German_credit

Michele Pesce

16 giugno 2019

Contents

1 Overview

The project is based on the study of the dataset **German credit data**, it has been downloaded from the UCI machine learning repository at **https://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29** It consists of a data text file containing 1k records and twentyone attributes and has been converted in a dataframe named *credit_original*. At a first sight there are factors and numbers, all the factors are coded data as "Axxx" format whose meanings are explained in a description text file "german.doc" located at the same repository; from the same file have been extracted the metadata and promptly converted in variables names. In the next section the activity of conversion/replacement of dataset is shown. This project relies on a typical binary classification problem: there is only a label "credit_response" whose possible values are "good" and "bad" which refers to a bank customer who asked for a credit. The bank board would wish to know in advance who would be good an who would be bad, on the basis of customer attributes. The target is to study and evalute best approaches to predict with high reliability the customer category on the base of attributes/factors after data manipulation/trasformation and a set of analysis tasks. This document reports the following sections:

- **Data cleaning** In this section, the data integrity is explored, and possibly executed all the operations to make the dataset assessable for further analysis.
- **Data exploration** Variables in the dataset are explored and checked in order to choose the relevant factors that could be used as predictors for the target label.
- **The modeling approach** ML algorithms that best fit the kind of problem are chosen to predict the outcome, on the basis of the predictors assesed on the data exploration section. Usually, more than an ML model is checked and tuned and chosen the one that best fits the business requirement.
- **Results** The ML outcomes are compared, and evaluated pros and cons of the different approaches.
- **Conclusions** To summary the operations, and take in consideration further different approaches that could improve the project and the results.

2 Data Cleaning

The first operation that have to be done on data is to check the presence of the inconsistent values that could affect the correctness of analysis and bias the ML model results. Generally, if we don't take in count incoherent values, mean and sd could be affected by bias, all the further analisys would be affected by some kind of error, and prediction would fail. First of all, the presence of NA's must be checked. Fortunately, the data frame does not contain NA's values as shown in the following R code, so all the metrics are calculated on true data.

```
#exploring NA'S

sum(is.na(credit_original))
```

```
## [1] 0
```

```
sum(complete.cases(credit_original))
```

```
## [1] 1000
```

```
glimpse(credit_original)
```

```
## Observations: 1,000
## Variables: 21
## $ checking_account  <fct> A11, A12, A14, A11, A11, A14, A14, A12, A14,...
## $ credit_duration   <int> 6, 48, 12, 42, 24, 36, 24, 36, 12, 30, 12, 4...
## $ Credit_history    <fct> A34, A32, A34, A32, A33, A32, A32, A32, A32,...
## $ purpose           <fct> A43, A43, A46, A42, A40, A46, A42, A41, A43,...
## $ credit_amount     <int> 1169, 5951, 2096, 7882, 4870, 9055, 2835, 69...
## $ savings_account   <fct> A65, A61, A61, A61, A61, A65, A63, A61, A64,...
## $ employment_since  <fct> A75, A73, A74, A74, A73, A73, A75, A73, A74,...
## $ percentage_income <int> 4, 2, 2, 2, 3, 2, 3, 2, 2, 4, 3, 3, 1, 4, 2,...
## $ personal_status   <fct> A93, A92, A93, A93, A93, A93, A93, A93, A91,...
## $ other_guarantors  <fct> A101, A101, A101, A103, A101, A101, A101, A1...
## $ residence         <int> 4, 2, 3, 4, 4, 4, 4, 2, 4, 2, 1, 4, 1, 4, 4,...
## $ property          <fct> A121, A121, A121, A122, A124, A124, A122, A1...
## $ age               <int> 67, 22, 49, 45, 53, 35, 53, 35, 61, 28, 25, ...
## $ other_plans       <fct> A143, A143, A143, A143, A143, A143, A143, A1...
## $ housing           <fct> A152, A152, A152, A153, A153, A153, A152, A1...
## $ existing_credits  <int> 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1,...
## $ job               <fct> A173, A173, A172, A173, A173, A172, A173, A1...
## $ house_manteinant  <int> 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ telephone         <fct> A192, A191, A191, A191, A191, A192, A191, A1...
## $ foreign_worker    <fct> A201, A201, A201, A201, A201, A201, A201, A2...
## $ credit_response   <int> 1, 2, 1, 1, 2, 1, 1, 1, 1, 2, 2, 2, 1, 2, 1,...
```

The next activity arranges data to get them more meaningful. The *credit_original* and the *key_map* dataframes are used as input for computing a new dataframe *credit_clear* whose values replace the coded "Axxx" values of the original one. The *german.txt* file has been made as a slightly transformation of the *german.doc* file. The *key_map* is extracted from the *german.txt*, that is a kind of hash-map with two variables: the keys as "Axxx" format and the values as meaningful descriptions. Next, the glimpse of *key_map* and *credit_clear* dataframes.

```
##   KEY               VALUE
## 1 A11              debtor
## 2 A12              enough
## 3 A13                good
## 4 A14          no_account
## 5 A30    no_credits_taken
## 6 A31 past_paid_back_duly
```

```
##                                        *key_map head*
## Observations: 1,000
## Variables: 21
## $ checking_account  <chr> "debtor", "enough", "no_account", "debtor", ...
## $ credit_duration   <int> 6, 48, 12, 42, 24, 36, 24, 36, 12, 30, 12, 4...
## $ Credit_history    <chr> "critical_account", "existing_paid_duly", "c...
## $ purpose           <chr> "radio/television", "radio/television", "edu...
## $ credit_amount     <int> 1169, 5951, 2096, 7882, 4870, 9055, 2835, 69...
## $ savings_account   <chr> "unknown/no_savings", "poor", "poor", "poor"...
## $ employment_since  <chr> "moreThan7years", "1to4years", "4to7years", ...
## $ percentage_income <int> 4, 2, 2, 2, 3, 2, 3, 2, 2, 4, 3, 3, 1, 4, 2,...
## $ personal_status   <chr> "male_single", "female_divorced/separated/ma...
## $ other_guarantors  <chr> "none", "none", "none", "guarantor", "none",...
## $ residence         <int> 4, 2, 3, 4, 4, 4, 4, 2, 4, 2, 1, 4, 1, 4, 4,...
## $ property          <chr> "real_estate", "real_estate", "real_estate",...
## $ age               <int> 67, 22, 49, 45, 53, 35, 53, 35, 61, 28, 25, ...
## $ other_plans       <chr> "none", "none", "none", "none", "none", "non...
## $ housing           <chr> "own", "own", "own", "for_free", "for_free",...
## $ existing_credits  <int> 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1,...
## $ job               <chr> "skilledemployee/official", "skilledemployee...
## $ house_manteinant  <int> 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ telephone         <chr> "yes", "none", "none", "none", "none", "yes"...
## $ foreign_worker    <chr> "yes", "yes", "yes", "yes", "yes", "yes", "y...
## $ credit_response   <chr> "good", "bad", "good", "good", "bad", "good"...

##                                      *credit_clear glimpse*
```

3 Data exploration

Prior to solve any ML problem, domain knowledge is essential, otherwise we will end up applying random algorithms and techniques blindly which may not give the right results. As a first step, to get a broad idea of what kind of data we are dealing with and summarize what has happened in the past, *descriptive analytics* will be applied. The different attributes of the data will be observed in order to extract meaningful features, use statistics and visualizations to understand what has already happened. The information obtained could be useful to select some predictive variables and exclude others that are not The following table shows the number of unique values for each variable

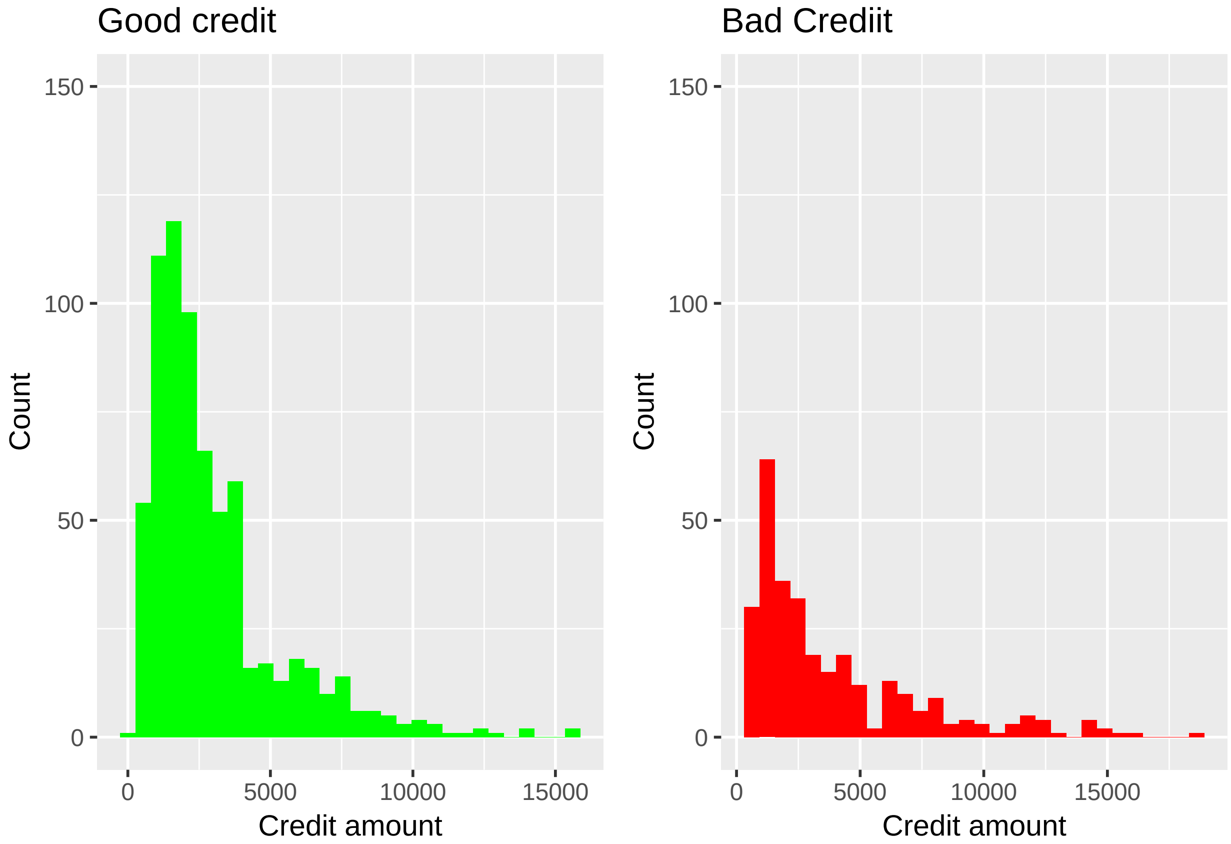
VARIABLE	UNIQUE_VAL
checking_account	4
credit_duration	33
Credit_history	5
purpose	10
credit_amount	921
savings_account	5
employment_since	5
percentage_income	4
personal_status	4
other_guarantors	3
residence	4
property	4
age	53
other_plans	3
housing	3
existing_credits	4
job	4

VARIABLE	UNIQUE_VAL
house_manteinant	2
telephone	2
foreign_worker	2
credit_response	2

The *credit_amount*, *credit_duration*, *age* variables are the numerical ones: the first concerns the credit required, it shows the most variability among all variables, the second about the credit_duration in weeks, the third is about the wide range of customer's age. The others are all categorical variables.

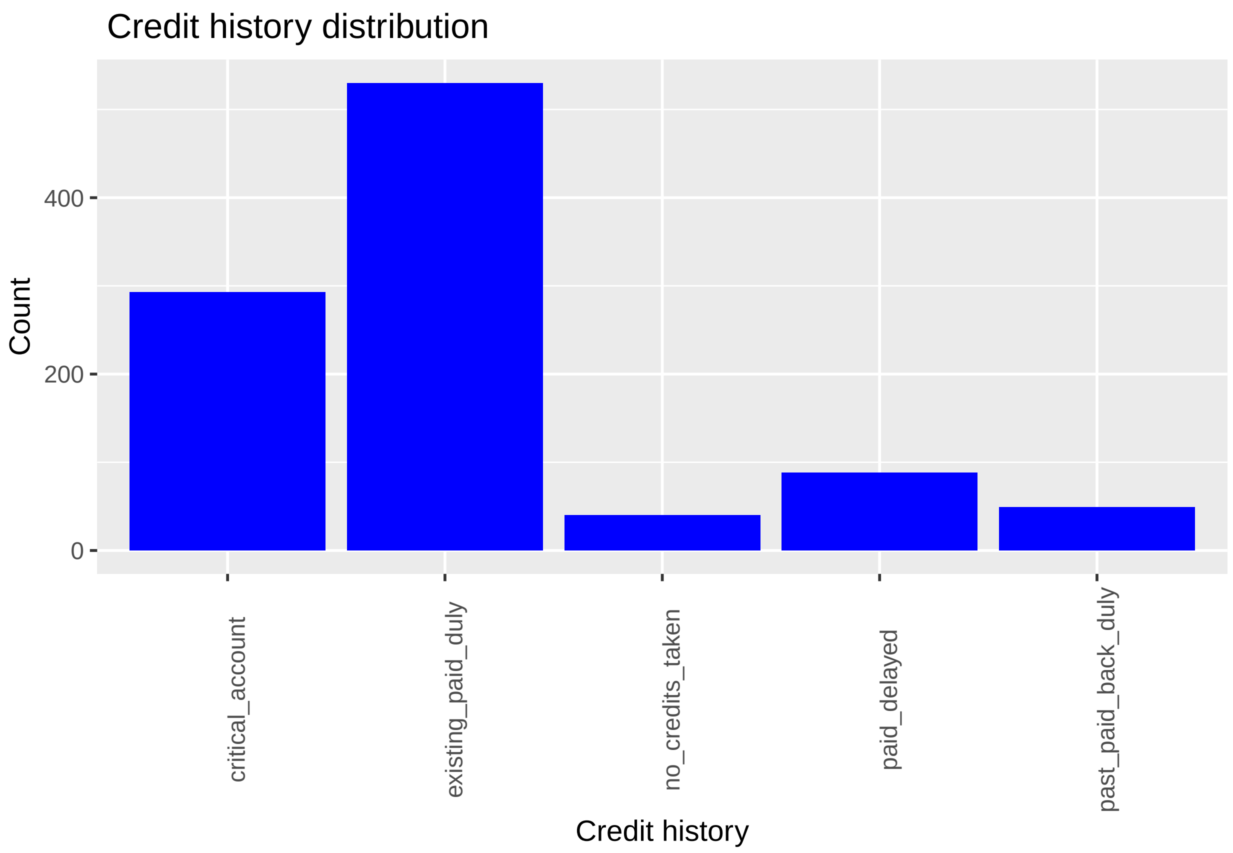
The dataset shows that "Good"" customers have 70% proportion, and bad 30%. This is a first simple hint of predicted proportion

