Seminar Economics (Statistics)

Classification of Credit Worthiness: A Neural Network Approach

Term Paper

Alexander Dalheimer

Supervisor: Hartmut Jakob Stenz[*]

Supervisor: Prof. Dr. Jörg Breitung

February 21, 2022

[*]I would like to thank Mr. Stenz for his great helpfulness at any time I needed his supervision.

Contents

Introduction **1**

Literature Review **2**

Method **4**

Data **10**

Results and Discussion **12**

Conclusion **16**

References **16**

Appendix **20**

Introduction

Assessing the credit risk – the probability that a person who applies for a credit will default – is one of the main interests of financial institutions. A default is associated with high costs. Therefore, credit institutions use different methods to assess the creditworthiness of credit applicants in order to filter borrowers with high probability of default. In the early days, the decision on loan applications was based solely on the subjective judgment by bankers (see e.g. Altman, 1968). Due to the obvious disadvantages of subjective judgment compared with data driven decisions about the creditworthiness, banks started to implement econometric / statistical methods to assess the credit risk in the 1960s. Popular methods were, and still are, discriminant analysis (see e.g. Frank and Cline, 1971; Grinols, 1975; Taffler and Abassi, 1984; Doumpos et al., 2001), logit (see e.g. Feder and Just, 1977; Rivoli and Brewer, 1998) and probit models (see e.g. Fisk and Rimlinger, 1979; Balkan, 1992; D., Siermann and Lubek, 1997) as well as cluster analysis (see e.g. Altman, 1968). In the late 1990s, and nowadays with the era of "deep learning", machine learning (ML) methods are popular to use for credit risk assessment. Classiscal ML models for classification such as Naive Bayes or k-Nearest Neighbor are most often only used as benchmark models. More widely used models are support vector machines (SVM) (see e.g. Desai, Crook and Overstreet, 1996; Davis, Edelman and Gammerman, 1992; Piramuthu, 1999; West, 2000) and (artificial) neural networks (NN) (see e.g. Desai, Crook and Overstreet, 1996; Davis, Edelman and Gammerman, 1992; Piramuthu, 1999; West, 2000). In order to overcome disadvantages of individual statistical and machine learning models, the literature came up with so-called hybrid models consisting of a combination of (1) a statistical and a ML model (see e.g. Yim and Mitchell, 2005; Markham and Ragsdale, 1995; Jo and Han, 1996; Lee et al., 2002; Hsieh, 2005) or (2) a combination of multiple ML models (see e.g. Tsai and Wu, 2008).

The prime objective of this research work is to predict the probability of default of potential borrowers. Since the data used in this paper is from a peer-to-peer lending company, the model is especially interesting for individual investors which have to decide which credit application they accept or reject. In order to facilitate this decision, a feed-forward neural network model is used for the prediction of credit defaults. In additon, the neural network is compared to a logistic regression with lasso penalty as a benchmark model.

The paper is structured as follows: first, the literature regarding the prediction of credit risk is summarized. Second, the neural network as a classification model is introduced. Subsequently, the data is described and a brief exploratory analysis is presented. Finally, the results are shown and discussed by comparing the results of the neural network with those of the logistic regression.

Literature Review

This literature review is focused on the comparison of different models in terms of their performance in predicting credit risk. However, it needs to be mentioned that it is difficult to compare multiple studies due to differences in the data they use and how they preprocess it (how missing values, outliers or imbalanced classes are handled), how they select the features (are all sensible variables used or is a dimensionality reduction technique used) or what evaluation metric (e.g accuracy or area under the curve) they use. All those differences need to be kept in mind when comparing studies.

There are multiple studies which compare the performance of different neural network models. For example, West (2000) compares five different neural network models: a multilayer perceptron, mixture-of-experts, radial basis function, learning vector quantization and fuzzy adaptive resonance which produced all accuracy values above 82%. The best model – the learning vector quantization – obtained an accuracy of 87%. Tsai and Wu (2008) compare the performance of deep neural network models with three hidden layers and different numbers of neurons as well as different numbers of training epochs. They also pool the predictions of multiple NN models in order to obtain a so-called multiple classifier. However, the results show that a single neural network outperforms a multiple classifier in terms of accuracy. Khashman (2009) uses a three-layer neural network and trains it under seven different learning schemes (ratio of learning and testing data).

Many studies have compared the performance of neural network models with other classical statistical models or other machine learning models. For example, Davis, Edelman and Gammerman (1992) use decision trees and a multi-layer perceptron NN. They find out that both models perform similarly. Desai, Crook and Overstreet (1996) compare the performance of a multi-layer perceptron neural network with a linear discriminant analysis and a logistic regression model. The authors report the result that the neural network outperforms both other models. A multi-layer perceptron NN is compared with a neural-fuzzy model in the study by Piramuthu (1999), who concluded that the neural network outperforms the neural-fuzzy model. Lee et al. (2006) argues that artificial neural networks are inferior to classification and regression trees (CART) and multi-variate adaptive regression splines (MARS). Abdou, Pointon and El-Masry (2008) compare a feed-forward neural network with discriminant analysis, logistic and probit regression, finding that the neural network outperforms all other models in terms of the average correct classification rate. A more recent study by Natasha, Prastyo and Suhartono (2019) compare classical econometric models such as a discriminant analysis and a binary logistic regression with machine learning approaches such as support vector machine and (deep) neural networks. They use the area under the receiver operating characteristic (AUC-ROC) curve which they claim to be an appropriate metric for imbalanced data.[1] Their findings suggest that the discriminant analysis model performs best in terms of AUC-ROC (0.71), followed by the binary logistic regression model (0.67). The deep neural network with two hidden layers with ten and three neurons respectively and Tanh activation function obtained an AUC-ROC value of 0.64. The worst performance were obtained for the neural network with one hidden layer with two neurons and ReLu activation function (0.61) followed by the support vector machine (0.5). Furthermore, Bayraci and Susuz (2019) compare a deep neural network with logistic regression, J48 decision tree, support vector machine model and a naive bayes classifier. The authors conclude that the support vector machine performs best followed by the deep neural network, logistic regression, J48 and naive bayes in terms of a accuracy measure that is weighted according to the class frequencies (results: 78.14%, 77.98%, 77.31%, 70.05%, 57.04%).

The literature review shows that there are many studies investigating the prediction performance of neural networks for the assessment of credit risk. However, the literature does not allow a comprehensive picture of the performance because, first, all studies use different evaluation measurements. This makes it difficult to compare results across studies and to draw a final conclusion. In addition, the used metrics are often not appropriate and hence the results can be biased and not informative since over-optimistic results are presented. Second, today more computational power is accessible compared to what was available in studies published twenty to thirty years ago. This makes it possible to train more complex models (with more parameters) and with more data, which both increases the computational requirements. This could lead to models with better performance, which will definitely be in the interest of financial institutions since credit risk assessment is still one of the most relevant topics for them.

[1]In the methods section of this paper I will argue that the area under the ROC is not an appropriate measure since it overrates correct predictions of the majority class.

Method

The feed-forward neural network: The objective of this research work is to predict the creditworthiness, or in other words, the risk of default. This boils down to be a binary classification problem: the dependent variable can either take on the value *fully paid* (0) or *default* (1). As already described in the introduction and literature review, there are many candidates of different models to assess the credit risk. However, this paper aims to evaluate the predictive power of a feed-forward neural network. Neural networks were developed in the mid 1980s and are inspired by the architecture of the human brain (Efron and Hastie, 2016). In order to understand how such a feed-forward neural network is constructed, one can visualize it in a neural network diagram as shown in figure 1. A neural network always consists of at least three layers: (1) an input layer, (2) a hidden layer and (3) an output layer. One speaks of a deep neural network if the number of hidden layers is larger or equal to two layers (Bayraci and Susuz, 2019). Each layer comprises neurons (illustrated by circles in figure 1). The neurons in the input layer represent the variables / features used as predictors. Hence, the number of neurons in the input layer is fixed by the number of available features. The number of neurons in the output layer is determined by the number of classes of the dependent variable. Since the dependent variable has two classes (default vs. fully paid), the output layer has exactly two neurons. The number of neurons in the hidden layer needs to be determined by the analyst and can have a significant effect on the performance of the neural network. A distinctiveness of the feed-forward neural network is that the neurons are only connected with all neurons of the previous layer. For example, if one considers the neurons $a_1, \dots, a_5$ in the hidden layer (L_2), one can see that the neurons are not connected with neurons of the same layer. But the neurons are connected with every neuron in the previous layer. This holds for the whole network. The connections of individual neurons (illustrated as arrows in figure 1) represent weight parameters. The neurons in the input layer contain no computation but represent the value of the corresponding feature. The neurons in the

