Peer-to-Peer Credit Risk Forecasting
a Deep Hybrid Learning Approach

Final Version

Master Economics and Business (Financial Economics)

Erasmus University Rotterdam

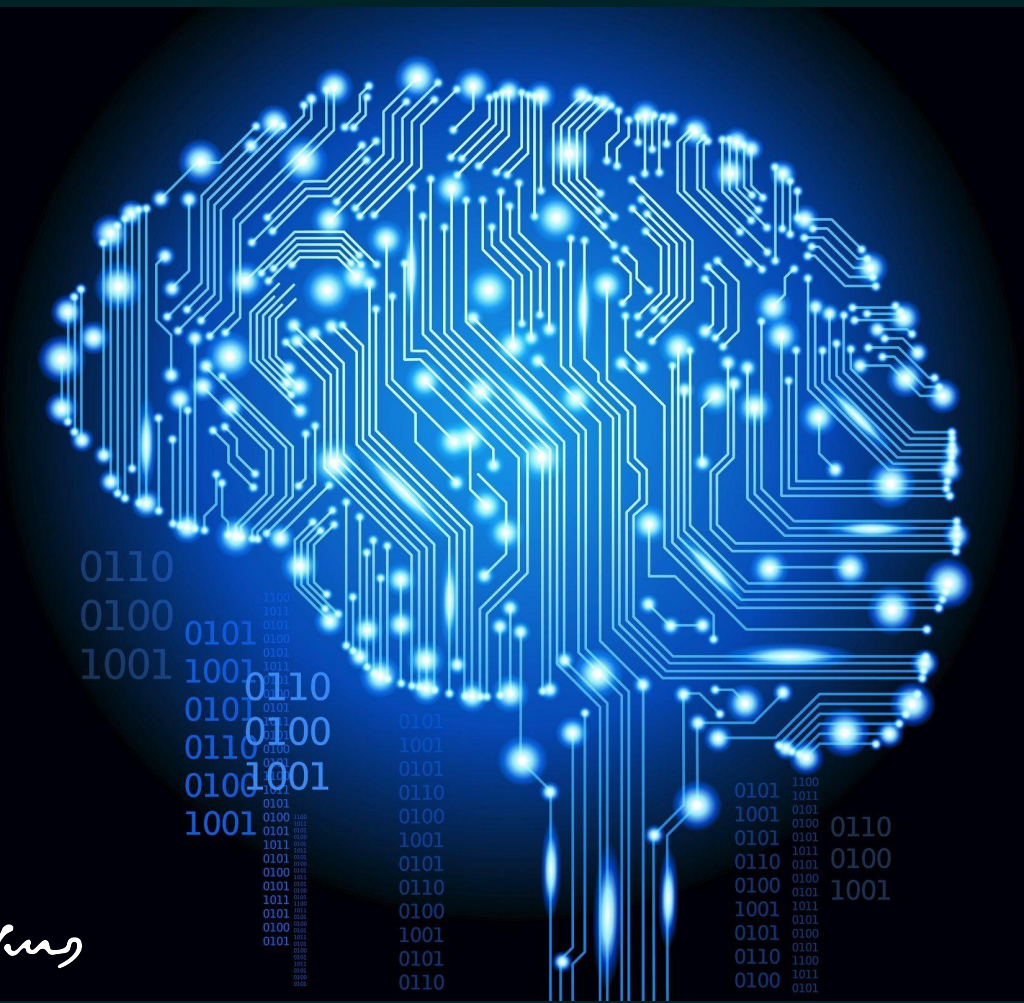
Erasmus School of Economics

M. B. Driesse 412208

Date: March 23, 2022

Supervisor: dr. R. Quaedvlieg

Second Assessor: dr. H. Zhu



Peer-to-Peer Credit Risk Forecasting
a Deep Hybrid Learning Approach

Master Thesis Economics and Business

by

M.B. Driesse 412208

For the track Financial Economics at Erasmus University Rotterdam.

The views stated in this thesis are those of the author and not necessarily those of Erasmus School of Economics or Erasmus University Rotterdam.

Credit title image: downloaded from: www.wallpaperaccess.com.

Erasmus University Rotterdam
Make it happen.

Abstract

The rapid growth in peer-to-peer lending has caused some concerns among financial researchers with regard to its effect on economic stability. This research applies deep hybrid learning to improve credit default risk assessment on an Estonian peer-to-peer lending platform called Bondora. The dataset contains 217,692 loans with 37 features. The method consists of a two step procedure where, in the first step, irrelevant features are eliminated from the dataset to decrease random noise. This prevents overfitting and improves the performance of the deep learning model that is applied in the second step. It is shown that the method can add value to current risk assessment and outperforms comparative techniques, particularly in large datasets. The validation AUC is 0.8284, which is promising, although metrics are difficult to compare across datasets and researches. The suggested method can contribute to mitigating information asymmetries between borrowers and investors which ultimately improves economic stability and may also decrease systemic risk.

