# "German Credit" scoring data analysis report

Marta Karaś, Jan Idziak

June 14, 2015

# Table of content

## Part I
# Introduction

In this part of the report we provide answers to the following questions about the "German Credit" data analysis we performed.

1. *Why was the study undertaken?*
2. *What was the purpose of the research? What research questions were stated?*

## 1 Data analysis context

### 1.1 Motivation

This report presents results of the "German Credit" scoring data analysis which was performed as a project assignment for the "Pozyskiwanie Wiedzy" course, which we attended at Wroclaw University of Technology, Faculty of Fundamental Problems of Technology (W-11), Mathematics programm (Master) in the 2014/15 summer semester. The lecturer of the course (both lectures and laboratories) is Ph.D. Adam Zagdański.

The main goal of the project is to make use of the variety of data-mining methods we have become familiar with during the course, in order to perform complete data analysis of selected data set. We also aim to pay attention to the practical appliacnces of some parts of our work.

### 1.2 Research questions

We stated the following research purposes for our analysis.

1. Find and describe relations in the data (relations bewteen explanatory variables and response variable, relations bewteen explanatory variables).
2. Compare different methods / algorithms to perform exploratory data analysis and predictive data analysis.
3. Provide a summary of the analysis, containing suggestions of practical appliance and remarks regarding possible further research.

## Part II

# Materials and methods

In this part of the report we describe the data set we obtained and the methods we use in the analysis.

This section is rather of the decriptional / theoretical character. For a list of actual analysis steps, the outputs of the methods and more, please refer to the III part of this report.

## 2 Data set

We perform analysis with the use of The (Statlog) German Credit Data we have obtained from the UCI Machine Learning Repository site.

### 2.1 Data set description

The data set contains data on 20 variables and the classification whether an applicant is considered a Good or a Bad credit risk for 1000 loan applicants. The file provided contains variables with values encoded accoring to the following schema:

- Attribute 1: (qualitative) Status of existing checking account
  A11 : ... < 0 DM
  A12 : 0 <= ... < 200 DM
  A13 : ... >= 200 DM / salary assignments for at least 1 year
  A14 : no checking account
- Attribute 2: (numerical) Duration in month
- Attribute 3: (qualitative) Credit history
  A30 : no credits taken/ all credits paid back duly
  A31 : all credits at this bank paid back duly
  A32 : existing credits paid back duly till now
  A33 : delay in paying off in the past
  A34 : critical account/ other credits existing (not at this bank)
- Attribute 4: (qualitative) Purpose
  A40 : car (new)
  A41 : car (used)
  A42 : furniture/equipment
  A43 : radio/television
  A44 : domestic appliances
  A45 : repairs
  A46 : education
  A47 : (vacation - does not exist?)
  A48 : retraining
  A49 : business
  A410 : others

- Attribute 5: (numerical) Credit amount
- Attribute 6: (qualitative) Savings account/bonds
  A61 : ... < 100 DM
  A62 : 100 <= ... < 500 DM
  A63 : 500 <= ... < 1000 DM
  A64 : .. >= 1000 DM
  A65 : unknown/ no savings account
- Attribute 7: (qualitative) Present employment since
  A71 : unemployed
  A72 : ... < 1 year
  A73 : 1 <= ... < 4 years
  A74 : 4 <= ... < 7 years
  A75 : .. >= 7 years
- Attribute 8: (numerical) Installment rate in percentage of disposable income
- Attribute 9: (qualitative) Personal status and sex
  A91 : male : divorced/separated
  A92 : female : divorced/separated/married
  A93 : male : single
  A94 : male : married/widowed
  A95 : female : single
- Attribute 10: (qualitative) Other debtors / guarantors
  A101 : none
  A102 : co-applicant
  A103 : guarantor
- Attribute 11: (numerical) Present residence since
- Attribute 12: (qualitative) Property
  A121 : real estate
  A122 : if not A121 : building society savings agreement/ life insurance
  A123 : if not A121/A122 : car or other, not in attribute 6
  A124 : unknown / no property
- Attribute 13: (numerical) Age in years
- Attribute 14: (qualitative) Other installment plans
  A141 : bank
  A142 : stores
  A143 : none
- Attribute 15: (qualitative) Housing
  A151 : rent
  A152 : own
  A153 : for free
- Attribute 16: (numerical) Number of existing credits at this bank

- Attribute 17: (qualitative) Job
  A171 : unemployed/ unskilled - non-resident
  A172 : unskilled - resident
  A173 : skilled employee / official
  A174 : management/ self-employed/ highly qualified employee/ officer
- Attribute 18: (numerical) Number of people being liable to provide maintenance for
- Attribute 19: (qualitative) Telephone
  A191 : none
  A192 : yes, registered under the customers name
- Attribute 20: (qualitative) Foreign worker
  A201 : yes
  202 : no

The classification variable states whether there was a default case ('bad' client - failed to pay off the credit) or not ('good' client).

- Classification: (qualitative) Default
  1 (default)
  0 (non-default)

# 3 Binning countinuous variables

In credit scoring, Information Value (IV) is frequently used to compare predictive power among variables. When developing new scorecards using logistic regression, variables are often binned and recoded using WoE concept.

## 3.1 Weight of Evidence (WoE)

One of our goals when binning variables is to maximize Information Value. Weight of Evidence (WoE) for single bin is defined as:

$$WoE = \left[ ln \left( \frac{\text{Relative Frequency of Goods}}{\text{Relative Frequency of Bads}} \right) \right] \times 100.$$

We can see that value of WoE will be 0 if the odds of Relative Frequency of Goods / Relative Frequency Bads is equal to 1. If the Relative Frequency of Bads in a group is greater than the Relative Frequency of Goods, the odds ratio will be less than 1 and the WoE will be a negative number; if the Relative Frequency of Goods is greater than the Relative Frequency of Bads in a group, the WoE value will be a positive number.

## 3.2 Information Value (IV)

We define Information Value of the variable as follow:

$$IV = \sum_{i=1}^{k} \left[ (\text{Relative Frequency of Goods}_i - \text{Relative Frequency of Bads}_i) \times ln \left( \frac{\text{Relative Frequency of Goods}}{\text{Relative Frequency of Bads}} \right) \right],$$

By convention the values of the IV statistic can be interpreted as follows. If the IV statistic is:

- Less than 0.02, then the predictor is not useful for modeling (separating the Goods from the Bads),
- 0.02 to 0.1, then the predictor has only a weak relationship to the Goods/Bads odds ratio,
- 0.1 to 0.3, then the predictor has a medium strength relationship to the Goods/Bads odds ratio,
- 0.3 or higher, then the predictor has a strong relationship to the Goods/Bads odds ratio.

## 3.3 Motivation

The WoE recoding of predictors is particularly well suited for subsequent modeling using Logistic Regression. Specifically, logistic regression will fit a linear regression equation of predictors (or WoE-recoded continuous predictors) to predict the logit-transformed binary Goods/Bads variable. Therefore, by using WoE-recoded predictors in logistic regression the predictors are all prepared and coded to the same WoE scale, and the parameters in the linear logistic regression equation can be directly compared. ([8])

# 4 Feature selection

Following [1], feature selection is essentially a task to remove irrelevant and/or redundant features. *Irrelevant features* cam be removed without affecting learning performance. *Redundant features* are a type of irrelevant feature. The distinction is that redundant feature implies the co-presence of another feature; individually, each feature is relevant, but the removal of one of them will not affect learning performance.

The selection of features may be achieved in two ways:

1. **Feature ranking**. The idea is to rank features according to some criterion and select the top $k$ features.

2. **Subset selection**. The idea is to select a minimum subset of features without learning performance deterioration.

In other words, subset selection algorithms can automatically determine the number of selected features, while feature ranking algorithms need to rely on some given threshold to select features.

The tree typical feature selection models are:

1. **Filter**. In a filter model, one selects the features firstly and then uses this subset to execute a classification algorithm.

2. **Wrapper**. In a wrapper model, one employs a learning algorithm and uses its performance to determine the quality of selected features.

3. **Embedded**. An embedded model of features selection integrates the selection of features in model builidng. An example of such model is a decision tree induction algorithm, in which at each branching node, a feature has to be selected.

In literature, various search strategies are proposed, including: forward, backward, floating, branch-and-bound, and randomized. A relevant issue, regarding exhaustive and heuristic searches is whether there is any reason to perform exhaustive searches if time complexity were not a concern. Research shows that exhaustive search can lead to the features that exacerbate data oerfitting, while heuristic search is less prone to data overfitting in feature selection, facing small data samples.

The evaluation of feature selection often entails two tasks:

1. One is to compare two cases: before and after feature selection. The goal of this task is to observe if feature selection achieves its intended objectives. The aspects of evaluation may include the number of selected features, time, scalability and learning model's performance.

2. The second task is to compare two feature selection algorithms to see if one is better than other for a certain task.

## 4.1 Feature selection algorithms

In this subsection we describe methods for feature selection we use in our analysis. In general, we use the FSelector R package exhaustively. This package contains both algorithms for filtering attributes and algorithms for wrapping classifiers and search attribute subset space.

### 4.1.1 Algorithms for filtering attributes

**CFS filter** CFS is a correlation-based filter method CFS from [2]. It gives high scores to subsets that include features that are highly correlated to the class attribute but have low correlation to each other. Let *Attribute* be an attribute subset that has $k$ attributes, $rcf$ models the correlation of the attributes to the class attribute, $rff$ - the intercorrelation between attributes. We define *Attribute* score as:

$$CfsScore(Attribute) = \frac{k\ rcf}{\sqrt{k + k(k-1)rff}}.$$

The algorithm from FSelector R package makes use of *Best-first search* for searching the attribute subset space. In *Best-first search*, the algorithm chooses the best node from all already evaluated ones and evaluates it. The selection of the best node is repeated approximately $max.brackets$ times in case no better node found.

**Chi-squared filter** The algorithm evaluates the worth of an attribute by computing the value of the chi-squared statistic with respect to the class.

**Information Gain filter** One of the entropy-based filters. Algorithm evaluates the worth of an attribute by measuring the information gain with respect to the class.

$$InfoGain(Class, Attribute) = H(Class) + H(Attribute) - H(Class|Attribute),$$

where $H$ is the information entropy.

**Gain Ratio filter** One of the entropy-based filters. Algorithm evaluates the worth of an attribute by measuring the gain ratio with respect to the class.

$$GainR(Class, Attribute) = \frac{H(Class) + H(Attribute) - H(Class|Attribute)}{H(Attribute)},$$

where $H$ is the information entropy.

**Symmetrical Uncertainty filter** One of the entropy-based filters. Algorithm evaluates the worth of a set attributes by measuring the symmetrical uncertainty with respect to another set of attributes.

$$SymmU(Class, Attribute) = 2\frac{H(Class) + H(Attribute) - H(Class|Attribute)}{H(Attribute) + H(Class)},$$

where $H$ is the information entropy.

**Linear Correlation filter** The algorithm finds weights of continous attributes basing on their Pearson's correlation with continous class attribute.

**Rank Correlation filter** The algorithm finds weights of continous attributes basing on their Spearman's correlation with continous class attribute.

**OneR algorithm** The algorithms find weights of discrete attributes basing on very simple association rules involving only one attribute in condition part. In other words, it uses the minimum-error attribute for prediction, discretizing numeric attributes. For more information, see [4].

**RReliefF filter** The algorithm evaluates the worth of an attribute by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class. Considering that result, it evaluates weights of attributes. Can operate on both discrete and continuous class data. For more information see [5,6,7].

**Consistency-based filter** Evaluates the worth of a subset of attributes by the level of consistency in the class values when the training instances are projected onto the subset of attributes. Consistency of any subset can never be lower than that of the full set of attributes, hence the usual practice is to use this subset evaluator in conjunction with a Random or Exhaustive search which looks for the smallest subset with consistency equal to that of the full set of attributes. The FSelector R package implementation makes use of *Best-first search* for searching the attribute subset space. Works for continuous and discrete data.

**RandomForest filter** It is a wrapper for variable importance measure produced by randomForest algorithm. The FSelector R package implementation allows for two types of importance measure:

1. mean decrease in accuracy,
2. mean decrease in node impurity.

The first measure is computed from permuting OOB (out-of-bound) data: For each tree, the prediction error on the out-of-bag portion of the data is recorded (error rate for classification, MSE for regression). Then the same is done after permuting each predictor variable. The difference between the two are then averaged over all trees, and normalized by the standard deviation of the differences. If the standard deviation of the differences is equal to 0 for a variable, the division is not done (but the average is almost always equal to 0 in that case).
The second measure is the total decrease in node impurities from splitting on the variable, averaged over all trees. For classification, the node impurity is measured by the Gini index. For regression, it is measured by residual sum of squares.

### 4.1.2 Algorithms for wrapping classifiers and search attribute subset space

In general, the wrapper approach depends on the so called *evaluation function* that is used to return a numeric value (a score) indicating how important a given subset of features is. Typically, one uses the classification-accuracy (usually based on cross-validation) as the score for the subset.

Below we provide a brief description of the algorithms for searching atrribute subset space.

**Greedy search** At first, greedy search algorithms expand starting node, evaluate its children and choose the best one which becomes a new starting node. This process goes only in one direction. *Forward search* starts from an empty and *backward search* from a full set of attributes.

**Best-first search** The algorithm is similar to *Forward search* besides the fact that is chooses the best node from all already evaluated ones and evaluates it. In the FSelector R package implementation, the selection of the best node is repeated approximately *max.brackets* times in case no better node found.

**Hill climbing search** The algorithm starts with a random attribute set. Then it evaluates all its neighbours and chooses the best one. It might be susceptible to local maximum.

**Exhaustive search** The algorithm searches the whole attribute subset space in breadth-first order.

# 5 Classification

## 5.1 Classification algorithms

**kNN k- nearest neighbours** Method is used for modeling in problem of regression or classification. It is simple algorithym using lazy learning. There is no actual model so all the computation is done while classification. In the problem of classification the result for every single observation is a class for which in k closest neighbours from the training set is the most popular.

**Decission trees** Decission tree is a method that perform recussive partition of the set for every predictor. In each step there is chosen split that separates the set between classes the most according to one of the meassures. The most populat measures are Information GAIN or GINI. For continous data it is desired to partition variable into categorical (It could cause loss of the information). Result is highly corelatet with the learning set. Nonetheless it is easy to interpret, and attractive visually. Another plus is that decission trees do not have any assumptions about distribution of the data and algorithms works fast.

**Random forest** It is curently one of the most popular method in machine learning. Its popularity grows thanks to good performance and small assumptions. However it performs well, it is hard to interpret the results, as long as model is complicated and consists many decission trees. For this method in each step of decission tree creation there is taken random subset of the features and then one of them is taken for split. This is done untli appropriate settled level. For Random forests the computatio time is much higher than for decission trees. Mostly it is because not only one tree is fitted but usualy much more. One of the biggest disadvantages of this model is hard interpretation of the output. Even though the subset of predictors is only taken it shows much better results than other regulat methods for different data sets.

**Logistic regression** The most popular method among application in banks, insurance companies and the industrie for modeling binary data (It could servs also for prediction multiclass data). It owes popularity to its simplicity, easy open form and straight interpretation. It is subject to produce Score Card. Method is a particular type of generalized linear model where link function has logit form $logit(p) = log(\frac{p}{1-p})$. It means that probability of occurance particular event, is modeled indirectly, as a appropriate transformation.

$$logit(p_j) = log(\frac{p_j}{1-p_j}) = \Sigma_{i=1}^{n}\beta_i X_{i,j}$$

Where $p_j$ is estimated probability, $\beta_i$ is factor for $X_{i,j}$ and X represents the features of observation.

**Linear discriminant analysis** It is another linear method. Under the assumption of normality and equality of covariance matrices within classes.

$$Pr(C = k|X = x) = \frac{f_k(x)\pi_k}{\Sigma_{l=1}^{K} f_l(x)\pi_l}$$

where $C = k$ represents particular class affiliation, $x$ is observation vector and $f_k(x)$, has appropriate Gaussian distribution with the mean $\mu_k$ and covariance matrixvariance $\Sigma$ and $\pi_k$ is a-priori classes probability. It is enough to compare numerator as long as denominator for all classes would be the same.

