Predictive Analytics: Credit Risk Scorecard Application Case Study: Group 15

Jacob Heye Hilbrands (12229285) Mustafa Alsudani (1214099)
Moritz Renkin (11807211) Nils Klüwer (12229263)

November, 2022

Abstract

Credit scorecards are crucial tools in the credit assessment process. They are based on the prior performance of clients that share the same traits as a new client. Consequently, the goal of a credit scorecard is to estimate risk because credit scorecards are based on the past behavior of clients that share the same qualities as a prospective client. Therefore, the primary purpose of it is to either approve or reject a new client's loan request. The scorecard's function is to support this option. In this assignment, we'll compare and contrast the binning techniques known as weight-of-evidence (woe) binning and group binning. That's why we have been using the existing scorecard. Moreover, certain model changes and binning splits will be made as a result of the addition of the 7 variables. To evaluate the effectiveness of the scorecard, the Gini coefficient is used. The major results of this research were: demonstrate that group binning outperforms woe binning in out-of-sample predictions as measured by the Gini coefficient. The dependent variable, creditability, may be predicted more precisely when more independent variables are included.

Contents

Abstract

Giving new customers credit results in two options: approve their loan request or deny their application. The scorecard's function is to support this choice. Consequently, credit scoring is a mechanism used to assess the degree of risk connected to a particular application. This tool consists of a set of statistically significant characteristics that may be used to predictably distinguish between good and negative things. Any sources of information that the lender has access to at the time of the application may be used to choose the scorecard variables.

A credit scorecard is, in a much more formal sense, a statistical model that forecasts the likelihood of default for a client applying a certain set of criteria. When a consumer misses a payment or otherwise violates the terms of a loan, this is known as defaulting in the banking industry.

1 Introduction

In the form of a score and a likelihood of default, credit scoring models and scorecards estimate the risk that a borrower won't return a loan.

For instance, a credit scorecard may award a borrower points based on the following table for their age and income. As previously noted, a single dependent variable, creditability, was predicted using seven independent variables. As a binning strategy, the Weight-of-Evidence technique was adopted.

Therefore, group binning will be used as an extra binning strategy for this project's execution. To increase the model's Gini coefficient of prediction, a seventh independent variable was added, and the binning breaks were modified.

2 Predictive Analytics Research Methodology

2.1 Predictor Variable Transformation

The commonly-used method of Weight of Evidence (WOE) is employed to get an estimate on the predictive power of the individual, independent attributes with regards to the credit risk. The WOE approach is used to change each independent variable. This technique analyzes the "strength" of grouping for separating good and bad risk based on the ratio of good applicants to bad applicants at each group level and seeks to establish a monotonic connection between the independent variables and the target variable.

$$WOE = ln(\frac{Distributionof Goods}{Distributionof Bads}) * 100 \tag{1}$$

The transformation steps can be considered as follow:

1. Divide the data into bins, often 10 or 20 at most.
2. Estimate the percentage of good and bad customers.
3. Take the natural log and use that to calculate the WOE.
4. Substitute the estimated WOE values for the original data.

Negative cases take precedence over positive ones if WOE values are negative. Positive cases take precedence over negative ones if WOE values are positive.

This accomplishes the following goals:

- All non-linear correlations are removed.
- All variables are scaled automatically to some extent.

- Categorical variables are converted to continuous variables.
- Missing Data may be treated as a simple factor value.

We could create a standalone scorecard that a person could manually fill out with a pen and a printout of all pertinent factors.

The following downsides apply to it:

- Binning always results in information loss.
- Score growth along single variables is not continuous and happens in stages.
- Binning calls for manual editing.
- Comparing logistic regression with classically scaled variables makes calculating variable relevance more difficult.

As an alternative to the transformation based on WOE-values, the variables can also be transformed with some pre-defined bin borders.

2.2 Logistic Regression Analysis

Logistic Regression (also known as Logit-Model) is a statistical Model that can estimate probability of a certain event happening based on one or more independent variables. Its application is widespread in various statistical methods, especially in classification problems and prediction analyses. Contrary to linear regression, the predicted variable in logistic regression is a Bernoulli variable, i.e. a binary random variable k with:

$$k \in \{0, 1\} \tag{2}$$

Formally, the logistic regression model estimates probability p of the Bernoulli k being 1, corresponding the the event in question happening. The logistic function defining $p(x)$ takes on the form:

$$p(k) = \frac{e^{\beta_0 + \beta_1 k}}{1 + e^{\beta_0 + \beta_1 k}} = \frac{1}{1 + e^{-}(\beta_0 + \beta_1 k)} \tag{3}$$

In the domain of Credit Scoring, logistic regression is one of the most widely used statistical models (Bolton u. a., 2009, p. 19), mainly because the are simple and easy to implement, with wide-spread support in programming languages and libraries. Another striking advantage in favor of logistic regression in this domain is the absence of necessary major assumptions for variables used.

