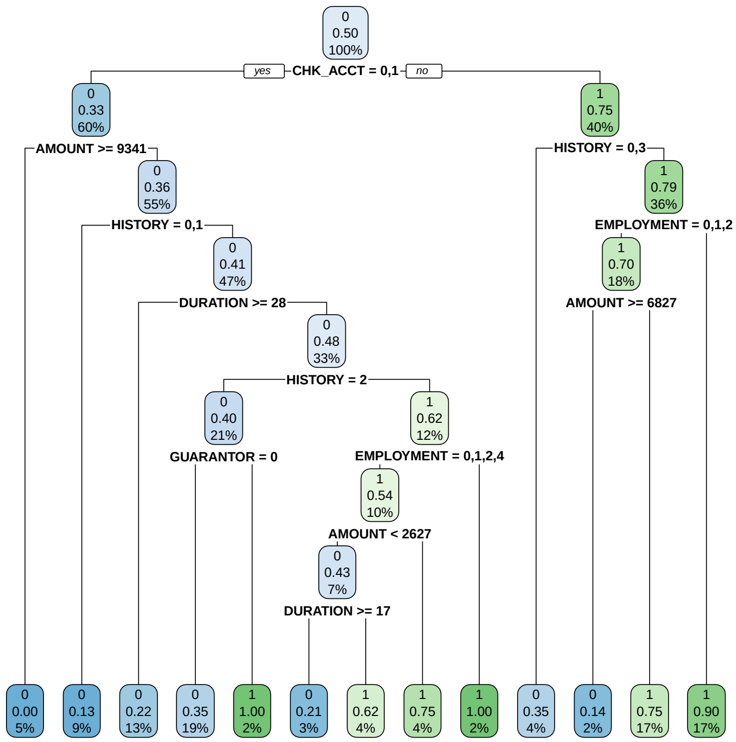
```
##
## Node number 189: 16 observations
##   predicted class=1  expected loss=0.25  P(node) =0.03587444
##     class counts:     4    12
##    probabilities: 0.250 0.750
##
## Node number 376: 14 observations
##   predicted class=0  expected loss=0.2142857  P(node) =0.03139013
##     class counts:    11     3
##    probabilities: 0.786 0.214
##
## Node number 377: 16 observations
##   predicted class=1  expected loss=0.375  P(node) =0.03587444
##     class counts:     6    10
##    probabilities: 0.375 0.625
```

Then we plot the classification tree.

```
par(pty = "s", mar = c(1, 1, 1, 1))
plot(german.ct, cex = 1)
text(german.ct, cex = 0.6)
```




```
rpart.plot(german.ct)
```



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

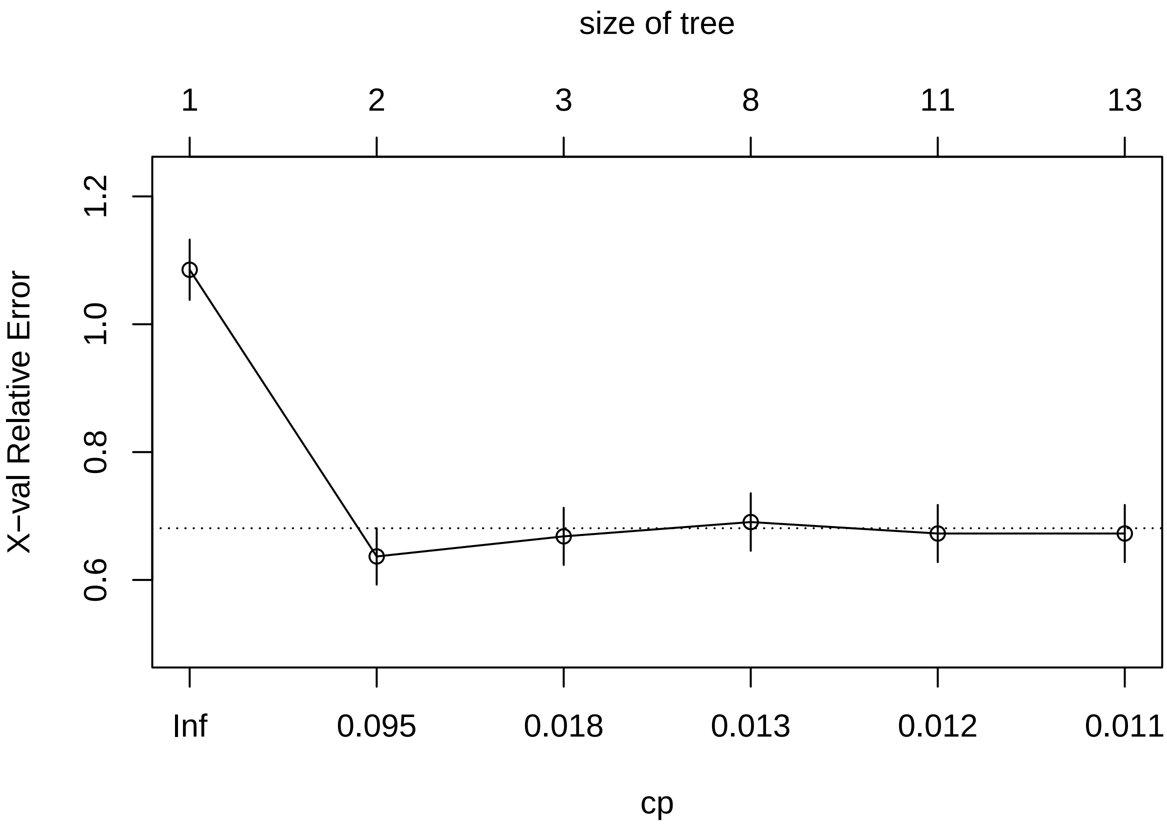
As the tree is quite big and we want to know if we can prune it.

```
printcp(german.ct)
```

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
```

```
plotcp(german.ct)
```

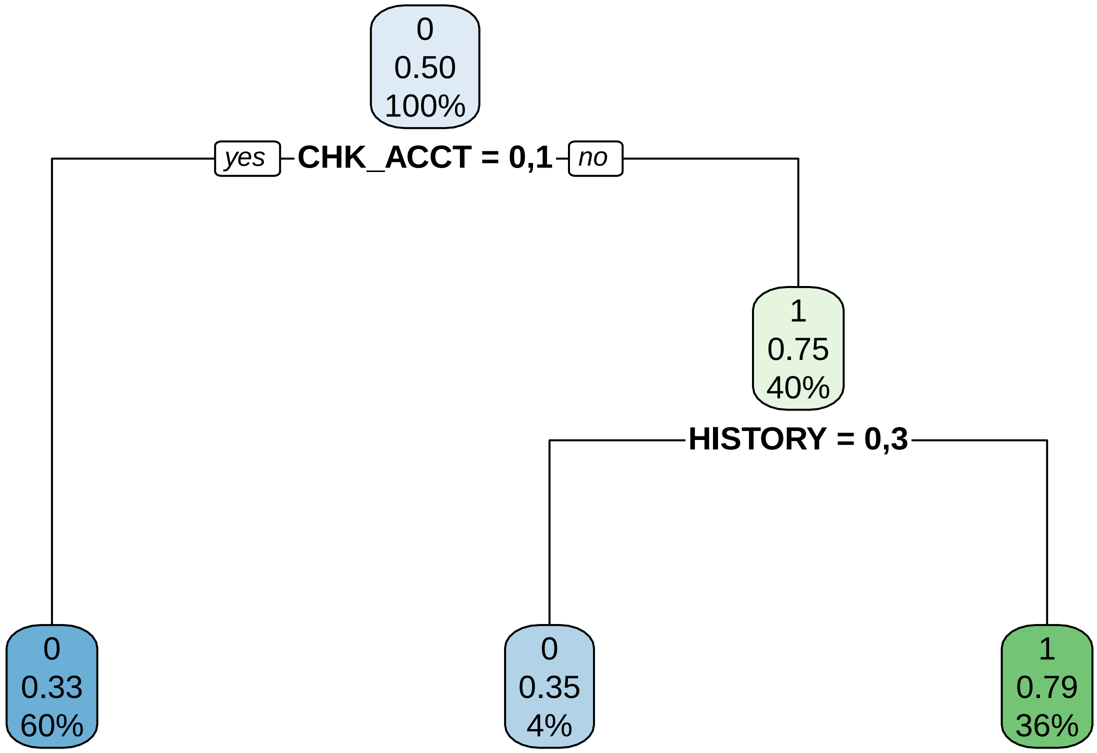


From the list of **cp** (complexity parameter), we would choose the line that has the lowest cross valiadation error. This can be seen on the column **xerror**. So the best cp would be 0.022422 with one split.

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.