The next plot shows the *credit amount* factor vs credit response, split in two parts: the good credit and the bad credit. As shown, the credit response does not seem to rely on credit amount, the plots look like quite similar. Most of applicants ask for low amount, i.e. the 75% is lower than 3972.25 DM and the average is 3271.258 DM



```
## *
```

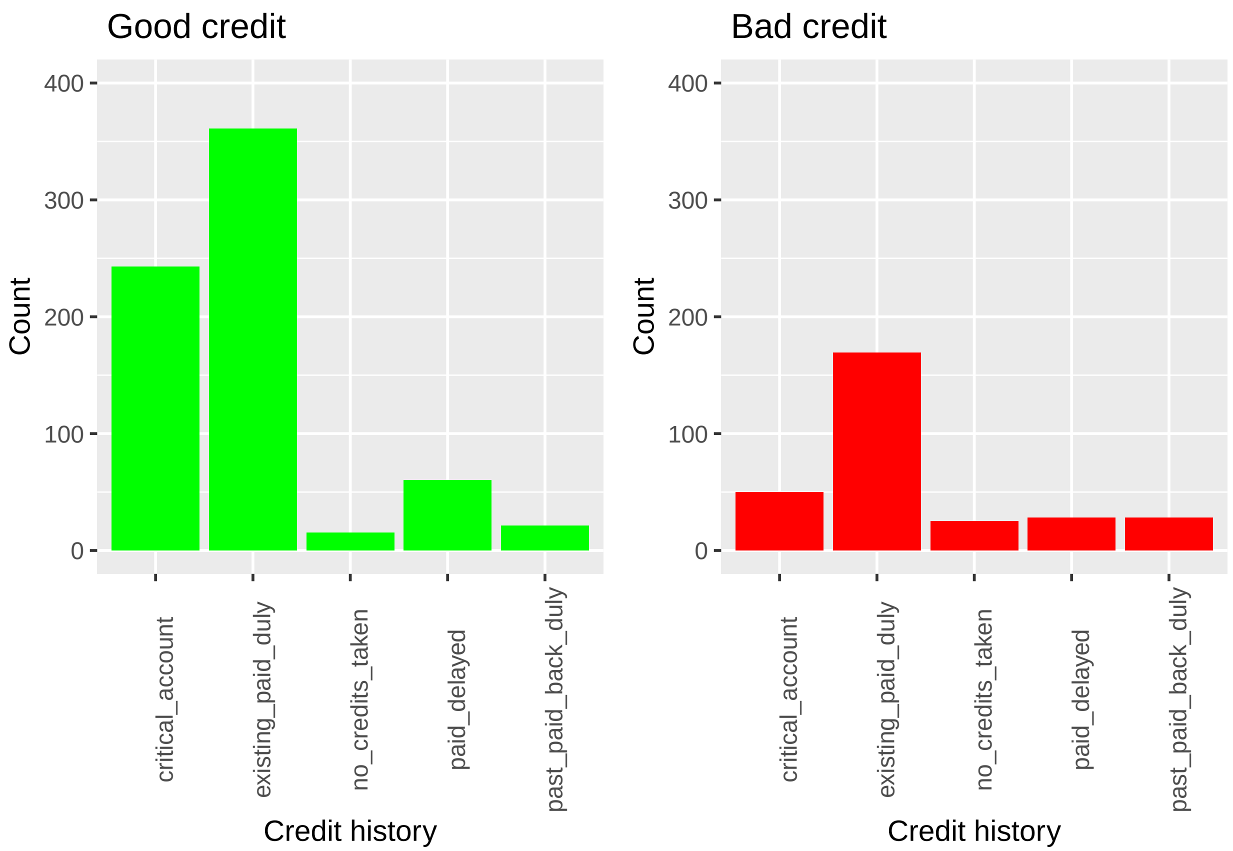
The following illustrations show graphically and as-a-table the *Credit_history* distribution and proportions. More than half are trusted customers who are reliable debtors, one-third have critical account so that they need a loan for some purchase.



Credit_history	amount	perc
critical_account	293	29.3
existing_paid_duly	530	53.0
no_credits_taken	40	4.0
paid_delayed	88	8.8
past_paid_back_duly	49	4.9

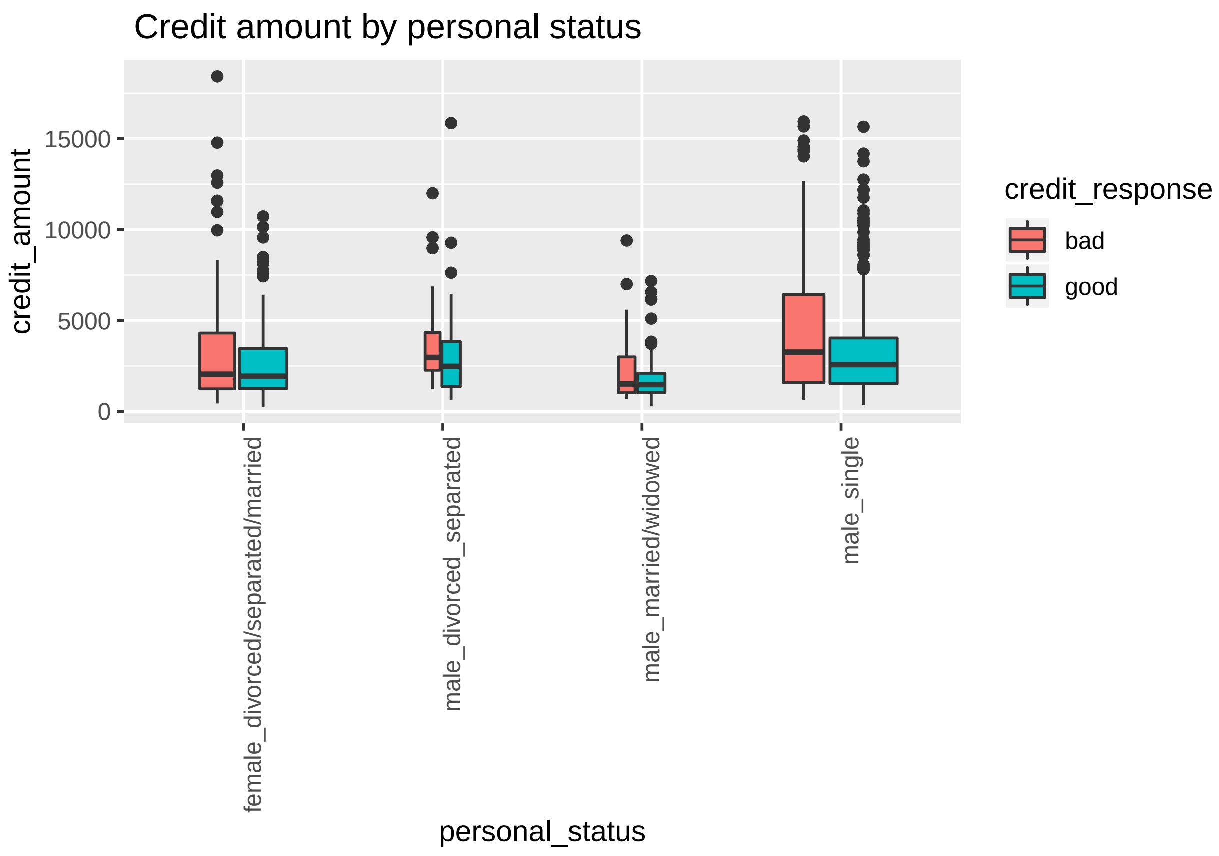
```
## *
```

The next plot show no direct relationship between *Credit_history* and credit response (the *credit_response* variable chosen as label). The two bar diagrams are quite similar and the different heights are releted to the proportion as shown in the table

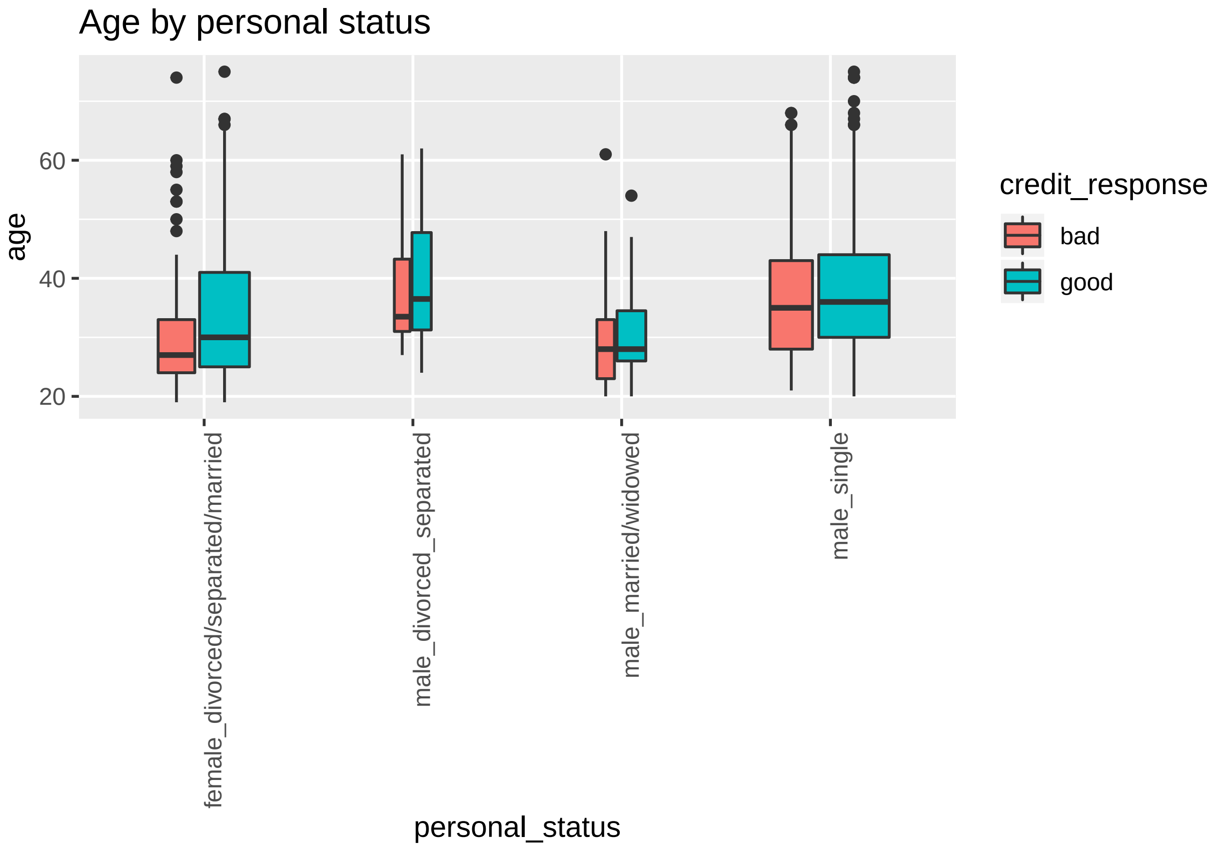


```
## *
```

The analysis next step focuses on *credit_amount*, *personal_status* and *age*. The table shows the proportion of personal status categories: more than half are male singles, the 30% are generally female. The majority of the first have bad credit, although 29 in 1000 is good for credit over 7500 DM as shown in the first plot. The plot age vs personal status does not seem of particular interest, but generally the proportion of females that has “good”" credit is higher than “bad”.



personal_status	amount	perc
female_divorced/separated/married	310	31.0
male_divorced_separated	50	5.0
male_married/widowed	92	9.2
male_single	548	54.8