Keywords: Machine Learning, Deep Hybrid Learning, Recursive Feature Elimination, Neural Network, Perceptron, Credit Default Risk, Credit Scoring, Information Asymmetry, Peer-to-Peer Lending, Bondora, FinTech

Contents

1

Introduction

After the financial crisis in 2008, peer-to-peer lending platforms have grown exponentially (Käfer, 2018). In the U.S., peer-to-peer lending has become the largest supplier of unsecured loans (Tang, 2019). As a result, many financial researchers have taken an interest in these developments. Some argue that peer-to-peer platforms are utilized to circumvent credit regulation (Braggion, Manconi and Zhu, 2018), others argue that they could be a source of systemic risk (Magnuson, 2018).

Due to information asymmetries, peer-to-peer lending possesses a high risk of investment failure. It is difficult for the lender to asses the borrowers' creditworthiness. The unsecured nature of the loans and the lack of regulation increase the credit risk of this kind of lending. In order to make an informed decision on whether to supply credit or not, lenders need to extract as much information as possible from the borrowers' loan application.

This research proposes a deep hybrid learning model that mitigates the information asymmetry between lenders and borrowers. The model helps lenders to extract information from the loan applications by eliminating irrelevant features from the dataset, thereby removing a substantial amount of random noise. Next, deep learning is applied on the less noisy features which prevents overfitting and results in superior model performance. The performance is assessed using out-of-sample (validation) AUC.

The data originates from Bondora, an Estonian peer-to-peer lending platform. Bondora is currently the 9'th largest peer-to-peer lending platform in the world in terms of total funding[1]. The platform provides only personal loans and allows all types of investors,

[1] See: https://p2pmarketdata.com/.

which makes it a "pure" peer-to-peer lending platform. This sets it apart from other platforms such as mintos which also provide business loans (and a number of other types of loans) and also from platforms such as Credimi, which only allow institutional investors. Furthermore, Bondora was founded in 2009 which makes it one of the older platforms in Europe with a relatively large track record.

Defaults are predicted using 37 a priori loan features as inputs for a sequential deep neural network. Two models are defined; one based on the complete dataset and another based on a subset of matured loans. The former has a validation AUC of 0.8284, the latter of 0.8000. Compared to other researches these results seem promising, although it is difficult to compare metrics across datasets. An estimation of the value added sketches the picture that deep hybrid learning can improve profits for Bondora investors by superior assessment of the creditworthiness of Bondora borrowers. The model outperforms other machine learning techniques on the complete dataset but not on the subset of matured loans.

This research fits between two literatures; the literature on credit scoring and the literature of peer-to-peer lending. In the second half of the 20th century statistical methods were introduced to estimate credit ratings and by the end of the century it has been show that machine learning methods yield superior results in many different settings (Huang, Chen, Hsu, Chen, and Wu, 2004). Moreover, in recent years, hybrid and ensemble machine learning methods have become particularly popular (Tripathi, Shukla, Reddy, Bopche and Chandramohan, 2022). In the literature on peer-to-peer lending, researchers have stressed the issue of information asymmetry that exists between borrowers and lenders. Many researchers are fearful that the rapid growth in peer-to-peer lending might decrease economic stability (Tarullo, 2019). This research tries to apply one of the more sophisticated methods from the credit scoring literature to decrease the problem of information asymmetry that worries researchers in the peer-to-peer lending literature.

2

Literature Review

2.1. Consumer Credit Risk Analysis

Credit risk is often defined as the potential that a borrower or counterparty fails to meet the obligations in accordance with the agreed terms[1] of a contract. This research will focus on consumer credit risk. Lenders have typically assessed this risk using consumer credit rating scores, where a lower rating reflects higher risk and increases the risk premium i.e., interest rate, on the contract. An example of these scores are FICO[2] scores which are most often used in the U.S. In the European Union a mixture of public and private institutions (Rothemund and Gerhardt, 2011, p. 1) determine consumer credit scores, these institutions all follow a directive issued by the European Commission (2021).

In the second half of the 20th century statistical methods were introduced to estimate credit ratings based on quantitative factors[3], these early methods applied Ordinary Least Squares (OLS) estimation (Fisher, 1959; Pogue and Soldofsky, 1969, Orgler, 1970), multiple discriminant analysis (MDA) (Durand, 1941; Myers and Forgy, 1963; Pinches and Mingo, 1973), logistic regression (Ederington, 1985, Steenackers and Goovaerts, 1989) and probit regression (Gentry, Whitford and Newbold, 1988). Prediction accuracy of these methods are typically between 50% and 70% percent (Huang, Chen, Hsu, Chen and Wu, 2004, p. 545).