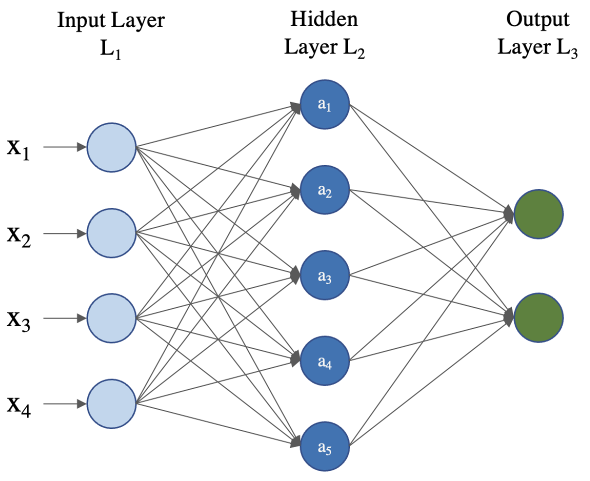


Figure 1: Feed-forward neural network diagram with four input neurons, one hidden layer with five neurons and two output neurons.

hidden layer can be computed as:

$$a_l = g(w_{l0}^{(1)} + \sum_{j=1}^{4} w_{lj}^{(1)} x_j) \tag{1}$$

where l refers to the l^{th} neuron and j refers to the j^{th} neuron of the previous layer which is equivalent to the j^{th} feature. The superscript indicates which layer is considered. Equation 1 shows that a neuron of the hidden layer a_j can be represented as a linear combination of the weights and neurons of the previous layer plus a bias (or intercept) $w_{l0}^{(1)}$. This linear combination is embedded into a non-linear function $g(\cdot)$ called activation function. Similarly, an individual output neuron, denoted as o, can be defined as

$$o = g(w_0^{(2)} + \sum_{j=1}^{5} w_{lm}^{(2)} a_l) \tag{2}$$

which is again a linear combination of the weights and neurons of the previous layer plus a bias embedded into an activation function. A so-called activation function is used since a neuron is supposed to be either activated or deactivated. Due to the fact that the weigthed sum can produce any value, the activation function transforms the value to a distinct range of values, e.g. between zero and one, where zero is associated with a deactivation and one with an activation of a neuron. A natural candidate for an activation function is the (logistic) sigmoid function which is a smooth s-shaped function defined as:

$$g(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t} \tag{3}$$

The sigmoid function can be easily differentiated (Nantomah, 2019) as shown in equation 4, which is necessary for the "learning" mechanism in the backpropagation algorithm.

$$\begin{aligned} g'(t) &= \frac{e^t}{(1+e^t)^2} = g(t)(1-g(t)) \\ g''(t) &= \frac{e^t(1-e^t)}{(1+e^t)^3} = g(t)(1-g(t))(1-2g(t)) \end{aligned} \tag{4}$$

Hornik, Stinchcombe and White (1989) proof that a feed-forward neural network with sigmoid activation function for the neurons in the hidden and output layer can, if enough data is available, approximate any smooth function. Therefore, neural networks are also called "universal approximators". The whole neural network can be described as a complex hierarchical function $f(x;W)$ where x is the feature vector and W the weigth matrix. Using a training set $\{x_i, y_i\}_1^n$ and an appropriate loss function $L[y, f(x)]$, the objective boils down to be a minimization problem:

$$\min_W \left\{ \frac{1}{N} \sum_{i=1}^{n} L[y_i, f(x_i; W)] \right\} \tag{5}$$

For a feed-forward neural network with two output neurons, hence a binary classification, a possible loss function is the binary cross-entropy also called log loss:

$$\text{Loss} = \frac{1}{N} \sum_{i=1}^{N} -(y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i)) \tag{6}$$

Note that the outputs of a neural network are probabilities at which an observation belongs to one of the two classes. The p_i in the loss function in equation 6 denotes the probability of belonging to the first class. Hence, $(1 - p_i)$ is the probability belonging to the second class. For example, if an observation belongs to the *default* class, hence $y_i = 1$, the second part vanishes and the first part of the equation becomes active. It is the other way around if an observation is of class *fully paid*. Therefore, if the neural network predicts the actual class with a high probability score, the loss will be low vice versa. Note, that the log is used since small differences between actual class and predicted probability are less penalized.

Up to this point mainly the feed-forward part of the neural network was considered. However, it was not discussed yet how the model computes its weights and biases. There the backpropagation algorithm comes into play. Using a training pair (x, y) the forward pass through the network is made were all weights and biases are initialized randomly. As shown in equations 1, 2 and 3, this computes all the activations at each neuron in every layer. Subsequently, the backpropagation algorithm goes back through the network and computes the gradient of the loss function for each weight and bias. Finally, the computed gradient is used by an optimization algorithm – which is usually the gradient descent algorithm – to update the weights and biases in order to minimize the loss function shown in equation 6 (Efron and Hastie, 2016).

Training of the network: The model used in this paper is a feed-forward neural network with sigmoid activation function for each neuron in the hidden and output layer, binary cross-entropy loss function, standard backpropagation algorithm and gradient descent for optimization of the weigths. Of course, all these specifications could have been considered as hyperparameters. This would mean that those could be tuned by replacing them by alternatives in order to optimize the prediction results. However, limited computational resources have led to the decision to consider those specifications as fixed[2]. Nevertheless, there are still hyperparameters to tune: the number of hidden layers and the number of neurons within the hidden layers. Unfortunately, also here limited computational resources restricted the number of combinations which could be computed. However, nine different network designs were created by specifying one hidden layer with five, ten or fifteen neurons and a second hidden layer with zero, five or ten neurons which sums up to nine combinations.

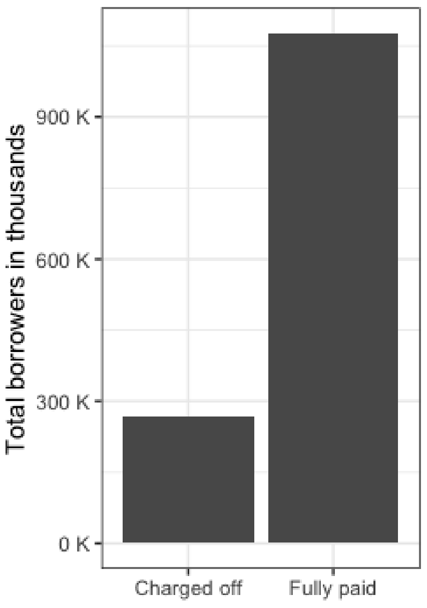


Figure 2: Class distribution of all borrowers included in the dataset.

For the training of a neural network balanced classes, meaning that both classes appear similarly frequent in the data, is of advantage. Unfortunately, this is usually not the case for credit data. Figure 2 shows that only about 20% of the cases are defaults while 80% of the borrowers have fully paid their loans. Such a skewed distribution of the classes can lead to a classifier which is biased towards the majority class. This is the case since the classifier was not able to "learn" the patterns of the minority class adequately due to the scarce occurrences. In a worst case, the classifier tends to misclassify the minority class entirely into the majority class. Possible solutions to this problem are up-sampling and down-sampling. Up-sampling means randomly drawing (with replacement) rows of the data of the minority class and duplicating those. In contrast, down-sampling implies the random deletion of observations of the majority class. The sampling is finished when the proportions of both classes are equal (for a more detailed discussion of this issue see e.g. Johnson and Khoshgoftaar, 2019). Both techniques are used withing the training procedure and results can be compared.