**Quadric discriminant analysis** It is similar method to the linear dyscriminant analysis. It keeps assumption about normality, but in this case covarance matrices could differ. Aproppriate probability function keep its form:

$$Pr(C = k|X = x) = \frac{f_k(x)\pi_k}{\Sigma_{l=1}^{K} f_l(x)\pi_l}$$

As before it is enough to compare numerators for all classes.

**Naive Bayes** Another method that uses Bayesian rule. It is called Naive Bayes as long as it has a naive assumption about loss of correlation between predictors. However this model has easy form, it also perform well in many appliactions.

$$P(Y = k|X = x) = \frac{P(X = x|Y = y)}{P(X = x)} = \frac{P(X_1 = x_1, \ldots X_n = x_n|Y = y)}{P(X = x)}$$

In this case probabilities are just taken as an empirical realisation of the data. It could also fall into problem of zero class frequencies. To omit this situation it is recommended to use one of the smoothing methods.

## 5.2 Classification performance metrics

**Confussion matrix** This is one of simple method to grade quality of the classification. It serves to compare acctual class of an observation to one predicted by model.

| | | Predicted Class | |
|---|---|---|---|
| | | True | False |
| Actuall Class | True | True Positive (TP) | False Negative (FN) |
| | False | False Positive (FP) | True Negative (TN) |

Using confussion matrix it is easier to calculate many of the goodnes of fit measures for the models such as sensitivity, specifity or many more.

**Sensitivity** One of the simple measures called also Recall or True Positive Rate. It measures proportion of predicted as true and actual true. Using confusion matrix it could be given as:

$$TPR = \frac{TP}{TP + FN}$$

**Specifity** This factor is also called True Negative Rate. It measures proportion predicted corretly as false and actuall number of false observations. It is given by:

$$SPC = \frac{TN}{TN + FP}$$

**Precission** Popular measure in information retrival. It repressents fraction of documents relevant to retrived. In binary classification it is defined as:

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

**False discovery rate** It is complementary to the Precission measure that is getting more popular thanks to groving dimensionality of data sets. In many applications it is of an interest of scientist to control this factor. The formula for this coefficient is subsequent:

$$FDR = \frac{FP}{TP + FP} = 1 - PPV$$

**Accuracy** It is simple measure that could be taken as a good indicator for model performance. It takes proportion of all positive clasified to total number of observations. For not equaly distributed observations between groups (for example in spam detection where there is many spam files classificator that predicts everything as a spam would have high Accuracy)

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}$$

**F-measure** F-measure is defined as a combination of Precission and Recall. Both of earlier described indexes gives some information about model, but using them separtely can effect with falling into missclassification for specific types of data.

$$F = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

### 5.2.1 Separation measures

**Kolmogorow Smirnov statistics and distributions** Kolmogorov Smirnov statistics is the biggest distance between distribution functions of scores (probabilities) for accutal groups of True and False. Distribitions shows just simple cumulative distribution functions for classes.

**ROC curve and GINI, AUROC indexes** ROC or Reciver Operating Characteristic is a graphical ilustration that represents preformance of the binary classifier. It is being used in medicine, radiology, biometrics and many more applications It shows proportion of the True Positive rate (on the vertical axis) and False Positive rate (on the horisontal axis) at various thresholds. AUC is factor strongly connected with the ROC curve. Abbreviation stands for area under curve. It could be calculated as follows:

$$AUC = \int_{-\infty}^{\infty} TPR(x)FPR(x)dx$$

**Histogram Good vs Bad** It is just histogram showing distribution of scores (probabilities) within classes. For good models there suppose to be visible difference between hight of the scores for different classes.

# 6 Principal Components Analysis

According to *Wikipedia* [10], Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of (possibly) correlated variables into a set of values of linearly uncorrelated variables called *principal components*. This transformation is defined in such a way that the first principal component is chosen to account for as much variance in plot dispersion from the centroid as possible; the second is chosen by the same criterion but subject to the constraint of being orthogonal the first, and so on. The principal components are the *eigenvectors of the covariance matrix* of distribution of the variables that a set of observations is distributed from.

The desired outcome of the principal component analysis is to project a feature space onto a smaller subspace that represents our data "well". Depending on how successful we are at reducing the data set, we can seek patterns among the distribution of plots in ordination space, and explore possible environmental correlates with these.

The results of a PCA are complex; we may be interested in knowing:

- variance explained by each eigenvector,
- cumulative variance explained by subsequents eigenvectors,
- *loadings* for each column; the loadings are the contribution of the column vector to each of the eigenvectors. A large positive component means that that column is positively correlated with that eigenvector; a large negative values is negative correlation; and small values mean that the species is unrelated to that eigenvector;
- *scores* for each row in the matrix or dataframe.

Generally, the vast majority of the variance is described on the first few eigenvectors, and we can save space by only calculating the scores for only the first few eigenvectors.

# 7 Cluster analysis

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). In this section we include some basic informations about cluster analysis algorithms and cluster analysis performance metrics. The notes below are based mainly on 3 different sources:

- "Data Clustering: A Review" article [9],
- notes from Laboratory for Dynamic Synthetic Vegephenonenology (LabDSV), the University of California [11],
- `clValid` (R package for cluster validation) vignette [12].

## 7.1 Introduction

Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity.

Typical pattern clustering activity involves the following steps:

1. pattern representation (optionally including feature extraction and/or selection),
2. definition of a pattern proximity measure appropriate to the data domain,
3. clustering or grouping,
4. data abstraction (if needed), and
5. assessment of output (if needed).

*Pattern representation* process that one usually performs is to gather facts and conjectures about the data, optionally perform feature selection and extraction, and design the subsequent elements of the clustering system. A careful investigation of the available features and any available transformations (even simple ones) can yield significantly improved clustering results. A good pattern representation can often yield a simple and easily understood clustering; a poor pattern representation may yield a complex clustering whose true structure is difficult or impossible to discern.

Since similarity is fundamental to the definition of a cluster, a *measure of the similarity* between two patterns drawn from the same feature space is essential to most clustering procedures. Because of the variety of feature types and scales, the distance measure (or measures) must be chosen carefully. It is most common to calculate the *dissimilarity* between two patterns using a distance measure defined on the feature space.

The *grouping* step can be performed in a number of ways. For example, we can distinguish between *hard* output clustering (a partition of the data into groups) or *fuzzy* (where each pattern has a variable degree of membership in each of the output clusters). We may also distinguish between *hierarchical* clustering algorithms that produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity and *partitional* clustering algorithms that identify the partition that optimizes (usually locally) a clustering criterion.

*Data abstraction* is the process of extracting a simple and compact representation of a data set. In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster prototypes or representative patterns such as the centroid.

Eventually, validity assessments are performed to determine whether the output is meaningful. A clustering structure is valid if it cannot reasonably have occurred by chance or as an artifact of a clustering algorithm. There are three types of validation studies. An *external* assessment of validity compares the recovered structure to an a priori structure. An *internal* examination of validity tries to determine if the structure is internally appropriate.

## 7.2 Similarity, Dissimilarity and Distance

*Similarity* is a characterization of the ratio of the number of attributes two objects share in common compared to the total list of attributes between them. Objects which have everything in common are identical, and have a similarity of 1.0. Objects which have nothing in common have a similarity of 0.0. There is a large number of similarity indices proposed and employed.

*Dissimilarity* is the complement of similarity, and is a characterization of the number of attributes two objects have uniquely compared to the total list of attributes between them. In general, dissimilarity can be calculated as *1 - similarity*.

*Distance* is a geometric conception of the proximity of objects in a high dimensional space defined by measurements on the attributes. `R` calculates distances with functions from at least a few packages. Among them there are:

- `stats::dist` - computes and returns the distance matrix computed by using the specified distance measure to compute the distances between the rows, $x$ and $y$ ($p$-elements vectors), of a data matrix; the distance measure to be used is one of:
  - `"euclidean"`:
    usual distance between the two vectors ($L_2$ norm): $\sqrt{\sum_i^p (x_i - y_i)^2}$,
  - `"maximum"`:
    maximum distance between two components of $x$ and $y$ (supremum norm),
  - `"manhattan"`:
    absolute distance between the two vectors ($L_1$ norm),
  - `"canberra"`:
    $\sum_i^p (|x_i - y_i| / |x_i + y_i|)$; terms with zero numerator and denominator are omitted from the sum and treated as if the values were missing,
  - `"binary"`:
    (*asymmetric binary*) the vectors are regarded as binary bits, so non-zero elements are "on" and zero elements are "off"; the distance is the proportion of bits in which only one is on amongst those in which at least one is on,
  - `"minkowski"`:
    the $p$ norm, the $p$th root of the sum of the $p$th powers of the differences of the components,

In practice, distances and dissimilarities are sometimes used interchangeably. They have quite distinct properties; e.g., dissimilarities are bounded within $[0,1]$ whereas distances are unbounded on the upper end.

## 7.3 Clustering Algorithms

`R` has wide variety of clustering algorithms available. We make use of different algorithms from the base distribution (`stats`) and add-on packages. A brief description of each clustering method and its availability is given below.

### 7.3.1 UPGMA

Unweighted Pair Group Method with Arithmetic Mean is probably the most frequently used clustering algorithm. It is an agglomerative, hierarchical clustering algorithm that yields a dendogram which can be cut at a chosen height to produce the desired number of clusters. Each observation is initially placed in its own cluster, and the clusters are successively joined together in order of their "closeness". The closeness of any two clusters is determined by a dissimilarity matrix, and can be based on a variety of agglomeration methods.

UPGMA is included with the base distribution of R in function `hclust()`, and is also implemented in the `agnes()` function in package `cluster`.

### 7.3.2 K-means

K-means is an iterative method which minimizes the within-class sum of squares for a given number of clusters. The algorithm starts with an initial guess for the cluster centers, and each observation is placed in the cluster to which it is closest. The cluster centers are then updated, and the entire process is repeated until the cluster centers no longer move. Often another clustering algorithm (e.g., UPGMA) is run initially to determine starting points for the cluster centers.

K-means is implemented in the function `kmeans()`, included with the base distribution of `R`.

### 7.3.3 Diana

Diana is a divisive hierarchical algorithm that initially starts with all observations in a single cluster, and successively divides the clusters until each cluster contains a single observation. Along with SOTA, Diana is one of a few representatives of the divisive hierarchical approach to clustering.

Diana is available in function `diana()` in package `cluster`.

### 7.3.4 PAM

Partitioning around medoids (PAM) is similar to K-means, but is considered more robust because it admits the use of other dissimilarities besides Euclidean distance. Like K-means, the number of clusters is fixed in advance, and an initial set of cluster centers is required to start the algorithm.

PAM is available in the `cluster` package as function `pam()`.

### 7.3.5 Clara

Clara is a sampling-based algorithm which implements PAM on a number of sub-datasets. This allows for faster running times when a number of observations is relatively large.

Clara is also available in package `cluster` as function `clara()`.

### 7.3.6 Fanny

This algorithm performs fuzzy clustering, where each observation can have partial membership in each cluster. Thus, each observation has a vector which gives the partial membership to each of the clusters. A hard cluster can be produced by assigning each observation to the cluster where it has the highest membership.

Fanny is available in the `cluster` package (function `fanny()`).

### 7.3.7 SOM

Self-organizing maps is an unsupervised learning technique that is popular among computational biologists and machine learning researchers. SOM is based on neural networks, and is highly regarded for its ability to map and visualize high-dimensional data in two dimensions.

SOM is available as the `som()` function in package `kohonen`.

### 7.3.8 SOTA

Self-organizing tree algorithm (SOTA) is an unsupervised network with a divisive hierarchical binary tree structure. It was originally proposed by Dopazo and Carazo (1997) for phylogenetic reconstruction, and later applied to cluster microarray gene expression data in (Herrero et al., 2001). It uses a fast algorithm and hence is suitable for clustering a large number of objects.

SOTA is included with the `clValid` package as function `sota()`.

## 7.4 Evaluation of clustering

In this analysis we focus on two types of clustering evaluation measurements: *internal* and *external* criterions.

### 7.4.1 Internal criterion

For internal validation, we use measures that reflect:

- compactness,
- connectedness,
- separation

of the cluster partitions.

*Connectedness* relates to what extent observations are placed in the same cluster as their nearest neighbors in the data space, and is here measured by the *connectivity*.

- *Connectivity*:

  Define $nn_{i(j)}$ as the $j$th nearest neighbor of observation $i$, and let $x_{i,nn_{i(j)}}$ be zero if $i$ and$nn_{i(j)}$ are in the same cluster and $1/j$ otherwise. Then, for a particular clustering partition $C = C_1, ..., C_K$ of the $N$ observations into $K$ disjoint clusters, the connectivity is defined as

  $$Conn(C) = \sum_{i=1}^{N} \sum_{j=1}^{L} x_{i,nn_{i(j)}},$$

  where $L$ is a parameter that determines the number of neighbors that contribute to the connectivity measure. The connectivity has a value between zero and $\infty$ and should be minimized.

*Compactness* assesses cluster homogeneity, usually by looking at the intra-cluster variance, while *separation* quantifies the degree of separation between clusters (by measuring the distance between cluster centroids). Since compactness and separation demonstrate opposing trends (compactness increases with the number of clusters but separation decreases), popular methods combine the two measures into a single score. The Dunn index and silhouette width are both examples of non-linear combinations of the compactness and separation, and with the connectivity comprise the three internal measures available in clValid. The details of each measure are given below.

- *Silhouette coefficient*:

  The silhouette coefficient contrasts the average distance to elements in the same cluster with the average distance to elements in other clusters. Objects with a high silhouette value are considered well clustered, objects with a low value may be outliers. This index works well with k-means clustering, and is also used to determine the optimal number of clusters.

- *Dunn index*:

  The Dunn index aims to identify dense and well-separated clusters. It is defined as the ratio between the minimal inter-cluster distance to maximal intra-cluster distance (in other words: minimum separation / maximum diameter). For each cluster partition, the Dunn index can be calculated by the following formula:

  $$D = \frac{\min_{1 \leq i < j \leq n} d(i,j)}{\max_{1 \leq k \leq n} d'(k)}$$

  where $d(i,j)$ represents the distance between clusters $i$ and $j$, and $d'(k)$ measures the intra-cluster distance of cluster $k$.

  Since internal criterion seek clusters with high intra-cluster similarity and low inter-cluster similarity, algorithms that produce clusters with high Dunn index are more desirable.

### 7.4.2 External criterion

An alternative to internal criteria is direct evaluation in the application of interest. In external evaluation, clustering results are evaluated based on data that was not used for clustering, such as known class labels. These types of evaluation methods measure how close the clustering is to the predetermined classes.

Some of the measures of quality of a cluster algorithm using external criterion include:

- *Rand index*:

  Given a set of $n$ elements $S = \{o_1, \ldots, o_n\}$ and two partitions of $S$ to compare, $X = \{X_1, \ldots, X_r\}$ which is a partition of $S$ into $r$ subsets, and $Y = \{Y_1, \ldots, Y_s\}$ which is a partition of $S$ into $s$ subsets, define the following:

  - $a$, the number of pairs of elements in $S$ that are in the same set in $X$ and in the same set in $Y$,
  - $b$, the number of pairs of elements in $S$ that are in different sets in $X$ and in different sets in $Y$,
  - $c$, the number of pairs of elements in $S$ that are in the same set in $X$ and in different sets in $Y$
  - $d$, the number of pairs of elements in $S$ that are in different sets in $X$ and in the same set in $Y$

  The Rand index, R, is:
  $$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}.$$

  Intuitively, $a + b$ can be considered as the number of agreements between $X$ and $Y$ and $c + d$ as the number of disagreements between $X$ and $Y$.

  The Rand index has a value between 0 and 1, with 0 indicating that the two data clusters do not agree on any pair of points and 1 indicating that the data clusters are exactly the same.

  The function `RRand()` in package `phyclust` implements this index.

### 7.4.3 Cluster stability

Let N denote the total number of observations (rows) in a dataset and M denote the total number of columns, which are assumed to be numeric (e.g., a collection of samples, time points, etc.).

The stability measures compare the results from clustering based on the full data to clustering based on removing each column, one at a time. (These measures work especially well if the data are highly correlated, which is often the case in high-throughput genomic data.)

The measures we used are:

- the average proportion of non-overlap (APN),
- the average distance (AD),
- the average distance between means (ADM),
- and the figure of merit (FOM).

In all cases the average is taken over all the deleted columns, and all measures should be minimized. Below we present the description of the measures listed above.