[1] Basle Committee on Banking Supervision and Bank for International Settlements (2000, p. 1)

[2] FICO scores are credit scores created by the Fair Isaac Corporation (FICO), the score is determined based on five major factors; payment history, accounts owed, length of credit history, credit mix and new credit. The resulting rating is scored somewhere between 300 and 850 where 650 is often used as the threshold that quantifies a "good" credit history.

[3] For a comprehensive literature review see: Abdou and Pointon (2011). For a review of consumer credit specifically see: Thomas (2000).

As an alternative to these statistical measures researchers have been using machine learning techniques such as; decision trees (Boyle, Crook, Hamilton and Thomas, 1992; Khandani, Kim and Lo, 2010), random forest and k-nearest neighbors (Kruppa, Schwarz, Arminger and Ziegler, 2013) Support Vector Machines (SVM) (Han, Han and Zhao, 2013), Least Absolute Shrinkage and Selection Operator (Lasso) (Wang, Xu and Zhou, 2015), hybrid measures (Zhu, Yang, Wang and Yuan, 2018), neural networks (Dutta and Shekhar, 1988; West, 2000; Tsai, Lin, Cheng and Lin, 2009) and many others. These machine learning techniques generally outperform statistical methods (Huang et al., 2004, p. 547), with prediction accuracy's between 70% and 90%. Many researchers find that neural networks are particularly effective (Abdou and Pointon, 2011, p. 73; Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen, 2003, p. 634; West, 2000).

However, it is important to note that there is no broad consensus in the literature on which algorithms have the highest performance, since the performance often depends on the dataset that is used and the specifics of the model. It is generally true that the variation in performance between datasets is larger than the variation between (machine learning) methods[4]. There is no systematic way in which these different methods are compared in the literature, mostly it is simple trial and error to determine which algorithm performs best. Machine learning is an active field of research and models are still in development, changing virtually every year. This makes it difficult to compare results in papers that are published some years apart.

In recent years the most common machine learning methods, applied for credit scoring, are ensemble learning and hybrid learning (Tripathi, Shukla, Reddy, Bopche and Chandramohan, 2022, p. 788), where two machine learning methods are combined to produce more accurate outcomes. The difference between ensemble learning and hybrid learning is that the two algorithms work independently with ensemble learning and dependently with hybrid learning. In this research hybrid learning is applied because recursive feature elimination is applied in the pre-processing stage and then a deep neural network is applied at the regression stage.

[4] e.g. Thomas (2000, p. 160), Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen (2003 p. 634), Tripathi, Shukla, Reddy, Bopche and Chandramohan (2022, p. 791)

There are some researchers that have used the same dataset (Bondora) as this research and whose methods are also closely related. Byanjankar, Heikkilä and Mezei (2015) use a neural network to predict which Bondora loans default and demonstrate that their neural network has less false positives than a comparative logistic regression model. However, their neural network does have more false negatives. Gavurova, Dujcak, Kovac and Kotásková (2018) research which factors are most important for successful loan application on Bondora, they find that especially credit history and debt to income ratio are important determinants. Our results can perhaps most easily be compared to the results in the researches mentioned above, since they ulitize the same dataset. Note, however, that those researches were done some years ago which greatly reduces the number of loans in the dataset. This can have significant consequences for model performance.

2.2. Peer-to-Peer Lending platforms

Peer-to-Peer (P2P) lending is the disintermediation of consumer finance using a social marketplace. Removing layers of intermediation between investors and borrowers (Morse, 2015). This kind of lending platforms provide an alternative to bank lending, the platforms operate with lower overhead costs and supply loans at lower rates than conventional retail lending (Basha, Elgammal and Abuzayed, 2021). The first P2P lending platforms appeared around 2005 and they became increasingly popular after the great recession, which may reflect the increase in popularity of alternatives to traditional financial institutions (Morse, 2015).

Over the last couple of years P2P lending platforms have become an attractive investment option. Technological advances in P2P platforms have facilitated better performance in credit scoring and a real time reporting supply of lending bids, which allows investors to diversify across loans. That way, borrower risk is spread across different investors (Namvar, 2014). Moreover, Morse (2015) calculates that the Internal Rate of Return (IRR) of prosper and lending club[5] loans are 3 percent point higher than standard[6] consumer Asset Backed Security (ABS) returns. As a result, the P2P lending market[7] has grown significantly over the

[5]The two larges P2P lending platforms in the U.S.
[6]Morse refers to the Barclays Capital Fixed ABS Index as a "standard" consumer ABS.
[7]Belleflamme, Omrani and Peitz (2015) estimate the worldwide P2P lending market in 2014 to be around 11

last couple of years, in the U.S. and Europe the size of market approximately doubles every year (Käfer, 2018, p. 4).