Before the neural network can be trained, the data set at hand is split into training and testing set. Since the data set is very large and more data increases the computational requirements for the training, I used a ratio of 50:50. For splitting the data, stratified

[2]Note that the mentioned specifications are the default arguments of the *RSNNS* R package which is used within the wrapper R package *caret*.

sampling was used which ensures that the proportions of the classes maintain in both sets. Furthermore, numerical features were standardized which is generally recommended since different scalings of features can influence the size of the weights. In additon, categorical features were converted to dummies with one category left out. Training and hyperparameter tuning is conducted by using k-fold cross-validation, where $k = 3$ was chosen also due to computational resource limitations. Usually five up to ten folds are chosen. First, I implemented stratified k-fold cross-validation, where the training set is split into k folds where all folds maintain the class proportions of the training set. The first $k - 1$ folds are used for training the neural network and the last k^{th} holdout fold is used for validation. This process is repeated k times until every fold was once the validation fold. This procedure is repeated for all nine network designs. The neural network which performs best in the cross-validation is selected as final model and will be used to make prediction on the testing set. Besides the stratified sampling also up- and down-sampling are implemented within this k-fold cross-validation. Note, that the up-/down-sampling is only done on the $k - 1$ training folds. The holdout fold is not sampled to prevent biased results.

As described only the best model of the cross-validations will be used for predictions on the testing set. However, what determines which model performs best? There are plenty of evaluation metrics available. However, metrics such as the standard accuracy

$$\text{ACC} = \left(\frac{\text{T}_p}{\text{T}_p + \text{F}_n}\right) + \left(\frac{\text{T}_n}{\text{T}_n + \text{F}_p}\right) \tag{7}$$

assume balanced data[3]. Otherwise, the accuracy will be biased towards the majority class due to the second fraction which considers the true negatives (majority class). Furthermore, the accuracy assumes same costs of both classes which is unrealistic in this application since false positives (predicting someone falsely as creditworthy) is associated with higher costs for a credit institute than false negatives (predicting someone falsely as not creditworthy). Another popular metric which is widely used and also often used for imbalanced data is the receiver operating characteristic (ROC) curve. However, also this metric is over-optimistic for imbalanced data since it also considers the true negatives. The metric I will use for both, the determination of the best model within the cross-validation and for the predictions on the test set, is the *area under the precision-recall (PR) curve*. The PR-curve shows the trade-off between precision (P) and recall (R) for

[3]Notation: $T_p \mathrel{\widehat{=}}$ true positive; $T_n \mathrel{\widehat{=}}$ true negative; $F_p \mathrel{\widehat{=}}$ false positive; $F_n \mathrel{\widehat{=}}$ false negative

different thresholds.

$$P = \frac{T_p}{T_p + F_p} \qquad\qquad R = \frac{T_p}{T_p + F_n} \tag{8}$$

A high precision relates to a low false positive rate. Hence, precision shows how many of the "default" predictions are actual defaults. In contrast, a high recall relates to a low false negative rate. Hence, recall takes into account how many defaults were not predicted. High scores for both metrics imply that the classifier is returning accurate results (high precision) and also returns a majority of all positive results (high recall). The PR-curve is a proper metric for imbalanced data since it does not consider true negatives and hence it concentrates on predictions of the class associated with the higher costs. An additional metric I use for comparing the model performances on the test set is a weighted accuracy which takes the class proportions into account.

Benchmark model: In order to judge the performance of the feed-forward neural network, a benchmark model is used for comparison. This benchmark model will be a logistic regression model with a L1 (Lasso) penalty term. The L1 norm $\lambda \sum_{j=1}^{p} |\beta_j|$, for $j = 1, \dots, p$ regressors and $\lambda > 0$, is helpful for feature selection since the L1 norm is a shrinkage method. If λ is sufficiently large, some of the coefficients will be shrank exactly to zero (James et al., 2013; Pereira, Basto and da Silva, 2016). In addition, the L1 regularization avoids overfitting by reducing the number of features and also avoids biased estimates through multicollinearity. The log-likelihood function extended by the L1 regularization term, which needs to be maximized, is defined as (Hastie, Tibshirani and Friedman, 2009):

$$\min_{\beta_0,\beta} \left\{ \sum_{i=1}^{n} \left[y_i(\beta_0 + \beta^T x_i) - \log\left(1 + e^{\beta_0 + \beta^T x_i}\right) \right] - \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{9}$$

The λ in the lasso penalty term is a hyperparameter which needs to be tuned. For this, again, k-fold cross-validation with an inclusion of stratified-, down- and up-sampling is used.

Data

Data source: The data used in this term paper is provided by the company *Lending Club* (LC)[4] which is the largest peer-to-peer lending marketplace in the United States of America (*Lending Club data*, 2021). Lending Club is a so-called platform firm which provides the online marketplace where investors can lend money to people who intend to take out a loan. Since Lending Club is just a platform, the risk that a borrower defaults is entirely bore by the investors. Therefore, a model which predicts the creditworthiness / the risk of default is especially interesting for individual investors. LC costumers which want to take out a loan have to apply for it by handing in information, such as the amount and purpose of the loan, personal information (adress, telephone number, mail) and a verifiable income source. Furthermore, Lending Club will gather credit history data for at least the last three years. This information can be used by individual investors to make a decision whom to grant a loan. In addition, Lending Club facilitates this decision by providing a *grade* that indicates the credit worthiness based on the historical credit data.

Data preprocessing: Before the models can be fitted to the data, it needs to be carefully inspected. After removing variables which either are not sensible due a measurement which took place after a borrower has defaulted or fully paid the loan, or because the documentation of variables in the codebook was unclear, the number of variables amounts to 25 and the number of rows to 1,345,310. Furthermore, the number of missing values for each feature was considered and is presented in table A1 in the appendix. The largest proportion of missing values has the variable *employment length* with 5.64%. All observations which have at least one missing value were removed since it was required by the used R-packaged *caret*. A further conducted preprocessing step was to look for outliers. Outliers can negatively influence the predictions of the neural network since very large values of features can lead to high weights associates with those features. Figure A1 in the appendix shows histograms for each continuous feature of the unprocessed data. Very skewed distributions with a few extrem values can be seen for nearly every variable. After checking if the outliers can be considered as missings-at-randoms by evaluating the correlation between outliers and the dependent variable, I had to conclude that there is some correlation structure. Therefore, I decided not to remove the outliers but to cap them for

[4]Anonymous data about credit history and if someone has fully paid or defaulted was available on the LC website for many years and could be scraped in order to have it in a data set. For this project, I use a ready-to-use data set which is publicly available on the machine learning competition website *kaggle.com* (*Lending Club data*, 2021).

all values larger than the 99th percentile at the value associated with the 99th percentile. The results can be inspected in figure A2 in the appendix. Although it is not important at all for a neural network I present in figure A3 in the appendix a plot often referred as "correlogram" that shows the pairwise correlations between all continuous features. One can see that only the features *loan amount* and *installment* are highly correlated with a coefficient of 0.95. This could lead to an issue of multicollinearity for the logistic regression model. However, for the neural network multicollinearity is not an issue since the neurons in the input layer are not connected and hence correlation does not matter.