2.3 Building Scorecards

Statistical and non-statistical methods can be used to create credit scorecards, respectively. Only statistical techniques, such as contingency tables and linear regression models, were employed. This is because, in the context of credit scoring, one might gain from understanding sample estimators and their characteristics, confidence intervals, and hypothesis testing. This information might be applied to determining the relative weights of various traits, both to confirm the relevance of pertinent factors and eliminate irrelevant ones. Here, describing categorical data and concentrating on logistic regression to build a scorecard are the major considerations.

From the standpoint of credit scoring, the response variable represents the client's quality (good or poor), and explanatory variables is a list of traits that have discriminating power, or, in other words, that significantly influence whether a customer is good or bad. Therefore, a statistical model demonstrates how explanatory factors affect the response variable.

$$Score = \sum_{i=1}^{n} (-(WOE_i * \beta_i + \frac{a}{n}) * factor + \frac{offset}{n}) \tag{4}$$

The regression coefficients are used to scale a scorecard, which is defined as bringing it into compliance with a specific range of points. The logit-transformed prediction probability of logistic regression models is a linear function of the predictor variable values, making them linear models. As a result, a final scorecard model created in this way has the advantageous property that the final credit score (credit risk) is a linear function of the predictors.

2.4 Forecasting Scorepoints and default Probabilities

As described in the previous sections, the devised scorecard model is based on a logistic regression function. The model can be used to predict the likelihood of a potential debtor defaulting on his/her loan, as well as the resulting scorepoints in the context of our scorecard. Based on the forecasted scorepoints a credit applicant can be categorized as either good or bad.

On a broader scale, the credit scorecard model, which is initially trained with a sample of historical borrower data, allows a banking institution to discriminate “good” from “bad” credit applicants, thereby making the decision whether a loan offer should be extended or not.

2.5 Forecasting Accuracy Testing

Testing the accuracy of forecasts is critical to any model that is sought to be applied in a real world scenario. In the banking industry, estimating credit risk is directly linked to a banks profitability as credit interest is usually calculated based on the risk of loan defaults. Since loan interest can be assumed to provide the main income stream for traditional banks (in contrast to investment banks), having even a small edge in risk forecasting accuracy can pose a tremendous competitive advantage. Therefore, this section defines the methodology to evaluate the accuracy of the model forecasts.

When working with predictive models, both statistical and machine-learning-based, the available data is usually split into a training and a test sample. This has become the de-facto standard over the last decades (Tan u. a. (2021)). The training sample is used to fit the model decide on potential hyper-parameters while the testing sample is used only on the final configured and fitted model to evaluate its accuracy. The rationale behind this split is to assess how the model performs on data it was not originally trained on.

Before deciding on how to split the data, it is important to determine if there are any hidden patterns in the data. For example, if the data was split based on a certain attribute, like date, this attribute has to be statistically independent to the target variable. Since no such attributes can be trivially found in the source data, the split is performed randomly with 75% being used as the training sample and the rest as the testing sample.

In general, the forecasting accuracy can be tested with both using the data it was fitted with (in-sample) and using the independent testing sample (out-of-sample). Discrepancies between the in-sample and out-of-sample results could hint at overfitting of the model. In order to ensure the resilience of the trained model in a real-world scenario, the testing sample shall not be considered for any decisions, or fittings regarding the our scorecard model, until the final out-of-sample test.

3 Empirical results

In this section the approach and the results of building a Out-of-Sample Gini coefficient maximizing scorecard model with only seven predictor variables are presented step-by-step. The scorecard is built on the “germancredit” data which is included in the “scorecard” library.