These developments have sparked interest among researchers in financial studies. In 2017 an editorial team of the Review of Financial Studies, Goldstein, Jiang and Karolyi (2019), launched a competition to develop research proposals on FinTech topics. Out of the 156 proposals, 47 mentioned P2P lending as the main topic of research. P2P was clearly the most popular topic among researches[8] followed by big data, which was mentioned as the main topic in 27 proposals.

Researchers have focused on different aspects of P2P lending. Tang (2019) researches to what extent P2P lending platforms serve as substitutes for bank lending and finds that P2P lending serves as a substitute for infra-marginal bank borrowers and as complements for small loans, indicating that the majority of the credit expansion as a result of P2P lending occurs among borrowers who already have access to bank credit. The shift of consumer finance from banks to P2P platforms worries some researchers. According to them, P2P lending is riskier than traditional banking (Käfer, 2018). P2P lending could be seen as a source of shadow banking which, according to Tarullo (2019), could decrease economic stability.

The growth in P2P lending might also increase systemic risk. After the financial crisis of 2008 many researchers argued financial institutions have become "too big to fail" which creates misincentives and increases systemic risk (Wilmarth, 2010). However, other researchers argue that the dispersed nature and small size of the FinTech industry raises its own systemic risk concerns. For example; the small size makes them very susceptible to adverse shocks, the shared susceptibility of hacking could serve as a propagation mechanisms for economic shocks and the significant information asymmetries allow offloading risks to counterparties (Magnuson, 2018, p. 1200). P2P lending can be a potential platform through which this offloading of risks could occur.

Furthermore, Braggion, Manconi and Zhu (2018) find that Chinese borrowers increase P2P loans when mortgage Loan-To-Value (LTV) caps increase. This indicates that P2P credit

billion USD.

[8] See Goldstein, Jiang and Karolyi (2019, p. 1653), figure 5.

is utilized to circumvent credit regulation. Through these channels P2P platforms can undermine financial regulation and decrease economic stability. This finding is in line with earlier research that concludes that developments in FinTech are often driven by regulatory arbitrage (Plantin, 2015; Buchak, Matvos, Piskorski and Seru, 2018).

Other researchers have taken a closer look at the disintermediation process of P2P lending (Balyuk and Davydenko, 2019). These researchers observed that retail investors are only playing a significant role as early adopters of the platforms. Over time, institutional investors rapidly start to dominate the supply of credit. The P2P platforms themselves are performing virtually all task related to loan evaluation with the (predominantly) institutional investors passively accepting almost all loan offers. For their efforts, the P2P platforms charge a service fee and de facto become the new intermediaries. This process is called reintermediation.

3

Data

This research uses data from a peer-to-peer online loan platform based in Tallinn (Estonia) named Bondora[1]. This platform publishes loan data that is not covered by data protection laws. The platform caters to both lenders and borrowers and lenders need to be able to make their own risk assessment. For this reason Bondora publishes the data.

Bondora is the 9'th largest peer-to-peer lending platform in the world and the largest "pure" platform, meaning that it only provides personal loans and puts no restriction on types of investors. It was founded by Pärtel Tomberg, Martin Rask and Mihkel Tasa in 2009 and restructured as a private limited company in 2015. Currently the total funding volume is €569.67 M[2]. Figure 3.1 reflects the monthly funding history.

The data was downloaded[3] on 17 February 2022 and covers loan data from 21 February 2009 onward, roughly 13 years in total. Over these years a total of 217,693 loans are recorded. It records data on the loan-level, Bondora supplies a number of different loan features such as; application date, amount, loan duration, age, income, liabilities, country, employment etc. The complete list, including descriptions, of all 37 features that are utilized in this research can be found in appendix A.1. Descriptive statistics of the continuous, ordinal and dummy features, including targets, can be found in table 3.1. For a summary of some of the nominal variables we refer to Bondora's statistics overview[4].

[1] See https://www.bondora.com/en/
[2] See: https://p2pmarketdata.com/platforms/bondora/statistics/
[3] From this web page: https://www.bondora.com/en/public-reports
[4] See: https://www.bondora.com/en/public-statistics.