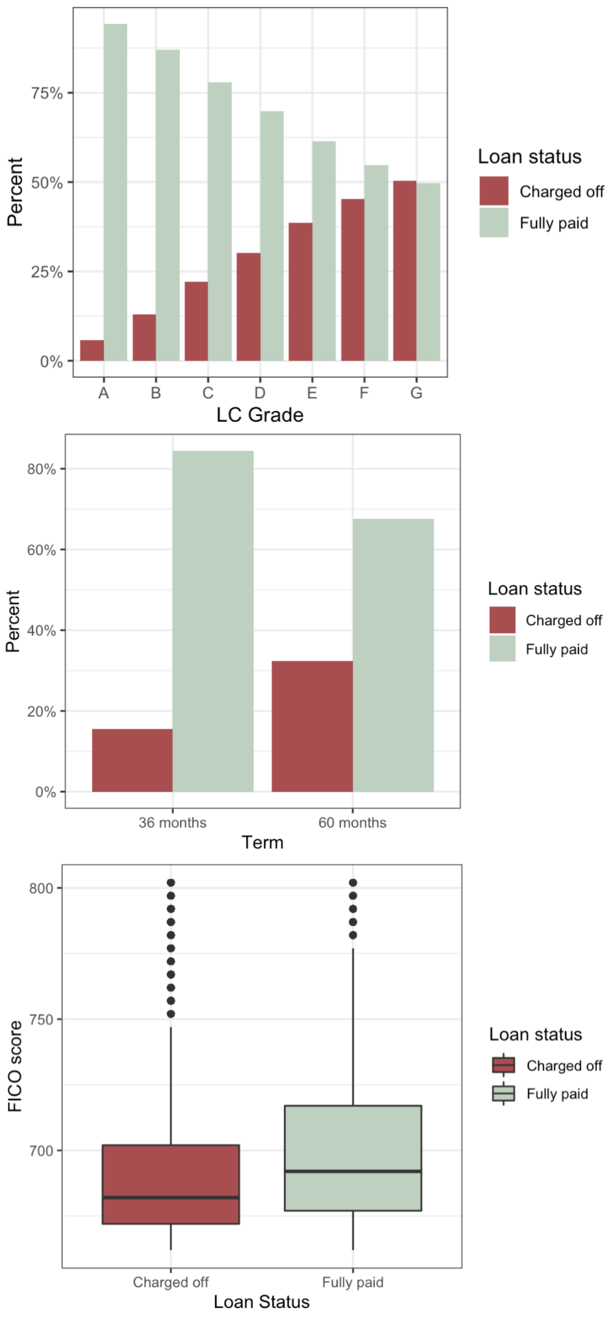


Figure 3: Exploratory analysis of the relation between the dependent variable and three selected features (Lending Club grade, duration of a credit and Fico score)

Exploratory analysis: Besides many advantages of a neural network, the largest disadvantage is the black-box architecture of the model. It is a highly parameterized model where the individual weights cannot be interpreted meaningfully since they are just adjusted and updated during the back-propagation in order to minimize the loss function. Therefore, one cannot tell why a neural network classifies an observation to the one or the other class. For this reason, an exploratory analysis is essential to get an idea how variables are related to each other. Due to limited space in this paper, I will only present three selected relationships which seems to be interesting. Figure 3 shows the relationship between the dependent variable (loan status) and the Lending Club grade, the duration of a credit and the Fico score. The first plot in figure 3 shows that the better the LC grade, the higher the proportion of borrowers which have fully paid their loan. The second plot reveals that the proportions of borrower which had a default is higher for longer credit duration. Finally, people with higher Fico score pay back the loan more frequently on average.

Results and Discussion

Training results of the neural network: The results from the *k*-fold cross-validation can be seen in table 1. The cross-validation with stratified sampling needed 7.5 hours to train. The best hyperparameter setting was found to be a two layer design where each layer contains ten neurons. Hence, the network is a "deep neural network". The area under the precision-recall curve (AUC-PR) is 0.371. For the cross-validation with down-sampling a training time of 1.6 hours was required which is much lower since many observations of the majority class were removed during the sampling which reduced the computational effort. A network design with only one layer with five neurons was found to have the highest performance with an AUC-PR of 0.372. Finally, the up-sampling within the cross-validation took the longest time to train (10.2 hours) since the models were trained with even more data due to the sampling method. Again, a deep network structure with two hidden layers with ten and five neurons respectively had the highest AUC-PR of 3.73. An interesting result is that the area under the curve is nearly the same for all of the three different sampling methods within the cross-validation. However, the network design differs. This results shows that, on the one hand, the neural network seems not to require a sampling method if the classes are imbalanced. On the other hand, using down-sampling, and hence decreasing the number of observations by removing random observations of the majority class, seems to be preferable due to much lower computational costs and same training results.

	Layer 1	**Layer 2**	**AUC-PR**	**Training time in hours**
Stratified sampling	10	10	0.371	7.5
Down-sampling	5	0	0.372	1.6
Up-sampling	10	5	0.373	10.2

Table 1: Results of the *k*-fold cross-validation of the feed-forward neural network.

Tesing results of the neural network: Figure 4 shows the precision-recall curves for all three models which performed best in the different cross-validations. The red curve represents the neural network model trained with stratified sampled folds in the cross-validation. The green curve represents the neural network trained with down-sampling and the blue curve belongs to the model trained with up-sampled folds within the cross-validation. As described in the methods section, the PR-curve shows the trade-off between precision (y-axis) and recall (x-axis). The horizontal line shows the proportion of the minority class (20%). One can see that with increasing recall the precision decreases, vice versa. This implies that with increasing number of predicted *defaults*, also the number of false positives increases. Interestingly, the curves are barely distinguishable which implies that the models perform equally in the test set. This and the result of the training (table 1) indicates that the different sampling methods do not influence the performance of the neural network.

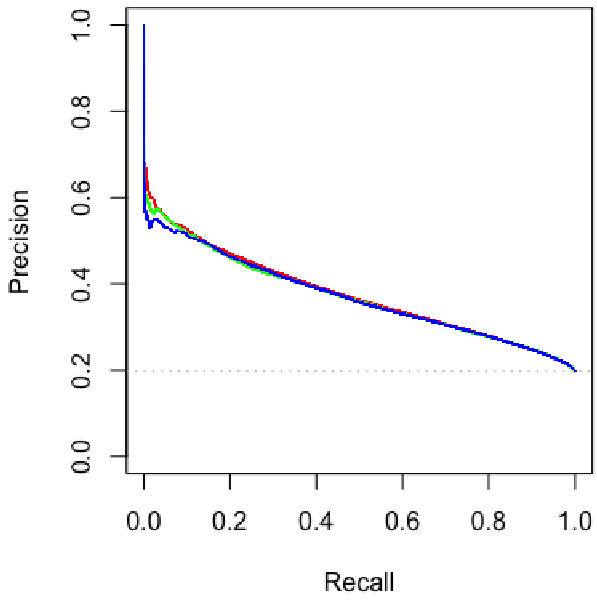


Figure 4: Precision-Recall (PR) curve of the predictions on the testing set by the three feed-forward neural networks which performed best in the cross-validations with different sampling methods. *Red curve*: NN trained with stratified sampled folds in CV; *Green curve*: NN trained with down-sampled folds in CV; *Blue curve*: NN trained with up-sampled folds in CV.

Logistic lasso regression results: For a fair comparison, the logistic lasso regression model is trained under the exact same conditions as the neural network: cross-validation with different sampling methods. The results of the training can be seen in figure 5 which

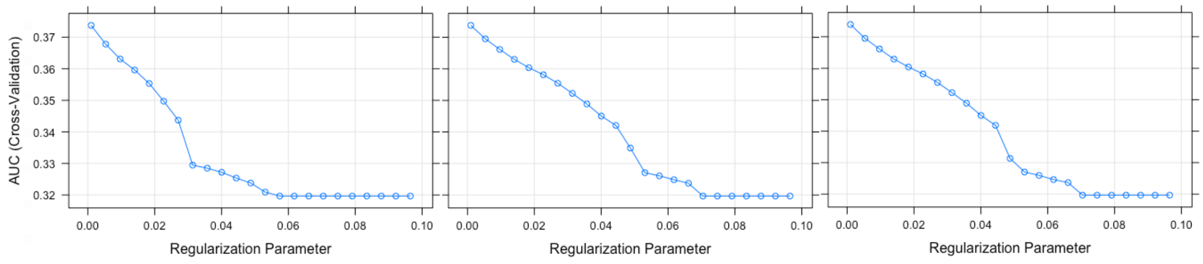


Figure 5: AUC-PR values for different values for the hyperparameter λ. *Left* plot: result of cross-validation(CV) with stratified sampling; *Middle*: CV with down-sampling; *Right*: CV with up-sampling.

shows how the AUC-PR changes with increasing values for λ. The results are surprising since the L1 penalty term is supposed to reduce the models' complexity. However, for every tried sequence of lambda values the algorithm has always chosen the value closest to zero, where zero would imply no penalty at all. Therefore, I decided to start with a lambda value of 0.001 which was then chosen during all cross-validations which is also showed in table 2 with the corresponding AUC-PR values. One possible explanation why the algorithm always favored the smallest value for λ is the large number of observations. For the cross-validation (CV) with stratified sampling five features were set to zero. For the CV with down- and up-sampling only two features were eliminated. In all three models the penalty term set the feature *loan amount* equal to zero which was highly correlated with the feature *installment*. The individual regression coefficients can be found in table B1 in the appendix.

	λ	AUC-PR
Stratified sampling	0.001	0.3737508
Down-sampling	0.001	0.3737865
Up-sampling	0.001	0.3739045

Table 2: Results of the *k*-fold cross-validation of the logistic lasso regression for 23 lambda values from 0.001 to 0.1.