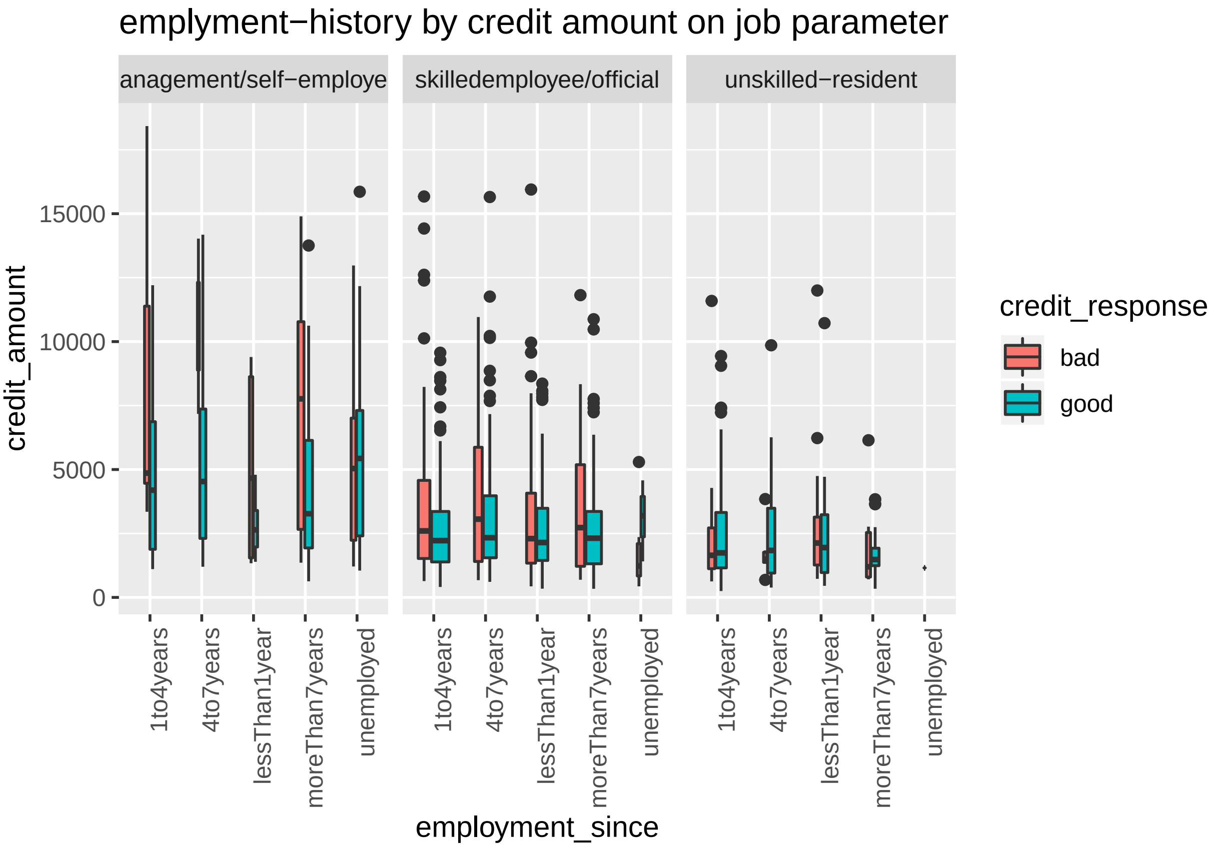


```
## *
```

Prior to start job and employment-history, two tables have been provided in order to observe proportions. It is very hard for unskilled and uemployed applicants to ask for loan, infact only 22 in 1000 are present as observation, so that these have been cut out from box-plot analysis. The data set top job is the skilled-employee, who more easly is a good credit candidate for middle/high credit amount. In proportion, managers/self-employed ask for high credit, but generally they are not considered very reliable. No way for unemployed, eventually skilled emplyees with consolidated employment history are favourite.

job	Total	perc
management/self-employed/	148	14.8
skilledemployee/official	630	63.0
unemployed/unskilled_non-resident	22	2.2
unskilled-resident	200	20.0

employment_since	Total	perc
1to4years	339	33.9
4to7years	174	17.4
lessThan1year	172	17.2
moreThan7years	253	25.3
unemployed	62	6.2



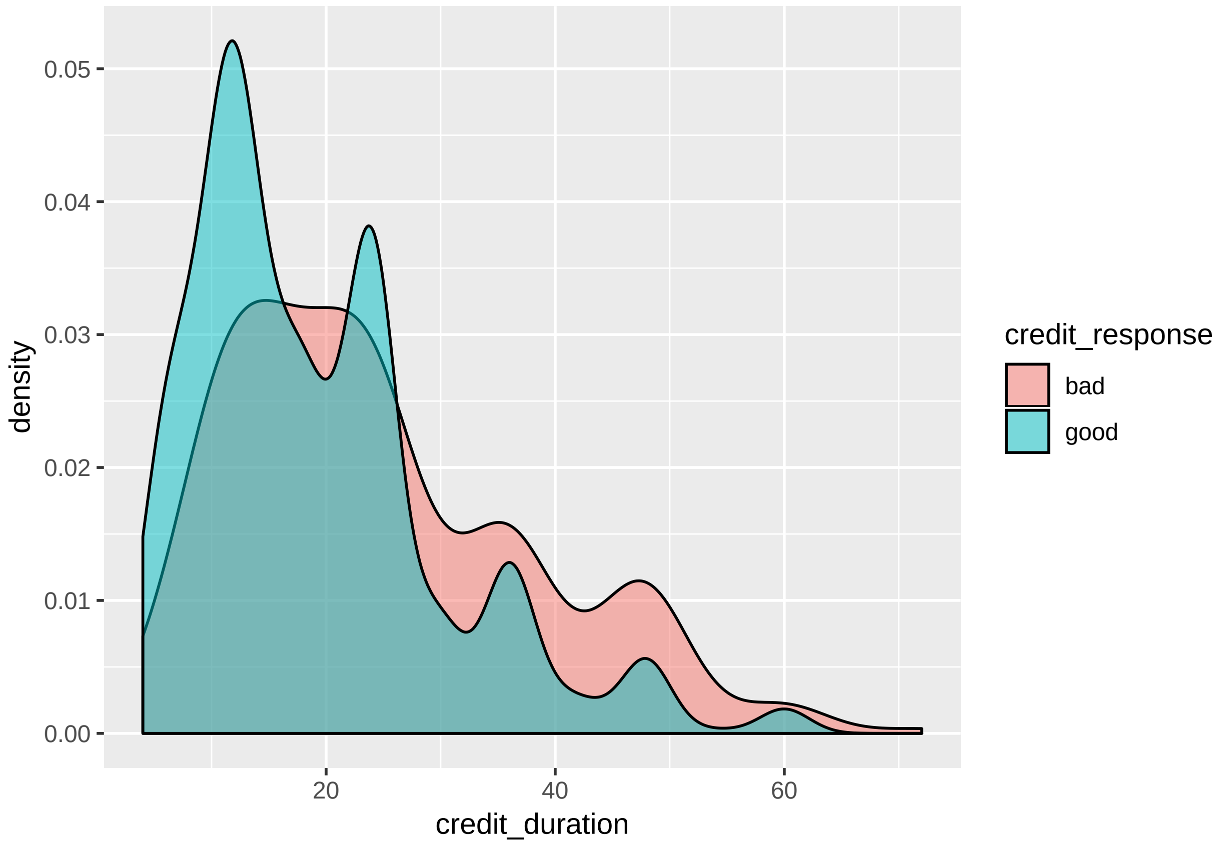
```
## *
```

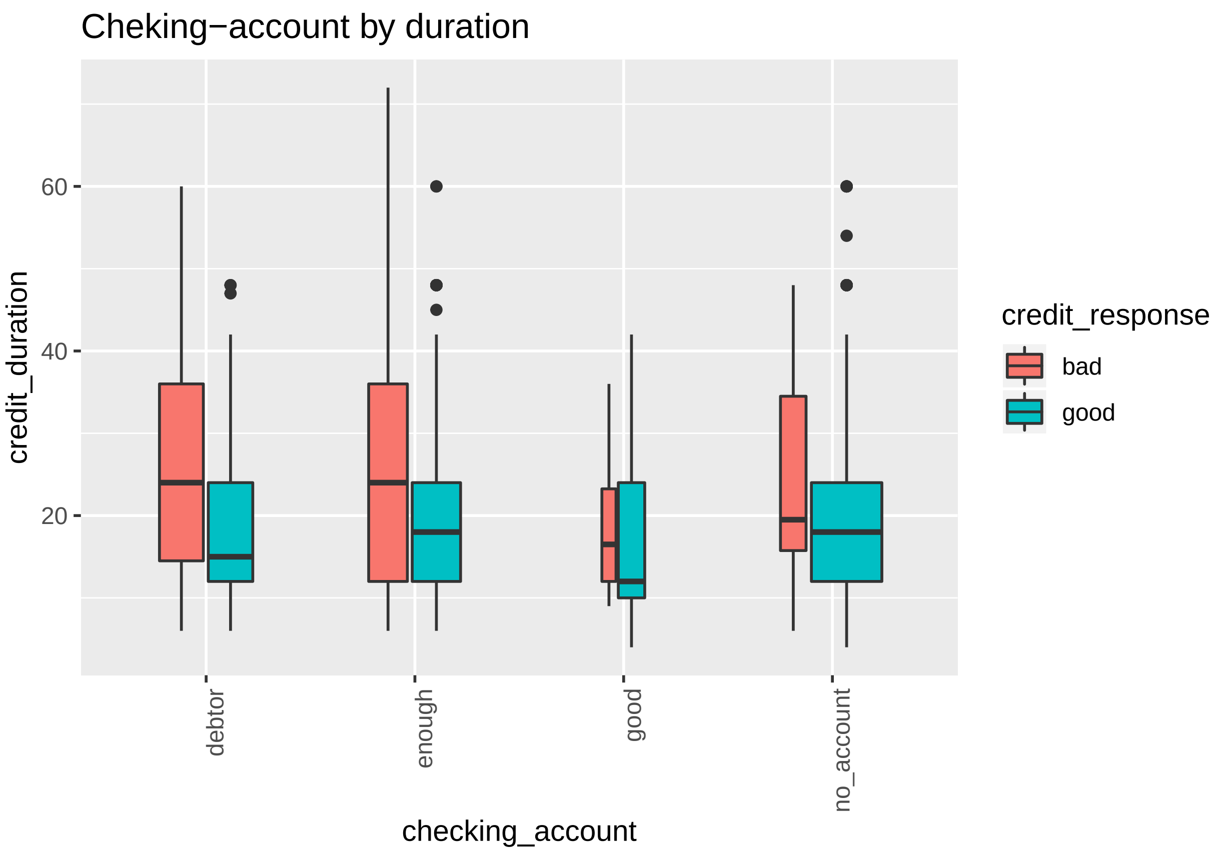
The credit-history is important to figure out the customer trust; The 53% of them are reliable applicants that pay duly existing credits; the 30% are customers with critical account either with other credits in different bank. Anyway, as shown in the table, except for a 6% that has a good bank account, the proportions are fairly partitioned among enough account, negative account, or even with no account. The following **credit duration density plot** shows that most of good credits last few weeks, whilst long time credits tend to be not so good. Looking at the **Cheking-account VS credit_duration** plot, except for customers with good account, the bank consider "bad" the credit_duration over 25 weeks: i.e. debtors,thin, and no_accounts are absolutely bad, while good account is trusted as expected. The plot with *credit_history* parameter shows that *existing_paid_duly* customers are the most reliable. Finally, in the last plot **Credit amount by Cheking-account** high *credit_amount* is approved for some with no account, but generally -if not with a good account- bad credit response is prevalent.

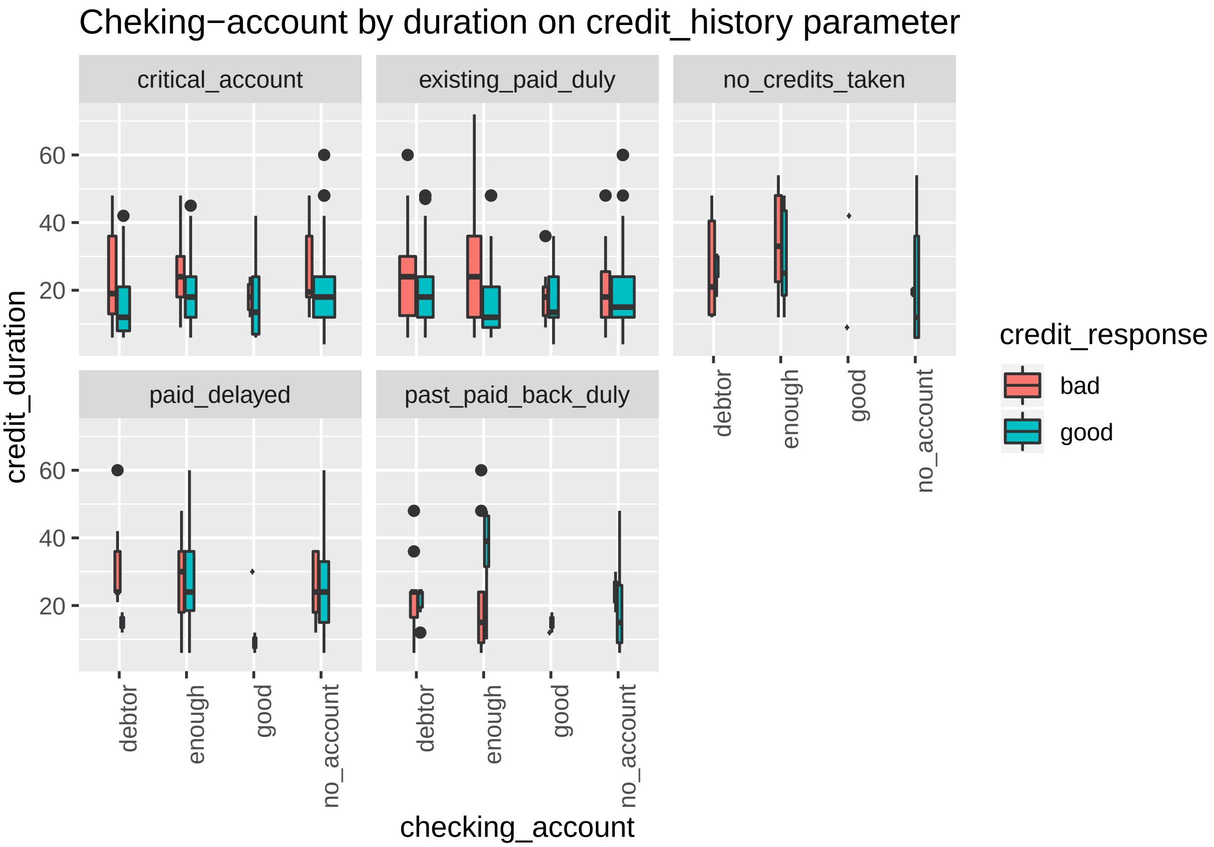
Credit_history	Total	perc
critical_account	293	29.3
existing_paid_duly	530	53.0
no_credits_taken	40	4.0
paid_delayed	88	8.8
past_paid_back_duly	49	4.9