```
german.ct.prune <- prune(german.ct, cp=0.022)
rpart.plot(german.ct.prune)
```



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

Using this pruned tree we made by hand, we can compute the prediction and build a confusion matrix.

```
german.ct.prediction <- predict(german.ct.prune, newdata=German_credit.te, type="class")

# Confusion Matrix
confusionMatrix(data=german.ct.prediction, reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 63 95
##          1 14 78
##
##                Accuracy : 0.564
##                  95% CI : (0.5001, 0.6264)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2083
##
##  Mcnemar's Test P-Value : 1.822e-14
##
##             Sensitivity : 0.8182
##             Specificity : 0.4509
##          Pos Pred Value : 0.3987
##          Neg Pred Value : 0.8478
##              Prevalence : 0.3080
##          Detection Rate : 0.2520
```

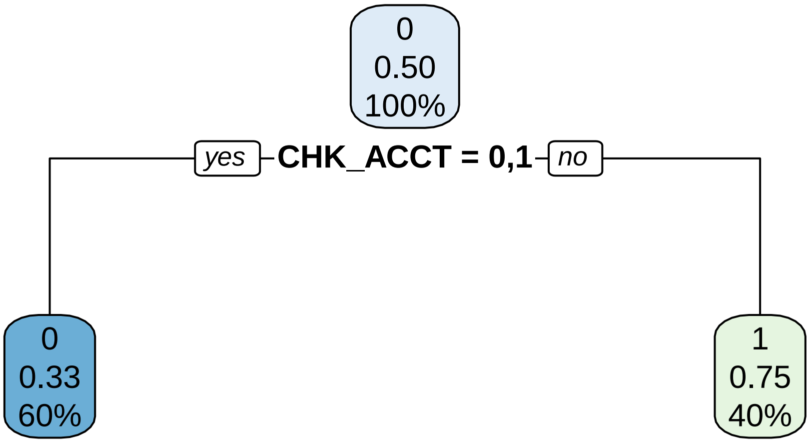
```
##    Detection Prevalence : 0.6320
##       Balanced Accuracy : 0.6345
##
##        'Positive' Class : 0
##
```

The accuracy of the classification tree model is not really good (0.564), however we see that the balanced accuracy (0.6345) is a little bit better but still not enough.

We decide to also look at what would an automatically pruned using 1-SE rule would give use and compare the model.

```
set.seed(1234)
german.ct.autoprune <- autoprune(RESPONSE ~ ., method = "class",
                                 data = German_Credit.tr.subs)
rpart.plot(german.ct.autoprune )
```

```
## Warning: Cannot retrieve the data used to build the model (so cannot determine roundint and is.binary
## To silence this warning:
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



Here, only the variable **CHK_ACCT** is used.

```
german.ct.autoprune.prediction <- predict(german.ct.autoprune,
                                          newdata=German_credit.te,
                                          type="class")

# Confusion Matrix
confusionMatrix(data=german.ct.autoprune.prediction,
                reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 61 88
##          1 16 85
##
##                Accuracy : 0.584
##                  95% CI : (0.5202, 0.6458)
##     No Information Rate : 0.692
```

```
##     P-Value [Acc > NIR] : 0.9999
##
##                   Kappa : 0.2251
##
##  Mcnemar's Test P-Value : 3.352e-12
##
##             Sensitivity : 0.7922
##             Specificity : 0.4913
##          Pos Pred Value : 0.4094
##          Neg Pred Value : 0.8416
##              Prevalence : 0.3080
##          Detection Rate : 0.2440
##    Detection Prevalence : 0.5960
##       Balanced Accuracy : 0.6418
##
##        'Positive' Class : 0
##
```

The accuracy of the classification tree model is not really good (0.584) althought it is a little bit better than the one we fitted by hand. We also see that the balanced accuracy (0.6418) is a little bit better than its accuracy and the balanced accuracy of the other model, but still not enough.

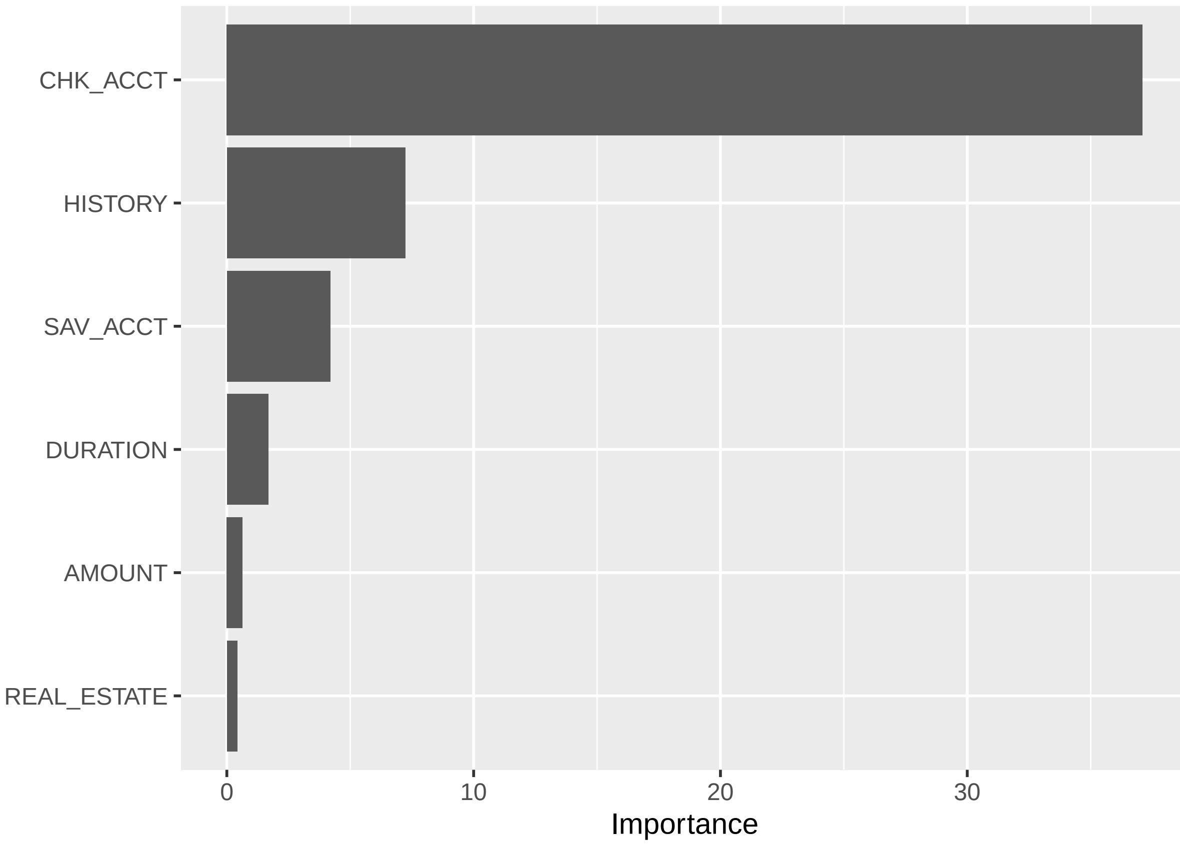
```
varImp(german.ct.prune)
```

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
```

```r
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.

Fitting another model : Logistic Regression

```r
# Logistic regression to see the significant variables (not working)
logreg <- glm(RESPONSE~., data = German_Credit.tr.subs, family= binomial)
summary(logreg)
```

```
##
## 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.

We can predict using the logistic regression.

```r
german.logrer.pred_prob <- predict(logreg, newdata = German_credit.te,
                                   type="response")

german.logrer.pred <- ifelse(german.logrer.pred_prob >= 0.5, 1, 0)

# Confusion Matrix
confusionMatrix(data=as.factor(german.logrer.pred),
                reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  49  46
##          1  28 127
##
##                Accuracy : 0.704
##                  95% CI : (0.6432, 0.7599)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.36896
##
##                   Kappa : 0.3479
##
##  Mcnemar's Test P-Value : 0.04813
##
##             Sensitivity : 0.6364
##             Specificity : 0.7341
##          Pos Pred Value : 0.5158
##          Neg Pred Value : 0.8194
##              Prevalence : 0.3080
##          Detection Rate : 0.1960
##    Detection Prevalence : 0.3800
##       Balanced Accuracy : 0.6852
##
##        'Positive' Class : 0
##
```

From the confusion matrix summary, this model is better than the classification tree as the accuracy is higher (0.704) and that the balanced accuracy seems to be almost good as well (0.6852)

Variable selection and interpretation with step method (AIC criteria) The stepwise variable selection is performed.