Figure 6 shows the precision recall curve for the logistic regression models. The curves are, again, very similar which implies that the different sampling methods within the cross-validation have almost no influence on the prediction performance of the models. Hence, the models perform equally regardless which sampling method was used during the tuning of the hyperparameter λ of the L1 penalty term.

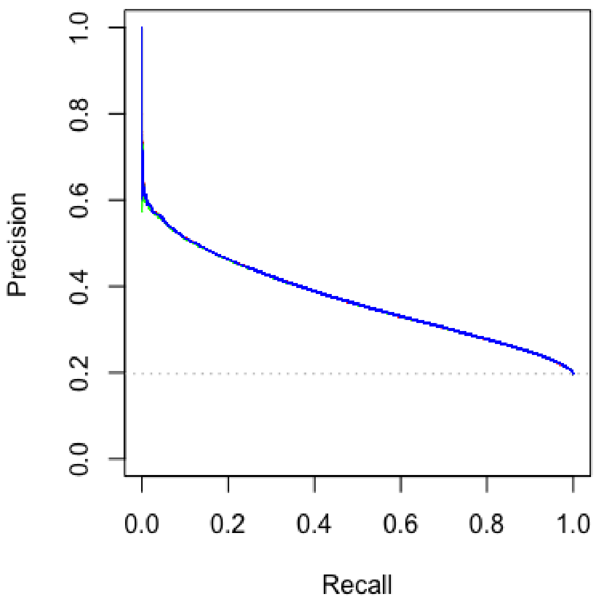


Figure 6: Precision-Recall (PR) curve of the predictions on the testing set by the three logistic regression (LR) models with L1 regularization, which performed best in the cross-validations with different sampling methods. *Red curve*: LR trained with stratified sampled folds in CV; *Green curve*: LR trained with down-sampled folds in CV; *Blue curve*: LR trained with up-sampled folds in CV.

Comparison and Discussion: Table 3 shows the values for the area under the precision recall curve as well as the weighted accuracy metric for all six models. The AUC-PR is the same for all models (0.37), except the neural network which was trained with cross-validation using stratified sampled folds. However, this

Model	**AUC-PR**	**WACC**
Neural Network strat. samp.	0.38	0.80
Neural Network down-samp.	0.37	0.70
Neural Network up-samp.	0.37	0.80
Lasso strat. samp.	0.37	0.80
Lasso down-samp.	0.37	0.67
Lasso up-samp.	0.37	0.66

Table 3: Comparison of prediction results of all neural network and logistic regression models. AUC-PR abbreviates area under the precision recall curve and WACC abbreviates weighted accuracy.

model performs only 0.01 better than the others. In contrast, the weighted accuracy measure shows more variance. The neural network models trained with cross-validation using stratified sampling and up-sampling as well as the logistic lasso regression model trained with stratified sampled folds performed best with an WACC of 0.80. The neural network trained with down-sampled folds achieved an accuracy of 0.70. The logistic lasso regression models, trained with down- and up-sampled folds in the cross-validation, performed worst with a weighted accuracy of 0.67 and 0.66 respectively. This difference is difficult to explain since AUC-PR of the training and testing is (nearly) the same for all models. However, the AUC-PR was used as evaluation metric in the training of the models and hence this metric was tried to maximized. One possible explanation for the larger variance of the weighted accuracy is that this difference would disappear if the WACC would have been used as evaluation metric within the cross-validation such that this metric would have been maximized during the parameter tuning procedure. Overall, both model types, the feed-forward neural network and the logistic regression model trained under cross-validation with stratified sampled folds, perform equally and best in terms of AUC-PR and WACC. Hence, one conclusion of this application is that sampling methods (up and down-sampling) are not required since the neural network can handle it by adjusting the its structure during the hyperparameter tuning. A further conclusion is that a logistic regression is preferable due to the same predictive power and much lesser computational costs than the neural network. However, a limitation of this analysis is that a simple architecture of the neural network was used and only a few different network designs could be trained due to limited computational resources. As shown by the literature, there are many examples of neural network models which outperform logistic regression models. A further limitation is that the standard backpropagation algorithm was used to update the weights of the neural network and there are algorithms which have shown better learning results (see e.g. Lee et al., 2015; Choromanska et al., 2019; Ma, Lewis and Kleijn, 2019).

Conclusion

The assessment of the credit worthiness of credit applicants is still one of the main concerns of financial institutions since credit defaults are associated with high costs. Therefore, this research project tried to predict the risk of default by using a neural network approach. Due to the skewed distribution of the classes a computationally heavy k-fold cross-validation approach for training and hyperparameter tuning was chosen. In order to judge the performance of the feed-forward neural network, a logistic regression model with L1 regularization was computed as a benchmark. The results suggest that down- and up-sampling are not required and made the predictions even worse in some cases. In contrast, stratified sampling during the cross-validation, that maintains the proportions of the classes, has led to the best results. The conclusion of this paper is that a logistic regression is in the era of *deep learning* still a powerful method for classification. However, the largest advantage of the logistic regression over the neural network are the much lower computational costs. Nevertheless, the results of this study should be considered carefully due to limitations such as a small number of tested network designs and the usage of the standard backpropagation algorithm instead of novel learning mechanisms.

References

Abdou, Hussein, John Pointon and Ahmed El-Masry. 2008. "Neural nets versus conventional techniques in credit scoring in Egyptian banking." *Expert Systems with Applications* 35:1275–1292.

Altman, E. I. 1968. "Financial ratios, discriminant analysis and the prediction of corporate bankruptcy." *The Journal of Finance* 23:589–609.

Balkan, E. M. 1992. "Political instability, country risk and probability of default." *Applied Economics* 24:999–1008.

Bayraci, Selcuk and Orkun Susuz. 2019. "A Deep Neural Network (DNN) based classification model in application to loan default prediction." *Theoretical and Applied Economics* 4:75–84.

Choromanska, Anna, Benjamin Cowen, Sadhana Kumaravel, Ronny Luss, Mattia Rigotti, Irina Rish, Brian Kingsbury, Paolo DiAchille, Viatcheslav Gurev, Ravi Tejwani and Djallel Bouneffouf. 2019. Beyond Backprop: Online Alternating Minimization with

Auxiliary Variables.
URL: *http://arxiv.org/abs/1806.09077*

D., Haan J., C. L. Siermann and E. V. Lubek. 1997. "Political instability and country risk: new evidence." *Applied Economics Letters* 4:703–707.

Davis, R. H., D. B. Edelman and A. J. Gammerman. 1992. "Machine-learning algorithms for credit-card applications." *IMA Journal of Management Mathematics* 4:43–51.

Desai, V. S., J. N. Crook and G. A. Overstreet. 1996. "A comparison of neural networks and linear scoring models in the credit union environment." *European Journal of Operational Research* 95:14–37.

Doumpos, M., K. Pentaraki, C. Zopounidis and C. Agorastos. 2001. "Assessing country risk using a multi-group discrimination method: a comparative analysis." *Financial Management* 27:16–34.

Efron, Bradley and Trevor Hastie. 2016. "Computer Age Statistical Inference.".

Feder, G. and R. E. Just. 1977. "A study of debt servicing capacity applying logit analysis." *Journal of Development Economics* 4:25–38.

Fisk, C. and F. Rimlinger. 1979. "Nonparametric estimates of LDC repayment prospects." *The Journal of Finance* 34:429–436.

Frank, C. R. and W. R. Cline. 1971. "Measurement of debt servicing capacity: an application of discriminant analysis." *Journal of International Economics* 1:327–344.

Grinols, E. 1975. *International debt rescheduling and discrimination using financial variables.* Earl Grinols.

Hastie, T., R. Tibshirani and J. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Second ed. Springer.

Hornik, Kurt, Maxwell Stinchcombe and Halbert White. 1989. "Multilayer Feedforward Networks are Universal Approximators." *Neural Networks* 2:359–366,.

Hsieh, N. C. 2005. "Hybrid mining approach in the design of credit scoring models." *Expert Systems with Applications* 28:655–665.

James, G., D. Witten, T. Hastie and R. Tibshirani. 2013. *An Introduction to Statistical Learning with Applications in R.* Springer.