- *Average proportion of non-overlap (APN)*:

  The APN measures the average proportion of observations not placed in the same cluster by clustering based on the full data and clustering based on the data with a single column removed. Let $C^{i,0}$ represent the cluster containing observation $i$ using the original clustering (based on all available data), and $C^{i,l}$ represent the cluster containing observation $i$ where the clustering is based on the dataset with column $l$ removed. Then, the APN measure is defined as
  $$APN(C) = \frac{1}{MN} \sum_{i=1}^{N} \sum_{l=1}^{M} \left(1 - \frac{n(C^{i,l} \cap C^{i,0})}{n(C^{i,0})}\right).$$
  The APN is in the interval $[0, 1]$, with values close to zero corresponding with highly consistent clustering results.

- *Average distance (AD)*:

  The AD measure computes the average distance between observations placed in the same cluster by clustering based on the full data and clustering based on the data with a single column removed. It is defined as
  $$AD(C) = \frac{1}{MN} \sum_{i=1}^{N} \sum_{l=1}^{M} \frac{1}{n(C^{i,l})n(C^{i,0})} \left[ \sum_{i \in n(C^{i,0}), j \in C^{i,l}} dist(i, j) \right].$$
  The AD has a value between zero and $\infty$, and smaller values are preferred.

- *Average distance between means (ADM)*:

  The ADM measure computes the average distance between cluster centers for observations placed in the same cluster by clustering based on the full data and clustering based on the data with a single column removed. It is defined as
  $$ADM(C) = \frac{1}{MN} \sum_{i=1}^{N} \sum_{l=1}^{M} dist(\overline{x}_{C^{i,l}}, \overline{x}_{C^{i,0}}),$$
  where $\overline{x}_{C^{i,0}}$ is the mean of the observations in the cluster which contain observation $i$, when clustering is based on the full data, and $\overline{x}_{C^{i,l}}$ is similarly defined.

  In the `clValid` package implementation, ADM uses the Euclidean distance. It also has a value between zero and $\infty$, and again smaller values are prefered.

- *Figure of merit (FOM)*:

  The FOM measures the average intra-cluster variance of the observations in the deleted column, where the clustering is based on the remaining (undeleted) samples. This estimates the mean error using predictions based on the cluster averages. For a particular left-out column $l$ the FOM is

  $$FOM(l,C) = \sqrt{\frac{1}{N}\sum_{k=1}^{K}\sum_{i\in C_k(l)} dist(x_{i,l}, \overline{x}_{C_k(l)})},$$

  where $x_{i,l}$ is the value of the $i$th observation in the $l$th column, and $\overline{x}_{C_k(l)}$ is the average of cluster $C_k(l)$.

  In the `clValid` package implementation, the only distance available for FOM is Euclidean. The FOM is multiplied by an adjustment factor $\sqrt{\frac{N}{N-K}}$ to alleviate the tendency to decrease as the number of clusters increases. The final score is averaged over all the removed columns, and has a value between zero and $\infty$, with smaller values equaling better performance.

# Part III
# Results

# 8 Data preprocessing results

In this section we present the results of data preprocessing we performed. The important parts of this process are: creating derived variables, binning countinuous variables and correcting bins (factor levels) if it seems reasonable.

The procedures and methods used to perform this part of the analysis include:

- visual inspection of density estimator plots and fitting distribution from different distribution families to numeric variables data,
- comparision of optimal discretization method (*smbinning* package) and equal frequency discretization with Information Value as criterion,
- using WoE criterion to define factor levels "similarity".

## 8.1 Searching for missing, corrupt and invalid data

We started the preprocessing in searching for missing, corrupt and invalid data. In our dataset most of the varaibles are factor variables ($PURPOSE$, $EMPLOYMENT$ etc.) Some of them are numeric but consist only of a few qunique values and thus may be seen rather like ordered factors ($NUM\_OF\_MAINTAINED\_PEOPLE$, $RESIDENCE$ etc.) We investigated frequency tables and did not notice anything anusual in the values.

Three variables in the data set are "truly" numeric: $DURATION$, $AMOUNT$ and $AGE$. We did not find anything particularly unusual in the values of these variables. To satisfy our curiosity, we tried to fit a probabilistic distribution to the values. We used $MASS :: fitdistr$ function to perform maximum-likelihood fitting of univariate distribution from selected distribution families. In each case we tried to fit parameters for three distributions families: *Gamma*, *Log-normal* and *Weibull*.

On the graphs below we can see kernel density estimates of variable density (black line) and curves representing density of the distribution fitted. We was not able to fit *Gamma* distribution to the $AMOUNT$ variable values (*Error in stats::optim(x = c(1169, 5951, 2096, 7882, 4870, 9055, 2835, : non-finite finite-difference value [1]*). It seems that *Log-normal* distribution fits quite well in each three cases.

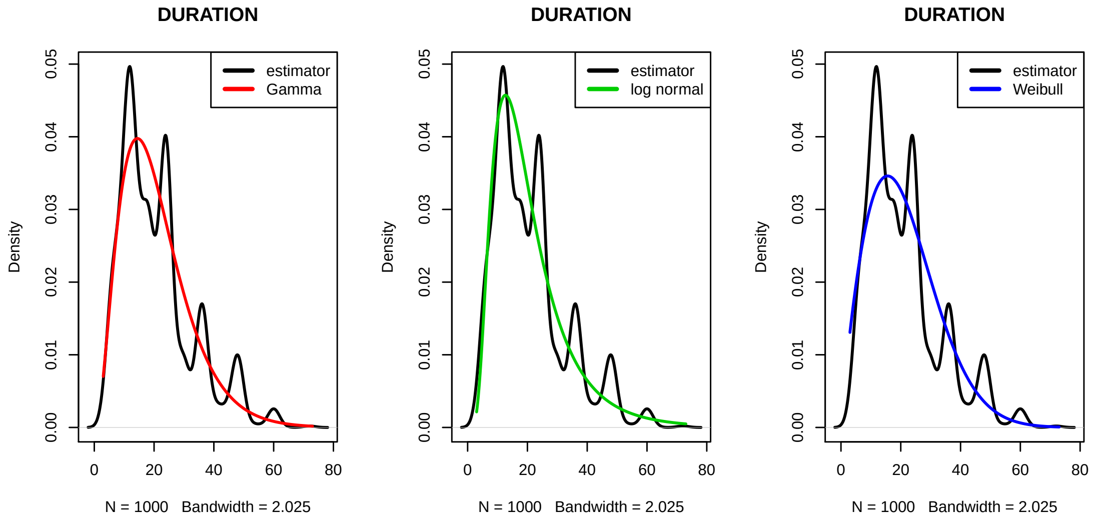


Figure 1: Kernel density estimate of $DURATION$ variable density (black line) and curves representing density of the distribution fitted.

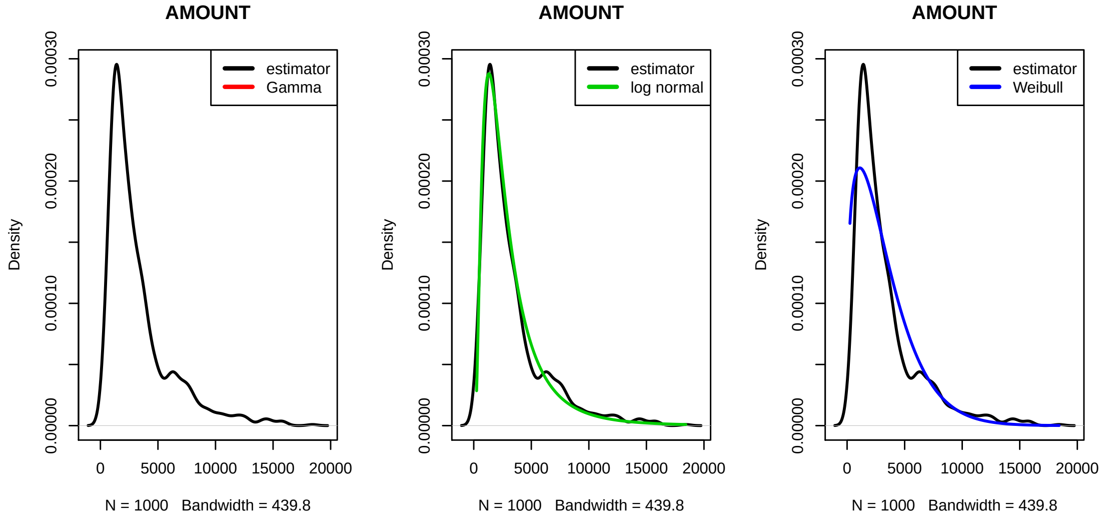


Figure 2: Kernel density estimate of $AMOUNT$ variable density (black line) and curves representing density of the distribution fitted.

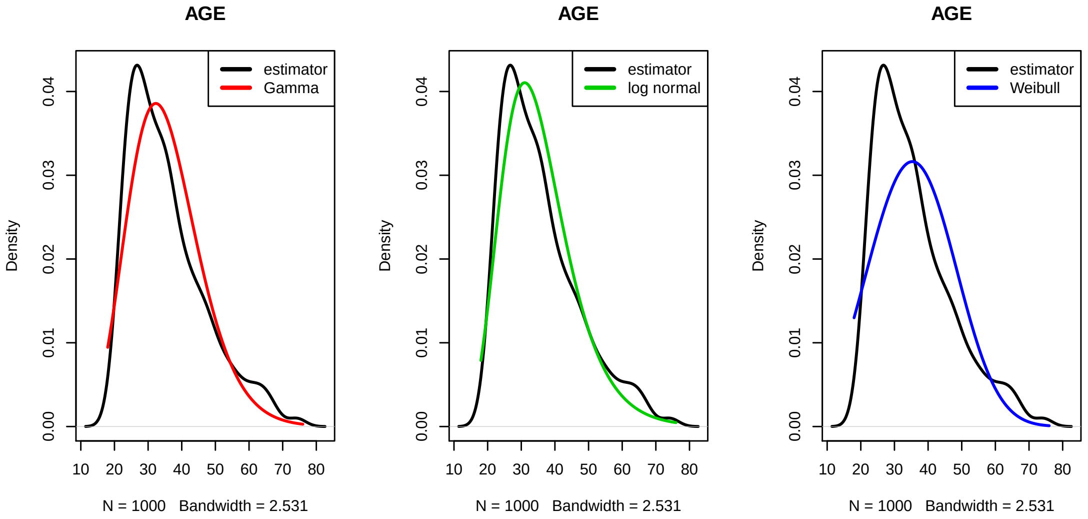


Figure 3: Kernel density estimate of $AGE$ variable density (black line) and curves representing density of the distribution fitted.

## 8.2 Creating derived variables

We were considering possibilities of creating derived variables. We came up with propositions of the following formulas:

$$AMOUNT\_TO\_DURATION = AMOUNT/DURATION,$$

$$DURATION\_TO\_AGE = DURATION/AGE,$$

$$AMOUNT\_TO\_AGE = AMOUNT/AGE.$$

On the *Figure 4.* we can see boxplots of $DURATION$, $AGE$ and $AMOUNT$ across two levels of response variable $RES$. On the *Figure 5.* we can see boxplots of derived variables. This comparision can give us intuition how well our new variables separate good and bad bank clients.

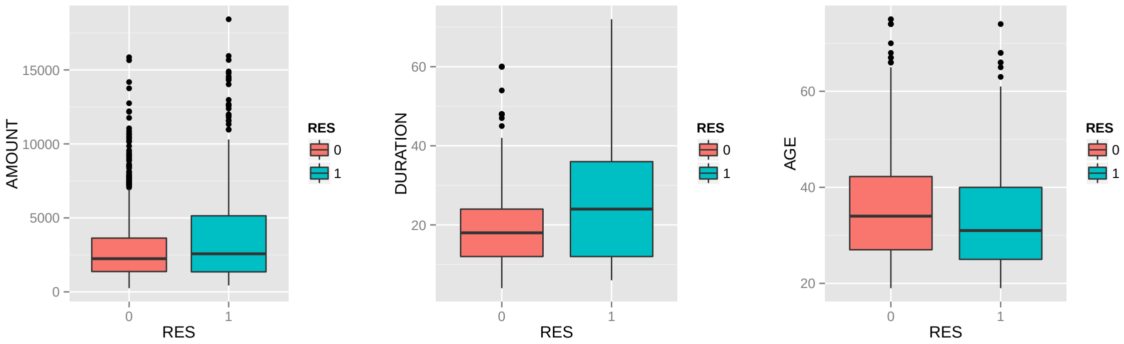


Figure 4: Boxplots of $AMOUNT$, $DURATION$ and $AGE$ variables across two levels of response variable $RES$.

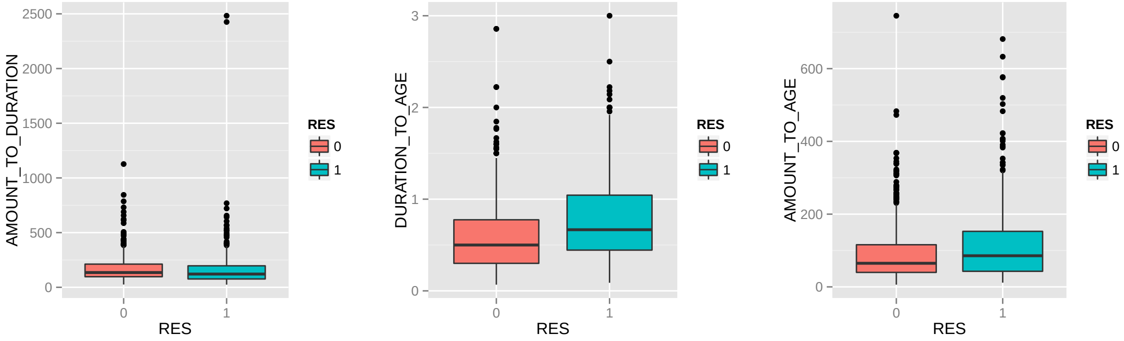


Figure 5: Boxplots of derived variables: $AMOUNT\_TO\_DURATION$, $DURATION\_TO\_AGE$ and $AMOUNT\_TO\_AGE$ across two levels of response variable $RES$.

The boxplots on the *Figure 5.* agree with our intuition: we expect higher *DURATION_TO_AGE* and *AMOUNT_TO_AGE* values for those clients who defaulted ($RES = 1$). On the other hand, we would expect the same for the *AMOUNT_TO_DURATION* variable whereas the plot shows something slightly opposite. However, the value differences on the *AMOUNT_TO_DURATION* boxplot are so fine that we suppose this variable is going to turn out to be of low information value and will not be consider as important one.

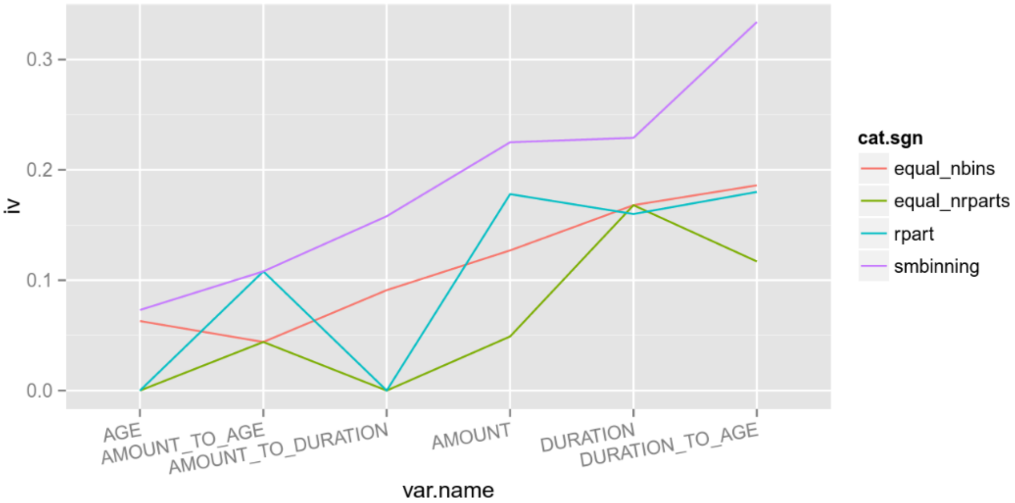
## 8.3 Binning continuous variables

In our analysis we compared three approaches of dividing continuous variables into categories:

1. equal frequency discretization (resulted bins are of equal number of observations),
2. supervised discretization which utilizes Recursive Partitioning to categorize the numeric characteristic and compute cutpoins based on Conditional Inference Trees algorithm (*smbinning* package),
3. categorize variable with simple tree model (from *rpart* package, with the use of default tree building parameters).