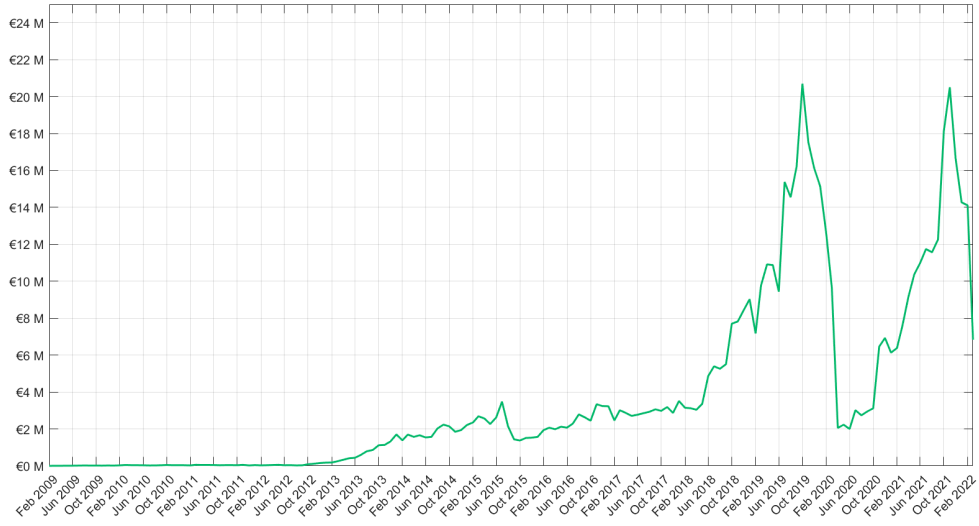


Figure 3.1: Monthly funding of the Bondora peer-to-peer lending platform since 2009.

3.1. Targets

The dataset of Bandora includes matured loans and unmatured loans. As our target we want to set loans that defaulted. But there are different ways to code defaults, two distinct defaults are utilized. The first one is based on a variable called "default date", this variable records the date on which the loan went into the default state. This means that the borrower had a late payment, it does not mean the loan defaulted indefinitely. When a borrower does not pay within 2.5 weeks a collection process is started. Sometimes the loan can be restructured, which means that the maturity date has been increased with more than sixty days. The target based on this variable will be a dummy variable called default_1 that equals one if a default date is recorded i.e., the borrower had at least one late payment, and is zero otherwise.

The previous measure simply forecasts the likelihood that a loan will default at least once before the maturity date[5]. Another approach is to consider matured loans which have defaulted and have a recorded principal write off. Bandora records the principal write offs after the loans have matured, so for this measure only matured loans are considered (which reduces the number of loans to about 55,000). This dummy is called default_2.

[5]Which loans will be classified as default depends on the chosen decision boundary.

3.2. Data Summary

Table 3.1 provides descriptive statistics of all variables that are pre-processed as continues variables, dummies or ordinal categorical variables.

Table 3.1: Descriptive Statistics.

Variable	type	Obs	Mean	S.D.	Min	Max
$default_1$	dummy	217,692	0.354	0.478	0	1
$default_2$	dummy	55,618	0.309	0.462	0	1
Bids Portfolio Manager	continuous	217,692	681.0	1154		10625
Bids Api	continuous	217,692	22.89	134.3	0	7570
Bids Manual	continuous	217,692	426.9	670.2	0	10630
New Credit Customer	dummy	217,692	0.560	0.496	0	1
Age	continuous	217,692	40.30	12.40	0	77
Education	ordinal	215,792	3.565	1.221	0	5
Number of Dependants	continuous	35,600	0.725	1.012	0	11
Applied Amount	continuous	217,692	2,708	2,317	10	10632
Loan Duration	continuous	217,692	48.51	16.95	1	120
Employment Duration Current Employer	ordinal	214,965	5.244	2.341	0	8
Work Experience	ordinal	36,522	2.915	1.494	0	5
Income From Principal Employer	continuous	217,692	270.1	1,420	0	228,400
Income From Pension	continuous	217,692	11.62	118.9	0	5,038
Income From Family Allowance	continuous	217,692	3.512	30.30	0	2,006
Income From Social Welfare	continuous	217,692	1.477	28.95	0	4,551
Income From Leave Pay	continuous	217,692	2.010	60.96	0	21,300
Income From Child Support	continuous	217,692	1.458	22.40	0	2,500
Income Other	continuous	217,692	26.64	361.5	0	50,000
Income Total	continuous	217,692	1,890	10,155	0	1,012,019
Debt to Income	continuous	217,642	4.801	13.33	0	198.0
Free Cash	continuous	217,642	75.53	547.6	-2,332	158,748
Verification Type	ordinal	217,642	3.192	1.291	0	4
Existing Liabilities	continuous	217,692	2.814	3.121	0	40
Liabilities Total	continuous	217,692	468.5	26,625	0	1.24×10^7
Refinance Liabilities	continuous	217,692	0.116	0.711	0	23

4

Methodology

In this research a combination of two different machine learning algorithms is applied. In this manner the attributes of both methods can be utilized. For the prediction of loan defaults we need an algorithm that scores high on forecasting accuracy, for this a neural network is applied. A disadvantage of neural networks is that they have a very low interpretability, the algorithm operates as a "black box". Moreover, neural nets tend to overfit when there is a lot of random noise in the dataset.