```r
mod.logreg.sel <- step(logreg) # store the final model into mod.logreg.sel
```

```
## Start:  AIC=482.97
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     FURNITURE + RADIO.TV + EDUCATION + RETRAINING + AMOUNT +
##     SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV + MALE_SINGLE +
```

```
##     MALE_MAR_or_WID + CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT +
##     REAL_ESTATE + PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT +
##     OWN_RES + NUM_CREDITS + JOB + NUM_DEPENDENTS + TELEPHONE +
##     FOREIGN
##
##                    Df Deviance    AIC
## - JOB               3   394.07 480.07
## - REAL_ESTATE       1   390.97 480.97
## - FURNITURE         1   390.98 480.98
## - NUM_DEPENDENTS    1   391.08 481.08
## - MALE_SINGLE       1   391.21 481.21
## - OWN_RES           1   391.35 481.35
## - RETRAINING        1   391.37 481.37
## - EDUCATION         1   391.50 481.50
## - NEW_CAR           1   391.57 481.57
## - DURATION          1   391.73 481.73
## - RADIO.TV          1   391.77 481.77
## - MALE_DIV          1   391.97 481.97
## - CO.APPLICANT      1   392.06 482.06
## - MALE_MAR_or_WID   1   392.07 482.07
## - AGE               1   392.39 482.39
## - NUM_CREDITS       1   392.46 482.46
## <none>                  390.97 482.97
## - RENT              1   393.15 483.15
## - FOREIGN           1   393.24 483.24
## - OTHER_INSTALL     1   395.01 485.01
## - TELEPHONE         1   395.01 485.01
## - PRESENT_RESIDENT  3   399.08 485.08
## - USED_CAR          1   395.81 485.81
## - PROP_UNKN_NONE    1   396.35 486.35
## - INSTALL_RATE      1   396.83 486.83
## - EMPLOYMENT        4   404.07 488.07
## - GUARANTOR         1   398.34 488.34
## - SAV_ACCT          4   404.98 488.98
## - HISTORY           4   405.97 489.97
## - AMOUNT            1   400.48 490.48
## - CHK_ACCT          3   443.96 529.96
##
## Step:  AIC=480.07
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     FURNITURE + RADIO.TV + EDUCATION + RETRAINING + AMOUNT +
##     SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV + MALE_SINGLE +
##     MALE_MAR_or_WID + CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT +
##     REAL_ESTATE + PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT +
##     OWN_RES + NUM_CREDITS + NUM_DEPENDENTS + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - FURNITURE         1   394.07 478.07
## - REAL_ESTATE       1   394.14 478.14
## - NUM_DEPENDENTS    1   394.36 478.36
## - EDUCATION         1   394.40 478.40
## - MALE_SINGLE       1   394.53 478.53
## - RETRAINING        1   394.54 478.54
## - RADIO.TV          1   394.64 478.64
```

```
## - OWN_RES           1   394.77 478.77
## - NEW_CAR           1   394.77 478.77
## - MALE_MAR_or_WID   1   395.00 479.00
## - DURATION          1   395.18 479.18
## - NUM_CREDITS       1   395.28 479.28
## - MALE_DIV          1   395.34 479.34
## - AGE               1   395.46 479.46
## - CO.APPLICANT      1   395.57 479.57
## <none>                  394.07 480.07
## - FOREIGN           1   396.86 480.86
## - TELEPHONE         1   396.96 480.96
## - RENT              1   397.22 481.22
## - PRESENT_RESIDENT  3   401.89 481.89
## - USED_CAR          1   397.93 481.93
## - OTHER_INSTALL     1   397.95 481.95
## - EMPLOYMENT        4   406.08 484.08
## - INSTALL_RATE      1   400.30 484.30
## - PROP_UNKN_NONE    1   400.37 484.37
## - GUARANTOR         1   401.43 485.43
## - SAV_ACCT          4   407.95 485.95
## - HISTORY           4   409.38 487.38
## - AMOUNT            1   403.88 487.88
## - CHK_ACCT          3   445.71 525.71
##
## Step:  AIC=478.07
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + EDUCATION + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_SINGLE + MALE_MAR_or_WID +
##     CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT + REAL_ESTATE +
##     PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT + OWN_RES + NUM_CREDITS +
##     NUM_DEPENDENTS + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - REAL_ESTATE       1   394.14 476.14
## - NUM_DEPENDENTS    1   394.36 476.36
## - MALE_SINGLE       1   394.53 476.53
## - EDUCATION         1   394.57 476.57
## - OWN_RES           1   394.77 476.77
## - RETRAINING        1   394.87 476.87
## - MALE_MAR_or_WID   1   395.02 477.02
## - DURATION          1   395.18 477.18
## - NUM_CREDITS       1   395.29 477.29
## - RADIO.TV          1   395.29 477.29
## - MALE_DIV          1   395.34 477.34
## - AGE               1   395.46 477.46
## - CO.APPLICANT      1   395.57 477.57
## - NEW_CAR           1   395.72 477.72
## <none>                  394.07 478.07
## - FOREIGN           1   396.86 478.86
## - TELEPHONE         1   396.96 478.96
## - RENT              1   397.22 479.22
## - PRESENT_RESIDENT  3   401.92 479.92
## - OTHER_INSTALL     1   397.95 479.95
## - USED_CAR          1   399.89 481.89
```

```
## - EMPLOYMENT        4   406.08 482.08
## - PROP_UNKN_NONE    1   400.37 482.37
## - INSTALL_RATE      1   400.39 482.39
## - GUARANTOR         1   401.44 483.44
## - SAV_ACCT          4   407.96 483.96
## - HISTORY           4   409.67 485.67
## - AMOUNT            1   403.93 485.93
## - CHK_ACCT          3   446.04 524.04
##
## Step:  AIC=476.14
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + EDUCATION + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_SINGLE + MALE_MAR_or_WID +
##     CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE +
##     AGE + OTHER_INSTALL + RENT + OWN_RES + NUM_CREDITS + NUM_DEPENDENTS +
##     TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - NUM_DEPENDENTS    1   394.44 474.44
## - MALE_SINGLE       1   394.60 474.60
## - EDUCATION         1   394.62 474.62
## - OWN_RES           1   394.83 474.83
## - RETRAINING        1   394.90 474.90
## - MALE_MAR_or_WID   1   395.04 475.04
## - DURATION          1   395.25 475.25
## - NUM_CREDITS       1   395.34 475.34
## - MALE_DIV          1   395.36 475.36
## - RADIO.TV          1   395.43 475.43
## - AGE               1   395.54 475.54
## - CO.APPLICANT      1   395.62 475.62
## - NEW_CAR           1   395.75 475.75
## <none>                  394.14 476.14
## - FOREIGN           1   396.89 476.89
## - TELEPHONE         1   396.96 476.96
## - RENT              1   397.28 477.28
## - PRESENT_RESIDENT  3   402.07 478.07
## - OTHER_INSTALL     1   398.14 478.14
## - USED_CAR          1   400.04 480.04
## - EMPLOYMENT        4   406.12 480.12
## - PROP_UNKN_NONE    1   400.61 480.61
## - INSTALL_RATE      1   400.63 480.63
## - GUARANTOR         1   401.92 481.92
## - SAV_ACCT          4   408.08 482.08
## - HISTORY           4   409.69 483.69
## - AMOUNT            1   404.35 484.35
## - CHK_ACCT          3   447.81 523.81
##
## Step:  AIC=474.44
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + EDUCATION + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_SINGLE + MALE_MAR_or_WID +
##     CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE +
##     AGE + OTHER_INSTALL + RENT + OWN_RES + NUM_CREDITS + TELEPHONE +
##     FOREIGN
```