3.1 Loading and Preparing the Data

The dataset "germancredit" is used from the library "scorecard". Below the import process is shown. The "germancredit" data has 21 variables int total. Seven predictor variables and a single dependent variable the "creditability" will remain for our scorecard model.

```
# Importing of library and data
library(scorecard)
data("germancredit")
names(germancredit)
```

```
##  [1] "status.of.existing.checking.account"
##  [2] "duration.in.month"
##  [3] "credit.history"
##  [4] "purpose"
##  [5] "credit.amount"
##  [6] "savings.account.and.bonds"
##  [7] "present.employment.since"
##  [8] "installment.rate.in.percentage.of.disposable.income"
##  [9] "personal.status.and.sex"
## [10] "other.debtors.or.guarantors"
## [11] "present.residence.since"
## [12] "property"
## [13] "age.in.years"
## [14] "other.installment.plans"
## [15] "housing"
## [16] "number.of.existing.credits.at.this.bank"
## [17] "job"
## [18] "number.of.people.being.liable.to.provide.maintenance.for"
## [19] "telephone"
## [20] "foreign.worker"
## [21] "creditability"
```

The first step of finding relevant predictor variables is done with the "var_filter()" function on the entire dataset. The default limits and rates for iv_limit, missing_rate & identical_rate mentioned below are used. The iv_limit excludes every variables whose information value is lower or equal to 0.02 in respect to our depedent variable "creditability". Through this filtering process 7 predictor variables are already excluded which are not eligible for the scorecard model as they cannot meet the explained criteria. In the "data_f.df" 14 variables are remaining, 13 possible predictor variables and one dependent variable "creditablitity".

```
# Filtering of variables with iv_limit >= 0.02, missing_rate <= 0.95
# and identical_rate <= 0.95
data_f.df = var_filter(germancredit, y="creditability")
```

```
ncol(data_f.df)
```

```
## [1] 14
```

3.2 Splitting the Data into Train and Test Samples

The remaining 13 possible predictor variables are split with the split_df function into train and test data with a ratio fo 75% train and 25% test data. Afterwards data_f.list is reformatted to be usable in the later process. The following scorecard model is only trained on the train data. The test data is later used to validate and test the performance of the scorecard model.

```
# Splitting data into train and test data with ratio 0.75
data_f.list = split_df(data_f.df,"creditability",ratio=c(0.75,0.25))
```

```
class(data_f.list)
lapply(data_f.list,class)
lapply(data_f.list, dim)
```

3.3 Weight-Of-Evidence (WOE)-Binning

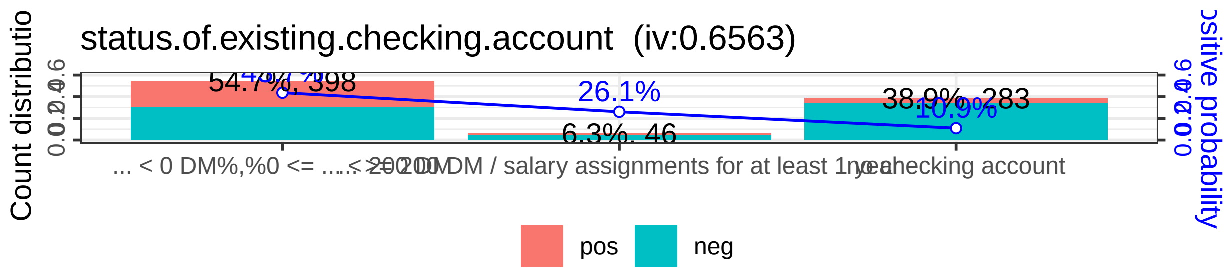
In the following the WOE-Binning is done with the 'woebin' function. For the 'woebin' function the default 'method="tree"' is used to generate the optimal binning of both numerical and categorical predictor variables. The breaks are generated automatically by the function and are saved in the "breaks.list" file.

```
# Binning of train data
# breaks.list is saved and imported seperately
bins.list = woebin(data_f.list$train,
                   "creditability",
                   save_breaks_list = "breaks.list")
```

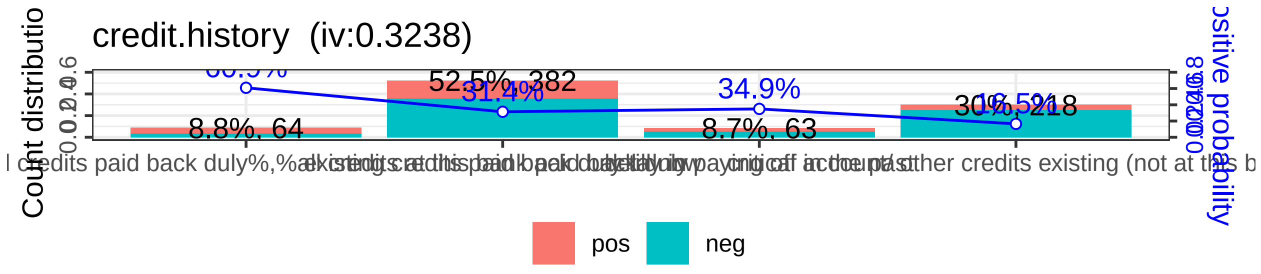
This breaks.list file needs to be run as well to generate a report on the entire scoreboard model process later on.