So, in the pre-processing stage, a single perceptron is used for feature selection. This has two main advantages. Firstly, it allows us to interpret which features are the most important determinants of loan defaults, which is difficult to identify applying only neural nets. Secondly, because of the pre-processing of categorical variables the number of features in the model increases rapidly. Not all these features provide useful information in the context of forecasting loan defaults. With many irrelevant features in the dataset neural nets tend to model random noise after repeated iterations. Moreover, a large number of irrelevant features increases the processing time of training the model which limits the scope of hyperparameter optimization.

Because the two methods are applied in sequence and are dependent this is called hybrid learning. In credit scoring, many researchers have applied this combination of machine learning with "deep" learning (Hsieh, 2005; Tsai and Chen, 2010; Xia, Liu, Li and Liu, 2017; Zhu, Yang, Wang and Yuan, 2018). It is also sometimes referred to as Deep Hybrid Learning (DHL).

4.1. Perceptron

4.1.1. Algorithm

The Perceptron was introduced by Rosenblatt (1957). It is a classification algorithm that predicts two possible outcomes based on a vector of input variables x_n[1]. Figure 4.1 is a schematic depiction of the algorithm.

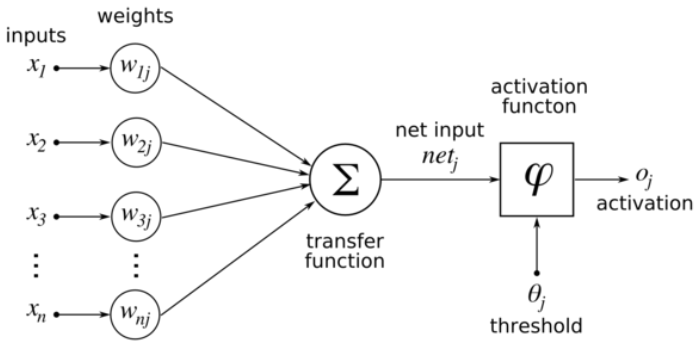


Figure 4.1: Schematic depiction of a Perceptron. Source: Mitchell (1997, p. 87)

It takes the dot product of the input vector x_n and a vector of "weights" $w_{n,j}$ to calculate a net input$_j$ (equation 4.1).

$$\text{Net Input}_j = \sum_{j=1}^{N} w_{n,j} x_n \tag{4.1}$$

This net input$_j$ goes through an activation function, if the threshold θ_j is met, the function activates and activation o_j takes on the value of 1 and is 0 otherwise. The algorithm can use different activation functions. A popular one is the given in equation 4.2.

$$\phi(x_n) = \begin{cases} 1 & \text{if } \sum_{j=1}^{N} w_{n,j} x_n + b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Here, a bias b, independent of the input vector x_n, is added to the net input$_j$. This effectively raises the threshold θ_j[2].

Initially, the weights $w_{n,j}$ are random small values, so the output o_j is also random. Next, the algorithm uses a variable d that equals 1 if the output o_j is equal to the outcome$_x$ and -1 otherwise. Using this variable the new weights $w'_{n,j}$ are calculated as follows:

$$w'_{n,j} = w_{n,j} + \eta d x_n \tag{4.3}$$

[1] outcome$_x \in \{0, 1\}$
[2] If b is negative it can also lower the threshold θ_j.

It can be seen that the weights increase when the output equals the outcome and decrease when the output does not equal the outcome. Furthermore, it can be seen in equation 4.2 that this algorithm creates a line in two dimensions (N = 2) and a (hyper)plane in higher dimensions (N > 2). This means that, in this form, it is only able to classify linearly separable classes.

η is the learning rate, which is a parameter that scales the weight correction. If η is too high it takes many iterations to find the optimal solution but if η is too low the algorithm might overshoot and not find the optimal solution. The algorithm has no analytical solution so it uses iterations instead. It keeps iterating until a previously defined number of iterations is met or until some stopping criterion is met.

This research applies the Perceptron module of sklearn[3] (in Python). All parameters are left as default meaning that the $\eta = 1$ and $b = 0$. The stopping criteria works as follows; the algorithm splits the input data into 90% training data and 10% test data and uses the test data to calculate the AUC (classification scoring metic, discussed in section 4.5.1), if this out of sample AUC does not improve for five sequential iterations the algorithm terminates with a maximum of 1000 iterations.