Jo, H. and I. Han. 1996. "Integration of case-based forecasting, neural network, and discriminant analysis for bankruptcy prediction." *Expert Systems with Applications* 11:415–422.

Johnson, Justin M. and Taghi M. Khoshgoftaar. 2019. "Survey on deep learning with class imbalance." *Journal of Big Data* 6:27.

Khashman, A. 2009. "A neural network model for credit risk evaluation." *International Journal of Neural Systems* 19:285–294.

Lee, Dong-Hyun, Saizheng Zhang, Asja Fischer and Yoshua Bengio. 2015. "Difference Target Propagation, in Machine Learning and Knowledge Discovery in Databases." *Springer International Publishing* pp. 498–515.
URL: *http://arxiv.org/abs/1412.7525*

Lee, T. S., C. C. Chiu, C. J. Lu and I. F. Chen. 2002. "Credit scoring using the hybrid neural discriminant technique." *Expert Systems with Applications* 23:245–254.

Lee, T. S., C. C. Chiu, Y. C. Chou and C. J. Lu. 2006. "Mining the customer credit using classification and regression tree and multivariate adaptive regression splines." *Computational Statistics & Data Analysis* 50:1113–1130.

Lending Club data. 2021.
URL: *https://www.kaggle.com/wordsforthewise/lending-club*

Ma, Wan-Duo Kurt, J. P. Lewis and W. Bastiaan Kleijn. 2019. "The HSIC Bottleneck: Deep Learning without Back-Propagation.".
URL: *http://arxiv.org/abs/1908.01580*

Markham, I. S. and C. T. Ragsdale. 1995. "Combining neural networks and statistical predictions to solve the classification problem in discriminant analysis." *Decision Sciences Journal* 26:229–242.

Nantomah, Kwara. 2019. "On Some Properties of the Sigmoid Function." *Asia Mathematica* pp. 1–11.

Natasha, Azaria, Dedy Dwi Prastyo and Suhartono. 2019. Credit scoring to classify consumer loan using machine learning. Vol. 2194 American Institute of Physics Inc. pp. 1–13.

Pereira, Jose Manuel, Mario Basto and Amelia Ferreira da Silva. 2016. "The Logistic Lasso and Ridge Regression in Predicting Corporate Failure." *Procedia Economics and Finance* 39:634–641.

Piramuthu, S. 1999. "Financial credit-risk evaluation with neural and neurofuzzy systems." *European Journal of Operational Research* 112:310–321.

Rivoli, P. and T. L. Brewer. 1998. "Political instability and country risk." *Global Finance* 8:309–321.

Taffler, R. J. and B. Abassi. 1984. "Country risk: a model for predicting debt servicing problems in developing countries." *Journal of the Royal Statistical Society* 147:541–568.

Tsai, C. F. and J. W. Wu. 2008. "Using neural network ensembles for bankruptcy prediction and credit scoring." *Expert Systems with Applications* 34:2639–2649.

West, D. 2000. "Neural network credit scoring models." *Computers & Operations Research* 27:1131–1151.

Yim, J. and H. Mitchell. 2005. "Comparison of country risk models: hybrid neural networks, logit models, discriminant analysis and cluster techniques." *Expert Systems with Applications* 28:137–148.

Appendix

A Descriptive Statistics

	NA Count	NA Proportion
emp_length	78511	5.84
mort_acc	47281	3.52
total_bal_ex_mort	47281	3.52
revol_util	857	0.06
pub_rec_bankruptcies	697	0.05
home_ownership	478	0.04
dti	414	0.03
annual_inc	0	0.00
delinq_2yrs	0	0.00
earliest_cr_line	0	0.00
grade	0	0.00
inq_last_6mths	1	0.00
loan_amnt	0	0.00
loan_status	0	0.00
open_acc	0	0.00
pub_rec	0	0.00
purpose	0	0.00
total_acc	0	0.00
verification_status	0	0.00
addr_state	0	0.00
issue_d	0	0.00
term	0	0.00
installment	0	0.00
int_rate	0	0.00
avg_fico	0	0.00

Table A1: Absolut number and percentage of missing values for each feature used in the analysis.

	Feature type	Feature	NA count	Unique categories	Category counts
1	factor	delinq_2yrs	0	2	no: 1086059, yes: 259251
2	factor	emp_length	78511	5	E: 442199, B: 229340, C: 227443, A: 196555
3	factor	grade	0	7	B: 392741, C: 381686, A: 235090, D: 200953
4	factor	home_ownership	478	3	MOR: 665579, REN: 534421, OWN: 144832
5	factor	loan_status	0	2	Ful: 1076751, Cha: 268559
6	factor	pub_rec	0	2	no: 1117425, yes: 227885
7	factor	purpose	0	3	Deb: 780321, Cre: 295279, Oth: 269710
8	factor	verification_status	0	3	Sou: 521273, Ver: 418336, Not: 405701
9	factor	addr_state	0	51	CA: 196528, TX: 110169, NY: 109842, FL: 95606
10	factor	issue_d	0	12	201: 375545, 201: 293095, 201: 223102, 201: 169300
11	factor	pub_rec_bankruptcies	697	2	no: 1176953, yes: 167660
12	factor	term	0	2	thi: 1020743, six: 324567

Table A2: Descriptive statistics of all preprocessed categorical features.

	Feature type	Feature	NA count	Mean	St. deviation	Min.	25th perc.	median	75th perc.	Max.
13	numeric	annual_inc	0	74571.30	42172.89	0.00	45780.00	65000.00	90000.00	250000.00
14	numeric	dti	414	18.13	8.44	0.00	11.79	17.61	24.06	38.46
15	numeric	earliest_cr_line	0	1998.68	7.59	1934.00	1995.00	2000.00	2004.00	2012.00
16	numeric	inq_last_6mths	1	0.65	0.92	0.00	0.00	0.00	1.00	4.00
17	numeric	loan_amnt	0	14391.65	8642.17	500.00	8000.00	12000.00	20000.00	35000.00
18	numeric	mort_acc	47281	1.65	1.92	0.00	0.00	1.00	3.00	8.00
19	numeric	open_acc	0	11.55	5.27	0.00	8.00	11.00	14.00	29.00
20	numeric	revol_util	857	51.79	24.46	0.00	33.40	52.20	70.70	98.20
21	numeric	total_acc	0	24.90	11.70	2.00	16.00	23.00	32.00	61.00
22	numeric	installment	0	437.44	259.44	4.93	248.48	375.43	580.73	1221.50
23	numeric	int_rate	0	13.21	4.68	5.31	9.75	12.74	15.99	26.30
24	numeric	total_bal_ex_mort	47281	48812.35	41999.09	0.00	20886.00	37296.00	62548.00	234494.32
25	numeric	avg_fico	0	698.05	31.38	627.00	672.00	692.00	712.00	802.00

Table A3: Descriptive statistics of all preprocessed continuous features.