Below plots of seven predictor variables with the highest information value are shown in descending order. This does not mean that these values are the final seven predictor variables.

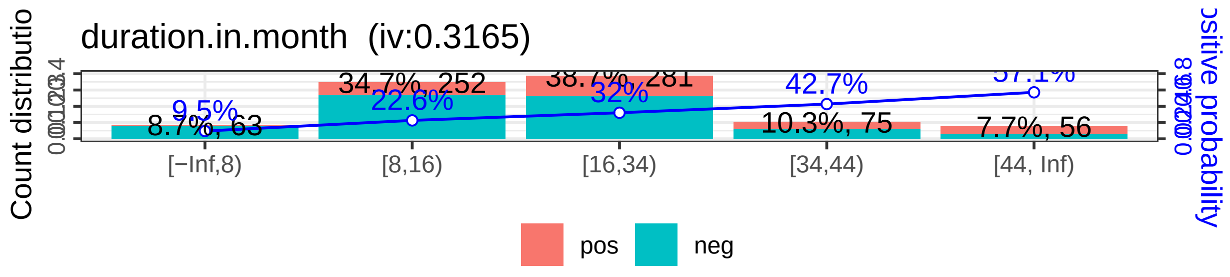
```
## $status.of.existing.checking.account
```



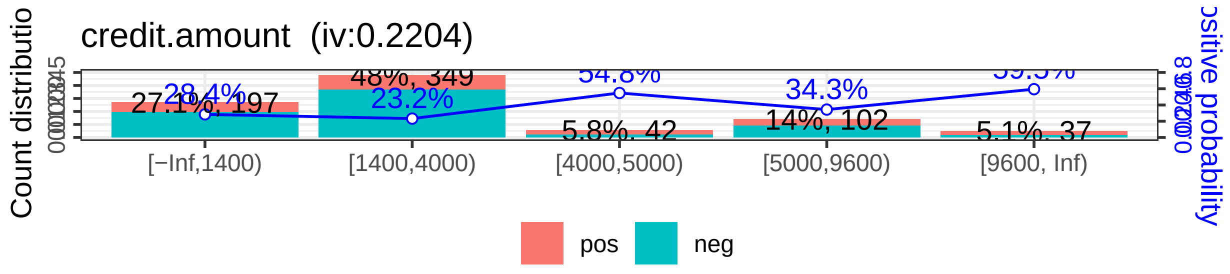
```
## $credit.history
```



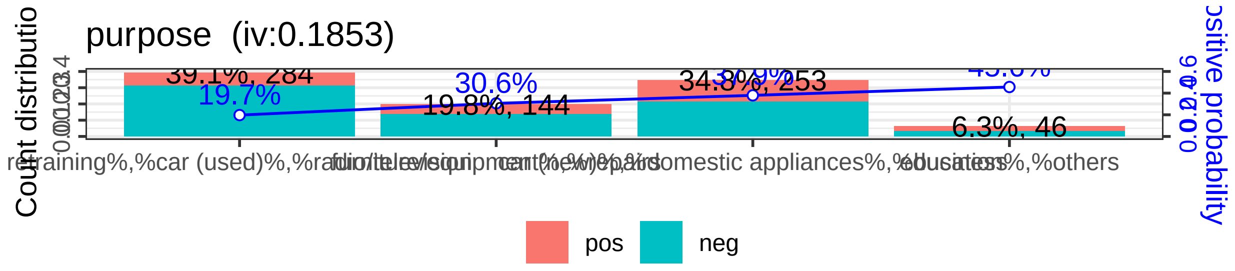
```
## $duration.in.month
```



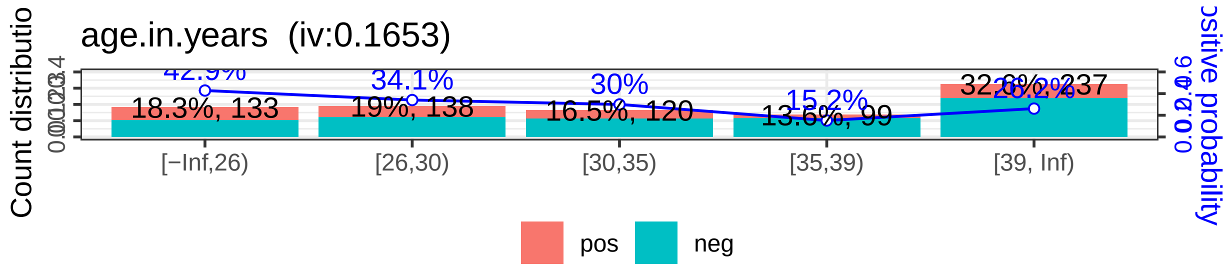
```
## $credit.amount
```



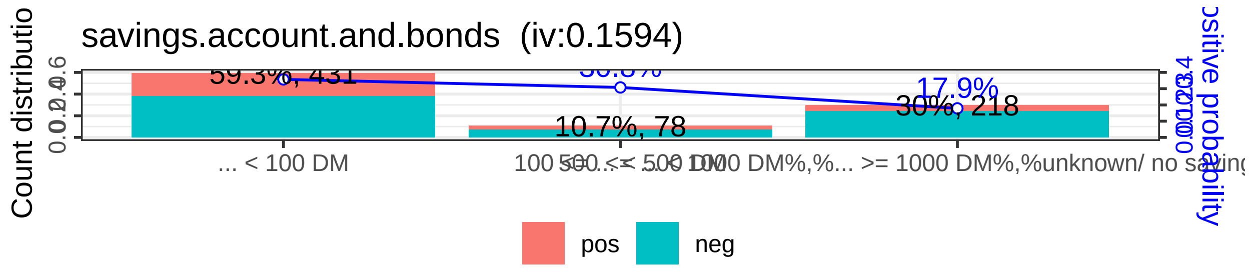
```
## $purpose
```



```
## $age.in.years
```



```
## $savings.account.and.bonds
```



3.4 Transformation of predictor variables

The train and test data is transformed with the generated information by ‘woebin’ into WOE-based predictor variables. These can be used later in glm and scorecard building. This is done by using the function ‘woebin_ply’.

```r
# WOE-Transformation of train and test data
data_woe.list = lapply(data_f.list,
                       function(x) woebin_ply(x, bins.list))
lapply(data_woe.list, class)
lapply(data_woe.list, dim)
```

In addition, the train and test data can be transformed into Bin-Group (GRP) based predictor variables using the bin borders or breaks. The functionality of building a scorecard with the GRP-Transforamtion is not yet implemented in the package at the current state, but the GRP-Transformation is later used for comparison and validation.

```
# Bin-Group (GRP) Transformation of train and test data
data_grp.list = lapply(data_f.list,
                       function(x) woebin_ply(x, bins.list, to = 'bin'))
lapply(data_grp.list, class)
lapply(data_grp.list, dim)
```

3.5 Generalized linear model (glm): Regressing response w.r.t. predictors

In this section, the logistic regression models are built. These models are used to select the seven most significant variables and subsequently to build the scorecard model with the chosen variables.

3.5.1 Selection of seven variables with Logistic regression w.r.t. WOE-transformed predictors

In order to select the seven most significant variables, a logistic regression is performed on the WOE-transformed predictor variables of the train data. The seven predictor variables which have the lowest significance value, are selected.

```
data_woe_first_iteration.glm <- glm(creditability ~ .,
                                    family = binomial(),
                                    data = data_woe.list$train)
summary(data_woe_first_iteration.glm)
```

```