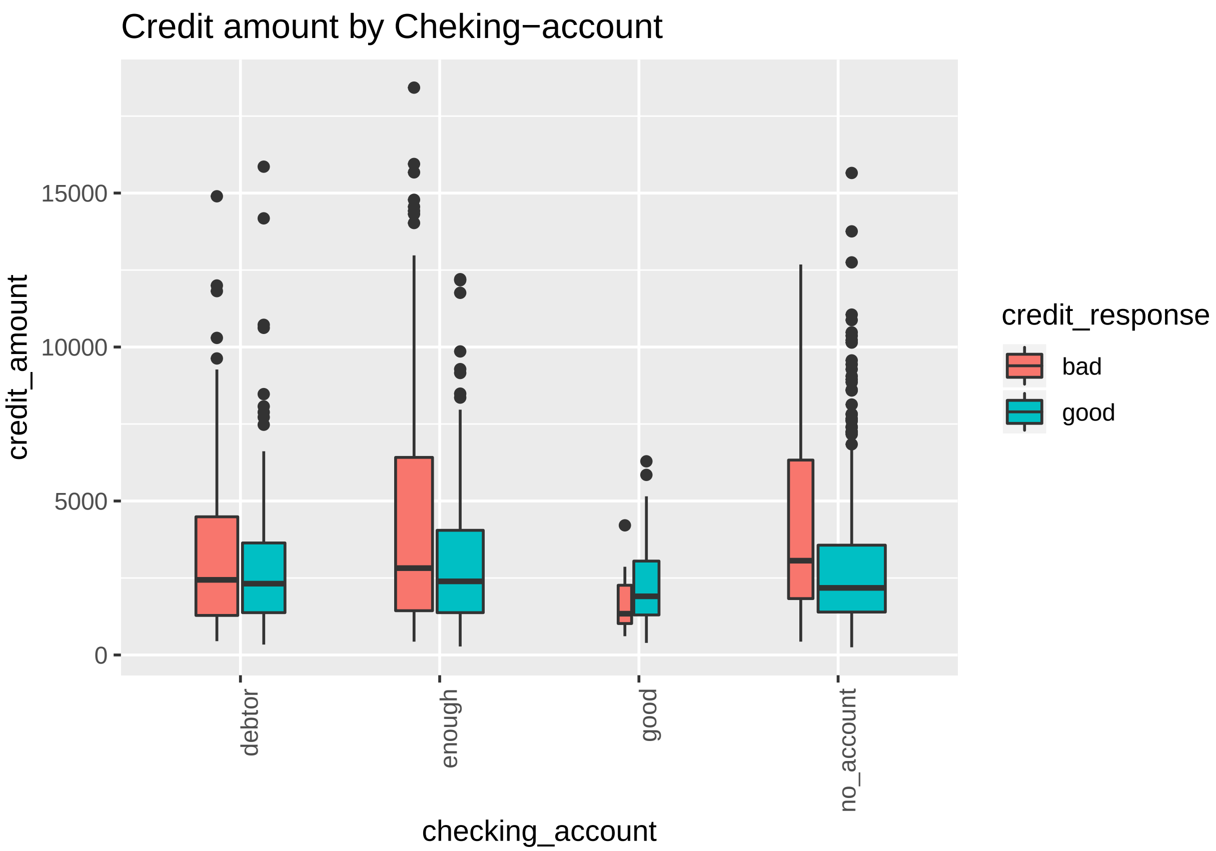
```
##                                            **********
##                                            **********
##                                            **********
```

checking_account	Total	perc
debtor	274	27.4
enough	269	26.9
good	63	6.3
no_account	394	39.4




Cheking−account by duration
credit_duration
60
40
20
debtor
enough
good
no_account
checking_account
credit_response
bad
good


Cheking−account by duration on credit_history parameter
critical_account
existing_paid_duly
no_credits_taken
paid_delayed
past_paid_back_duly
credit_duration
60
40
20
checking_account
debtor
enough
good
no_account
credit_response
bad
good



As analysis last step, other meaningful attributes vs *credit_amount* have been illustrated. The *saving_accounts* proportion table shows that poors are the absolute majority: it is quite normal that who has low savings (and possibly no other neither little property) tends to ask for a loan. The 60% of total ask loans for cars or radio/television. In the **Credit amount vs Property on saving-account parameter** the "excellent" *saving_account* value has been excluded, because it is a very few amount, also "good" and "very good" have very few applicants; poor or no-savings with no-property/unkown have favour but not for high amount. The **Credit amount vs age on saving-account parameter** plot shows a different perspective based on age: generally the most of customers are in the 20-45 age rank, anyway who has poor or no-saving could be older. Only a little proportion of poor-saving-customer gets credit over 7500 DM.On the other hand, customers with no or unknown savings have high chance to get positive classification. Seemingly, as savings increase, the proportion of good evaluation increases too regardless age; anyway the correlation between savings and credit amount is 0.034938 which is too low and it would depend, among others, on different age and saving proportions.

savings_account	Total	perc
excellent	48	4.8
good	103	10.3
poor	603	60.3
unknown/no_savings	183	18.3
very_good	63	6.3

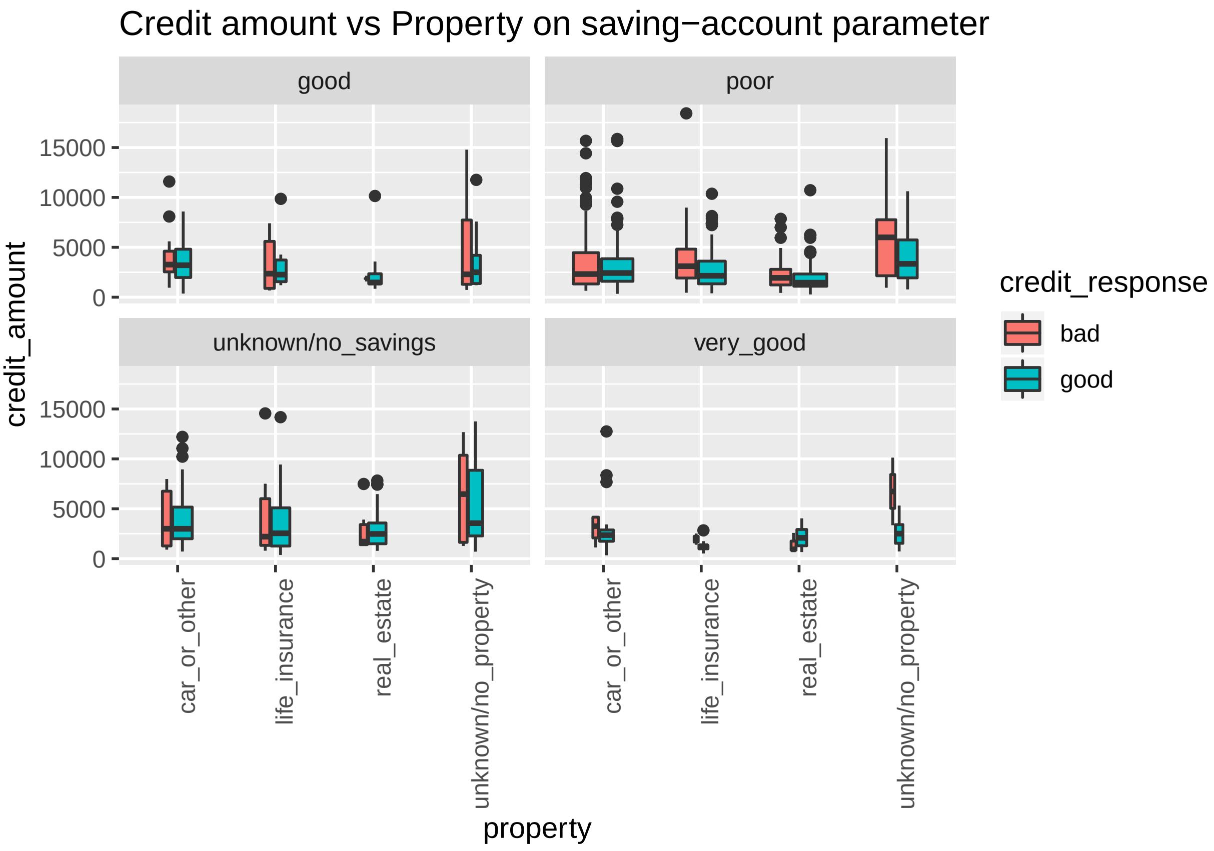
```
## **********
```

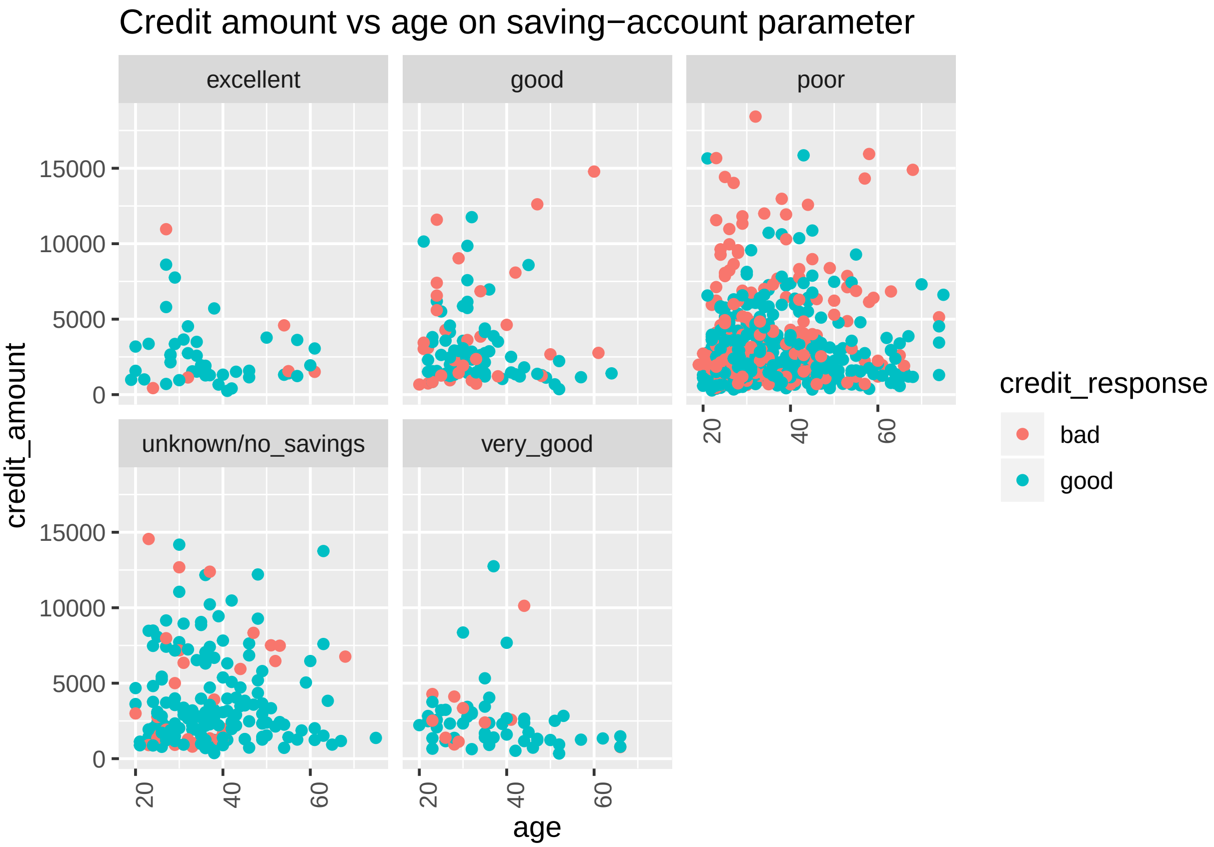
```
## **********
```

```
## **********
```