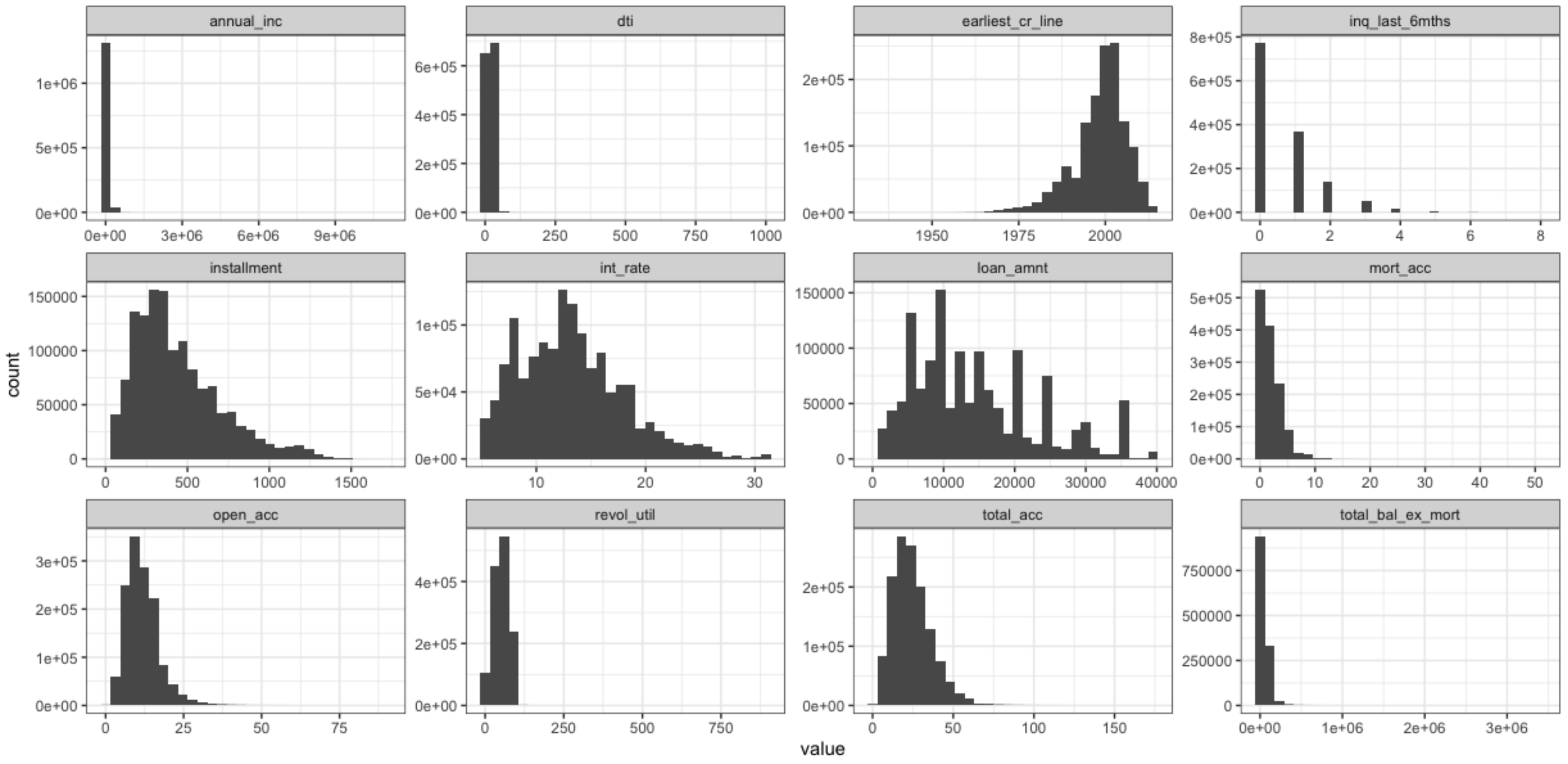


Figure A1: Histograms for all continuous features *before* outliers were capped.

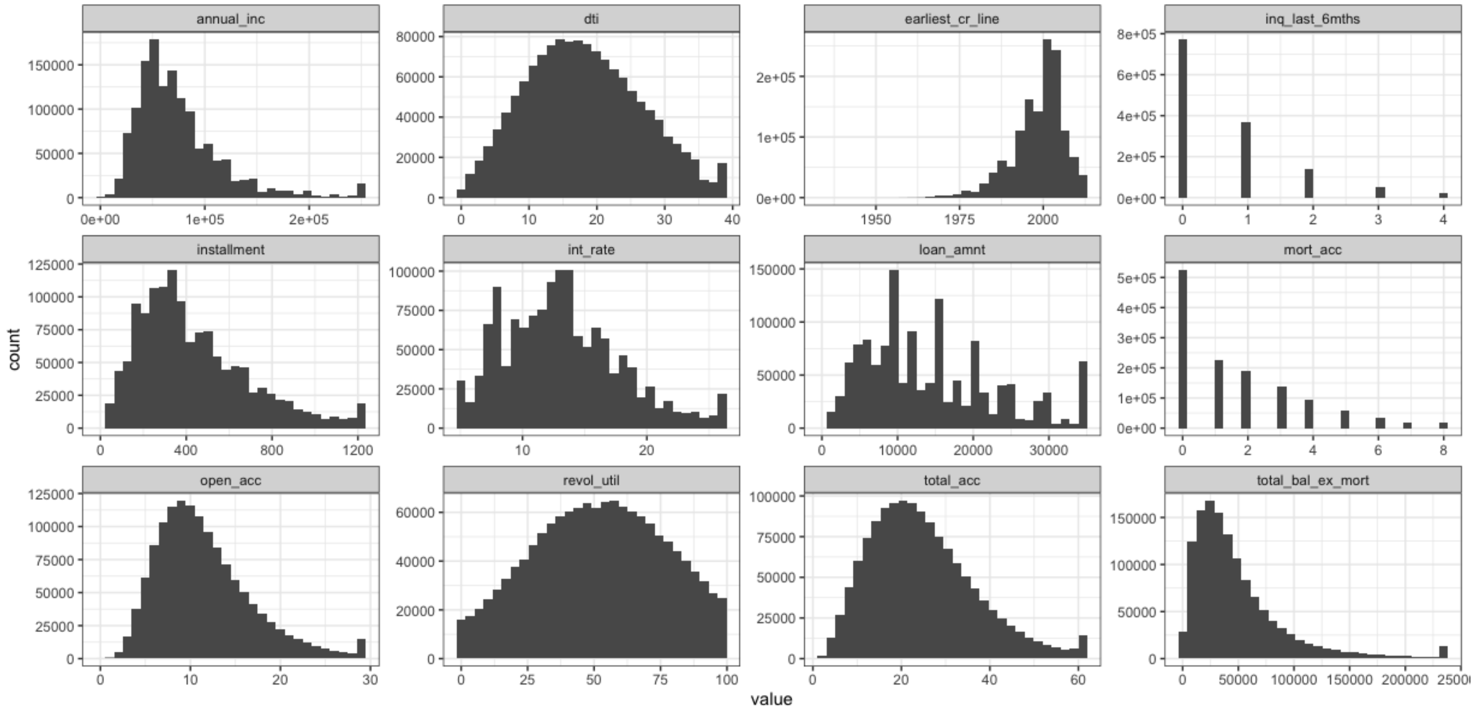


Figure A2: Histograms for all continuous features *after* outliers were capped.