##
## Call:
## glm(formula = creditability ~ ., family = binomial(), data = data_woe.list$train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0742  -0.6943  -0.3671   0.7495   2.4750
##
## Coefficients:
##                                                     Estimate Std. Error
## (Intercept)                                         -0.85979    0.09958
## status.of.existing.checking.account_woe              0.82357    0.12779
## duration.in.month_woe                                0.77204    0.19631
## credit.history_woe                                   0.66714    0.18079
## purpose_woe                                          1.01485    0.23555
## credit.amount_woe                                    0.73487    0.21701
## savings.account.and.bonds_woe                        0.80377    0.26016
## present.employment.since_woe                         0.59775    0.33021
## installment.rate.in.percentage.of.disposable.income_woe  2.95015    1.62616
## other.debtors.or.guarantors_woe                      1.17259    0.60659
## property_woe                                         0.12139    0.39847
## age.in.years_woe                                     0.96868    0.26093
## other.installment.plans_woe                          0.62812    0.58047
## housing_woe                                          0.37643    0.40013
##                                                     z value Pr(>|z|)
## (Intercept)                                          -8.634  < 2e-16 ***
## status.of.existing.checking.account_woe               6.445 1.16e-10 ***
## duration.in.month_woe                                 3.933 8.40e-05 ***
## credit.history_woe                                    3.690 0.000224 ***
## purpose_woe                                           4.308 1.64e-05 ***
## credit.amount_woe                                        3.386 0.000708 ***
## savings.account.and.bonds_woe                            3.089 0.002005 **
## present.employment.since_woe                             1.810 0.070265 .
## installment.rate.in.percentage.of.disposable.income_woe   1.814 0.069650 .
## other.debtors.or.guarantors_woe                          1.933 0.053226 .
## property_woe                                             0.305 0.760642
## age.in.years_woe                                         3.712 0.000205 ***
## other.installment.plans_woe                              1.082 0.279215
## housing_woe                                              0.941 0.346826
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 886.32  on 726  degrees of freedom
## Residual deviance: 658.56  on 713  degrees of freedom
## AIC: 686.56
##
## Number of Fisher Scoring iterations: 5
```