The IV comparision is presented on the figure below. Note that equal frequency discretization results are presented for:

- the same number of bins as in variable resulted from *smbinning* method; signature: *equal_nbins*,
- the same number of bins as in variable resulted from *rpart* method; signature: *equal_nrparts*.



We can see a few interesting things from the plot above:

- The *smbinning* function from the *smbinning* seems to beat other methods in terms of IV of resulted binned variable.

- Derived variable $DURATION\_TO\_AGE$ has a strong relationship to the Goods/Bads odds ratio (over 0.33 IV value), whereas two other derived variables ($AMOUNT\_TO\_AGE$, $AMOUNT\_TO\_DURATION$) seem to have not.

## 8.4 Correcting bins (levels) of categorical variables

The last element of data pre-processing we performet os correcting bins (levels) of categorical variables. In the domain of particular variable's levels, we searched for levels which:

- reasonalby similar in terms of their meaning (e.g. for $PURPOSE$ variable we have logically similar levels: *car (new)* and *car (old)*,
- do not vary in terms of $GOOD$ / $BAD$ observations fraction.

To assess whether or not levels vary in terms of $GOOD$ / $BAD$ observations fraction we wrote a function which compute Information Value for non-changed variable and the same variable with a par of levels joined, for all possible pair of levels. Below we present the head table output of this function and WoE plot for each of the levels for a particular variable.

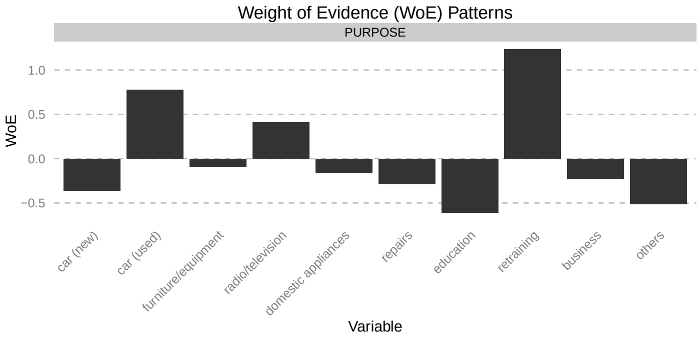
We include results for the two variables in case of which we decided to perform bin correcting.

1. $PURPOSE$

   Below we can see what is the $PURPOSE$ variable IV ($3^{rd}$ column) after joining a pair of levels ($1^{st}$ and $2^{nd}$ column).

|  | lvl1 | lvl2 | iv |
|---|---|---|---|
| 46 | NONE | NONE | 0.1692 |
| 21 | furniture/equipment | domestic appliances | 0.1692 |
| 37 | business | repairs | 0.1691 |
| 36 | business | domestic appliances | 0.1691 |
| 16 | education | others | 0.1691 |
| 28 | car (new) | repairs | 0.1691 |

   We can also investigate the WoE values plot:



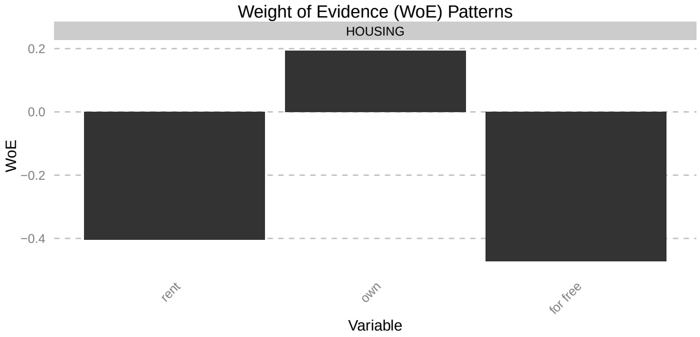
   Judging from the above study we concluded that it is reasonable to join $furniture/equipment$ and $domesticappliances$ levels and we performed it.

2. *HOUSING*

   Below we can see what is the *HOUSING* variable IV ($3^{rd}$ column) after joining a pair of levels ($1^{st}$ and $2^{nd}$ column).

| | lvl1 | lvl2 | iv |
|---|---|---|---|
| 4 | NONE | NONE | 0.0833 |
| 3 | for free | rent | 0.0830 |
| 1 | own | for free | 0.0389 |
| 2 | own | rent | 0.0296 |

   We can also investigate the WoE values plot:



   Judging from the above study we concluded that it is reasonable to join $for free$ and $rent$ levels and we performed it.

## 8.5 Removing variables of 0.0 Information Value

After the transformations described above, we chcecked what are IV for each of the resulting variable. The values are presented below.

| Variable | InformationValue | Bins | ZeroBins | Strength |
|---|---|---|---|---|
| CHK_ACCT | 0.6660 | 4 | 0 | Very strong |
| DURATION_TO_AGE | 0.3338 | 3 | 0 | Strong |
| HISTORY | 0.2932 | 5 | 0 | Strong |
| DURATION | 0.2293 | 3 | 0 | Strong |
| AMOUNT | 0.2251 | 4 | 0 | Strong |
| SAVINGS_ACCT | 0.1960 | 5 | 0 | Average |
| PURPOSE | 0.1692 | 9 | 0 | Average |
| AMOUNT_TO_DURATION | 0.1575 | 5 | 0 | Average |
| PROPERTY | 0.1126 | 4 | 0 | Average |
| AMOUNT_TO_AGE | 0.1082 | 2 | 0 | Average |
| EMLOYMENT | 0.0864 | 5 | 0 | Weak |
| HOUSING | 0.0830 | 2 | 0 | Weak |
| AGE | 0.0732 | 2 | 0 | Weak |
| OTHER_INSTALLMENT_PLANS | 0.0576 | 3 | 0 | Weak |
| STATUS_AND_SEX | 0.0447 | 4 | 0 | Weak |
| IS_FOREIGN_WORKER | 0.0439 | 2 | 0 | Weak |
| OTHER_DEBTORS | 0.0320 | 3 | 0 | Weak |
| RATE_TO_DISP_INCOME | 0.0239 | 2 | 0 | Weak |
| NUM_OF_THIS_BANK_CREDITS | 0.0101 | 2 | 0 | Wery weak |
| JOB | 0.0081 | 3 | 0 | Wery weak |
| TELEPHONE | 0.0064 | 2 | 0 | Wery weak |
| NUM_OF_MAINTAINED_PEOPLE | 0.0000 | 1 | 0 | Wery weak |
| RESIDENCE | 0.0000 | 1 | 0 | Wery weak |

We can see that two variables: $NUM\_OF\_MAINTAINED\_PEOPLE$ and $RESIDENCE$ have 0.0 information value. We decided to remove them from the data set as we assume they are kind of useless data.

## 8.6 Recoding variables to WoE

After the operations descripted above, we made a copy of our data and converted all of the categorical variables in this copy into their WoE representatives. Since then we have been working on two data sets:

1. *german_data_cat.txt* - categorized data set,
2. *german_data_woe.txt* - categorized and recoded to WoE data set (besides the response variable, this data set contains numeric variables only).

## 8.7 Data preprocessing summary

To conclude, the data preprocessing we performed above results in:

- creating derived variables and including them into the data set: $AMOUNT\_TO\_DURATION$, $DURATION\_TO\_AGE$ and $AMOUNT\_TO\_AGE$,
- binning continuous variables with the use of *smbinning* method and keeping them in those binned form in the data set; related to: $AGE$, $AMOUNT$, $DURATION$, $AMOUNT\_TO\_DURATION$, $DURATION\_TO\_AGE$ and $AMOUNT\_TO\_AGE$,

- correcting bins (levels) of categorical variables and keeping them in those transformed form in the data set: *PURPOSE* and *HOUSING*,
- removing variables of 0.0 information value: *NUM_OF_MAINTAINED_PEOPLE* and *RESIDENCE*,
- creating separated data set which contains all the categorical variables in the recoded to WoE form.

# 9 Feature Selection

## 9.1 Filter comparison

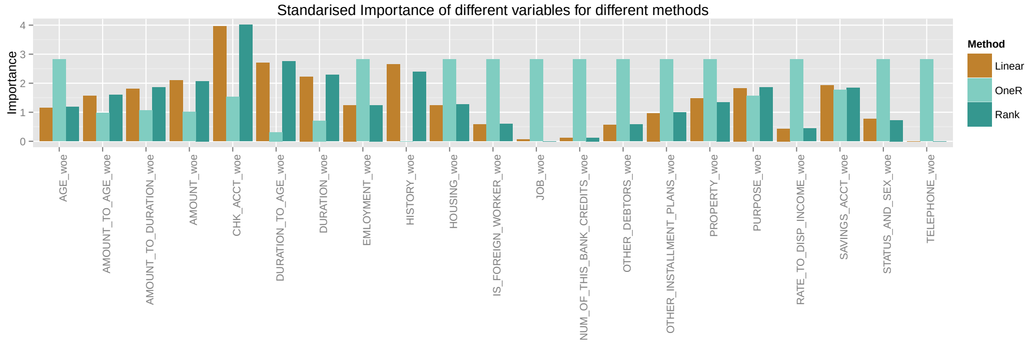


Figure 6: Histogram of the importance of different variables for woe transformation data

There had been involved many measures of the variable importance. Most of them are giving similar results. The most important Variable seems to be CHK_ACCT and the least important variables such as: IS_FOREIGN_WORKER (which is for 0.95 cases constant), TELEPHONE or JOB. Linear and Rank corelation methods gave also some idea about correlation WOE transformation of the data with the response variable.

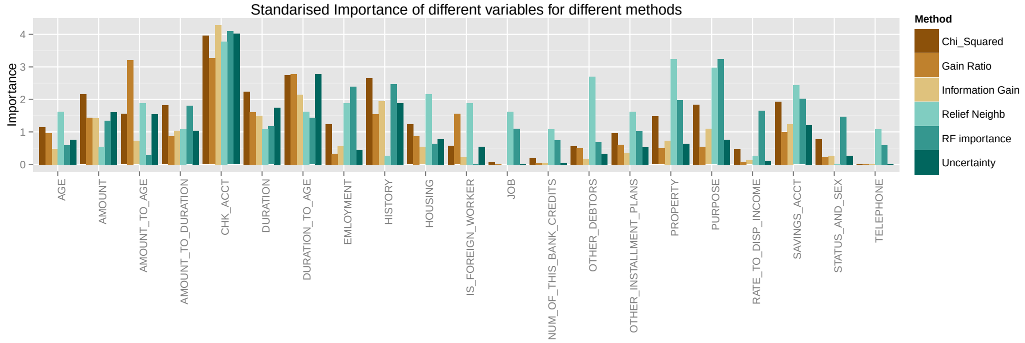


Figure 7: Histogram of the importance of different variables for categorical data

However most of the results for the diffenert methods are similar, some might be interesting. For example high contribution of the variables PURPOSE and PROPERTY difrenciate RF importance and eighbarhood methods from the others.

## 9.2 Models without features selection

| | Label | ACC | TPR | SPC | PPV | F1 | FDR |
|---|---|---|---|---|---|---|---|
| 1 | Boosting | 0.79 | 0.81 | 0.69 | 0.90 | 0.85 | 0.10 |
| 2 | Random Forest | 0.76 | 0.78 | 0.67 | 0.91 | 0.84 | 0.09 |
| 3 | AIC Logistic regression | 0.78 | 0.82 | 0.65 | 0.88 | 0.85 | 0.12 |
| 4 | Logistic regression | 0.78 | 0.82 | 0.67 | 0.88 | 0.85 | 0.12 |
| 5 | K-NN | 0.80 | 0.84 | 0.67 | 0.89 | 0.87 | 0.11 |
| 6 | Linear Dyscryminant Analysis | 0.78 | 0.83 | 0.63 | 0.88 | 0.85 | 0.12 |
| 7 | Quadric Dyscriminant Analysis | 0.78 | 0.87 | 0.59 | 0.82 | 0.84 | 0.18 |

Table 1: Basic statistics for models with all features

First step in the modeling part for this case had been fitting simple models for all features. It had been expected that soe of the methods would perform well under this constraints, as long as they have its own feature selection methodology. One of such methods is Random Forest. Suprisingly it had just 0.76 Accuracy. Method that were supposed to perform not to good (KNN) had extremally good Accuracy of 0.8.

## 9.3 CSF method

| | Label | ACC | TPR | SPC | PPV | F1 | FDR |
|---|---|---|---|---|---|---|---|
| 1 | AIC Logistic Regression | 0.76 | 0.82 | 0.60 | 0.83 | 0.83 | 0.17 |
| 2 | Logistic Regression | 0.75 | 0.76 | 0.66 | 0.93 | 0.84 | 0.07 |
| 3 | K-NN | 0.76 | 0.79 | 0.59 | 0.90 | 0.84 | 0.10 |
| 4 | Linear Dyscriminant Analysis | 0.78 | 0.83 | 0.61 | 0.87 | 0.85 | 0.13 |
| 5 | Quadric Dyscriminant Analysis | 0.80 | 0.84 | 0.65 | 0.88 | 0.86 | 0.12 |

Table 2: Basic statistics for models with features from CFS method

First regular method of subset selection had been CFS. Is gave conservative results of just few features for models:

- DURATION_TO_AGE, CHK_ACCT HISTORY and SAVINGS_ACCT for WOE transformation of the data
- DURATION_TO_AGE, CHK_ACCT and HISTORY categoregorical data

At this place interesting observation is that only 3 or 4 variables could give almost the same Accuracy as 21. According to Razor rule eventhough there is a small diffenence in the performance f the models, preffered would be those with much less predictors. QDA results with the accuracy 0.8 are even similar to those resulted from KNN using all variables.

## 9.4 Consistency method

| | Label | ACC | TPR | SPC | PPV | F1 | FDR |
|---|---|---|---|---|---|---|---|
| 1 | AIC Logistic Regression | 0.81 | 0.83 | 0.72 | 0.90 | 0.87 | 0.10 |
| 2 | Logistic Regression | 0.80 | 0.83 | 0.71 | 0.90 | 0.86 | 0.10 |
| 3 | K-NN | 0.79 | 0.83 | 0.63 | 0.88 | 0.86 | 0.12 |
| 4 | Linear Dyscriminant Analysis | 0.81 | 0.84 | 0.70 | 0.91 | 0.87 | 0.09 |
| 5 | Quadric Dyscriminant Analysis | 0.79 | 0.83 | 0.64 | 0.88 | 0.86 | 0.12 |

Table 3: Basic statistics for models with features from consistency method

Using consistency method there had been another set obtained. This time more quantitative.

- DURATION, AMOUNT, AMOUNT_TO_DURATION, DURATION_TO_AGE, AMOUNT_TO_AGE, PURPOSE, CHK_ACCT, HISTORY, SAVINGS_ACCT for WOE transformation of the data
- RATE_TO_DISP_INCOME, NUM_OF_THIS_BANK_CREDITS, AGE, DURATION_TO_AGE, PURPOSE, CHK_ACCT, HISTORY, SAVINGS_ACCT, EMLOYMENT, JOB , OTHER_DEBTORS, PROPERTY and STATUS_AND_SEX for categoregorical data

This time results are better than ever before. It shows the importance of the feature selection during the model fitting process.