```
##
##                    Df Deviance    AIC
## - EDUCATION         1   394.92 472.92
## - MALE_SINGLE       1   395.09 473.09
## - OWN_RES           1   395.17 473.17
## - RETRAINING        1   395.17 473.17
## - MALE_MAR_or_WID   1   395.38 473.38
## - NUM_CREDITS       1   395.62 473.62
## - DURATION          1   395.62 473.62
## - MALE_DIV          1   395.65 473.65
## - RADIO.TV          1   395.72 473.72
## - AGE               1   395.86 473.86
## - CO.APPLICANT      1   395.92 473.92
## - NEW_CAR           1   395.93 473.93
## <none>                  394.44 474.44
## - FOREIGN           1   397.19 475.19
## - TELEPHONE         1   397.23 475.23
## - RENT              1   397.64 475.64
## - OTHER_INSTALL     1   398.27 476.27
## - PRESENT_RESIDENT  3   402.33 476.33
## - USED_CAR          1   400.66 478.66
## - EMPLOYMENT        4   406.68 478.68
## - PROP_UNKN_NONE    1   400.76 478.76
## - INSTALL_RATE      1   401.31 479.31
## - GUARANTOR         1   402.15 480.15
## - SAV_ACCT          4   408.32 480.32
## - HISTORY           4   409.71 481.71
## - AMOUNT            1   404.85 482.85
## - CHK_ACCT          3   448.11 522.11
##
## Step:  AIC=472.92
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_SINGLE + MALE_MAR_or_WID +
##     CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE +
##     AGE + OTHER_INSTALL + RENT + OWN_RES + NUM_CREDITS + TELEPHONE +
##     FOREIGN
##
##                    Df Deviance    AIC
## - MALE_SINGLE       1   395.58 471.58
## - OWN_RES           1   395.70 471.70
## - MALE_MAR_or_WID   1   395.82 471.82
## - RADIO.TV          1   395.83 471.83
## - DURATION          1   396.03 472.03
## - RETRAINING        1   396.05 472.05
## - NUM_CREDITS       1   396.12 472.12
## - MALE_DIV          1   396.16 472.16
## - AGE               1   396.46 472.46
## - CO.APPLICANT      1   396.54 472.54
## <none>                  394.92 472.92
## - NEW_CAR           1   397.36 473.36
## - FOREIGN           1   397.66 473.66
## - TELEPHONE         1   397.84 473.84
## - RENT              1   398.20 474.20
```

```
## - OTHER_INSTALL     1   398.71 474.71
## - PRESENT_RESIDENT  3   402.81 474.81
## - USED_CAR          1   400.66 476.66
## - PROP_UNKN_NONE    1   400.97 476.97
## - EMPLOYMENT        4   407.23 477.23
## - INSTALL_RATE      1   401.81 477.81
## - GUARANTOR         1   402.42 478.42
## - SAV_ACCT          4   409.38 479.38
## - HISTORY           4   410.48 480.48
## - AMOUNT            1   405.60 481.60
## - CHK_ACCT          3   448.27 520.27
##
## Step:  AIC=471.58
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_MAR_or_WID + CO.APPLICANT +
##     GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE + AGE + OTHER_INSTALL +
##     RENT + OWN_RES + NUM_CREDITS + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - OWN_RES           1   396.32 470.32
## - RADIO.TV          1   396.61 470.61
## - RETRAINING        1   396.63 470.63
## - DURATION          1   396.64 470.64
## - NUM_CREDITS       1   396.75 470.75
## - CO.APPLICANT      1   397.14 471.14
## - MALE_MAR_or_WID   1   397.23 471.23
## - AGE               1   397.36 471.36
## <none>                  395.58 471.58
## - MALE_DIV          1   397.73 471.73
## - NEW_CAR           1   397.94 471.94
## - FOREIGN           1   398.40 472.40
## - TELEPHONE         1   398.56 472.56
## - RENT              1   398.98 472.98
## - OTHER_INSTALL     1   399.33 473.33
## - PRESENT_RESIDENT  3   403.51 473.51
## - USED_CAR          1   401.28 475.28
## - PROP_UNKN_NONE    1   401.47 475.47
## - INSTALL_RATE      1   401.91 475.91
## - GUARANTOR         1   403.51 477.51
## - EMPLOYMENT        4   409.53 477.53
## - SAV_ACCT          4   409.88 477.88
## - HISTORY           4   411.06 479.06
## - AMOUNT            1   405.84 479.84
## - CHK_ACCT          3   448.88 518.88
##
## Step:  AIC=470.32
## RESPONSE ~ CHK_ACCT + DURATION + HISTORY + NEW_CAR + USED_CAR +
##     RADIO.TV + RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT +
##     INSTALL_RATE + MALE_DIV + MALE_MAR_or_WID + CO.APPLICANT +
##     GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE + AGE + OTHER_INSTALL +
##     RENT + NUM_CREDITS + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
```

```
## - DURATION          1   397.20 469.20
## - RADIO.TV          1   397.29 469.29
## - NUM_CREDITS       1   397.39 469.39
## - RETRAINING        1   397.39 469.39
## - MALE_MAR_or_WID   1   397.86 469.86
## - CO.APPLICANT      1   397.91 469.91
## <none>                  396.32 470.32
## - AGE               1   398.39 470.39
## - MALE_DIV          1   398.40 470.40
## - NEW_CAR           1   398.57 470.57
## - FOREIGN           1   398.92 470.92
## - TELEPHONE         1   399.22 471.22
## - OTHER_INSTALL     1   400.14 472.14
## - PRESENT_RESIDENT  3   404.24 472.24
## - RENT              1   402.09 474.09
## - USED_CAR          1   402.59 474.59
## - INSTALL_RATE      1   402.66 474.66
## - PROP_UNKN_NONE    1   403.68 475.68
## - EMPLOYMENT        4   410.14 476.14
## - SAV_ACCT          4   410.22 476.22
## - GUARANTOR         1   404.24 476.24
## - HISTORY           4   411.37 477.37
## - AMOUNT            1   406.67 478.67
## - CHK_ACCT          3   449.30 517.30
##
## Step:  AIC=469.2
## RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR + RADIO.TV +
##     RETRAINING + AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE +
##     MALE_DIV + MALE_MAR_or_WID + CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT +
##     PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT + NUM_CREDITS +
##     TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - RADIO.TV          1   398.00 468.00
## - NUM_CREDITS       1   398.24 468.24
## - CO.APPLICANT      1   398.68 468.68
## - RETRAINING        1   398.75 468.75
## - MALE_MAR_or_WID   1   398.82 468.82
## <none>                  397.20 469.20
## - MALE_DIV          1   399.36 469.36
## - NEW_CAR           1   399.59 469.59
## - AGE               1   399.75 469.75
## - FOREIGN           1   400.30 470.30
## - TELEPHONE         1   400.54 470.54
## - OTHER_INSTALL     1   401.18 471.18
## - PRESENT_RESIDENT  3   405.32 471.32
## - RENT              1   402.71 472.71
## - USED_CAR          1   403.35 473.35
## - EMPLOYMENT        4   410.61 474.61
## - SAV_ACCT          4   410.86 474.86
## - GUARANTOR         1   404.97 474.97
## - PROP_UNKN_NONE    1   405.48 475.48
## - INSTALL_RATE      1   405.49 475.49
## - HISTORY           4   412.02 476.02
```

```
## - AMOUNT             1   418.20 488.20
## - CHK_ACCT           3   452.31 518.31
##
## Step:  AIC=468
## RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR + RETRAINING +
##     AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV +
##     MALE_MAR_or_WID + CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT +
##     PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT + NUM_CREDITS +
##     TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - NUM_CREDITS       1   398.89 466.89
## - MALE_MAR_or_WID   1   399.37 467.37
## - CO.APPLICANT      1   399.50 467.50
## <none>                  398.00 468.00
## - MALE_DIV          1   400.33 468.33
## - AGE               1   400.43 468.43
## - RETRAINING        1   400.66 468.66
## - FOREIGN           1   401.11 469.11
## - TELEPHONE         1   401.60 469.60
## - OTHER_INSTALL     1   401.93 469.93
## - PRESENT_RESIDENT  3   406.08 470.08
## - NEW_CAR           1   403.02 471.02
## - USED_CAR          1   403.47 471.47
## - RENT              1   403.84 471.84
## - SAV_ACCT          4   411.51 473.51
## - INSTALL_RATE      1   405.70 473.70
## - EMPLOYMENT        4   411.78 473.78
## - GUARANTOR         1   406.17 474.17
## - HISTORY           4   412.43 474.43
## - PROP_UNKN_NONE    1   406.54 474.54
## - AMOUNT            1   419.79 487.79
## - CHK_ACCT          3   453.56 517.56
##
## Step:  AIC=466.89
## RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR + RETRAINING +
##     AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV +
##     MALE_MAR_or_WID + CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT +
##     PROP_UNKN_NONE + AGE + OTHER_INSTALL + RENT + TELEPHONE +
##     FOREIGN
##
##                    Df Deviance    AIC
## - MALE_MAR_or_WID   1   400.21 466.21
## - CO.APPLICANT      1   400.43 466.43
## <none>                  398.89 466.89
## - MALE_DIV          1   401.09 467.09
## - AGE               1   401.28 467.28
## - RETRAINING        1   401.54 467.54
## - FOREIGN           1   402.04 468.04
## - TELEPHONE         1   402.43 468.43
## - OTHER_INSTALL     1   402.95 468.95
## - PRESENT_RESIDENT  3   407.00 469.00
## - NEW_CAR           1   404.15 470.15
## - USED_CAR          1   404.45 470.45
```

