KDD Process

Data Transformation

Data Encoding (Scaling of continuous variables, coding of categorical variables)

Scaling of continuous variables

The data encoding during the first iteration concerns just about the scaling of our continuous data. For scaling of our continuous data we considered two methods. On the one hand the Min/Max method and on the other hand the z-Transformation. Z-Transformation (standardization) is used to scale our data. Due to the Min/Max method is more sensitive for outliers. The method is more sensitive because the result is bounded.

The first step to implement the z-Transformation was to detect the continuous variables. To detect the continuous variables we considered two ways of coding. The first way was to implement an algorithm which detects the continuous variables by comparing the number of different numeric attributes of a variable with a specific number. If the specific number is lower than the number of different numeric attributes of a variable, the variable is continuous. The second way was to implement an algorithm which detects the continuous variables of the data by searching for the continuous variable name. We got the continuous variable name out of the description table.   
The second algorithm is used to detect the continuous variables. Due to the first algorithm leads to a misinterpretation of the variable if the prescribed number of different numeric attributes is lower than the number of categories represented by numeric attributes of a variable.

The second step to implement standardization was to code a function which scales a vector by z-Transformation. Instead of using the scale() function we coded the function z.scale() to standardize a vector.

The final step was to code a function which scales just the continuous variables of the data by standardization. We used therefore a for-loop which applies the algorithm out of step one as well as the algorithm out of step two. The first algorithm detects the i-th continuous variable of the table. The algorithm out of step two standardizes the data of the i-th continuous variable. After it finished the loop the function returns the standardized data table.

**Evaluation**

An evaluation of the first iteration requires that we look at the preprocessing, transformation as well as the choice of algorithm under the point of view of a first attempt to predict the churn. The first attempt is applied in form of an univariate outlier detection, the approach of replace missing values with the median, without a treatment of categorical variables, a feature selection using univariate filter approach (Nadaraya-Watson estimator) and a data set without outlier for unrobust models and a data set with outliers for robust models. We consider taking a smaller subset (5.000 samples) to compare several models as well as the meta parameters. We decided to take a smaller subset in order to save computational cost. We considered the trade-off between saving computational cost and the accuracy. The best three models of the first iteration have classification accuracy between 60 and 62 per cent.

The classification accuracy of our first iteration depends on a simple attempt to predict the churn of 5.000 samples. To improve the prediction we take the following factors in account: A multivariate outlier detection, an advance imputation procedures for missing values, tune the meta parameters and to take the 50.000 samples for the final model.

The following figures **eight** and **nine** showing the accuracy depending on meta parameters.



Figure



Figure nine:

Based on comparing figure nine and eight with the figure **FREDDY CAPITEL ,** it can be concluded that further meta parameter tuning lead to a higher accuracy.

The more complex code would be the second iteration of our KDD Process. We could also compare the results of the second iteration with the first iteration to answer the question: “Is there an improvement?”