## 9.5 Best subset method

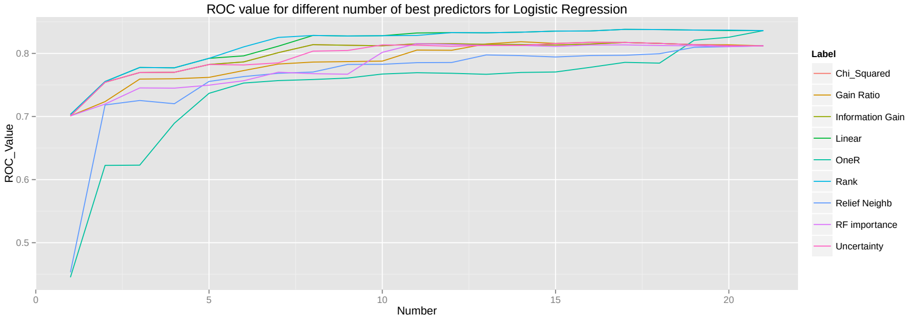


Figure 8: Comparison of the first n best variables for different methods using GLM model

We developed another method to choice the best and moderately big subset for modeling. Using all filters from the first subsection we went to show how good results would be acheived using just first n best predictors for given measure. Results are interesting. For some algorythms the adequate maximum is achieved for relatively small number of predictors. We choose subset having only 8 predictors based on Information gain. In following paragraphs we would see how does it work for different algorithms.

|   | Label | ACC | TPR | SPC | PPV | F1 | FDR |
|---|---|---|---|---|---|---|---|
| 1 | Logistic Regression | 0.81 | 0.84 | 0.72 | 0.90 | 0.87 | 0.10 |
| 2 | K-NN | 0.79 | 0.82 | 0.66 | 0.90 | 0.86 | 0.10 |
| 3 | Linear Dyscriminant Analysis | 0.80 | 0.84 | 0.66 | 0.89 | 0.86 | 0.11 |
| 4 | Quadric Dyscriminant Analysis | 0.79 | 0.86 | 0.63 | 0.86 | 0.86 | 0.14 |

Table 4: Basic statistics for models with features from filtered best method

First important and interesting information is that using consistency rule and best subset for the AIC based Logistic Regression we got the same results. They would be discussed more in next paragraphs.

# 10 Classification modelling results

## 10.1 Analysis of fitted models

At first we would show some comparison analysis between some of the fitted models. In the next sections there would be discussed each model separately. Interesting observation is that Random Forest lost a lot comparing to the Boosting model (GBM). Rest of the algorithms at first looks as tough the results are similar.

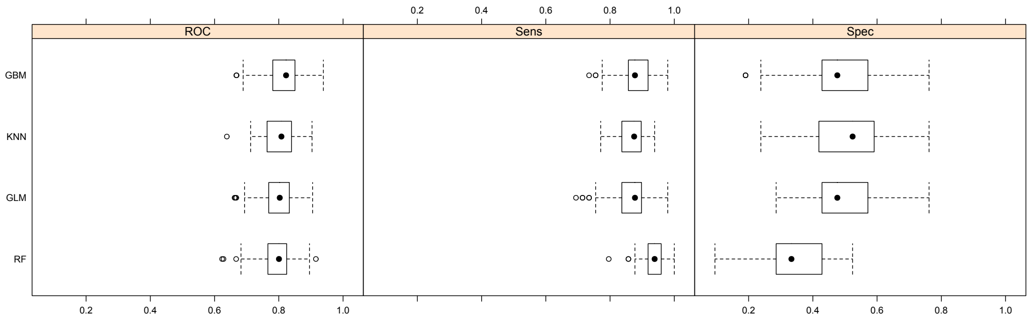


Figure 9: Histogram of the importance of different variables for categorical data

Here are boxplots of the ROC values, Sensitiviti, and Specifity. It had been calculated using 10 times 10-fold classification for all models. It looks like most of the models have similar statistics besides Specifity and Sensitivity for Random Forest. Given good values for Sensitivity it looses some value on the second index.

Nonetheless Random Forest looks worse at first sight, the values of the ROC are very simmilar to the Boosting model. The biggest skeewnes is visible for Spec and Sens for those two models. There are differences in the classifications. That could suggest that usage of ensemble method could effect in improvement of the model performance.

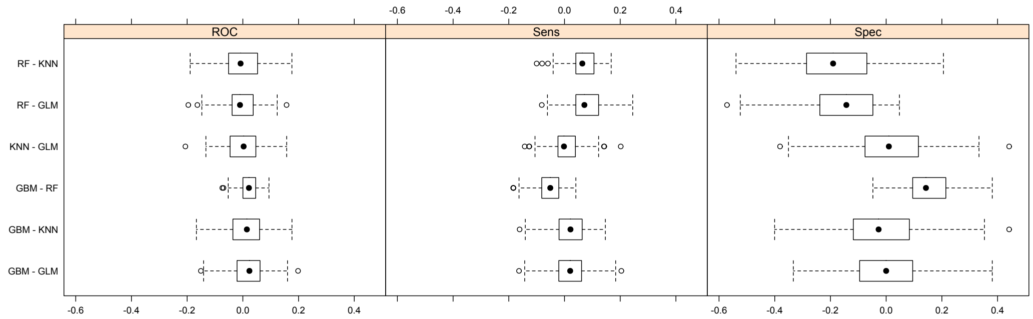


Figure 10: Comparison of the first n best variables for different methods using GLM model

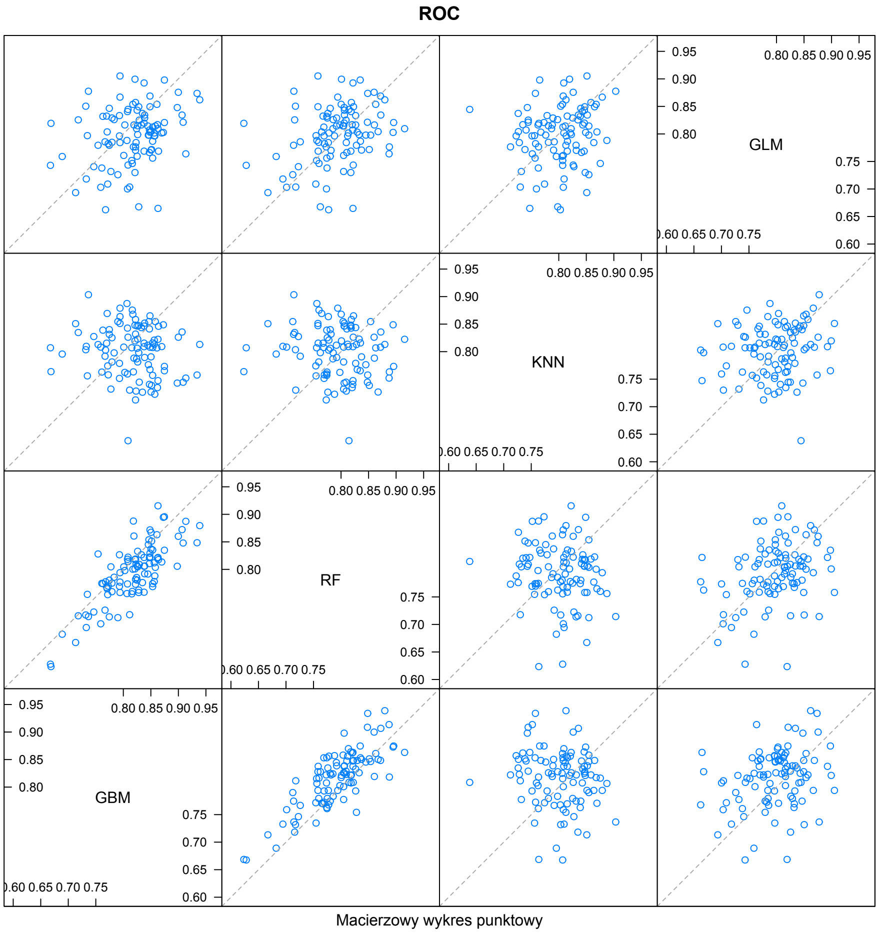


Figure 11: Histogram of the importance of different variables for woe transformation data

## 10.2 Logistic Regression

Logistic regression is the model that is used in industrie the most often. As it had been said before it thanks its popularity to easiness in interpretation of the results. Based oon this approach it is possible to produce scorecard which could be used to clasiffy approprietly cases.

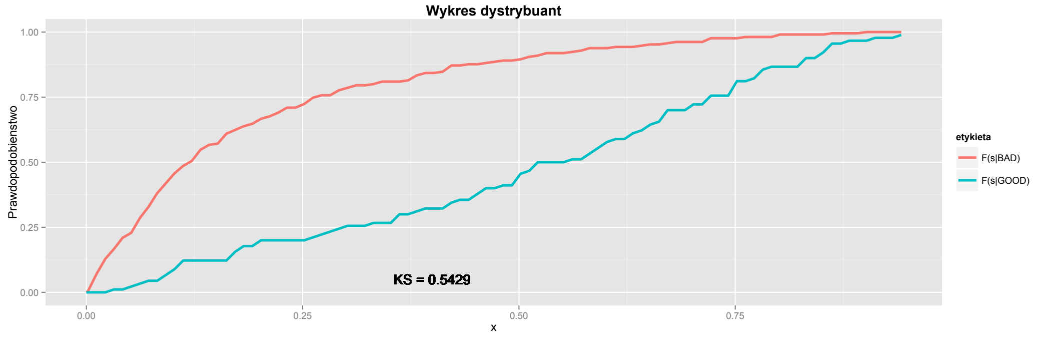
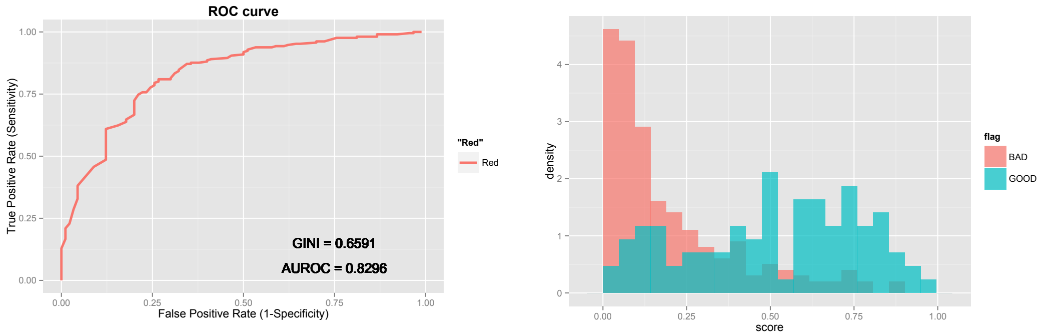


Figure 12: Distribution plot for good and Bad class with KS statistic for GLM method

Plot abowe shows distributions for good and bad cases. The KS- statistics is high. That indicates good separation between classes for this particular model.



(a)

Figure 13: Plot of Roc curve and histogram for GLM method

Another two plots - ROC curve and histogram can give us better understanding of the model performance. AUROC with value 0.83 is said to be really good. Interesting is that most of the bad values having very small score values.

All simulations in this section had been done for the testing data sets.

## 10.3 Scorecard for Logistic Regression

| | Variable | Value |
|---|---|---|
| 1 | (Intercept) | 511.93 |
| 2 | ‘CHK_ACCT>=200 DM‘ | -30.32 |
| 3 | ‘CHK_ACCT0-200 DM‘ | -14.95 |
| 4 | ‘CHK_ACCTno checking account‘ | -50.64 |
| 5 | ‘DURATION_TO_AGE01 DURATION_TO_AGE <= 1.21875‘ | 26.56 |
| 6 | ‘DURATION_TO_AGE02 DURATION_TO_AGE <= 3‘ | 67.92 |
| 7 | ‘HISTORYcritical account/ other credits existing (not at this bank)‘ | -52.15 |
| 8 | ‘HISTORYdelay in paying off in the past‘ | -37.21 |
| 9 | ‘HISTORYexisting credits paid back duly till now‘ | -36.26 |
| 10 | ‘HISTORYno credits taken/ all credits paid back duly‘ | -6.03 |
| 11 | ‘DURATION01 DURATION <= 33‘ | 12.13 |
| 12 | ‘DURATION02 DURATION <= 72‘ | 2.37 |
| 13 | ‘AMOUNT01 AMOUNT <= 3913‘ | -59.83 |
| 14 | ‘AMOUNT02 AMOUNT <= 7824‘ | 14.85 |
| 15 | ‘AMOUNT03 AMOUNT <= 18424‘ | 47.81 |
| 16 | ‘SAVINGS_ACCT>=1000 DM‘ | -28.61 |
| 17 | ‘SAVINGS_ACCT100-500 DM‘ | -1.22 |
| 18 | ‘SAVINGS_ACCT500-1000 DM‘ | -14.45 |
| 19 | ‘SAVINGS_ACCTunknown/ no savings account‘ | -30.90 |
| 20 | ‘PURPOSEcar (new)‘ | 16.19 |
| 21 | ‘PURPOSEcar (used)‘ | -20.29 |
| 22 | PURPOSEeducation | 31.75 |
| 23 | ‘PURPOSEfurniture/equip_domestic applia‘ | 11.91 |
| 24 | PURPOSEothers | -10.23 |
| 25 | ‘PURPOSEradio/television‘ | -4.87 |
| 26 | PURPOSErepairs | 12.99 |
| 27 | PURPOSEretraining | -50.63 |
| 28 | ‘AMOUNT_TO_DURATION01 AMOUNT_TO_DURATION <= 250.944444444444‘ | -24.12 |
| 29 | ‘AMOUNT_TO_DURATION02 AMOUNT_TO_DURATION <= 291.583333333333‘ | -48.23 |
| 30 | ‘AMOUNT_TO_DURATION03 AMOUNT_TO_DURATION <= 358.090909090909‘ | -54.94 |
| 31 | ‘AMOUNT_TO_DURATION04 AMOUNT_TO_DURATION <= 2482.66666666667‘ | 3.82 |

Table 5: Scorecard table

Scorecard gives us some information about contribution of single variables in the model. The bigger value off the score the bigger chance of client to be good. The calibration for this scorecard is standard wich means that for point value of 500 chance for being good is 0.5.

Big contribution in this scorecard has for example AMOUNT. However interesting is that middle Amount gets the most points and oth smaller and bigger are worse.

Here is important to note that for models in this section excluding Random Forest and Boosting the same variablees set had been used as for Logistic Regression.

## 10.4 Linear Dyscryminant Analysis

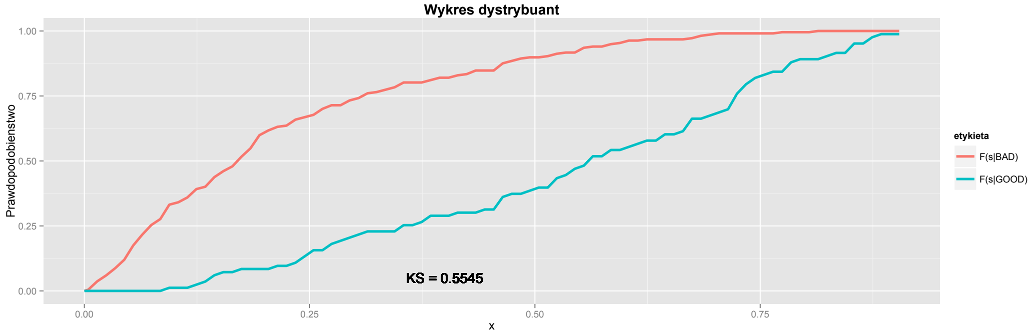
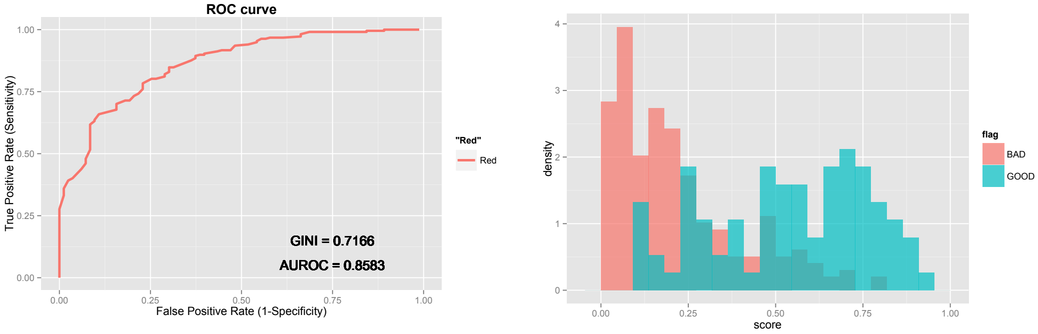


Figure 14: Distribution plot for good and Bad class with KS statistic for LDA method

It is also important to be addressed that the data for this model had been transformated into the WOE. It is also assumed that appropriate assumptions are fulfilled. With the value of KS styatistics 0.5545, LDA gives much better results than Logistic Regression.



(a)

Figure 15: Plot of Roc curve and histogram for LDA method

ROC curve also gives better results. AUROC statistic is equal 0.8583 which is very high. For the histogram interestting is that for two smallest classes there is no good observation in the bands. The same there is no bad observations for very large scores.