4.1.2. Recursive Feature Elimination

A perceptron is in some ways similar to linear regression, where the weights of the perceptron can, for the purpose of this research, be interpreted as the coefficients/estimates in linear regression. Both algorithms essentially create a line of best fit. The difference is that with linear regression this line is calculated by minimizing the sum of squared residuals, which can be done analytically by solving the first order conditions of the OLS estimates. With a binary dependent variable (linear probability model), this line can be used to classify the new inputs. More suitable would be a probit or logit model where the coefficients are first normalized using cumulative distribution functions.

A perceptron, on the other hand, creates a line of best fit based on the iterative process discussed above. This line divides the featurespace so that new inputs can be mapped and

[3]See: https://scikit-learn.org/stable/modules/linear_model.html#perceptron

classified. The commonality between linear regression and perceptrons is that, once the input vector x_n has been normalized, the coefficients/weights can be interpreted as feature importance. This is the goal of this first step of the hybrid model.

Next, the weights from the perceptron are used for recursive feature elimination (Guyon and Elisseeff, 2003). All features are ranked[4] from largest to smallest based on their absolute weights[5]. Now we run the perceptron and calculate the AUC using 5-fold cross validation. Then, the feature with the smallest coefficient is eliminated and we run the algorithm again, if the AUC does not decrease, that feature remains excluded. If the AUC decreases after the elimination, the feature will be included. This process will be repeated over all features. Using the AUC, or some other scoring metric, of a machine learning model to decide which features to eliminate is called the "wrapper" method of recursive feature elimination and was popularized by Kohavi and John (1997). The features selected by this process are used as inputs for our final model, discussed in the next section.

4.2. Neural Network

Once the input vector has been trimmed down to the essentials, the neural network can be build. Perceptrons can be viewed as building blocks of neural networks. The difference is that they are called neurons and that there is a multitude of them. The connections between neurons are called synapses and have a weight and the neurons themselves have an activation function and a bias, same as in the perceptron. The architecture of a neural network is depicted in figure 4.2. As can be seen, there are multiple neurons in the network. Each input neuron is connected to all neurons in the first layer, called the hidden layer h_1. Another difference with the perceptron is that there are multiple layers behind the first layer. In this research we use "dense" layers which means that each neuron of the input layer is connected to all neurons in the first hidden layer h_1 and each neuron in the first hidden layer is connected to all neurons in the second hidden layer h_2, and so forth. The

[4] After pre-processing (discussed in section 4.3), 173 features remain. One can construct 2^{173} different subsets from these features (the total number of subsets (n) that can be constructed without limiting the size of each subset (k) can be calculated as: $\sum_{k=0}^{n} \frac{n!}{k!(n-k)!}$ which in this particular case equals n^2-1, this can be proven using the binomial theorem where $x = y = 1$). In order to know which subsets to test with cross validation, the discussed ranking of weights is constructed. It reduces the number of times the model has to run to 173, which is feasible using a single perceptron.

[5] Weights can be negative.

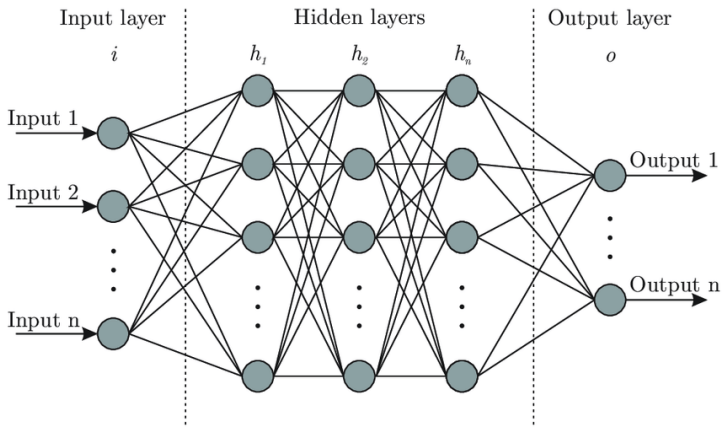


Figure 4.2: Schematic depiction of the architecture of an artificial neural network . Source: Bre, Gimenez and Fachinotti (2018, p. 1430, figure 1)

last layer consist of one neuron in regression and multiple neurons in classification, where the number of neurons in the output layer equals the number of classes.

Since the layers are connected in sequence this architecture is sometimes referred to as a sequential neural network or a multilayer perceptron. The structure of multiple layers was first introduced by Ivakhnenko and Lapa (1965)[6]. When a model has no hidden layers it is called a shallow neural network and when it has one or more hidden layers it is called a "deep" neural network. These kind of networks have been shown to outperform alternative machine learning methods in numerous important applications (Schmidhuber, 2015).