The following seven predictor variables are selected as they have the lowest significance value (<0.01):

- status.of.existing.checking.account
- duration.in.month
- credit.history
- purpose
- credit.amount
- savings.account.and.bonds
- age.in.years

3.5.2 Logistic regression w.r.t. selected WOE-transformed predictors

The logistic regression model with the seven selected WOE-transformed predictor variables is built on the train data and saved in "data_woe_second_iteration.glm".

```
data_woe_second_iteration.glm <- glm(creditability ~
                                      status.of.existing.checking.account_woe
                                      + duration.in.month_woe
                                      + credit.history_woe
                                      + purpose_woe
                                      + credit.amount_woe
                                      + savings.account.and.bonds_woe
                                      + age.in.years_woe,
                                        family = binomial(),
                                        data = data_woe.list$train)
```

3.5.3 Logistic regression w.r.t. selected GRP-transformed predictors

The logistic regression model with the seven selected GRP-transformed predictor variables is built on the train data and saved in "data_grp_second_iteration.glm".

```
data_grp_second_iteration.glm <- glm(creditability ~
                                      status.of.existing.checking.account_bin
```

```
                                             + duration.in.month_bin
                                             + credit.history_bin
                                             + purpose_bin
                                             + credit.amount_bin
                                             + savings.account.and.bonds_bin
                                             + age.in.years_bin,
                                            family = binomial(),
                                            data = data_grp.list$train)
```

3.6 Building the scorecard-model

The scorecard is built with the logistic regression model of the seven WOE-transformed variables and is saved in "scorecard_woe_second_iteration.scm". The scorecard contains the baseline points and the points associated with every bin. In the output of the command `scorecard_woe_second_iteration.scm$purpose`, the points for each bin of the "purpose" predictor variable is displayed. When the purpose of a credit is e.g. furniture, equipment or repairs, the credit applicant receives -3 points for this predictor variable.

```
scorecard_woe_second_iteration.scm <- scorecard(bins.list,
                                               data_woe_second_iteration.glm)
names(scorecard_woe_second_iteration.scm)
```

```
## [1] "basepoints"                          "status.of.existing.checking.account"
## [3] "duration.in.month"                   "credit.history"
## [5] "purpose"                             "credit.amount"
## [7] "savings.account.and.bonds"           "age.in.years"
```

```
scorecard_woe_second_iteration.scm$purpose
```

```
##    variable                                        bin count count_distr neg
## 1:  purpose retraining%,%car (used)%,%radio/television   284  0.39064649 228
## 2:  purpose              furniture/equipment%,%repairs   144  0.19807428 100
## 3:  purpose car (new)%,%domestic appliances%,%business   253  0.34800550 157
## 4:  purpose                          education%,%others    46  0.06327373  25
##    pos   posprob         woe       bin_iv  total_iv
## 1:  56 0.1971831 -0.54948057 0.1038486990 0.1853448
## 2:  44 0.3055556  0.03353282 0.0002242187 0.1853448
## 3:  96 0.3794466  0.36261576 0.0487911020 0.1853448
## 4:  21 0.4565217  0.68015999 0.0324807583 0.1853448
##                                        breaks is_special_values points
## 1: retraining%,%car (used)%,%radio/television             FALSE     43
## 2:              furniture/equipment%,%repairs             FALSE     -3
## 3: car (new)%,%domestic appliances%,%business             FALSE    -28
## 4:                         education%,%others             FALSE    -53
```

The scores for the entire "germancredit" data are calculated and saved in "score_woe_second_iteration.df"

```
score_woe_second_iteration.df = scorecard_ply(germancredit,
                                             scorecard_woe_second_iteration.scm,
                                             only_total_score = FALSE)
```

The scores for the splitted "germancredit" data (train and test) are calculated and saved in "score_woe_second_iteration.list".

```
score_woe_second_iteration.list <- lapply(data_f.list,
                                          function(x) scorecard_ply(x,
                                             scorecard_woe_second_iteration.scm))
```