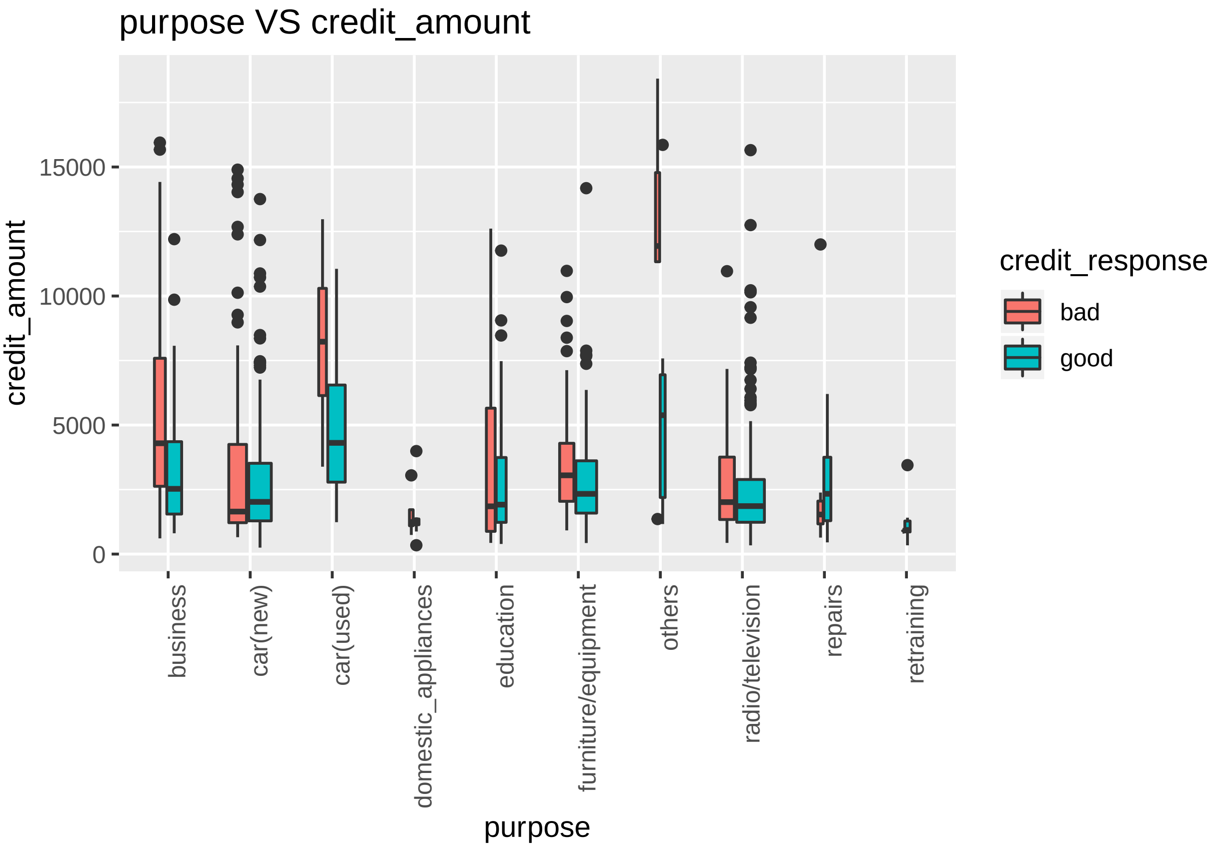
property	Total	perc
car_or_other	332	33.2
life_insurance	232	23.2
real_estate	282	28.2
unknown/no_property	154	15.4

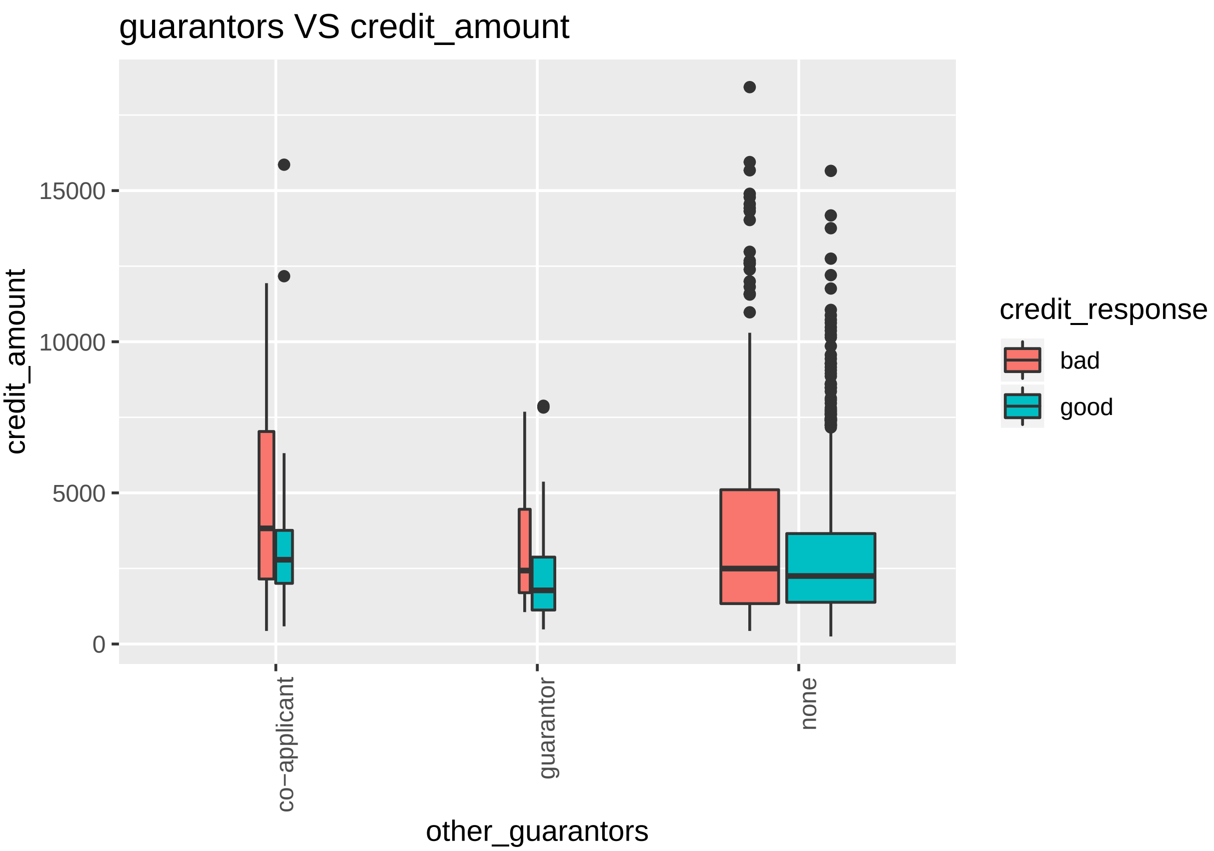
purpose	Total	perc
business	97	9.7
car(new)	234	23.4
car(used)	103	10.3
domestic_appliances	12	1.2
education	50	5.0
furniture/equipment	181	18.1
others	12	1.2
radio/television	280	28.0
repairs	22	2.2
retraining	9	0.9

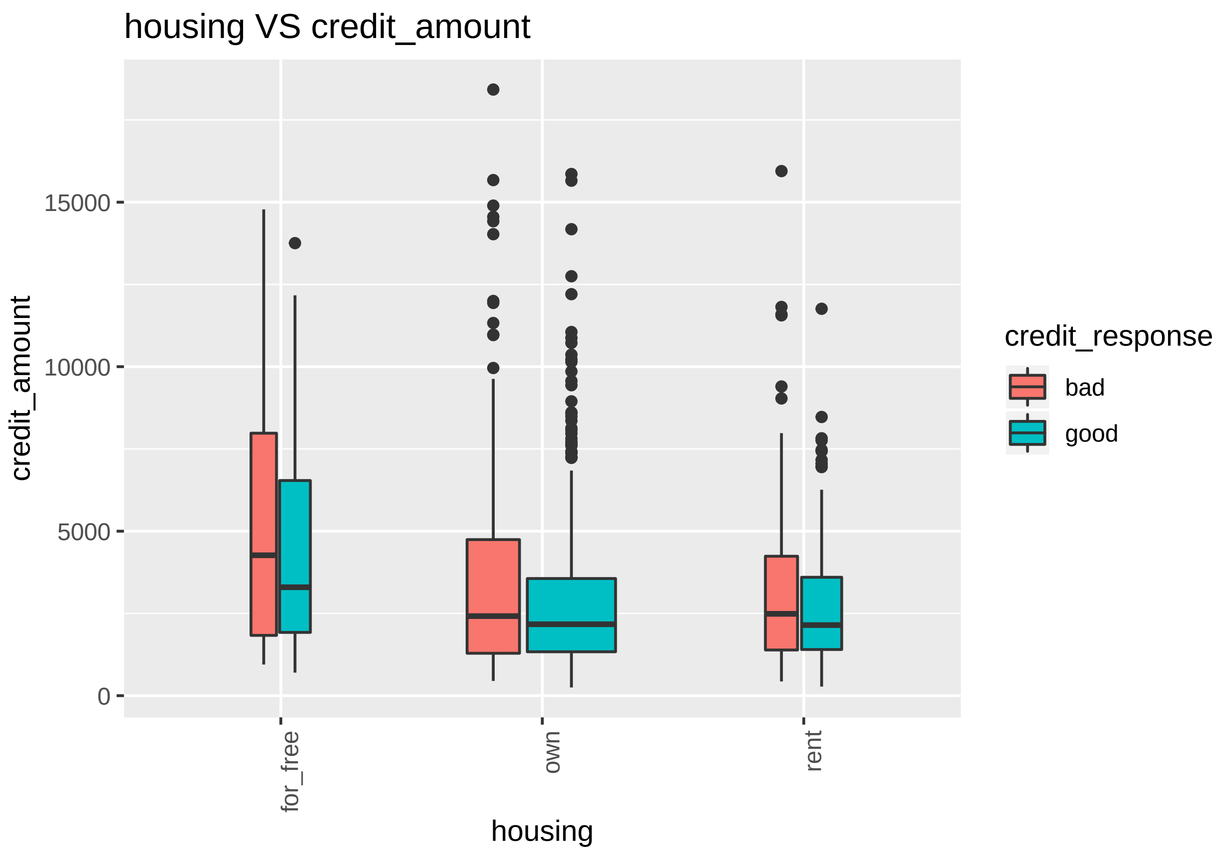




The *purpose VS credit_amount* plot shows that only for a very few cases it is possible to get a credit over 7500 DM just for radio/television or a new car; over 5000 for a used car, but never over 7500. The data are very heterogeneous, so It is very difficult to observe a general behaviour. The *guarantors VS credit_amount* does not help more, generally is better not to have other guarantors. Looking at the *housing VS credit_amount* plot it is easy to observe that who pays the rent would not ask for high credit, similarly the owner that would pay for mortgage (who would not, probably could ask for high loan). The favourite is a "for_free" customer that does not pay any rent or mortgage, even though high credit could not be ammitted.


purpose VS credit_amount
credit_amount
15000
10000
5000
0
business
car(new)
car(used)
domestic_appliances
education
furniture/equipment
others
radio/television
repairs
retraining
purpose
credit_response
bad
good


guarantors VS credit_amount
credit_amount
15000
10000
5000
0
co−applicant
guarantor
none
other_guarantors
credit_response
bad
good