## 10.5 KNN Method

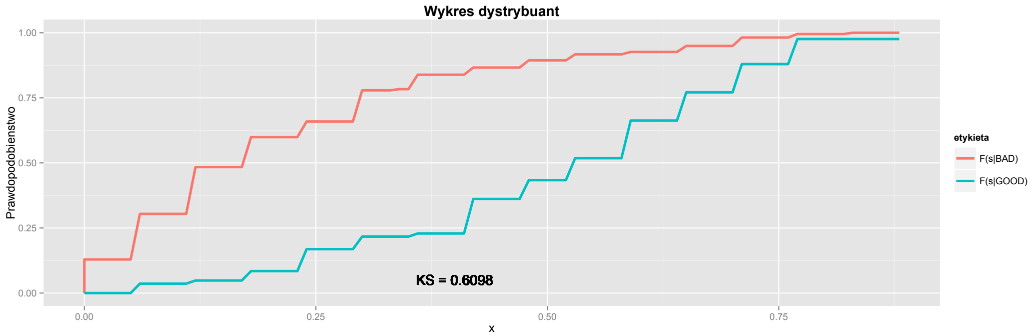
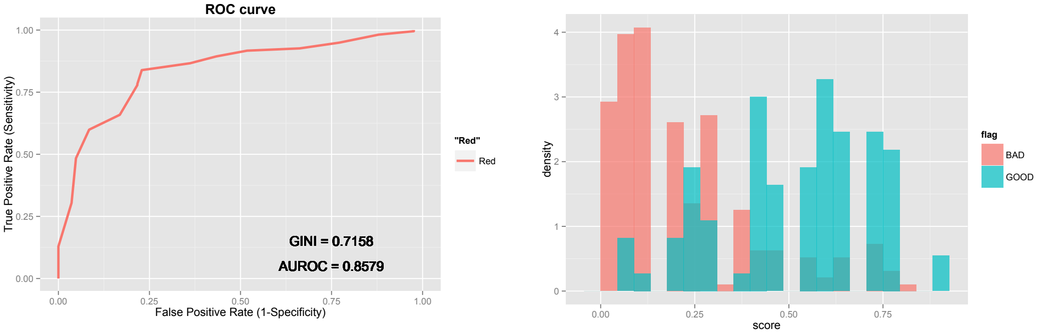


Figure 16: Distribution plot for good and Bad class with KS statistic for KNN method

Similarily to the LDA method, KNN had been produced using WOE transformation of the data. The set of features is choosen to be the best for this task. nonethelss, estimation for the all predictor variables included gave better results. This could have effected in long time of computation for bigger data sets.



(a)

Figure 17: Plot of Roc curve and histogram for KNN method

On the histogram we could observe sparse bins. It is caused by small number of different possible differences between distances for different observations. AUROC value is moderatelly good.

It is important to say that tuning of the number of neighbours used in this model had been performed using 10 times repeated 10-k fold crossvalidation. The best model had been one with 19 neighbours.

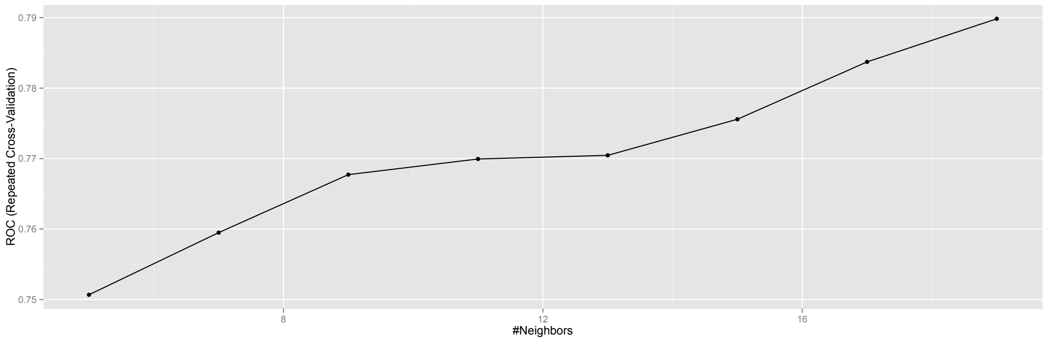


Figure 18: Tuning of the nieghbours number ROC results

Trend for different number of neighbours is visible upsising. We can expect that for bigger number, algorithm could have even better results, However for 700 observations in the training set, bigger number of observations could lead to overfitting of the model. Differences in the models performance are visible but not extrime.

## 10.6 Random Forest

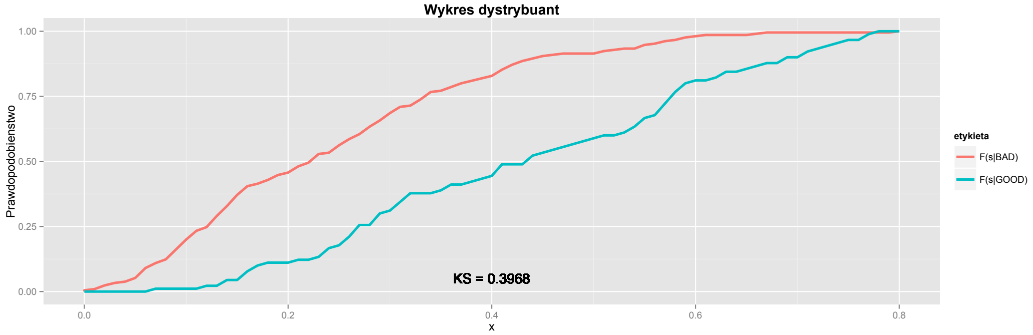
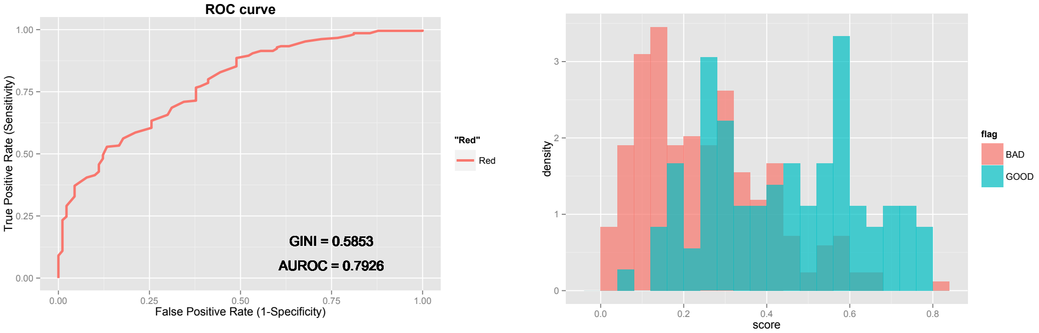


Figure 19: Distribution plot for good and Bad class with KS statistic for RF method

Suprising fact is that random forest model is behaving not to good. Separation based on the KS statistic is very weak and much worse than simple logistic regression. It could be caused also by over fitting for this model. Even though it is overfitted it performs moderately well.



(a)

Figure 20: Plot of Roc curve and histogram for RF method

For this figure we see aggain that comparing to the other models, RF gives bad results. On the histogrm, even for small score bands there is a lot of the Good observations.

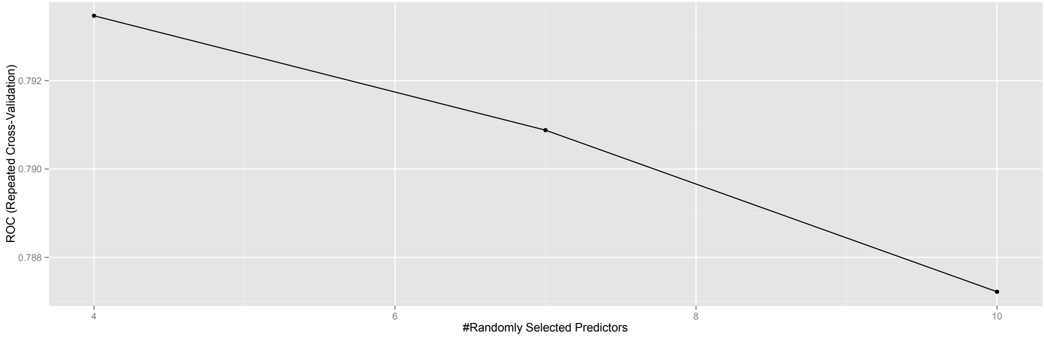


Figure 21: Tuning of the trees number ROC results

Tuning is done similarily to the KNN model. There had been performed 10 times repeated 10 fold crossvalidation. This figure shows that for the bigger number of the trees for single step the power of the model is falling. It shows also that the best model and at the same time final model had been one with the 4 features sampeled at each time.

# 11 Boosting

Boosting is another benchmark method. It had not been considered at beggining but as a extremaly good considered method it would serve as benchmark to compare with the other methods improved with the feature selection. For this method ther was no step of feature selection. Model had been tuned and calculated for the full data set.

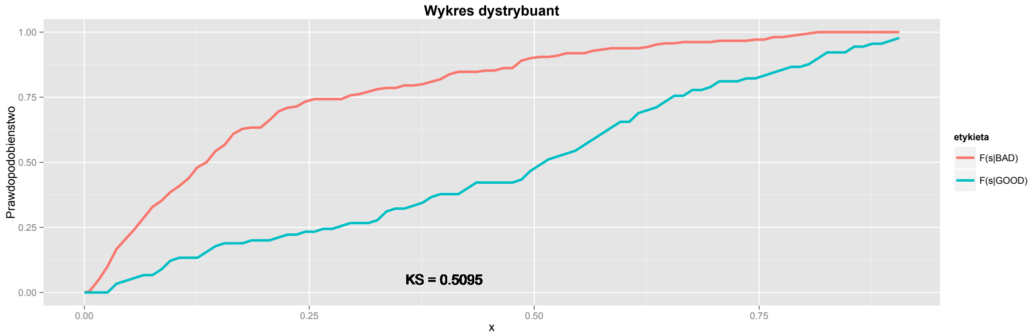
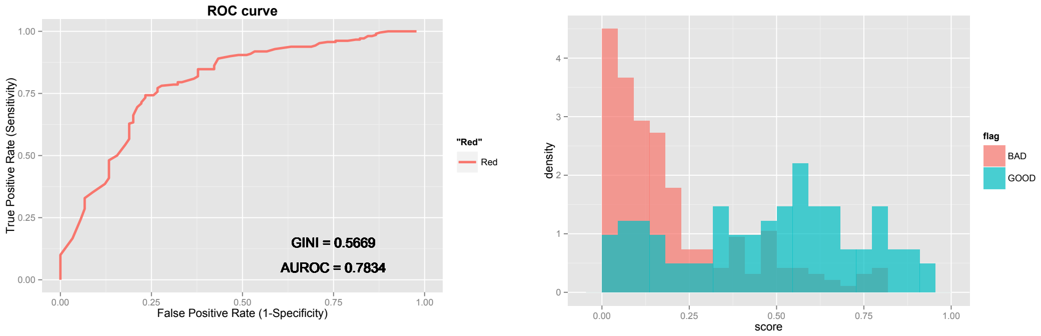


Figure 22: Distribution plot for good and Bad class with KS statistic for GBM method

Similarily with the Random Forestt method in this case results are not too good comparing with other methods. Although the KS Value is better than for the RF, it is still smaller than any other method with performed feature selection.



(a)

Figure 23: Plot of Roc curve and histogram for GBM method

The histogram for this model is even worse than for the Random Forest. For first four scorebands number of Good observations is very high. Value of AUROC is good.

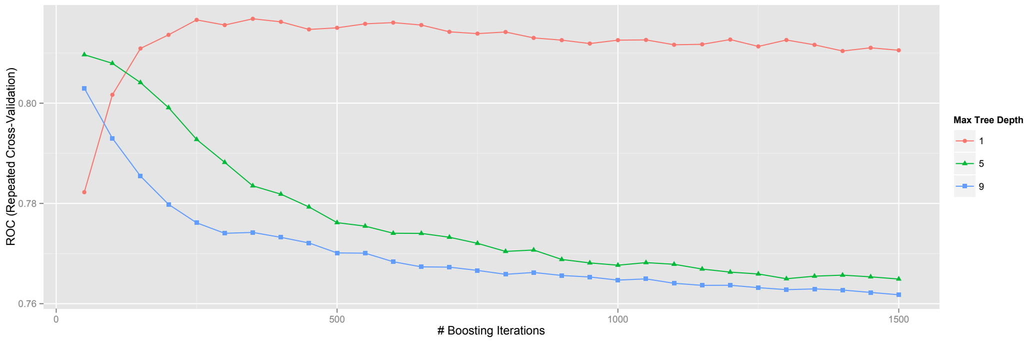


Figure 24: Tuning of the tree depth ROC results

For the boosting method there are two tuning parameters. Number of iterations in the algorithm and depth of the tree. We could see that for the deepth equal just one the results are the best. What is more, there is no significant improvement for bigger iteration number than 500 for any algorithm.

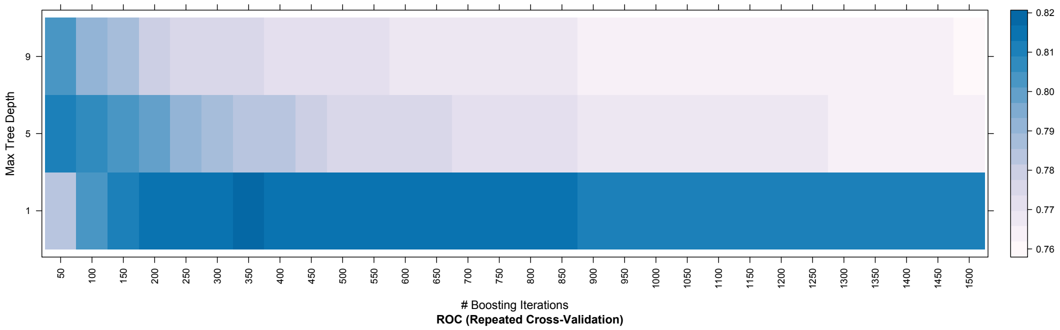


Figure 25: Tuning of the tree depth ROC results

What is more all of the algorithms are getting worse at some point. It could mean that for such values we start to overfit (the extreme observatioins are tried to be fitted better what results in drop of power of the algorithm.) Best tune had been achieved for the 250 iterations with depth just 1. ROC for this setup is being estimated as around 0.8. oFor Bigger depths of the trees we expect big overfit.

## 11.1 Summary of Modeling and Feature Selection

In last two sections there had been many interesting patterns discovered. The most interestting results sre that:

- Ofen including not relevant predictors in the data set could effect in the loss of the power for the model.
- Even for very small predictors set, results could be very optimistic as it had been shown for CFS set of features.
- Proper feature selection can boost Logistic Regressioin to be better than methods with automatic feature selection and parameter tuning.
- Using Scorecard interesting relations within the data can be discovered. It also sometime can be used to manipulate with the score. For example what happened if I am Married , would I get higher or lower credit offer.

# 12 Cluster analysis results

In this section we present our data clustering analysis results. In analysis, we assume a priori knowledge about the number of expected clusters (which equals 2) and we want to investigate:

- How cluster compactness, connectedness, separation and stability depend on feature selection performed before data clustering and how they depend on clustering method choice?
- Whether or not employing dimensionality reduction PCA method improves the above measures?
- How accurate is the clustering result in terms of separating BAD and GOOD observations (default and no default clients)?

In general, we follow the believe that first of all we want clustering schema to be of high stability - we would not trust even some 'beautiful' results if they come from the method which has turned out to be unstable in the analysis. Moreover, we decided to narrow down the analysis to the data set with variables transformed to WOE, as we believe it brings more information about variable levels (normalized, numerical relation) than factor variables (where something either is or is not of the same level for a given variable).