```
## - RENT              1   404.91 470.91
## - SAV_ACCT          4   412.22 472.22
## - INSTALL_RATE      1   406.47 472.47
## - HISTORY           4   412.50 472.50
## - EMPLOYMENT        4   412.60 472.60
## - GUARANTOR         1   407.16 473.16
## - PROP_UNKN_NONE    1   407.24 473.24
## - AMOUNT            1   421.00 487.00
## - CHK_ACCT          3   454.25 516.25
##
## Step:  AIC=466.21
## RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR + RETRAINING +
##     AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV +
##     CO.APPLICANT + GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE +
##     AGE + OTHER_INSTALL + RENT + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - CO.APPLICANT      1   401.67 465.67
## - MALE_DIV          1   402.12 466.12
## <none>                  400.21 466.21
## - AGE               1   402.75 466.75
## - FOREIGN           1   402.78 466.78
## - RETRAINING        1   402.90 466.90
## - TELEPHONE         1   403.60 467.60
## - OTHER_INSTALL     1   404.12 468.12
## - PRESENT_RESIDENT  3   408.28 468.28
## - NEW_CAR           1   405.42 469.42
## - USED_CAR          1   405.66 469.66
## - RENT              1   406.25 470.25
## - SAV_ACCT          4   412.79 470.79
## - HISTORY           4   413.36 471.36
## - INSTALL_RATE      1   408.02 472.02
## - GUARANTOR         1   408.12 472.12
## - PROP_UNKN_NONE    1   408.46 472.46
## - EMPLOYMENT        4   414.61 472.61
## - AMOUNT            1   421.27 485.27
## - CHK_ACCT          3   455.55 515.55
##
## Step:  AIC=465.67
## RESPONSE ~ CHK_ACCT + HISTORY + NEW_CAR + USED_CAR + RETRAINING +
##     AMOUNT + SAV_ACCT + EMPLOYMENT + INSTALL_RATE + MALE_DIV +
##     GUARANTOR + PRESENT_RESIDENT + PROP_UNKN_NONE + AGE + OTHER_INSTALL +
##     RENT + TELEPHONE + FOREIGN
##
##                    Df Deviance    AIC
## - MALE_DIV          1   403.37 465.37
## <none>                  401.67 465.67
## - AGE               1   404.05 466.05
## - RETRAINING        1   404.28 466.28
## - FOREIGN           1   404.40 466.40
## - TELEPHONE         1   405.19 467.19
## - PRESENT_RESIDENT  3   409.37 467.37
## - OTHER_INSTALL     1   405.66 467.66
## - NEW_CAR           1   406.70 468.70
```

```
## - USED_CAR          1   407.58 469.58
## - RENT              1   408.22 470.22
## - HISTORY           4   414.55 470.55
## - SAV_ACCT          4   414.84 470.84
## - INSTALL_RATE      1   409.33 471.33
## - PROP_UNKN_NONE    1   409.84 471.84
## - GUARANTOR         1   410.10 472.10
## - EMPLOYMENT        4   416.15 472.15
## - AMOUNT            1   424.34 486.34
## - CHK_ACCT          3   458.18 516.18
##
## Step:  AIC=465.37
## 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
##
##                    Df Deviance    AIC
## <none>                 403.37 465.37
## - AGE               1   405.39 465.39
## - FOREIGN           1   406.08 466.08
## - PRESENT_RESIDENT  3   410.70 466.70
## - RETRAINING        1   406.78 466.78
## - TELEPHONE         1   406.78 466.78
## - OTHER_INSTALL     1   407.15 467.15
## - NEW_CAR           1   408.67 468.67
## - USED_CAR          1   409.53 469.53
## - HISTORY           4   415.74 469.74
## - RENT              1   410.11 470.11
## - SAV_ACCT          4   416.43 470.43
## - INSTALL_RATE      1   410.48 470.48
## - PROP_UNKN_NONE    1   411.18 471.18
## - EMPLOYMENT        4   417.60 471.60
## - GUARANTOR         1   411.85 471.85
## - AMOUNT            1   426.51 486.51
## - CHK_ACCT          3   460.85 516.85
```

```
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**).

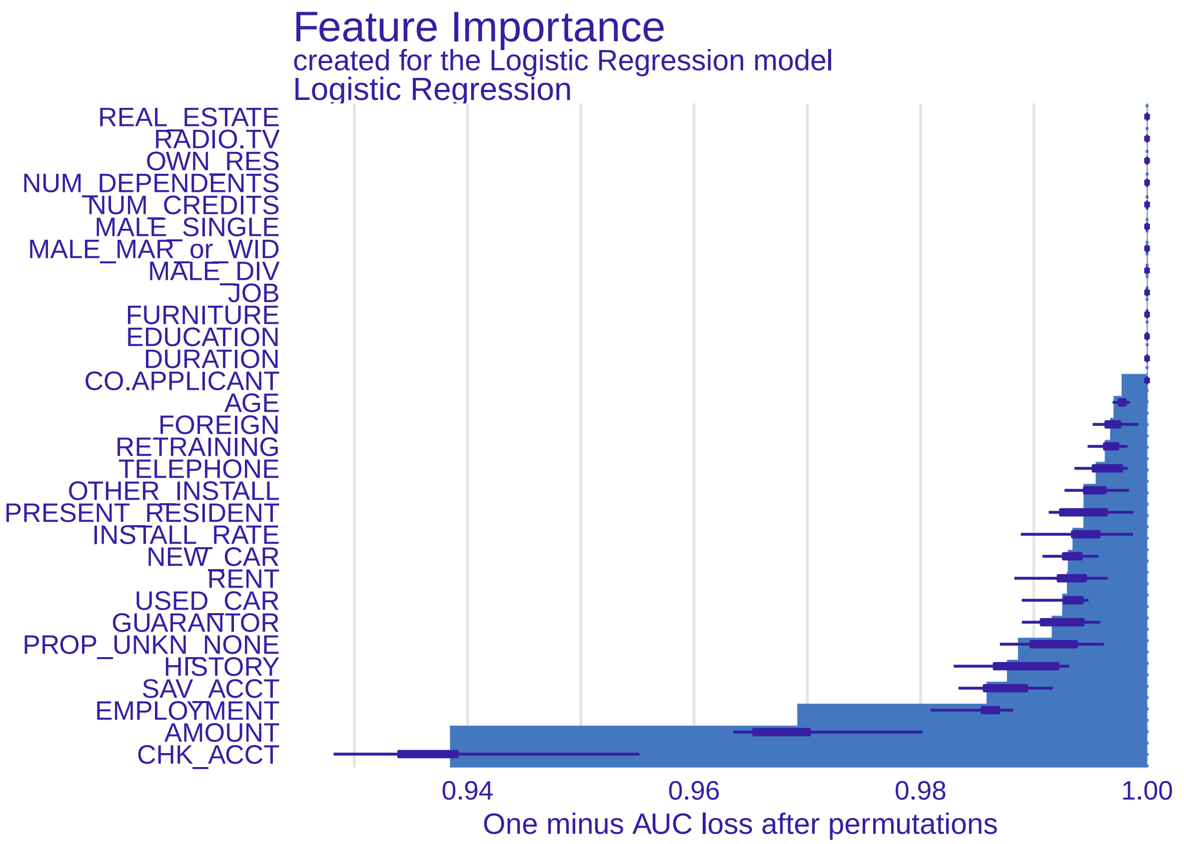
- The predicted probability of being a good applicant for **HISTORY1** is lower than for **HISTORY0**.
- The predicted probability of being a good applicant for **HISTORY4** is higher than for **HISTORY0** (and also higher than for **HISTORY2** and **HISTORY3**).
- The predicted probability of being 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**.

```
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 = "Logistic Regression")
```