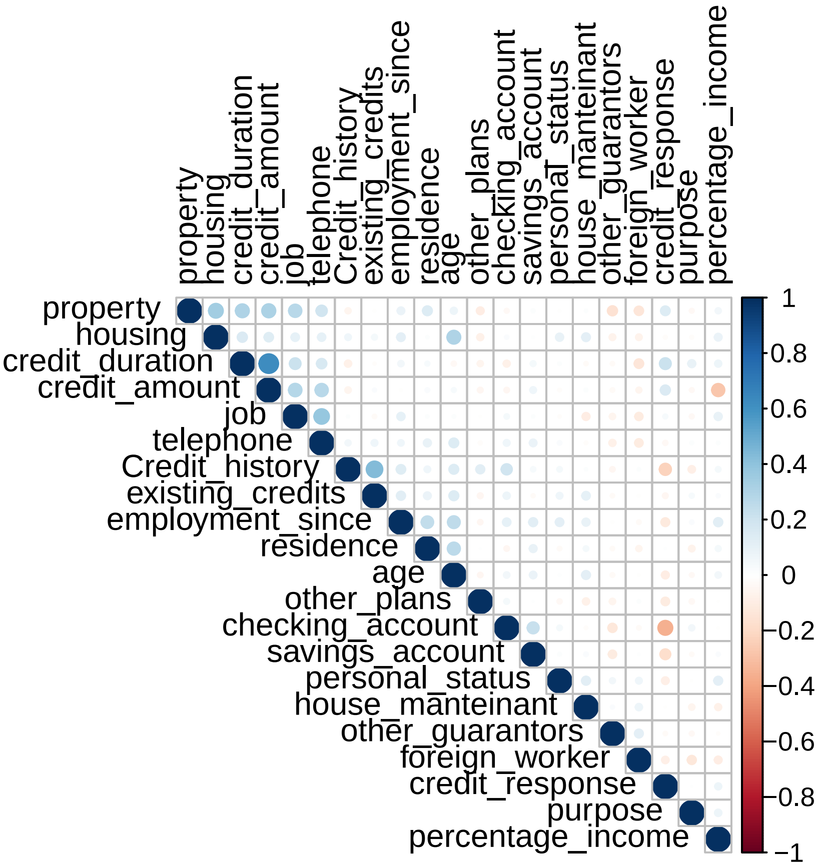


4 The modeling approach

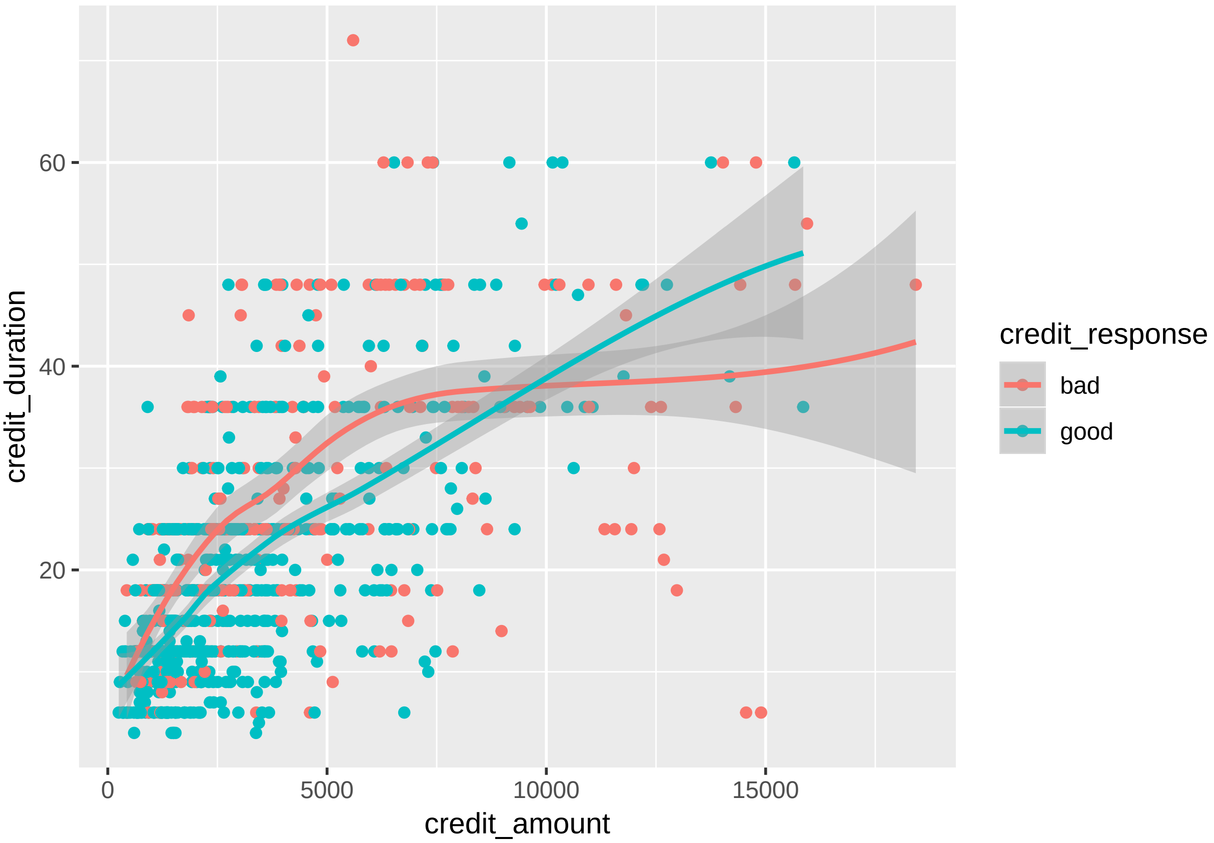
Preprocessing

At the end of data exploration, it's important to inspect the relationship between variables, if two or more of them are correlated, they would give quite same information about prediction, so we should choose only the ones with no redundant information. To apply the correlation function, The *credit_original* must be converted in df numeric, i.e. all the factor variable are transformed in numeric, so we have a *credit_num* dataframe as shown in a glimpse. The correlation illustrated graphics show correlation between variables ordered by correlation rate. That couple that appears most correlated is *duration/credit_amount* whose value is 0.6249842. Next, the related scatterplot that includes smoothing function; as shown, there is a natural straight relationship between amount and duration, the great is the credit the more are instalment (i.e. longer duration) when the credit is bad the trend looks like quite different; anyway the general trend seems to be too approximate, so that both should be considered as independent factors.

```
## Observations: 1,000
## Variables: 21
## $ checking_account  <dbl> 1, 2, 4, 1, 1, 4, 4, 2, 4, 2, 2, 1, 2, 1, 1,...
## $ credit_duration   <int> 6, 48, 12, 42, 24, 36, 24, 36, 12, 30, 12, 4...
## $ Credit_history    <dbl> 5, 3, 5, 3, 4, 3, 3, 3, 3, 5, 3, 3, 3, 5, 3,...
## $ purpose           <dbl> 5, 5, 8, 4, 1, 8, 4, 2, 5, 1, 1, 10, 5, 1, 1...
## $ credit_amount     <int> 1169, 5951, 2096, 7882, 4870, 9055, 2835, 69...
## $ savings_account   <dbl> 5, 1, 1, 1, 1, 5, 3, 1, 4, 1, 1, 1, 1, 1, 1,...
## $ employment_since  <dbl> 5, 3, 4, 4, 3, 3, 5, 3, 4, 1, 2, 2, 3, 5, 3,...
## $ percentage_income <int> 4, 2, 2, 2, 3, 2, 3, 2, 2, 4, 3, 3, 1, 4, 2,...
## $ personal_status   <dbl> 3, 2, 3, 3, 3, 3, 3, 3, 1, 4, 2, 2, 2, 3, 2,...
## $ other_guarantors  <dbl> 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ residence         <int> 4, 2, 3, 4, 4, 4, 4, 2, 4, 2, 1, 4, 1, 4, 4,...
## $ property          <dbl> 1, 1, 1, 2, 4, 4, 2, 3, 1, 3, 3, 2, 3, 3, 3,...
## $ age               <int> 67, 22, 49, 45, 53, 35, 53, 35, 61, 28, 25, ...
## $ other_plans       <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,...
## $ housing           <dbl> 2, 2, 2, 3, 3, 3, 2, 1, 2, 2, 1, 1, 2, 2, 1,...
## $ existing_credits  <int> 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1,...
## $ job               <dbl> 3, 3, 2, 3, 3, 2, 3, 4, 2, 4, 3, 3, 3, 2, 3,...
## $ house_manteinant  <int> 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ telephone         <dbl> 2, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 2, 1, 1,...
## $ foreign_worker    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ credit_response   <int> 1, 2, 1, 1, 2, 1, 1, 1, 1, 2, 2, 2, 1, 2, 1,...

##                                  *credit_num glimpse*

##                                  **
```



```
##                                  *CORRELATION MAP*

##                                  **
```



To further inspect if some variable could be cut out from predictors, caret package *nzv* function is applied. Next, the table that shows detail of function' s computation and the the 'nzv' foreign_worker variable that has to be excluded. The dataframe credit_clean, from which the nzv has been removed, is the final dataset to use for modeling approach, while the normalized *credit_num* dataset is used as input for PC Analysis.

```
##                    freqRatio percentUnique zeroVar   nzv
## checking_account    1.437956           0.4   FALSE FALSE
## credit_duration     1.027933           3.3   FALSE FALSE
## Credit_history      1.808874           0.5   FALSE FALSE
## purpose             1.196581           1.0   FALSE FALSE
## credit_amount       1.000000          92.1   FALSE FALSE
## savings_account     3.295082           0.5   FALSE FALSE
## employment_since    1.339921           0.5   FALSE FALSE
## percentage_income   2.060606           0.4   FALSE FALSE
## personal_status     1.767742           0.4   FALSE FALSE
## other_guarantors   17.442308           0.3   FALSE FALSE
## residence           1.340909           0.4   FALSE FALSE
## property            1.177305           0.4   FALSE FALSE
## age                 1.020000           5.3   FALSE FALSE
## other_plans         5.856115           0.3   FALSE FALSE
## housing             3.983240           0.3   FALSE FALSE
## existing_credits    1.900901           0.4   FALSE FALSE
## job                 3.150000           0.4   FALSE FALSE
## house_manteinant    5.451613           0.2   FALSE FALSE
## telephone           1.475248           0.2   FALSE FALSE
## foreign_worker     26.027027           0.2   FALSE  TRUE
## credit_response     2.333333           0.2   FALSE FALSE
```

The goal of *normalization* is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges. In the german credit dataset all the numeric variables are scaled and centered, while the factors have levels, that are numerical representations of classes.

The last step before applying ML models, is to partition the preprocessed dataset: the 70% of the entire datset as train set and the 30% for test set; the train has been split in the vector outcome (the *credit_response* variable as y) and the nineteen predictor variables as x

ML models

The following Machine learning algorithms have been chosen to fit train data and predict outcome on test data. These are among the most used, and fit quite well with the dataset; on the other hand the *knn* does not because it is not particularly effective on variables where there is a luck of hierarchical relationship between levels or numbers, so it has not been chosen as model.

- **GLM**
- **Random Forest**
- **Rpart**
- **Naive bayes**

To explore the variable reduction approach, in the end of section the PCA model has been also inspected.

GLM

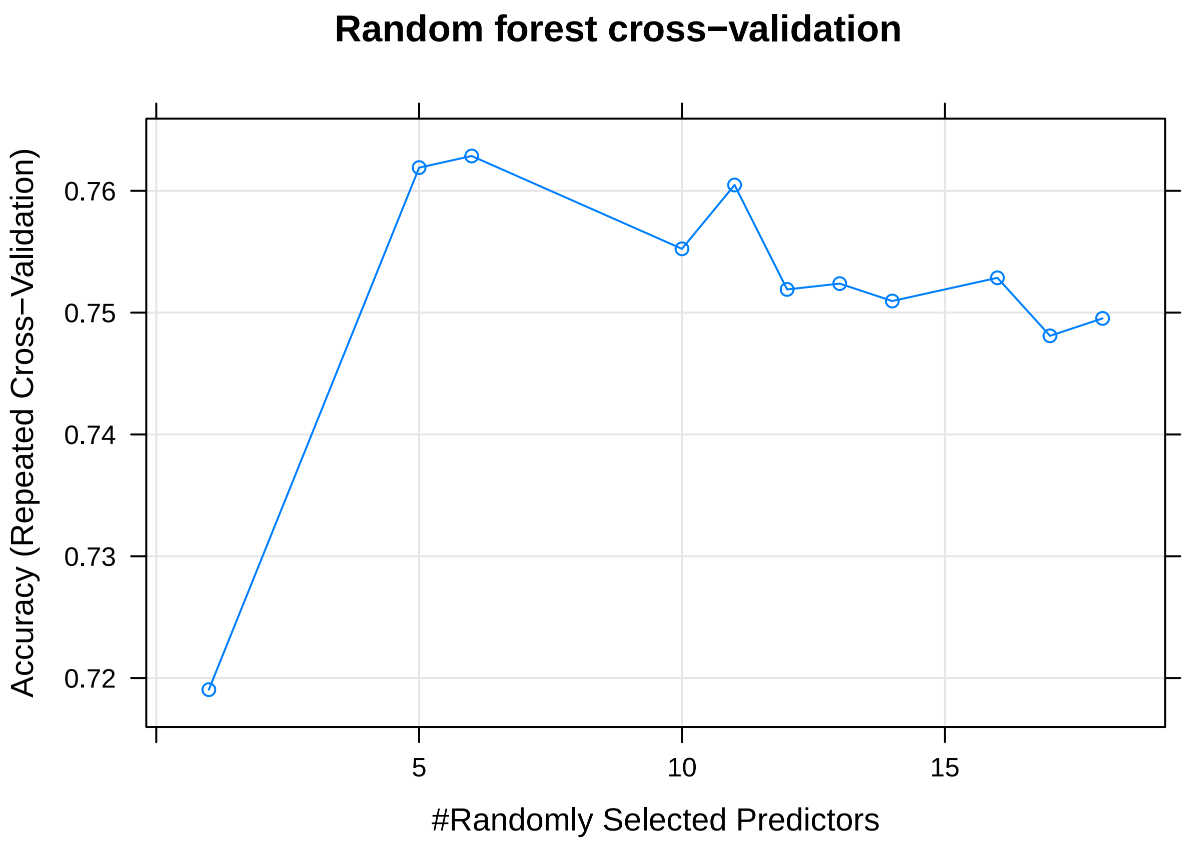
This model works by trying to find out the relationship between the class variable and the other independent feature variables by estimating probabilities. Glm does not predict classes directly but the probability of the outcome, so that for the metrics evaluation the probability is converted in outcome classes. In this case, since this is a binary classification problem, the approach is by binomial logistic regression. This model does not require cross validation because there are not parameters to tune.

Method	Accuracy	Sensitivity	Specificity
GLM	0.78	0.8952381	0.5111111

The table above shows the summarized metrics: the accuracy is not excellent. The sensitivity is good, the specificity is good too, so the FPR=0.4888889 is quite low. This approach seems to be quite satisfactory.