## 12.1 Cluster validation

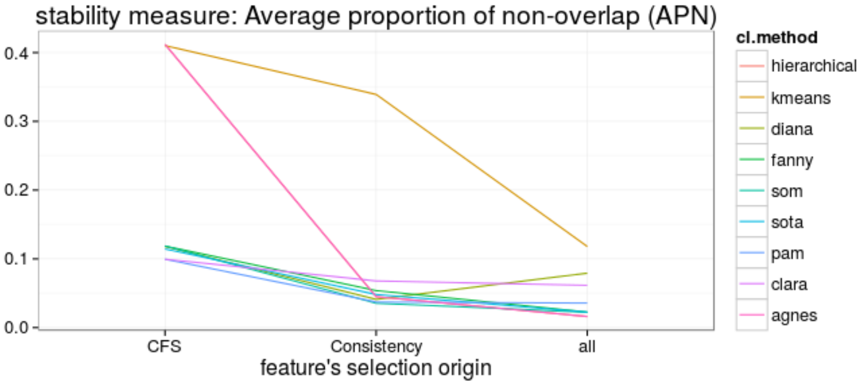
We performed the comparision of:

- clustering different data subsets that were created with the use of feature selection results from the previous section:
  - *CFS* method for WOE recoded data (variables included in the data set: DURATION_TO_AGE, CHK_ACCT, HISTORY and SAVINGS_ACCT),
  - *Consistency* method for WOE recoded data (variables included in the data set: DURATION, AMOUNT, AMOUNT_TO_DURATION, DURATION_TO_AGE, AMOUNT_TO_AGE, PURPOSE, CHK_ACCT, HISTORY, SAVINGS_ACCT),
  - *all* variables from the data set,
- clustering with the use of different algorithms.

In analysis, we assume the default `clValid::clValid` method parameters except from the fact that we use `method = "ward"` as the agglomeration method used for hierarchical clustering (`hclust` and `agnes`). The metrics used for computing dissimilarity matrix is `"euclidean"`.

### 12.1.1 Stablility validation

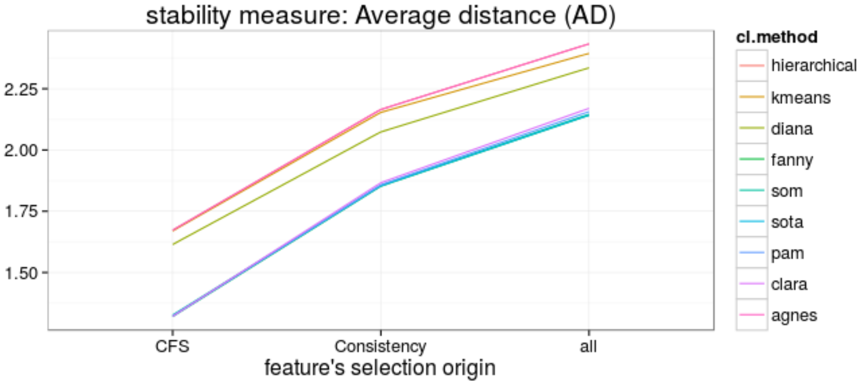
Below we can see 4 plots presenting stablility validation comparision results for clustering algorithms, across 3 different subsets of data set (*CFS*-based selection, *Consistency*-based selection, *all* variables (full data set)).



### Average proportion of non-overlap (APN)

We can see that:

- In general, average proportion of non-overlap is decreasing as the number of variables in the data set is growing - when we use *all* variables in the data set, we obtain highly consistent clustering results for the majority of algirithms. However, for majority of algorithms those differences depending on feature selection are not striking.
- *kmenas* method provides us with distinctively high values for the cases when not all variables are used. The same situation appears with *agnes* when clusters are built on a few variables (*CFS* features selection origin.)

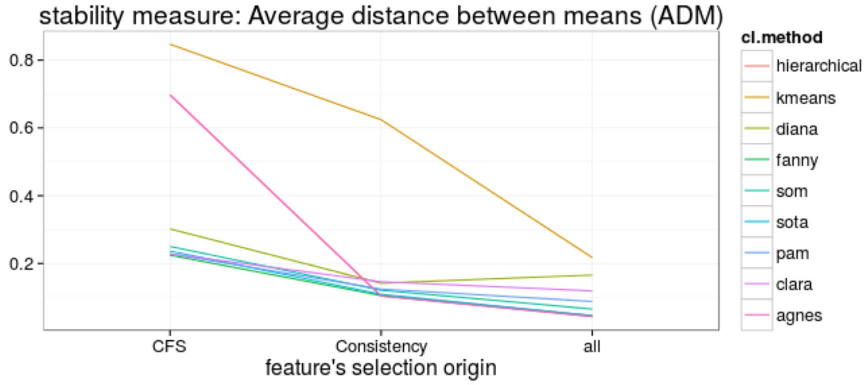


### Average distance (AD)

We observe that:

- The average distance between observations placed in the same cluster is growing with the growth of number of variables in the data set (which is quite intuitive).

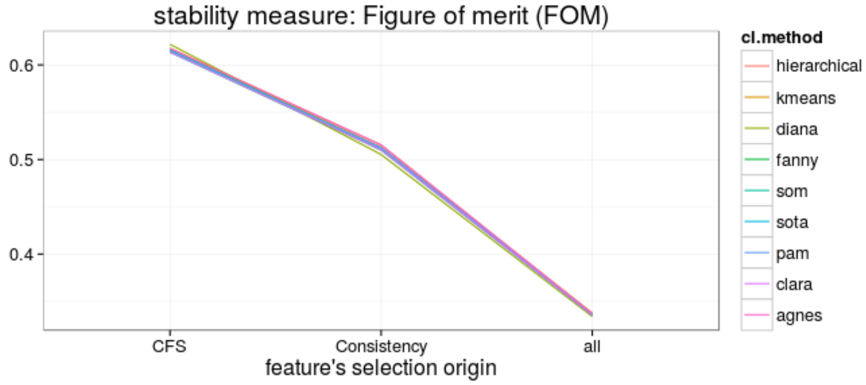
- We can also see like two groups of algorithms with similar performance. "Worse" group is the group with *kmeans*, *diana* and *agnes* algorithms (higher AD values).



### Average distance between means (ADM)

We see that:

- ADM measure gives results very similar to those of APN measure. Similarly, as smaller measure values are prefered, we see that *kmenas* and *agnes* methods provide us with relatively high values in case when clusters are built on a few variables.



### Figure of merit (FOM)

We see that:

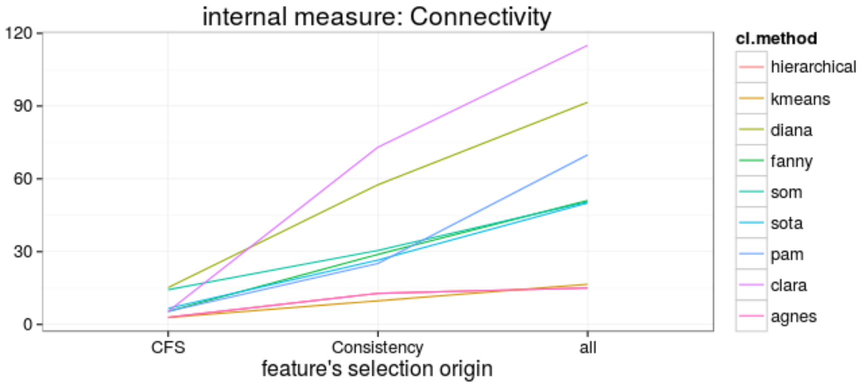
- FOM measure results depend not really much on algorithm selection; they depend rather on the number of variables taken into clustering analysis.

- We observe that with the growth of the number of variables in the data set, FOM measure is decreasing. We quess that it may be the result of incorporating the scaling factor (mentioned in the measure description).

To sum up, whereas it is hard to conclude whether or not more or less variables should be used in the clustering (without any suggested threshold for the measure values), we can make an observation that *kmeans*, *agnes* and *diana* are the algorithms that performed worse than the others in terms of at least one measure.

### 12.1.2 Internal validation

We continue the clustering validation with the use of internal measures. Below we can see 3 plots presenting internal validation comparision results for clustering algorithms, across 3 different subsets of data set.

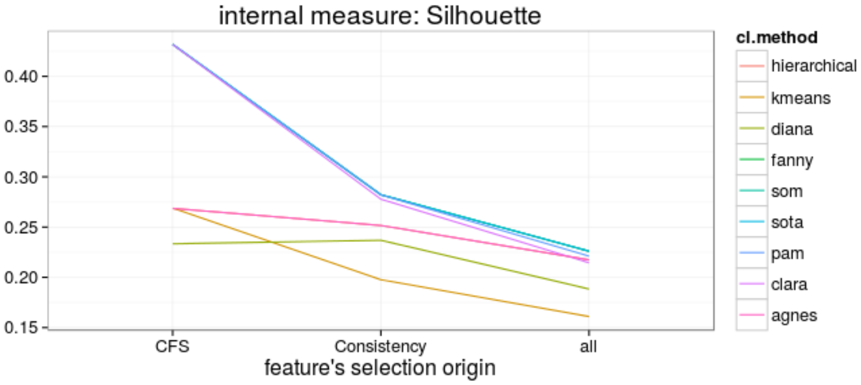


#### Connectivity

We want to minimize the value of Connectivity - when minimized, it suggests that observations are placed in the same cluster as their nearest neighbors in the data space quite frequently.

We observe that:

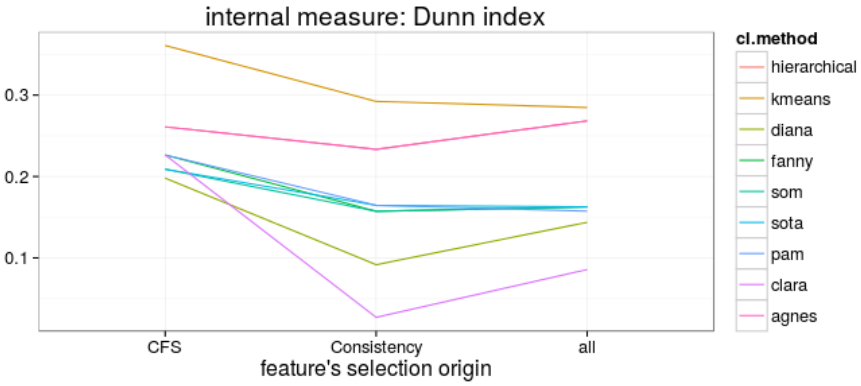
- We can distinguish about 3 groups of algorithms with similar Connectivity values tendency. We can see that *clara* and *diana* have higher Connectivity values, and *agnes*, *hierarchical* (they are overlaping) and *kmenas* have relatively lower Connectivity values.



### Silhouette

As objects with a high silhouette value are considered well clustered, we can conclude that:

- silhouette value is the lowest for all algorithms in the case where *all* varaibles are in the data set,
- *som*, *sota*, *pam* and *clara* have relatively high silhouette values.



### Dunn Index

As we remember, since internal criteria seek clusters with high intra-cluster similarity and low inter-cluster similarity, algorithms that produce clusters with high Dunn index are more desirable. From the plot presented above we may conclude that:

- What is interesting, the number of variables (in the data set subset) seems to have much lower impact on Dunn index than the clustering algorithm choice.
- *kmenas*, *hierarchical* and *agnes* have relatively high Dunn Index values.

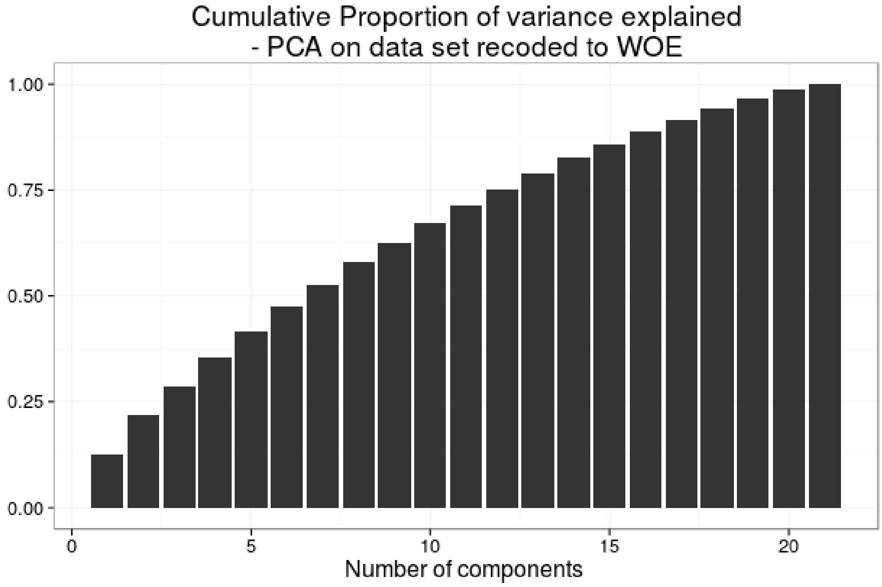
It is hard to conclude that one / a few algorithms give better validation results that others *in general* - in some cases some methods performs better and in other cases the same methods seem to be among the worst. We find it difficult to put up with some straightforward suggestions and see this situation as an argument in favour of the opinion that clustering is difficult in its plurality of possible approaches and interpretations.

### 12.1.3 Incorporating the dimensiality reduction PCA algorithm

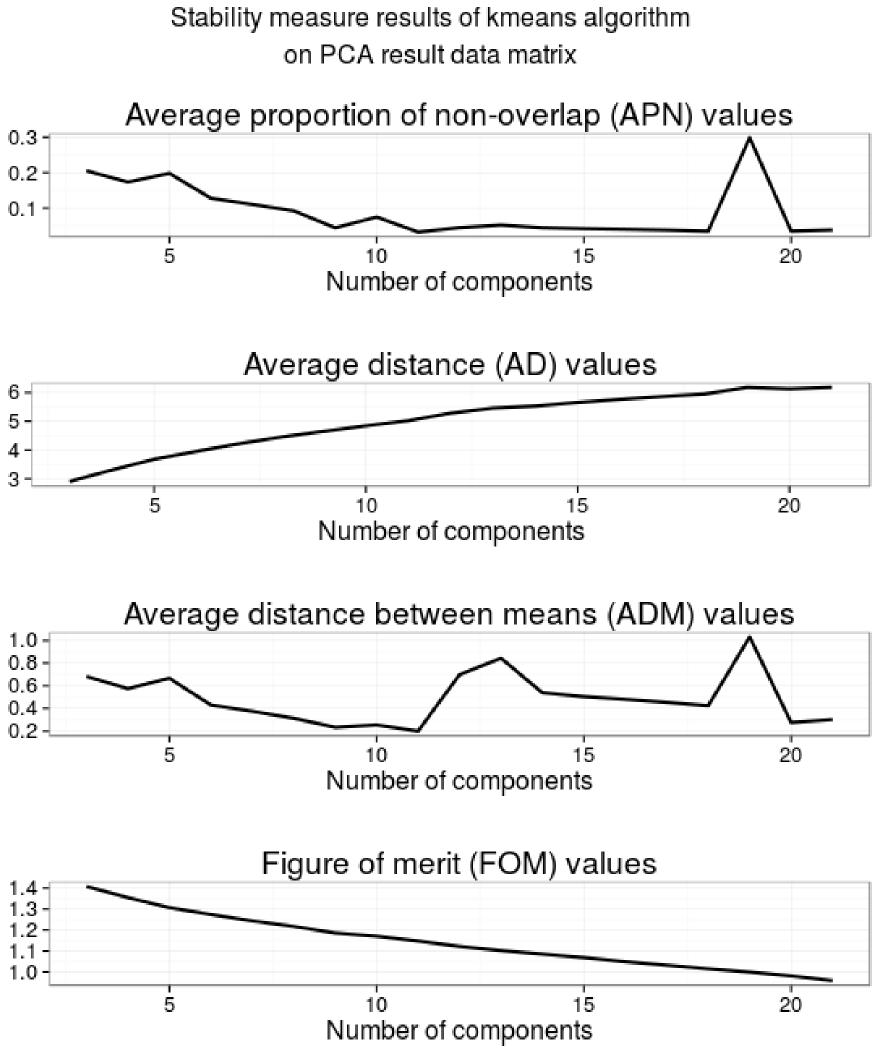
In this subsection we present results of our attempt to incorporate the dimensionality reduction PCA algorithm into clustering analysis. Our primary goal is to observe how the number of principal components kept in the data set affects the cluster validation results.

We performed dimensionality reduction on data set with variables recoded to WOE with the use of `stats::prcomp` method. As before, we investigate only the cases that results in 2 clusters output.

Below we can see a plot presenting cumulative proportion of variance explained, for subsequent numbers of PCA output components included. We can observe that we need 5 first components (out of all 21) to explain 50% variance after the transformation and 12 first components to explain 75% variance. It may suggest that we can have different patterns in our data to explain which we may need relatively large part of all varaibles avaliable.