A report can also be generated for this scorecard model which includes information on the dataset, model coefficients, model performance, WOE binning, scorecard, population stability, and gains.

```r
# Report of Scoreboard with WOE-transformed predictor variables
y<-"creditability"
x<-c("status.of.existing.checking.account","duration.in.month","credit.history",
      "purpose","credit.amount","savings.account.and.bonds",
      "age.in.years")

report(data_f.list,
       y,
       x,
       breaks.list,
       seed = NULL,
       save_report = "Report_WOE_second_iteration")
```

Building a scorecard with GRP-transformed predictor variables is unfortunately not supported by the "scorecard" library. Therefore, no report can be created as well. Nevertheless, the performance of the logistic model of the GRP-transformed predictor variables is compared with the logistic model of the WOE-transformed predictor variables in the next section.

3.7 Predicted probabilities and scorepoints

With the scorecard of the logistic regressions of both WOE- and GRP-transformed predictors and the WOE-transformed predictor variables, the predicted probabilities and scorepoints can be calculated. The predicted probabilities forecast the probability of default for each applicant in the "germancredit" data. The first predicted probabilities of both transformations are displayed below.

```r
# Predicted probabilties w.r.t. WOE-transformed predictors
predProb_woe.list <- lapply(data_woe.list,
                        function(x) predict(data_woe_second_iteration.glm,
                                            type = 'response',x))
head(predProb_woe.list$train)
```

```
##          1          2          3          4          5          6
## 0.03371439 0.58246350 0.74532855 0.13486467 0.06681654 0.02574071
```

```r
# Predicted probabilties w.r.t. GRP-transformed predictors
predProb_grp.list <- lapply(data_grp.list,
                            function(x) predict(data_grp_second_iteration.glm,
                                                type = 'response',x))
head(predProb_grp.list$train)
```

```
##          1          2          3          4          5          6
## 0.03045587 0.57359188 0.79067183 0.11022337 0.07227866 0.02260710
```

The calculated scorepoints of the applicants of the "germancredit" data can only be calculated for WOE-transformed predictor variables because no scorecard could be built for the GRP-transformed predictors.

```r
# Predicted scorepoints w.r.t. WOE-transformed predictors
head(score_woe_second_iteration.list$train)
```

```
##    score
## 1:   630
## 2:   363
## 3:   309
## 4:   522
## 5:   577
## 6:   650
```

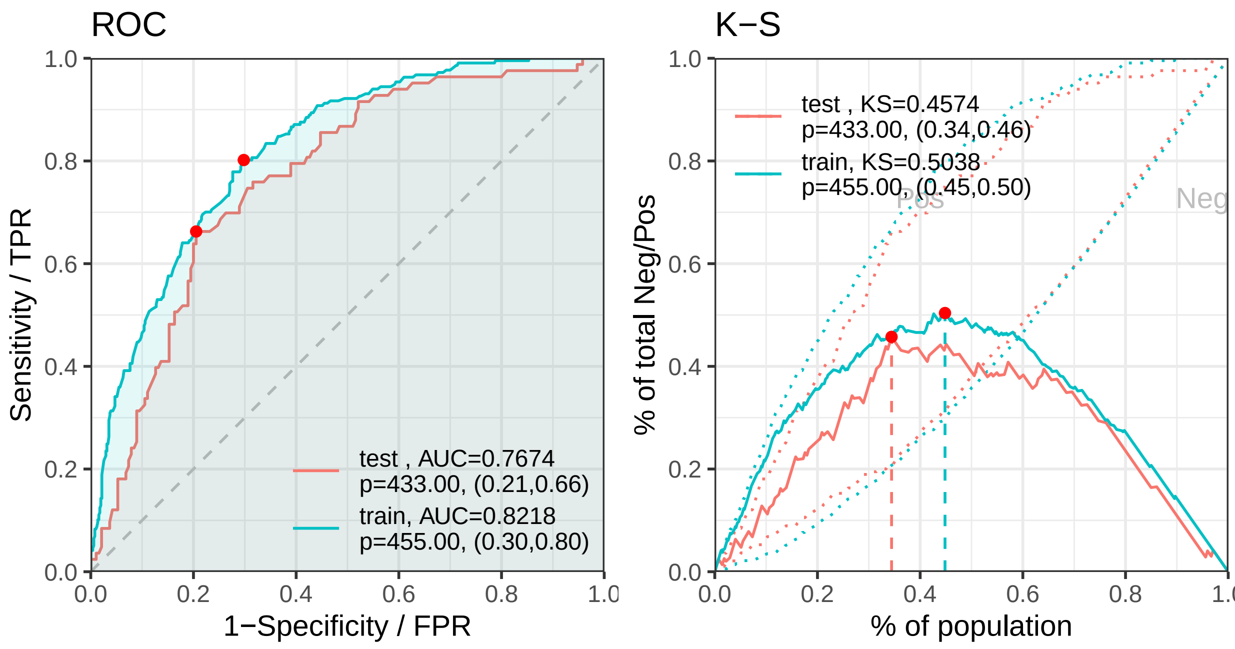
From the report that was created above for the scorecard with WOE-transformed predictor variables, it can be can be seen that the target score is 600 and the target predicted probability is 5,26%. That means, that applicants with a score below 600 and with a predicted probability of default higher than 5,26% are declined.