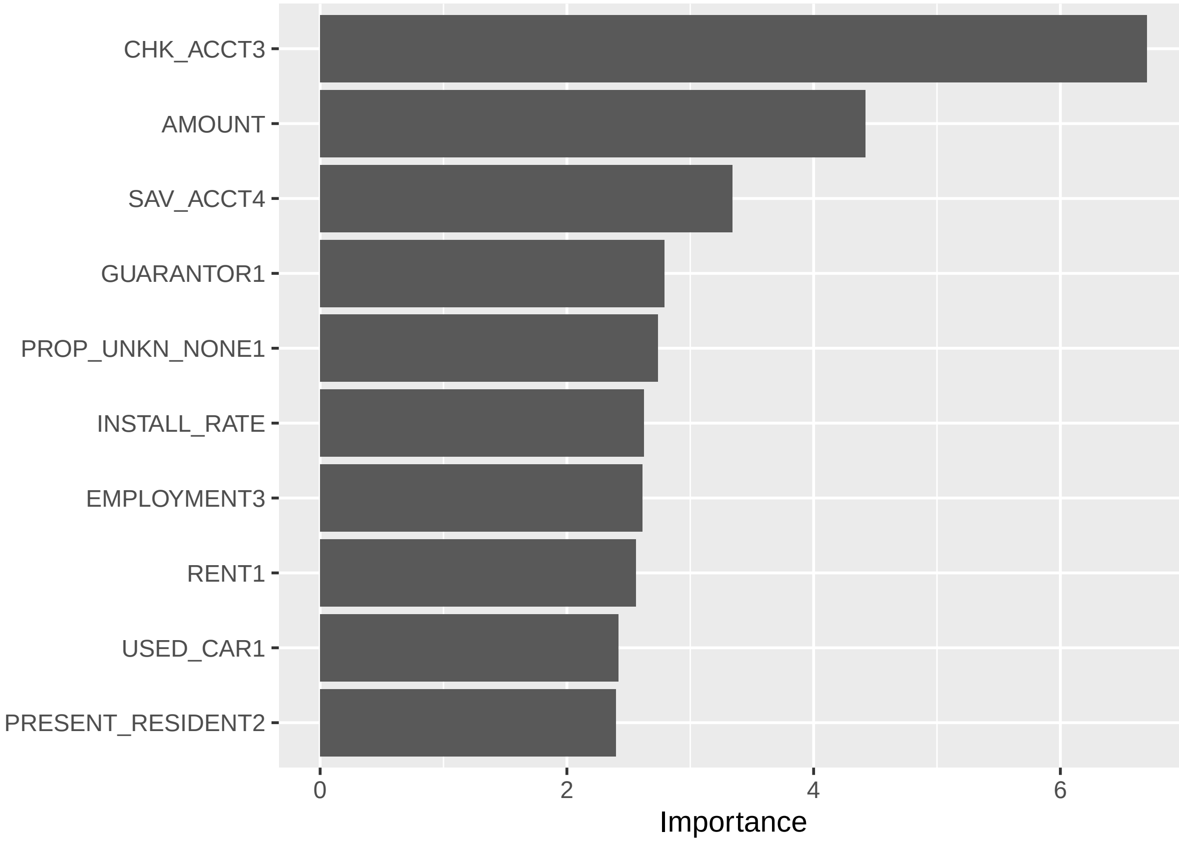
Variable importance for logistic regression

```
## Preparation of a new explainer is initiated
##   -> model label       :  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.

Fitting another model : KNN

To perform a k-nearest neighbor method, we do not need to balance the data so we will use the unbalanced training set. Now, we make the prediction using a 2-NN (with Euclidean distance).

```
German_credit.num.response <- German_credit_cleaned %>%
  select('RESPONSE','DURATION', 'AMOUNT',
         'INSTALL_RATE', 'AGE', 'NUM_CREDITS','NUM_DEPENDENTS')
```

```
set.seed(123)
## build the K-NN model with k = 2
German_credit.knn <- knn3(data=German_credit.num.response, RESPONSE~., k=2)

German_credit.knn.pred <- predict(German_credit.knn,
                                  newdata = German_credit.num.response,
                                  type="class")
```

Now we can use the 2-NN to predict the test set using the training set. Note that the model is fitting on the training set and the predictions are computed on the test set.

```
set.seed(123)
# applying Knn model with k = 2 on the training set
German_credit.knn.tr <- knn3(data=German_credit.tr, RESPONSE~., k=2)

German_credit.knn.tr.pred <- predict(German_credit.knn.tr,
                                     newdata = German_credit.te, type="class")
```

To compare the predictions above and the true state of the applicant (the one in the test set), we can build a table. It is called a **confusion matrix** (again, this will be detailed later on).

```
table(Pred=German_credit.knn.tr.pred, Observed=German_credit.te$RESPONSE)
```

```
##     Observed
## Pred   0   1
##    0  21  45
##    1  56 128
```

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.

```
# Confusion Matrix
confusionMatrix(data=German_credit.knn.tr.pred,
                reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  21  45
##          1  56 128
##
##                Accuracy : 0.596
##                  95% CI : (0.5323, 0.6574)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.9995
##
##                   Kappa : 0.0131
##
##  Mcnemar's Test P-Value : 0.3197
##
##             Sensitivity : 0.2727
##             Specificity : 0.7399
##          Pos Pred Value : 0.3182
##          Neg Pred Value : 0.6957
##              Prevalence : 0.3080
##          Detection Rate : 0.0840
##    Detection Prevalence : 0.2640
##       Balanced Accuracy : 0.5063
##
##        'Positive' Class : 0
##
```

The accuracy (0.596) and the unbalanced accuracy (0.5063) are both too low.

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

```
set.seed(123)
# applying Knn model with k = 3 on the training set
German_credit.knn3.tr <- knn3(data=German_credit.tr, RESPONSE~., k=3)
```

```r
German_credit.knn3.tr.pred <- predict(German_credit.knn3.tr,
                                      newdata = German_credit.te, type="class")

table(Pred=German_credit.knn3.tr.pred, Observed=German_credit.te$RESPONSE)
```

```
##     Observed
## Pred   0   1
##    0  14  28
##    1  63 145
```

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.

```r
# Confusion Matrix
confusionMatrix(data=German_credit.knn3.tr.pred,
                reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  14  28
##          1  63 145
##
##                Accuracy : 0.636
##                  95% CI : (0.573, 0.6957)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.975213
##
##                   Kappa : 0.0229
##
##  Mcnemar's Test P-Value : 0.000365
##
##             Sensitivity : 0.1818
##             Specificity : 0.8382
##          Pos Pred Value : 0.3333
##          Neg Pred Value : 0.6971
##              Prevalence : 0.3080
##          Detection Rate : 0.0560
##    Detection Prevalence : 0.1680
##       Balanced Accuracy : 0.5100
##
##        'Positive' Class : 0
##
```

Both the accuracy (0.636) and the balanced data (0.5100) improved a little bit with k = 3 compared to k = 2. The accuracies might still be a bit low.

Linear Support Vector Machine

…

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

```
##
## 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
```

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

# confusion matrix
table(Pred=German_credit.svm.pred, obs=German_credit.te$RESPONSE)
```

```
##     obs
## Pred   0   1
##    0  50  50
##    1  27 123
```

```r
confusionMatrix(data=German_credit.svm.pred, reference = German_credit.te$RESPONSE )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  50  50
##          1  27 123
##
##                Accuracy : 0.692
##                  95% CI : (0.6307, 0.7486)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.53077
##
##                   Kappa : 0.3328
##
##  Mcnemar's Test P-Value : 0.01217
##
##             Sensitivity : 0.6494
##             Specificity : 0.7110
##          Pos Pred Value : 0.5000
##          Neg Pred Value : 0.8200
##              Prevalence : 0.3080
##          Detection Rate : 0.2000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.6802
##
##        'Positive' Class : 0
##
```

The accuracy (0.692) and the balanced accuracy (0.6802) are lower than 0.75 which means that it might not be good enough.

Radial basis Support Vector Machine

We try now with a radial basis kernel (the default).

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

```
##
## 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
```

```
German_credit.pred <- predict(German_credit.rb, newdata = German_credit.te)
confusionMatrix(data=German_credit.pred, reference = German_credit.te$RESPONSE )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  54  52
##          1  23 121
##
##                Accuracy : 0.7
##                  95% CI : (0.6391, 0.7561)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.421916
##
##                   Kappa : 0.3628
##
##  Mcnemar's Test P-Value : 0.001224
##
##             Sensitivity : 0.7013
##             Specificity : 0.6994
##          Pos Pred Value : 0.5094
##          Neg Pred Value : 0.8403
##              Prevalence : 0.3080
##          Detection Rate : 0.2160
##    Detection Prevalence : 0.4240
##       Balanced Accuracy : 0.7004
##
##        'Positive' Class : 0
##
```

The accuracy is better, 70%, compared to 69% with the linear method.

Tunning the hyperparameters of Linear SVM

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

```r
German_Credit.trctrl <- trainControl(method = "cv", number=10)
```

```r
set.seed(143)
svm_Linear <- train(RESPONSE ~., data = German_Credit.tr.subs, method = "svmLinear",
                    trControl=German_Credit.trctrl)
svm_Linear
```

```
## 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).