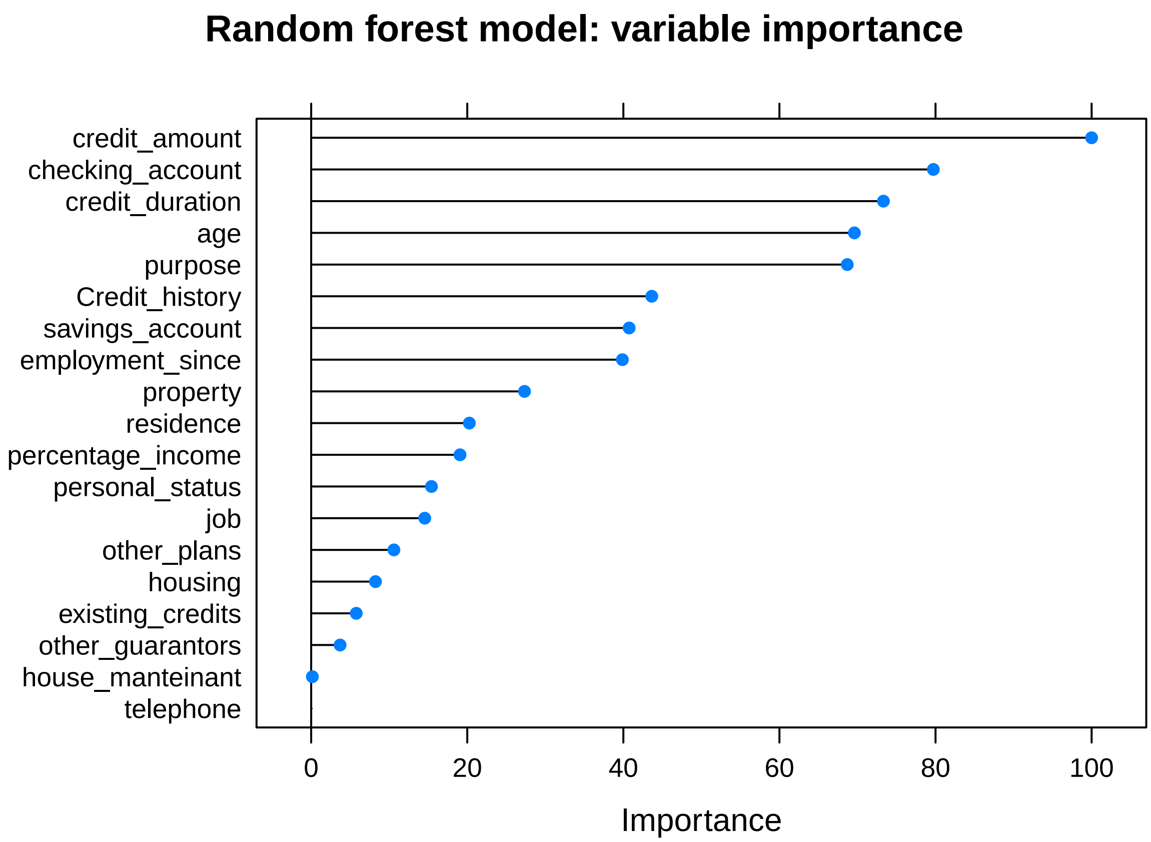
Random Forest

Random forest is a machine learning algorithm that comes from the family of ensemble learning algorithms. At any point in time,each tree in the ensemble of decision trees is built from a bootstrap sample, which is basically sampling with replacement. The randomness of the algorithm increases the bias of the model slightly but decreases the variance and then prevents overfitting. The first approach takes into account all the variables, the second only a subset of them chosen after variable importance computed by cross-validated training



```
##                                                    ---
##                                                    ---
```

Method	Accuracy	Sensitivity	Specificity
RF	0.7666667	0.9190476	0.4111111



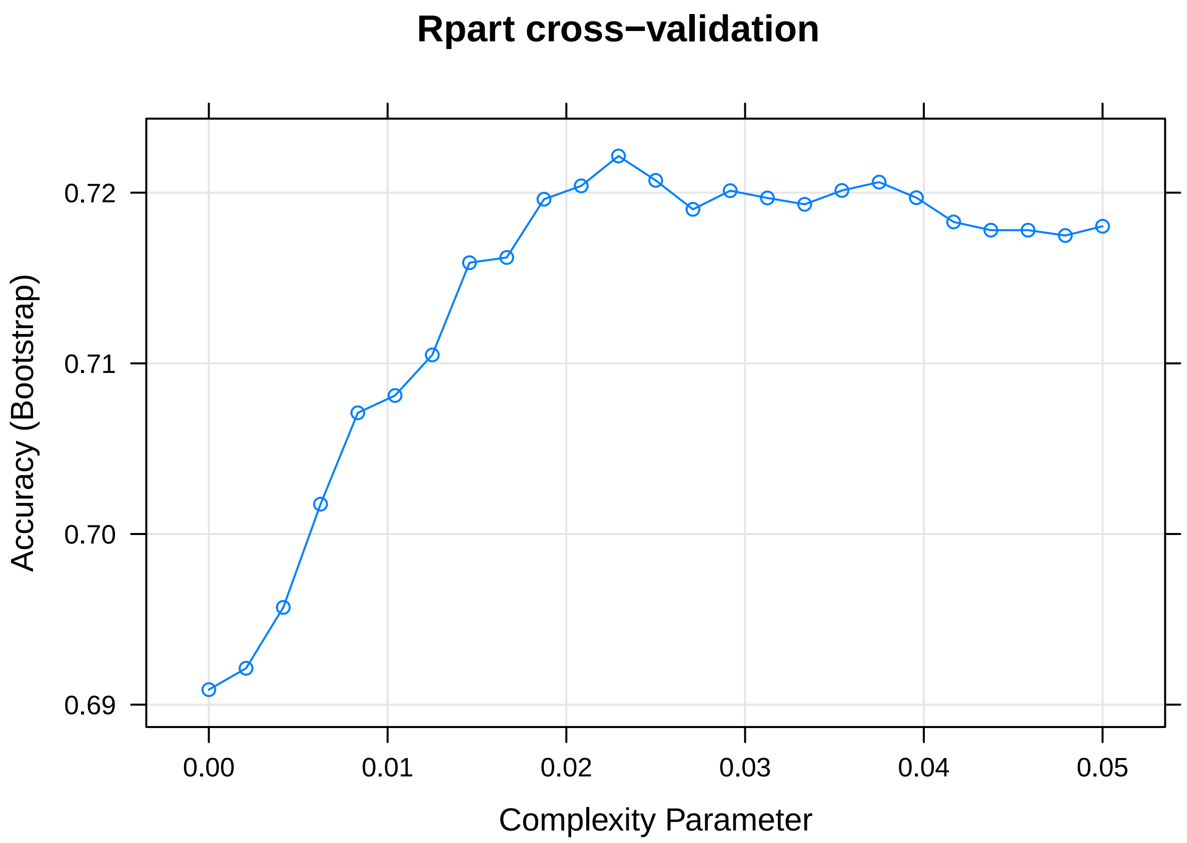
```
##                                                 ---
##                                                 ---
```

The model, after cross-validation, is trained by randomForest randomForest's library function to tune both ntree and randomly selected predictors. The metrics obtained with all predictors are quite similar to the GLM apporach. The accuracy (0.76) is not so satisfactory; the sensitivity better than GLM, but GLM wins on specificity . In the second RF model, the nine most important predictors have been chosen for training: except a slightly reduction in sensitivity, there are improvements.

Method	Accuracy	Sensitivity	Specificity
RF_IMP	0.7733333	0.9095238	0.4555556

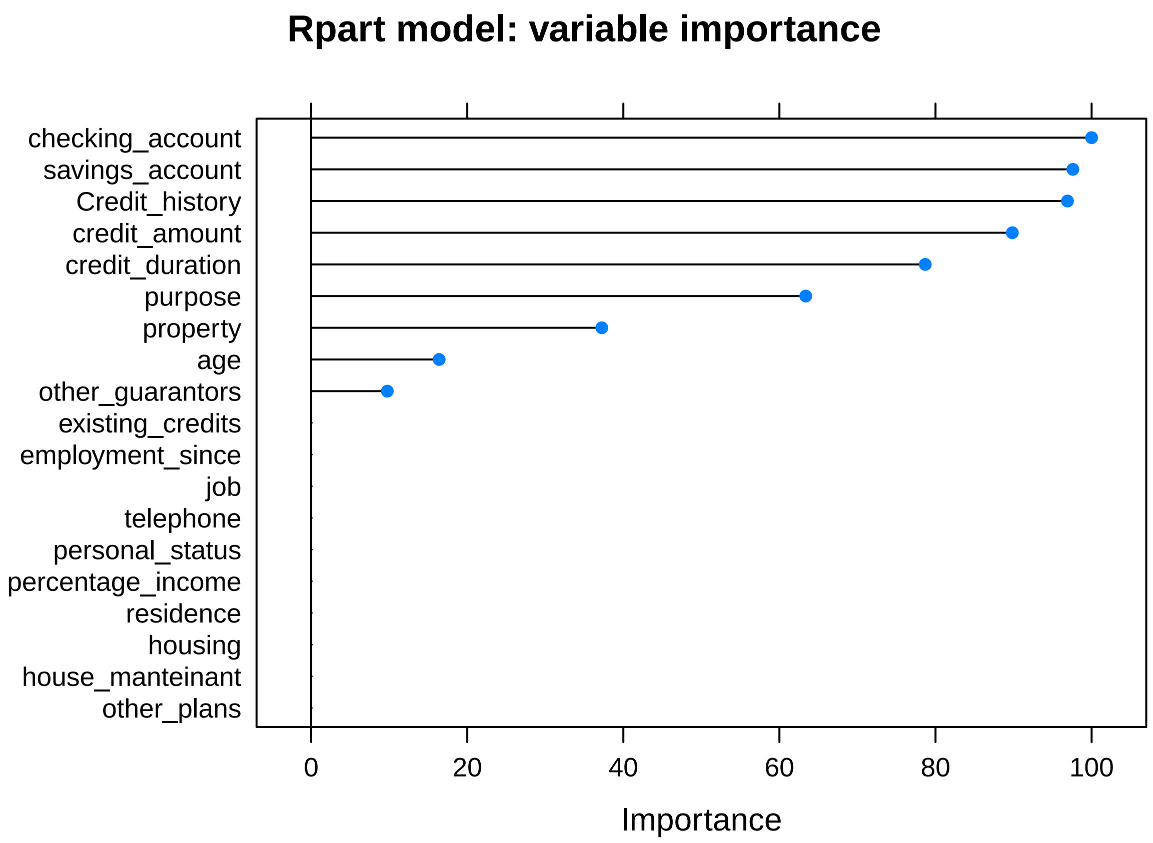
RPart

Rpart is an algorithm that is a part of the decision trees family, that is CART family. Decision trees are mainly used for making decisions that would be most useful in reaching some objective and designing a strategy based on these decisions. They are easy to represent, construct, and understand. However, the drawback is that they are very prone to overfitting. As shown in the metrics table, when using all variables as predictors, there are worsenings compared to GML and RF except for the sensitivity



```
##                                                ---
##                                                ---
```

Method	Accuracy	Sensitivity	Specificity
RPART	0.74	0.9142857	0.3333333



The VarImp results are clear: the Rpart algorithm, for fitting, takes into account mainly six over nineteen variables; anyway the prediction after the retraining done with the five-variables subset of predictor i slightly better baut there is the same low specificity, so this model hardly would be chosen.

Method	Accuracy	Sensitivity	Specificity
RPART_IMP	0.75	0.9285714	0.3333333

Naive Bayes

The Naïve Bayes classifier is a simple probabilistic classifier which is based on Bayes theorem but with strong assumptions regarding independence. Bayesian probability incorporates the concept of conditional probability, the probabilty of event A given that event B has occurred. This approach is based on the processing of the following formula:

$P(C_k|X) = \frac{P(C_k) \cdot P(X|C_k)}{P(X)}$

Where:

- $P(C_k)$ = the *prior* probability of the outcome Y. 70% “good” and 30% “bad”

- $P(X)$ = the probability of the predictor variables (same as $P(C_k|x_1, \ldots, x_p)$) is the probability of each observed combination of predictor variables.

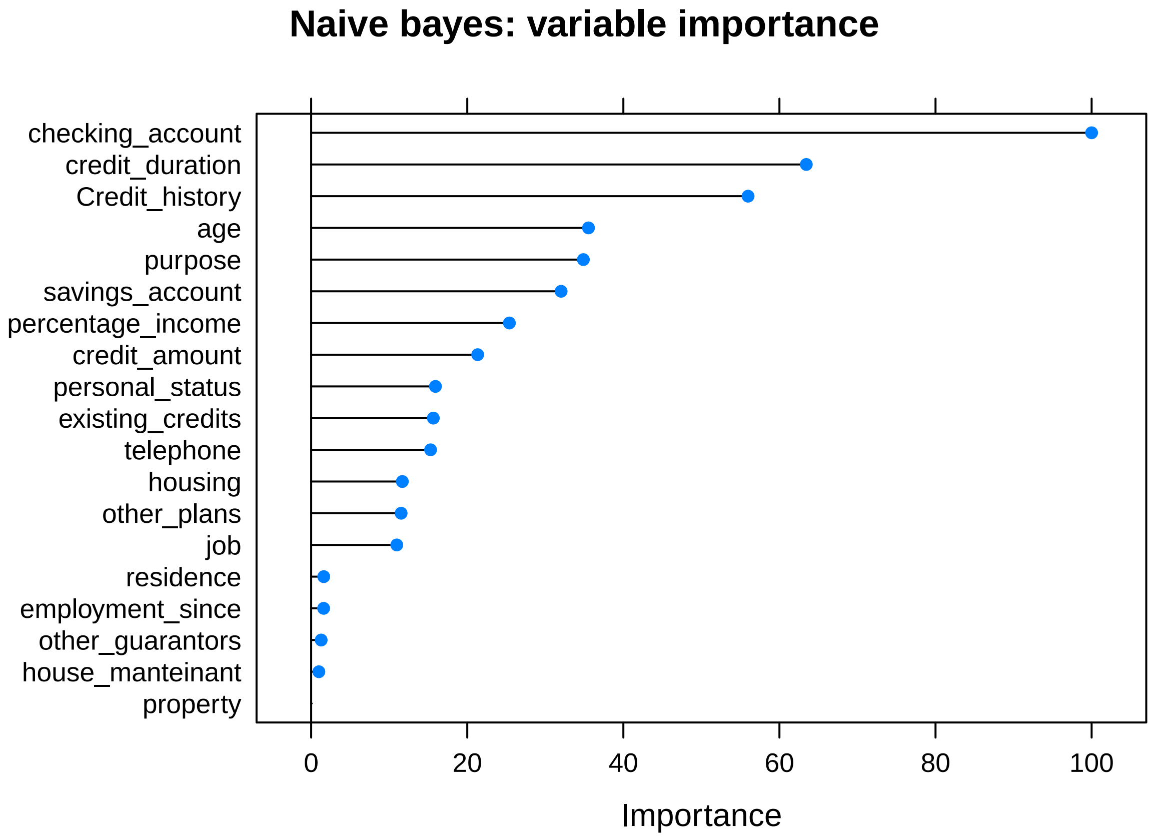
- $P(X|C_k))$ = the conditional probability or *likelihood.* Essentially, for each class of the response variable (i.e. “good” or “bad”), what is the probability of observing the predictor values.