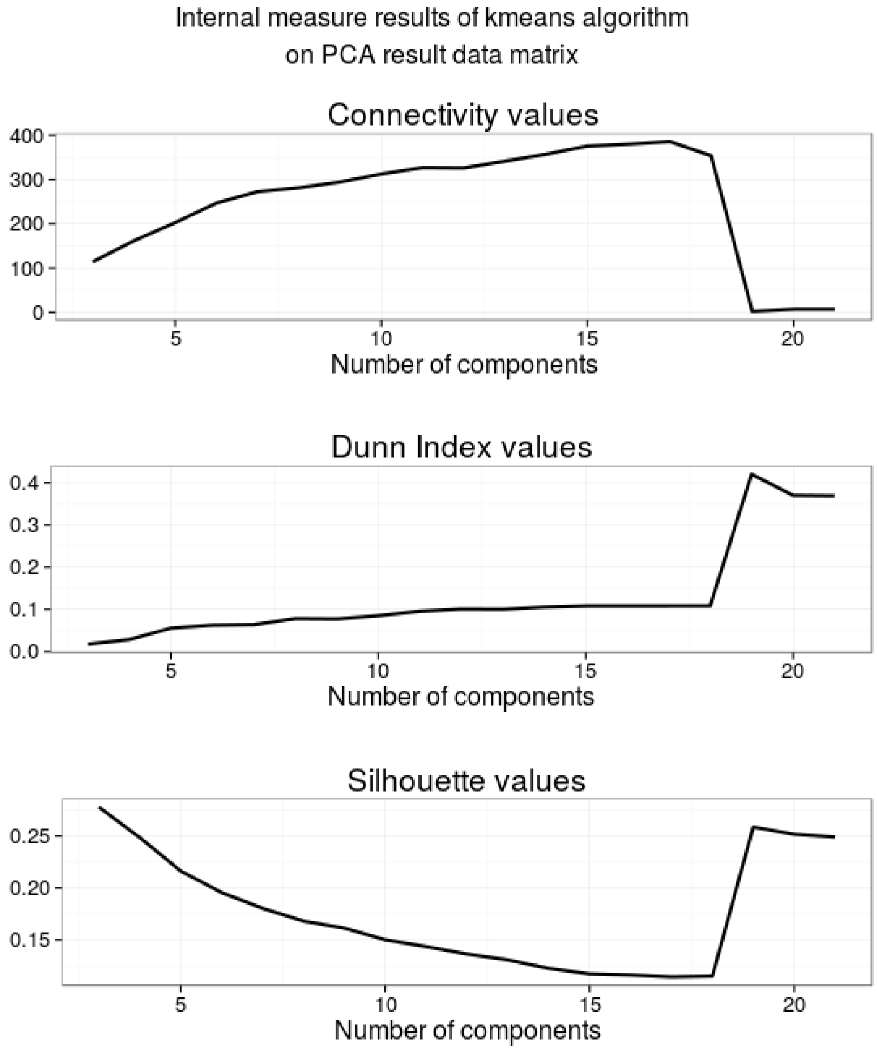
Below we can see a plot presenting stability measure results for clusters built with the use of *kmenas* algorithm, for different numbers of components included in the input data set.



**Stability measures comparision**

From the above plot we can observe that:

- With the increase of number of components included in the data set the AD values tends to increase (bad thing) and FOM values tends to decrease (good thing), both in a "regular" way. APN values decrease (good thing) with the number of components increasing, but the plot behaves "irregularly" for the large number of components (note "peak" around the 19 value on OX axis). ADM values does not seem to depict any pattern in this comparision.



**Internal measures comparision**

From the above plot we can observe that:

- There is an interesting plots "pattern regularity" decline: in each of three cases we can see regular increase / decrease of the measure values with the number of components increasing, but this regularity goes down from 19 value on the OX axis on.
- Before the "pattern regularity" decline mentioned above, we can see that Connectivity values increase (bad thing) with the number of components. Dunn Index is sightly increasing (good thing) whereas Silhouette values are decreasing (bad values).
- After "pattern regularity" decline, all of the 3 compared values are moving very quickly in the desired direction (they increase in case of Dunn Index and Silhouette measure and they

decrease in case of Cinnectivity). It may suggest that the data is really complex (as the cumulative proportion of variance explained plot suggested) and we receive good unternal measures results when we include relatively large number of components.

Once again, we find it difficult to conclude the results from the above comparision. Nevertheless, the "pattern regularity" decline noticed on the internal measures comparision plot makes us inclined to make use of PCA output with 19 components in further analysis.

### 12.1.4 External validation

In this point of our analysis we planned to employ an algorithm with best stability and internal measure performance so as to build clusters and investigate their practical properties (by which we mean external validation with the use of Rand Index and other measures).

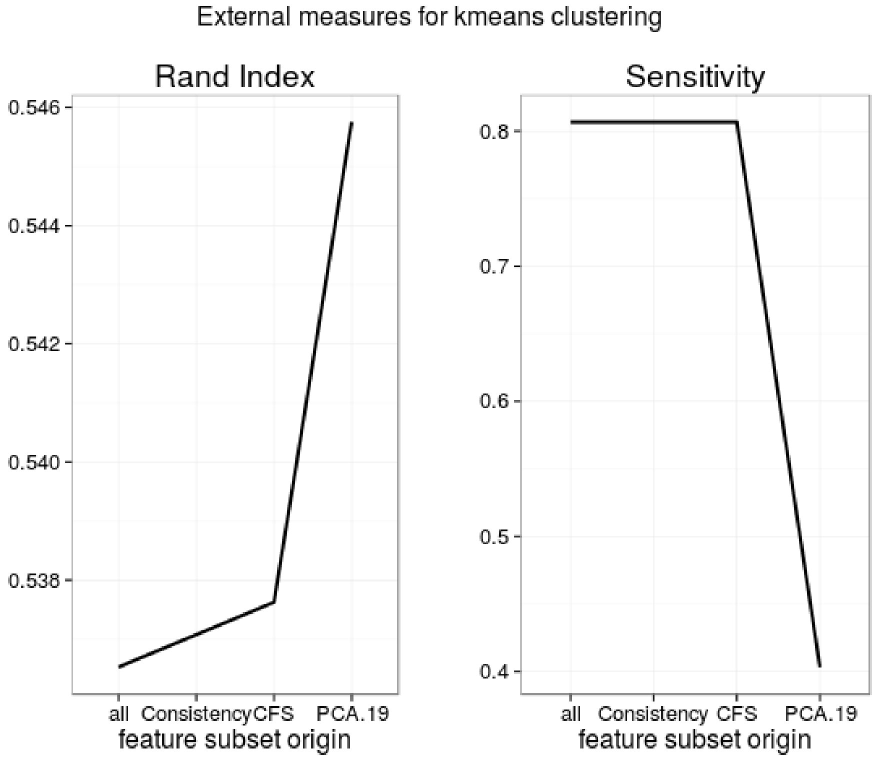
However, we were not sure to decide on the "best" algorithm; eventually, we decided to perform comparision with the use of *kmeans* algorithm, as it presented good properties in some cases in the previous comparision and because of the fact it is quite commonly used algorithm, in general.

The plots below present Rand Index values and Sensitivity values for *kmeans* clustering performed on data subsets of different origin (see OX axis labels).

- Cluster labes and real response variable labels are matched in self-written function which checks which of two resulted clusters has greater *fraction* of BAD observations (defaulted clients). This cluster is assumed to be the cluster that gathers BAD observations and it is assigned a label equal with the label of real response variables BAD value (here: $RES = 1$). Respectively, the other cluster is assumed to be the cluster with GOOD observations and it is assigned a label equal with the label of real response variable GOOD value (here: $RES = 0$).

- Sensitivity is computed as the fraction: (# True Positives)/(# True Positives + # False Negatives).

We can see from the plot above that Rand Index values do not vary that much across different methods of subset feature selection. When it comes to Sensitivity, we have similar (quite high) results except from clusters obtained from $PCA$ transformation with 19 components included in the input data set.


External measures for kmeans clustering
Rand Index
0.546
0.544
0.542
0.540
0.538
all
Consistency
CFS
PCA.19
feature subset origin
Sensitivity
0.8
0.7
0.6
0.5
0.4
all
Consistency
CFS
PCA.19
feature subset origin

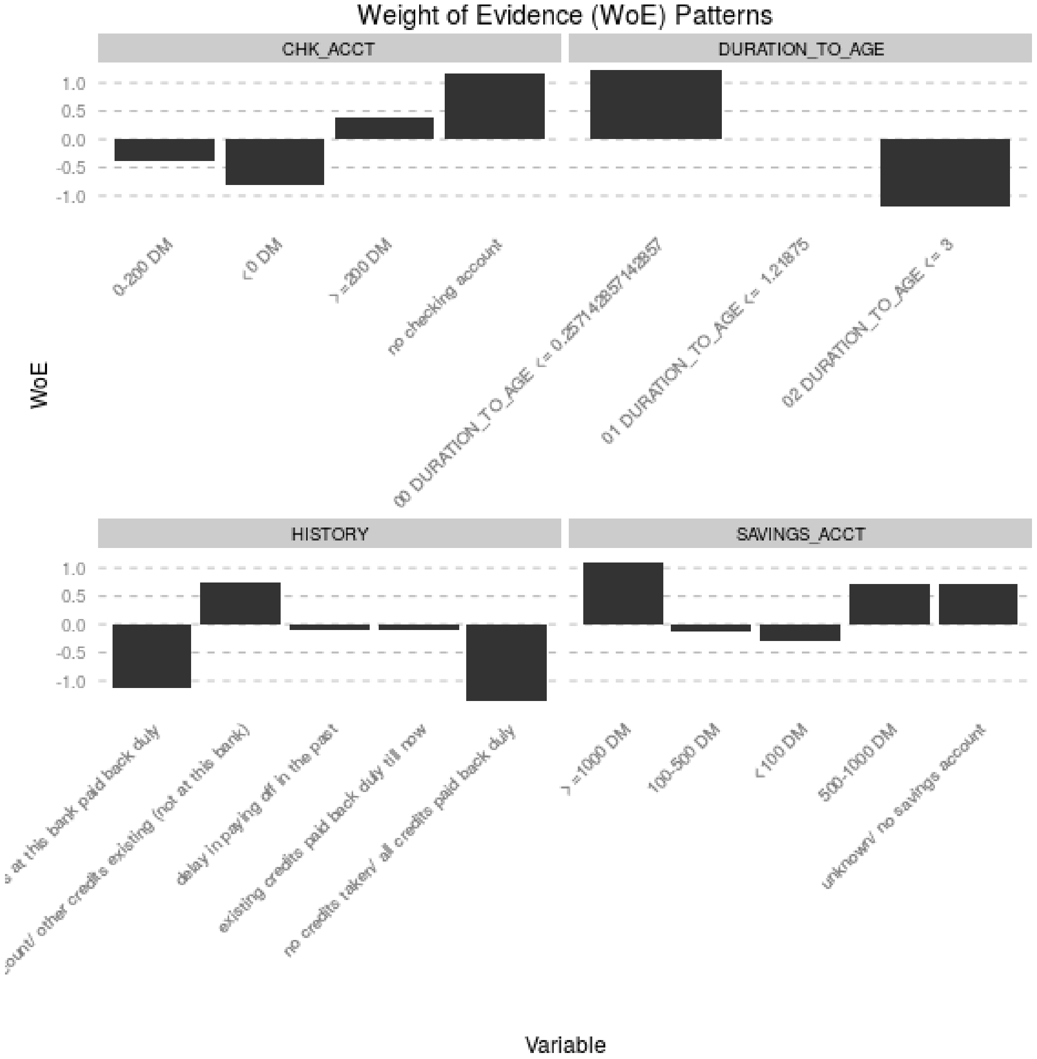
## 12.2 Cluster representative patterns

In this section we give an example of data abstraction - extracting a simple and compact representation of a data set. We present a centroids of each of two clusters resulted from *kmeans* algorithm performed on data set recoded to WOE, with feature selection based on the *CFS* method result. Set of selected variables includes: DURATION_TO_AGE_woe, CHK_ACCT_woe, HISTORY_woe and SAVINGS_ACCT_woe. The centroids are summarized in the table below.

| cluster | DURATION_TO_AGE_woe | CHK_ACCT_woe | HISTORY_woe | SAVINGS_ACCT_woe |
|---|---:|---:|---:|---:|
| default | 0.012 | -0.606 | -0.054 | -0.041 |
| no default | 0.136 | 1.074 | 0.177 | 0.155 |

We can obtain more "human-redable" interpretation of these centroids with the use of relation between WOE and levels of categirized variables - by investigating the plot above.

We can see that the differences in WOE values for particular varaibles between centroids of the two clusters do not differ that much, which is quite disappointing.


Weight of Evidence (WoE) Patterns
CHK_ACCT
DURATION_TO_AGE
HISTORY
SAVINGS_ACCT
WoE
1.0
0.5
0.0
-0.5
-1.0
0-200 DM
<0 DM
>=200 DM
no checking account
00 DURATION_TO_AGE <= 0.257142857142857
01 DURATION_TO_AGE <= 1.21875
02 DURATION_TO_AGE <= 3
s at this bank paid back duly
count/ other credits existing (not at this bank)
delay in paying off in the past
existing credits paid back duly till now
no credits taken/ all credits paid back duly
>=1000 DM
100-500 DM
<100 DM
500-1000 DM
unknown/ no savings account
Variable

## 12.3 Cluster analysis summary

Cluster analysis appears to be quite difficult to perform as there are a lot of *different* measures indicating cluster "quality", which are not easy to interpret. Moreover, we can see that among 8 different clustering algorithms that we have compared, there is no one approach that "beats" others in each of the validation comparision.

External validation of the *kmeans* algorithm performance shows about 0.8 Sensitivity score for some data subsets (with out approach of matching real and cluster labels), which is rather satysfying. However, in our opinion the whole procedure requires to much of arbitrary assessments to employ it as a "serious" source of information in credit scoring decision process.

## Part IV
# Discussion

There are a lot of different areas that might be investigated further when it comes to clustering analysis results.

- One of the major issues is the cluster algorithms parameter tunning. We expect the methods to perform better in terms of both internal and external measures after more careful arguments selection.
- One may find it interesting to investigate how the results are affected by the metrics choice in computing similarity / dissimilarity objects describing the data.
- Another interesting thing to compare would be the results of clutering without the restriction to the 2-cluster results. Maybe there are better (in terms of quality) outputs, containing 3 or more clusters?

# References

[1] Computational Methods of Feature Selection (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series), Huan Liu, Hiroshi Motoda, 2007, ISBN-13: 978-1584888789

[2] Hall, M. A., Smith, L. A. (1998). Practical feature subset selection for machine learning. Australian Computer Science Conference. Springer. 181-191.

[3] Liu, H. and Setiono, R., Chi2: Feature selection and discretization of numeric attributes, Proc. IEEE 7th International Conference on Tools with Artificial Intelligence, 338-391, 1995

[4] R.C. Holte (1993). Very simple classification rules perform well on most commonly used datasets. Machine Learning. 11:63-91.

[5] Kenji Kira, Larry A. Rendell: A Practical Approach to Feature Selection. In: Ninth International Workshop on Machine Learning, 249-256, 1992.

[6] Igor Kononenko: Estimating Attributes: Analysis and Extensions of RELIEF. In: European Conference on Machine Learning, 171-182, 1994.

[7] Marko Robnik-Sikonja, Igor Kononenko: An adaptation of Relief for attribute estimation in regression. In: Fourteenth International Conference on Machine Learning, 296-304, 1997.

[8] StatSoft, Formula Guide Weight of Evidence Module http://documentation.statsoft.com/portals/0/formula

[9] Data Clustering: A Review, A.K. Jain, M.N. Murty, P.J. Flynn, 1999, https://www.cs.rutgers.edu/ mlittman/courses/lightai03/jain99data.pdf

[10] Principal component analysis From Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Principal_component_analysis

[11] notes from Laboratory for Dynamic Synthetic Vegephenonenology (LabDSV), the University of California, http://ecology.msu.montana.edu/labdsv/R/labs/

[12] clValid, an R package for cluster validation, Guy Brock, Vasyl Pihur, Susmita Datta, and Somnath Datta, Department of Bioinformatics and Biostatistics, University of Louisville, http://cran.r-project.org/web/packages/clValid/vignettes/clValid.pdf