```r
grid <- expand.grid(C = c(0.01, 0.1, 1, 10, 100, 1000))
grid
```

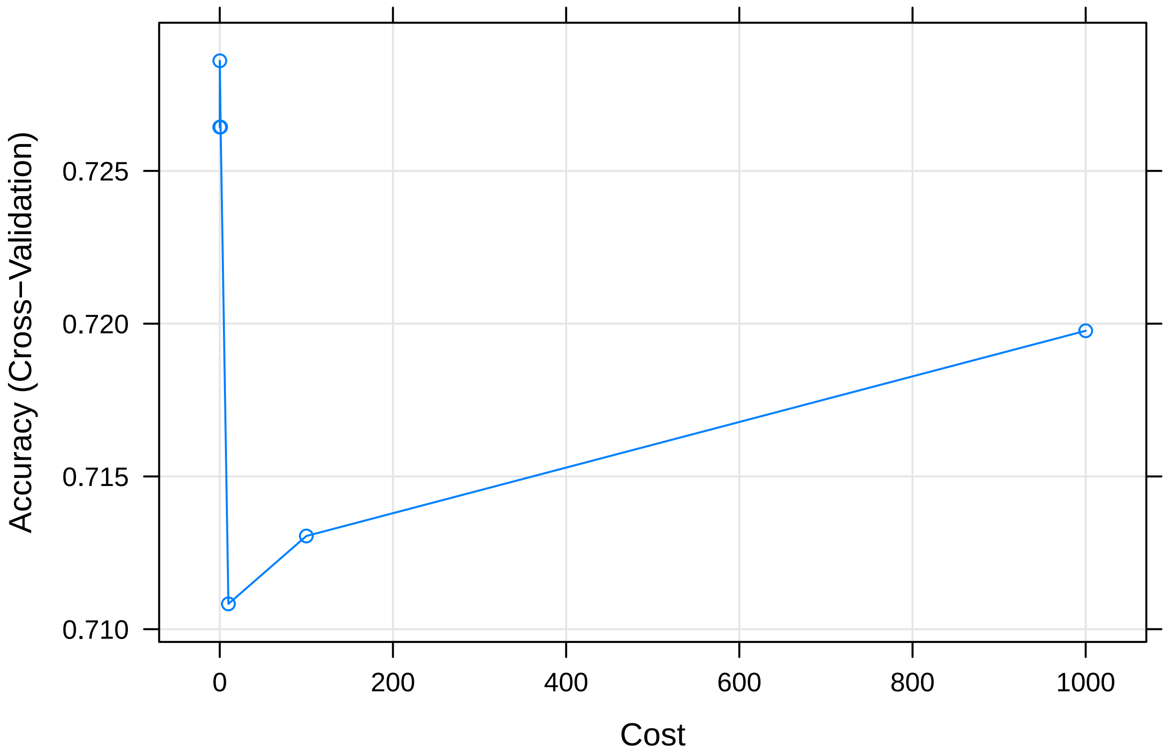
```
##       C
## 1 1e-02
## 2 1e-01
## 3 1e+00
## 4 1e+01
## 5 1e+02
## 6 1e+03
```

```r
set.seed(143)
svm_Linear_Grid <- train(RESPONSE ~., data = German_Credit.tr.subs,
                         method = "svmLinear",
                         trControl=German_Credit.trctrl,
                         tuneGrid = grid)
svm_Linear_Grid
```

```
## 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.
```

```r
plot(svm_Linear_Grid)
```




```r
svm_Linear_Grid$bestTune
```

```
##     C
## 2 0.1
```

Tunning the hyperparameters of Radial basis SVM

```r
grid_radial <- expand.grid(sigma = c(0.01, 0.02, 0.05, 0.1),
                           C = c(1, 10, 100, 500, 1000))
grid_radial
```

```
##    sigma    C
## 1   0.01    1
## 2   0.02    1
## 3   0.05    1
## 4   0.10    1
## 5   0.01   10
## 6   0.02   10
## 7   0.05   10
## 8   0.10   10
```

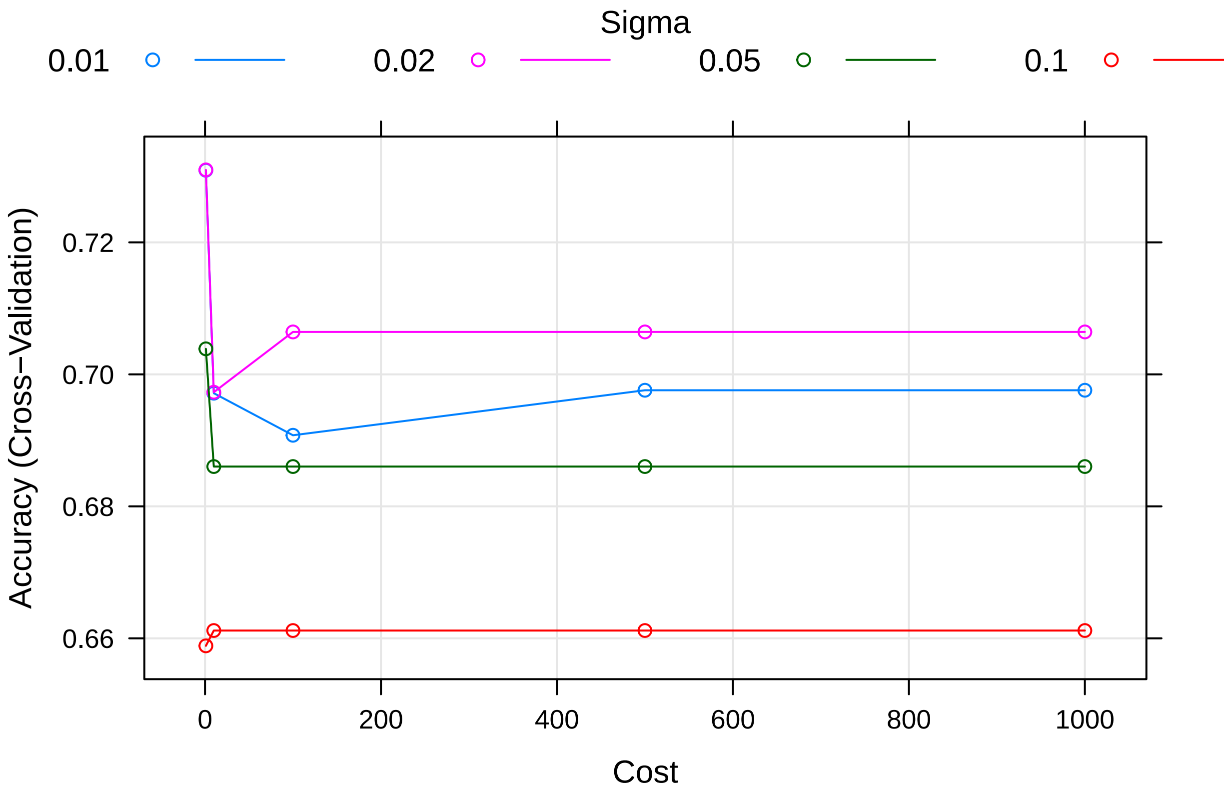
```
## 9   0.01  100
## 10  0.02  100
## 11  0.05  100
## 12  0.10  100
## 13  0.01  500
## 14  0.02  500
## 15  0.05  500
## 16  0.10  500
## 17  0.01 1000
## 18  0.02 1000
## 19  0.05 1000
## 20  0.10 1000
```

```r
set.seed(143)

svm_Radial_Grid <- train(RESPONSE ~., data = German_Credit.tr.subs,
                         method = "svmRadial",
                         trControl=German_Credit.trctrl,
                         tuneGrid = grid_radial)

svm_Radial_Grid
```

```
## 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.
```

```r
plot(svm_Radial_Grid)
```




```r
svm_Radial_Grid$bestTune
```

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

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

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

confusionMatrix(data=German_credit.rb.tuned.pred,
                reference = German_credit.te$RESPONSE)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  54  53
##          1  23 120
##
##                Accuracy : 0.696
##                  95% CI : (0.6349, 0.7524)
##     No Information Rate : 0.692
```

```
##     P-Value [Acc > NIR] : 0.4761878
##
##                   Kappa : 0.3564
##
##  Mcnemar's Test P-Value : 0.0008794
##
##             Sensitivity : 0.7013
##             Specificity : 0.6936
##          Pos Pred Value : 0.5047
##          Neg Pred Value : 0.8392
##              Prevalence : 0.3080
##          Detection Rate : 0.2160
##    Detection Prevalence : 0.4280
##       Balanced Accuracy : 0.6975
##
##        'Positive' Class : 0
##
```