- $P(C_k|X)$ = posterior probability. By combining our observed information, we are updating our *a priori* information on probabilities to compute a posterior probability that an observation has class C_k

The NB approach using all predictors gives quite good results, but lose compared to GLM. The specificity is slighty better than RF

Method	Accuracy	Sensitivity	Specificity
NB	0.7633333	0.8809524	0.4888889

```
##                                     ---

##                                     ---
```



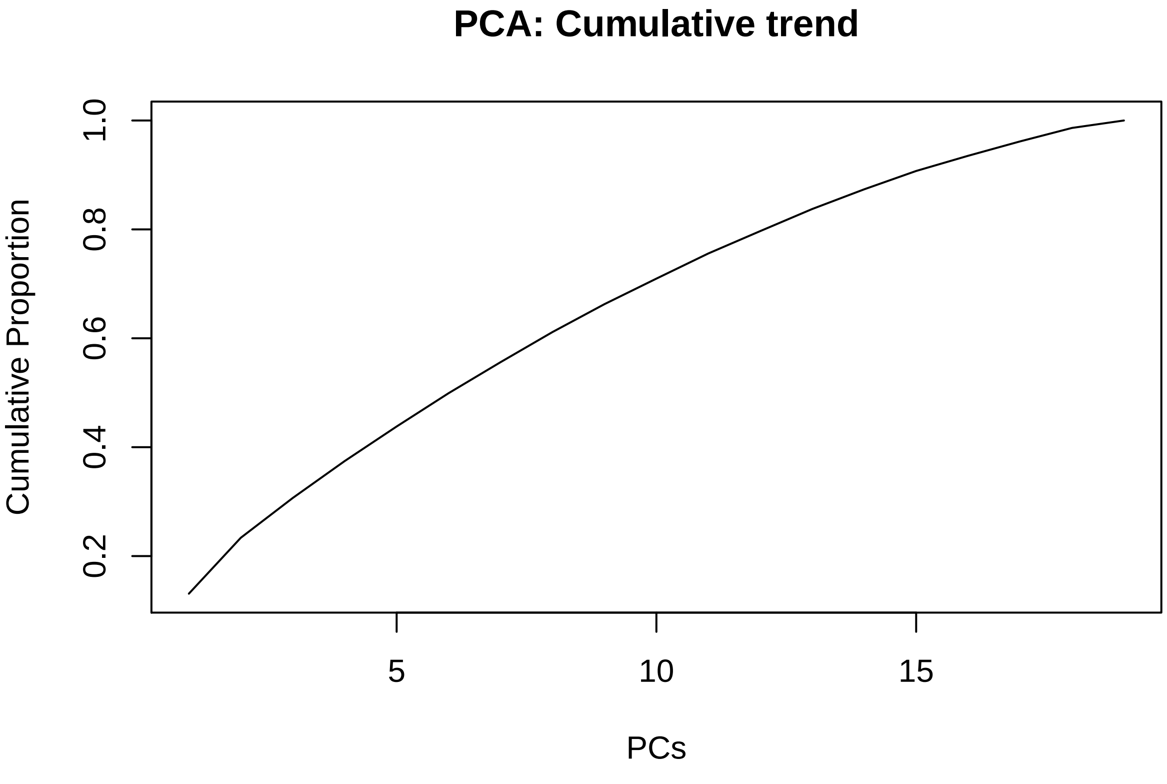
The model has been retrained using the most five important predictors computed by varimp: there is general worstening compared to the all-variables NB model.

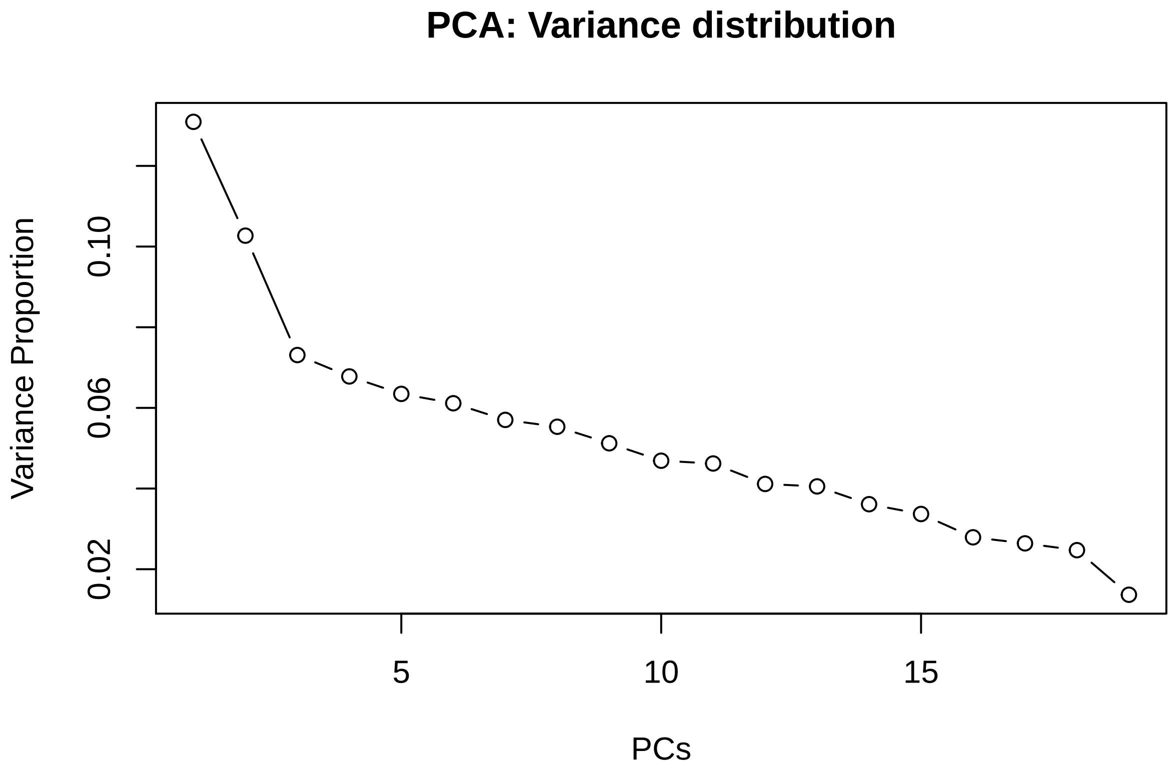
Method	Accuracy	Sensitivity	Specificity
NB-IMP	0.7466667	0.8904762	0.4111111

Principal Component Analysis

The idea of PCA is simple: reduce the number of variables of a data set, while preserving as much information as possible. This model is based on a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. In this particular case, there are 1.000 observations covered by 19 variables. As seen before, none of them is correlated to each other, so we expect no great advatage using PCA. The input dataset has to be numeric and normalized, so this has be done. Next, the pca summary shows numerical evidence of quite uniform variability distribution, that is also possible to apprecciate by two plots: the cumulative one illustrates that variability sums up almost as a stright line trend following a gentle *positive* slope.

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     1.5772 1.3969 1.1785 1.1350 1.09817 1.07787 1.04089
## Proportion of Variance 0.1309 0.1027 0.0731 0.0678 0.06347 0.06115 0.05702
## Cumulative Proportion  0.1309 0.2336 0.3067 0.3745 0.43799 0.49914 0.55616
##                            PC8     PC9   PC10    PC11    PC12    PC13
## Standard deviation     1.02526 0.98647 0.9440 0.93696 0.88417 0.87750
## Proportion of Variance 0.05532 0.05122 0.0469 0.04621 0.04114 0.04053
## Cumulative Proportion  0.61149 0.66270 0.7096 0.75581 0.79695 0.83748
##                           PC14    PC15    PC16    PC17    PC18    PC19
## Standard deviation     0.82844 0.80008 0.72819 0.70839 0.68528 0.50968
## Proportion of Variance 0.03612 0.03369 0.02791 0.02641 0.02472 0.01367
## Cumulative Proportion  0.87360 0.90729 0.93520 0.96161 0.98633 1.00000

##                                      ---

##                                      ---
```





The variance distribution shows that beyond the first three component there is a great amount of variability, i.e. the principal three component have only the 31% of total variability. On the basis of this evidence, no training/predictions have been done using PCA.

5 Results

Prior to choose the best approach, we should keep always keep in mind the business requirements related to the particular project. In other words, the decision is to choose the best model to maximize profits and minimize losses for the bank. These would be the main rules to follow:

- To predict a customer with bad credit rating as good, the bank will end up losing the whole credit amount lent to him since he will default on the payment and so loss for the bank is 100%, so the FPR should be as low as possible (specificity as high as possible)

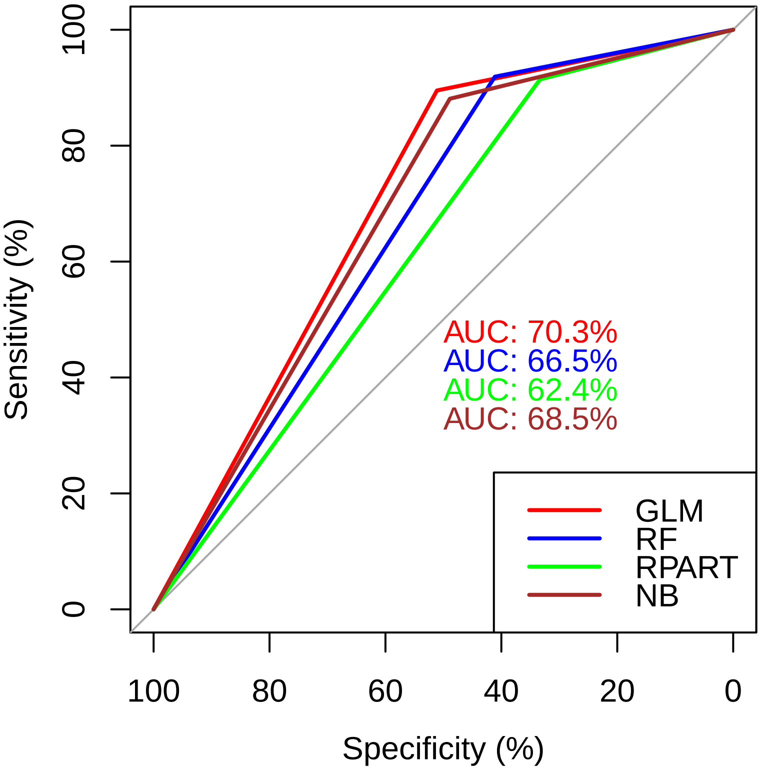
- To incorrectly predict a customer with good credit rating as bad, means to deny him the credit loan but there is neither any profit nor any loss involved in this case. High sensitivity is appreciated, but specificity shoud more important .

- To correctly predict a customer with bad credit rating as bad, means correctly deny him a credit loan and so there is neither any loss nor any profit. The results as the same as the previous conditions, but there is better accuracy and better prediction skills.

- To correctly predict a customer with good credit rating as good, means, of course, the main target, if the TPR is much more higher than FPR.

Considering all the above rules, none of the model so far used and shown in the table would perfectly fit, so we should choose a trade-off approach.

GLM seems to be a good approach: best AUC and Accuracy, good Specificity, but has not the best *sensitivity*; NB has good *specificity* and AUC (68.5%) lower than GLM's AUC(70.3%) but higher that RF's AUC(66.5). In this case, a deep analisys should be done in order to understand the weight of *credit_amount* (the most important variable in all methods) and look for a better tradeoff between sensitivity, specificity and AUC.



Method	Accuracy	Sensitivity	Specificity
GLM	0.7800000	0.8952381	0.5111111
RF	0.7666667	0.9190476	0.4111111
RF_IMP	0.7733333	0.9095238	0.4555556
RPART	0.7400000	0.9142857	0.3333333
RPART_IMP	0.7500000	0.9285714	0.3333333
NB	0.7633333	0.8809524	0.4888889
NB-IMP	0.7466667	0.8904762	0.4111111

6 Conclusions

The data scientist should always keep in mind that the business requirement set is the main milestone when he works on whatever project. The techineques adopted in this project would aim to better explore the business rules and the problem domain and eventually would find out the solution that best fits the use cases. Altough GLM could be fair solutions for this use case, as seen in the results section, none of the models perfectly fits the problem: for example, further algorithms could be used and benchmarked (i.e. Neural networks); deeper analysis on important variable could be done and different seeds tested, ensemble tecniques should be helpful. Some or a combination of various approaches could drammaticaly improve metrics and reliability.