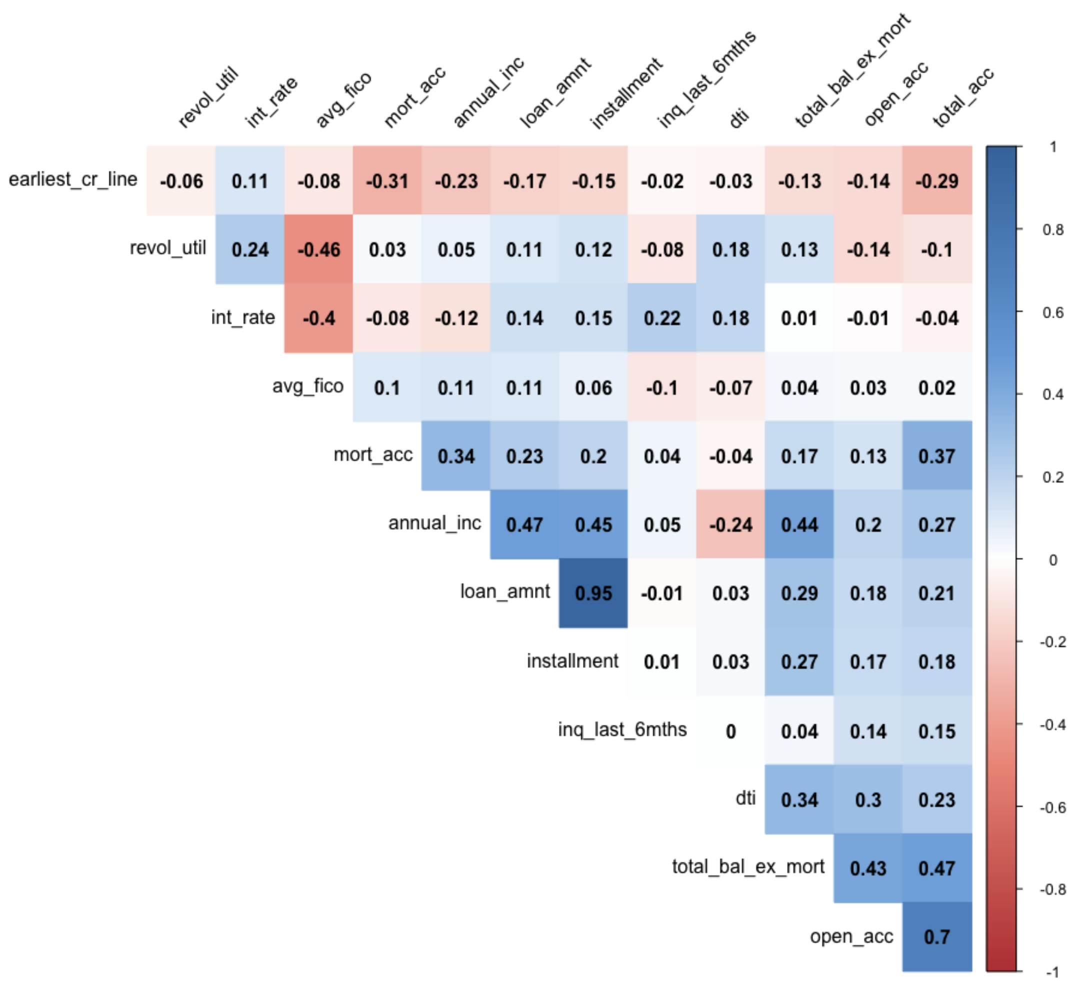
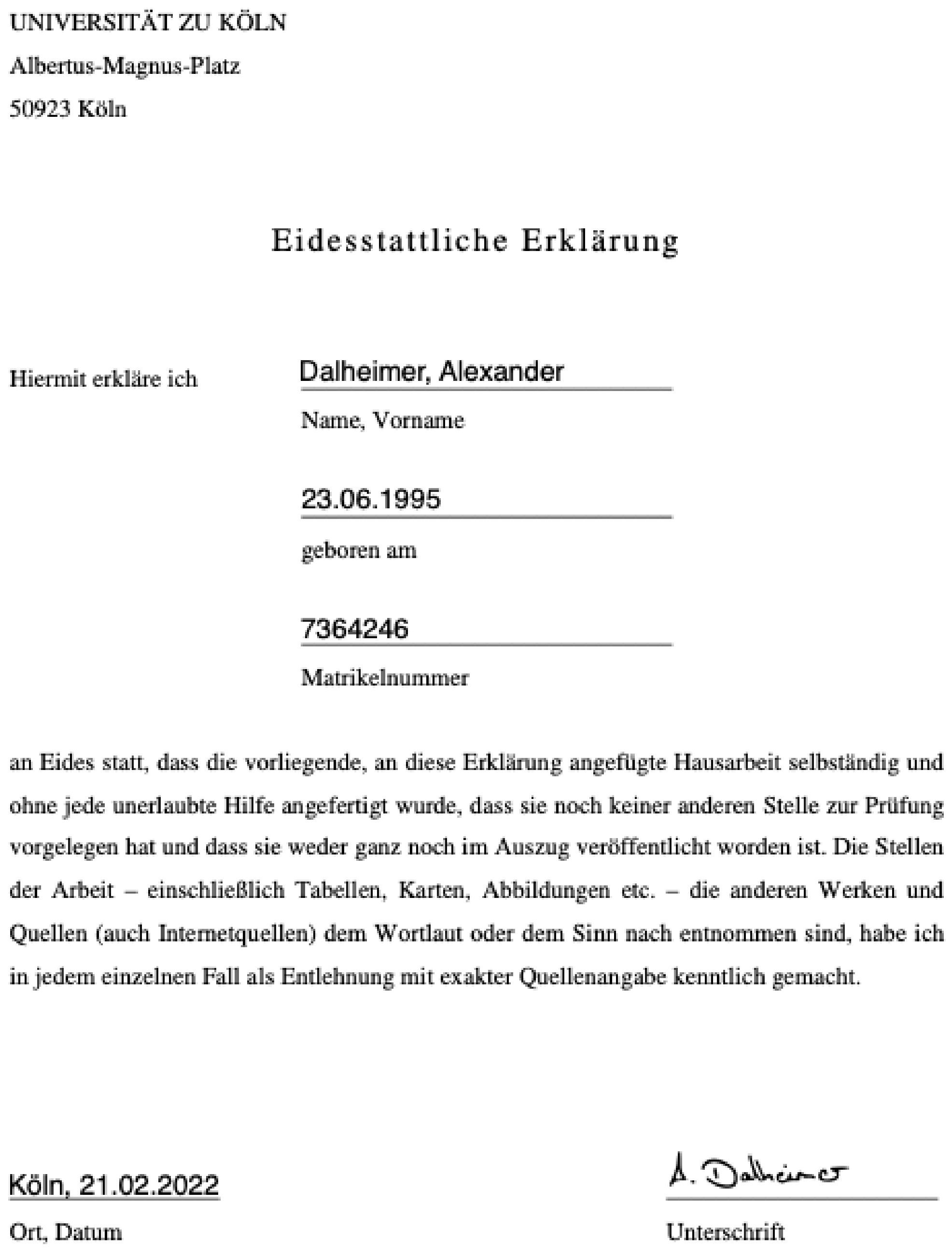


Figure A3: Correlation between all continuous features.

B Results of the Logistic Lasso Regression

	Coef. CV strat.	**Coef. CV down**	**Coef. CV up**
(Intercept)	1.59	0.01	0.01
annual_inc	0.06	0.06	0.06
delinq_2yrsyes	-0.02	-0.02	-0.02
dti	-0.17	-0.17	-0.17
earliest_cr_line	-0.03	-0.04	-0.03
emp_lengthB	**0.00**	0.00	**0.00**
emp_lengthC	**0.00**	**0.00**	0.00
emp_lengthD	**0.00**	0.00	0.00
emp_lengthE	0.01	0.01	0.01
gradeB	-0.06	-0.09	-0.10
gradeC	-0.14	-0.18	-0.19
gradeD	-0.12	-0.16	-0.17
gradeE	-0.07	-0.10	-0.12
gradeF	-0.02	-0.03	-0.05
gradeG	-0.01	-0.02	-0.03
home_ownershipOWN	-0.03	-0.04	-0.04
home_ownershipRENT	-0.12	-0.12	-0.12
inq_last_6mths	-0.07	-0.07	-0.07
loan_amnt	**0.00**	**0.00**	**0.00**
mort_acc	0.11	0.10	0.10
open_acc	-0.09	0.10	0.10
pub_recyes	-0.02	-0.03	-0.02
purposeDebt_consolidation	-0.01	-0.02	-0.02
purposeOther_expenditures	-0.04	-0.05	-0.05
revol_util	0.02	0.03	0.02
total_acc	0.06	0.07	0.07
verification_statusSource_Verified	-0.06	-0.06	-0.06
verification_statusVerified	-0.03	-0.02	-0.03
pub_rec_bankruptciesyes	**0.00**	0.01	0.01
termsixty_mth	-0.27	-0.30	-0.29
installment	-0.12	-0.13	-0.13
int_rate	-0.32	-0.34	-0.32
total_bal_ex_mort	0.05	0.05	0.06
avg_fico	0.19	0.18	0.18

Table B1: Results of the logistic regression models with optimized hyperparamter λ of the lasso regularization term, which was found during k-fold cross-validation. Note that only the bold written zeros are set exactly to zero by the L1 penalty. All other zeros are only close to zero but unequal to zero.



UNIVERSITÄT ZU KÖLN
Albertus-Magnus-Platz
50923 Köln

Eidesstattliche Erklärung

Hiermit erkläre ich	Dalheimer, Alexander
	Name, Vorname
	23.06.1995
	geboren am
	7364246
	Matrikelnummer

an Eides statt, dass die vorliegende, an diese Erklärung angefügte Hausarbeit selbständig und ohne jede unerlaubte Hilfe angefertigt wurde, dass sie noch keiner anderen Stelle zur Prüfung vorgelegen hat und dass sie weder ganz noch im Auszug veröffentlicht worden ist. Die Stellen der Arbeit – einschließlich Tabellen, Karten, Abbildungen etc. – die anderen Werken und Quellen (auch Internetquellen) dem Wortlaut oder dem Sinn nach entnommen sind, habe ich in jedem einzelnen Fall als Entlehnung mit exakter Quellenangabe kenntlich gemacht.

Köln, 21.02.2022	[illegible]
Ort, Datum	Unterschrift