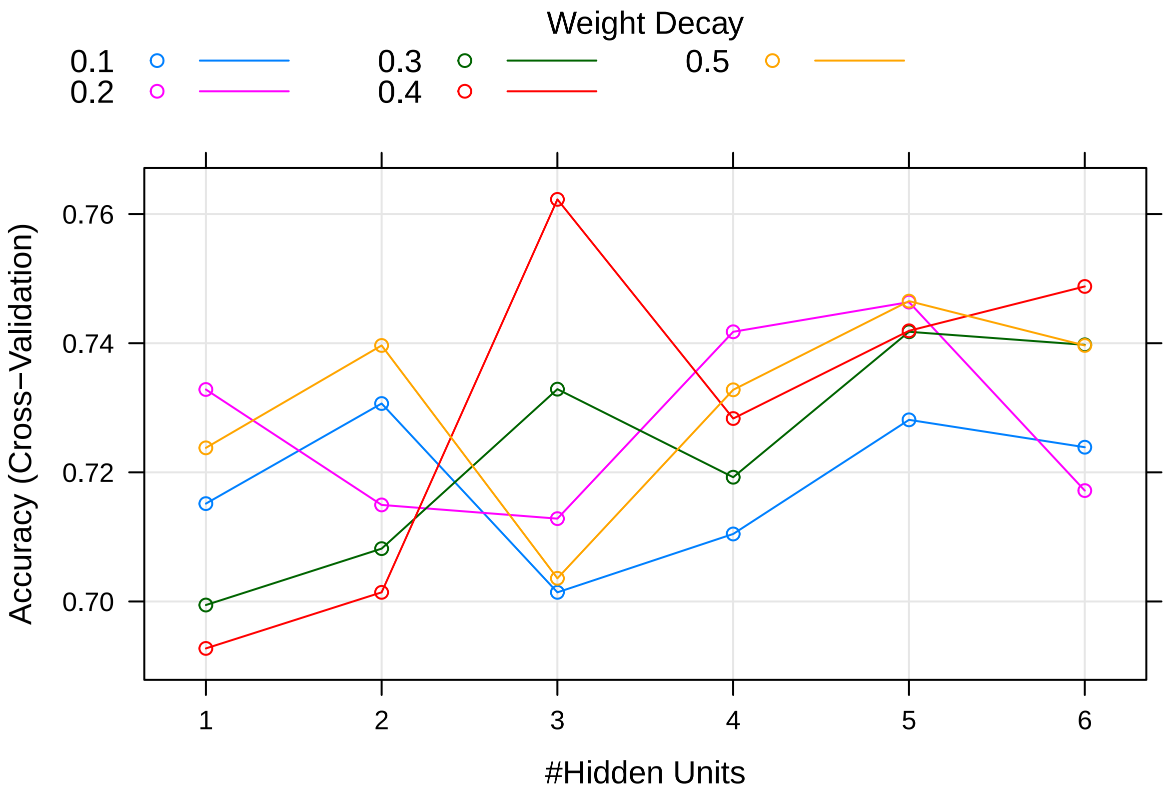
We have an accuracy of 0.696 and a balanced accuracy of 0.6975.

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.

```r
set.seed(1)
fitControl <- trainControl(method = "cv",
                           number = 10)

nnetGrid <-  expand.grid(size = seq(from = 1, to = 6, by = 1),
                        decay = seq(from = 0.1, to = 0.5, by = 0.1))

nnetFit <- train(RESPONSE ~ .,
                 data = German_Credit.tr.subs,
                 method = "nnet",
                 metric = "Accuracy",
                 tuneGrid = nnetGrid,
                 trControl = fitControl)
```

```r
plot(nnetFit)
```



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

```
set.seed(345)
nn4 <- nnet(RESPONSE ~ ., data=German_Credit.tr.subs, size=3, decay = 0.4)
```

```
## # weights:  142
## initial  value 319.952116
## iter  10 value 308.907473
## iter  20 value 304.936051
## iter  30 value 282.481326
## iter  40 value 232.152300
## iter  50 value 220.473107
## iter  60 value 219.575976
## iter  70 value 218.752380
## iter  80 value 218.532186
## iter  90 value 218.306805
## iter 100 value 217.488600
## final  value 217.488600
## stopped after 100 iterations
```

```
nn4_pred <- predict(nn4, type="class")
tab4 <- table(Obs=German_Credit.tr.subs$RESPONSE, Pred=nn4_pred) # confusion matrix
tab4
```

```
##    Pred
## Obs   0   1
##   0 182  41
##   1  41 182
```

```
(acc4 <- sum(diag(tab4))/sum(tab4)) # accuracy
```

```
## [1] 0.8161435
```

The accuracy of the neural network is of 81.61%

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.

```
Gradient_boost.y_train <- as.integer(German_Credit.tr.subs$RESPONSE)-1
Gradient_boost.y_test <- as.integer(German_credit.te$RESPONSE)-1

Gradient_boost.X_train <- sparse.model.matrix(RESPONSE ~ .-1,
                                              data = German_Credit.tr.subs )
Gradient_boost.X_test <- sparse.model.matrix(RESPONSE ~ .-1,
                                             data = German_credit.te )
```

```
set.seed(123)
xgb_train <- xgb.DMatrix(data = Gradient_boost.X_train,
                         label = Gradient_boost.y_train)

xgb_test <- xgb.DMatrix(data = Gradient_boost.X_test,
                        label = Gradient_boost.y_test)

xgb_params <- list(
  booster = "gbtree",
  eta = 0.01,
  max_depth = 8,
  gamma = 4,
  subsample = 0.75,
  colsample_bytree = 1,
  objective = "multi:softmax",
  eval_metric = "mlogloss",
  num_class = 2
  )

xgb_model <- xgb.train(
  params = xgb_params,
  data = xgb_train,
  nrounds = 5000,
  verbose = 1
  )

xgb_model
```

```
## ##### 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
```

```r
xgb_preds <- predict(xgb_model, Gradient_boost.X_test, reshape = TRUE)

# confusion matrix
xgb_tab <- table(Obs=Gradient_boost.y_test, Pred=xgb_preds)
xgb_tab
```

```
##    Pred
## Obs   0   1
##   0  57  20
##   1  59 114
```

```r
# accuracy
(xgb_acc <- sum(diag(xgb_tab))/sum(xgb_tab))
```

```
## [1] 0.684
```

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

```r
confusionMatrix(data=as.factor(xgb_preds),
                reference = as.factor(Gradient_boost.y_test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  57  59
##          1  20 114
##
##                Accuracy : 0.684
##                  95% CI : (0.6224, 0.7411)
##     No Information Rate : 0.692
##     P-Value [Acc > NIR] : 0.6369
##
##                   Kappa : 0.35
##
##  Mcnemar's Test P-Value : 1.909e-05
##
##             Sensitivity : 0.7403
##             Specificity : 0.6590
##          Pos Pred Value : 0.4914
##          Neg Pred Value : 0.8507
##              Prevalence : 0.3080
##          Detection Rate : 0.2280
##    Detection Prevalence : 0.4640
##       Balanced Accuracy : 0.6996
##
##        'Positive' Class : 0
##
```

Next Analysis

- Balance the data using either method.
- Then, using **caret** and either CV or Bootstrap, put several models in competition.
- Select the best one according to the choice of score.

- Finally, use the test set to see if the best model does not overfit the training set.

By doing this, we will have achieved a complete supervised learning task from A to Z.

Cross-validation with caret

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

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

```
German_Credit.trctrl <- trainControl(method = "cv", number=10)
```

```
German.Credit.cv <- train(RESPONSE ~., data = German_credit.tr,
                method = "glmStepAIC",
                family="binomial",
                trControl=German_Credit.trctrl, trace=0)
German.Credit.cv
```

```
## 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, 674, 675, 675, 675, 674, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.746834  0.3576738
```

```
German_Credit.pred <- predict(German.Credit.cv, newdata = German_credit.te)

confusionMatrix(data=German_Credit.pred, reference = German_credit.te$RESPONSE)
```

```
## 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

```
trctrl <- trainControl(method = "boot632", number=10)
German_credit.boot <- train(RESPONSE ~., data = German_credit.tr,
                            method = "glmStepAIC", family="binomial",
                            trControl=trctrl, trace = 0)

German_credit.boot
```

```
## 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
##   0.7609583  0.4037715
```