3.8 Gini Coefficient In-Sample and Out-of-Sample

Below the prediction power of the proposed predictor variables is validated through the In-Sample and Out-of-Sample testing. The full validation is done with the predicted scores in "score_woe_second_iteration.list" and the function perf_eva. To further validate the scorecard the Gini-Coefficient of "data_woe_second_iteration.glm" and "data_grp_second_iteration.glm" is calculated as well.

```
#Scorecard Model second iteration
perf_eva(pred = score_woe_second_iteration.list,
         label = default.list,
         binomial_metric = c("gini","auc","r2", "rmse"),
         show_plot=c("roc","ks"),
         confusion_matrix = TRUE)
```

```
## [INFO] The threshold of confusion matrix is 448.0000.
```



```
## $binomial_metric
## $binomial_metric$train
##         Gini       AUC
## 1: 0.6436704 0.8218352
##
## $binomial_metric$test
##         Gini       AUC
## 1: 0.5348129 0.7674065
##
##
## $confusion_matrix
## $confusion_matrix$train
##    label pred_0 pred_1     error
## 1:     0    369    141 0.2764706
## 2:     1     48    169 0.2211982
## 3: total    417    310 0.2599725
##
## $confusion_matrix$test
##    label pred_0 pred_1     error
## 1:     0    135     55 0.2894737
## 2:     1     24     59 0.2891566
## 3: total    159    114 0.2893773
##
##
## $pic
## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

WOE-Binning, second iteration GLM with 7 predictor variables

```
##         Gini
## 1: 0.6444203
```

```
##         Gini
## 1: 0.5346227
```

GRP-Binning, second iteration GLM with 7 predictor variables

```
##         Gini
## 1: 0.6502033
```

```
##         Gini
## 1: 0.5352568
```

Method	In-Sample Gini-Coefficient	Out-of-Sample Gini-Coefficient
Second iteration Scorecard	0.6436704	0.5348129
Second iteration WOE-Binning	0.6444203	0.5346227
Second iteration GRP-Binning	0.6502033	0.5352568

As expected the In-Sample Gini-Coefficient is much higher as the OoS-Gin-Coefficient, but still performing well with over 50%. The Out-of-Sample testing represents a real world data set much better than the In-Sample testing. The differences in the OoS testing are marginal ranging from 0.5346227 (WOE-Binning) to 0.5352568 (GRP-Binning) having only a difference of 0.0006341. As stated by Peussa (2016) the OoS-Gini Coefficient should lie between 10% and 50%, having 53.53% shows that the used approach is working. (Peussa u. a., 2016, p. 24)

4 Summary

The main objective of this paper is the conception and implementation of a scorecard prediction model as well as examine the impact of different binning methods on the model performance. The initial data sample consisting of past credit information was split into training and testing samples with both in-sample and out-of-sample Gini-coefficients being calculatated in order to quantify the predictive accuracy of the model.

The seven most significant attributes have been selected out of the germancredit dataset based on their significance value in the logistic regression models. The scorecard model was then fitted with these attributes.

The Gini-coefficients from in-sample and out-of-sample testing indicate a solid predictive accuracy with even the out-of-sample gini-coefficient reaching 53.35%.

References

[Bolton u. a. 2009] Bolton, C. ; Mathematics, University of Pretoria. Department o. ; Mathematics, Applied: *Logistic Regression and Its Application in Credit Scoring.* University of Pretoria, 2009 https://books.google.at/books?id=K7B3MwEACAAJ

[Peussa u. a. 2016] Peussa, Aleksandr u. a.: Credit risk scorecard estimation by logistic regression. (2016)

[Tan u. a. 2021] Tan, Jimin ; Yang, Jianan ; Wu, Sai ; Chen, Gang ; Zhao, Jake: *A critical look at the current train/test split in machine learning.* http://dx.doi.org/10.48550/ARXIV.2106.04525. Version: 2021