The activation functions in the neurons of the neural network are slightly different than the one in the perceptron but equally simple. This research applies ReLU (Rectified Linear Unit) activation functions which can simply be expressed as $f(x) = \max(0, x)$. Hence, if all weights coming into the neuron add up to more than 0 the output is the same as the input. If the weights add up to less than 0, the output is zero. Instead of ReLU the logistic sigmoid function[7] used to be the most popular option. This function normalizes all inputs between zero and one, a form of cumulative distribution functions, which seems perhaps more intuitive. However, in a famous paper by Glorot, Bordes and Bengio (2011) it was found that ReLU actually performs better than sigmoid and as of 2017 ReLU is the most popular activation function for deep neural networks (Ramachandran, Zoph and Le, 2017). The output layer has one neuron and uses the sigmoid activation function.

[6] According to Schmidhuber (2015, p. 90).
[7] $f(x) = \frac{1}{1+e^{-x}}$

An important deviation from a single perceptron is the way the weights get updated. In a neural network this happens in a process called backpropagation[8]. A loss function is calculated, which basically takes the difference between the predicted output of the model and the actual outcome$_x$. Next, the weights in the last hidden layer are updated with the gradient of that loss function. These adjustments are summed at each neuron and that sum functions as the loss function for the next hidden layer. This process is iterated over all observations in the dataset. The details of this process are beyond the scope of this research but it can be assumed that through this process the loss functions essentially gets minimized. This process of finding the local minimum of the loss function is know as gradient descent. A popular loss function for regression is the squared error which is, obviously, simply the square of the difference between output and outcome$_x$. In classification cross-entropy is often used. The loss function applied in this research is binary cross entropy[9].

4.3. Pre-processing

Before the Bondora data can be used in the perceptron and neural network some pre-processing is necessary. Firstly, all categorical variables need to be encoded. If the categorical variable is ordinal, such as work experience, all categories are simply coded accordingly; smallest is one, the second smallest is two and so forth. This cannot be done with nominal variables since there is no predefined order. An example is the day of the week on which the borrower applied for the loan. For those variables dummies are constructed for each category. This process inflates the number of features in the model which is a motivation for recursive feature elimination.

Secondly, the data needs to be normalized. The optimal weights in the deep neural network are interdependent, meaning that we cannot scale individual weights to match the average of each input in the input vector x_n as can be done with, for example, OLS. If

[8] The backpropagation algorithm was introduced by Rumelhart, Hinton and Williams (1986).

[9] Also known as "log loss", the formula is given:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \times log(p(y_i)) + (1-y_i) \times log(1-p(y_i))$$

Here, y_i is the target i.e., default or non-default. and $p(y_i)$ is the predicted probability for the target to be 1 for all N points.

we do not normalize the data the algorithm has trouble optimizing the loss function (as a matter of fact, Keras cannot even calculate the loss function) due to the difference between the mean of, for example, applied amount and age. In short, all data is normalized. The min-max-scaler of sklearn[10] is used. Each input x is normalized to x' according to equation 4.4[11].

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4.4}$$

Finally, neural networks cannot handle missing values so we need some way to deal with them. One way is to simply drop each observation that contains a missing value, which greatly reduces the number of observations. Quick and dirty ways to deal with missing values are to replace them with some constant or the column average. This research replaces missing values with the constant -1, expecting that the network will learn to ignore the values[12]. A more sophisticated way would be to forecast the missing values using, for example, regression (Little and Rubin, 2019). Missing values can also contain information, when occupation is left blank for example. Recently developed advanced neural nets utilize this "informative missingness" to improve model performance (Che, Purushotham, Cho, Sontag and Liu, 2018). As a final remark, officially, recursive feature elimination is also part of pre-processing.

4.4. Hyperparameter Optimization

The weights of the model are parameters which are optimized through the process of back-propagation. These parameters are known as trainable parameters to which we can apply gradient descent. There are other parameters in the model such as; the number of layers, the number of neurons in each layer, the learning rate, the activation function, the loss function etc. These parameters, which define the architecture of the model, are known as

[10] See: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

[11] This normalization is heavy on skewed data such as refinance liabilities and income from leave pay. To deal with skewed features, log transformations can be applied or, alternatively, the feature values can be ranked before normalizing them. This is not done in this research but ideally it should have been done. Applying ranks to the most skewed features in the pre-processing phase does not change the validation AUC or feature selection. However, it does change the magnitude (not sign) of some feature weights in the perceptron model depicted in figure 5.1.

[12] François Chollet (2021), the creator of Keras, suggests that missing values can be replaced with values that are not already meaningful in the dataset. In this research the value -1 is chosen because the features with lots of missing values are strictly positive. According to Chollet, the network will learn that values of -1 are not informative, conditional on the data being missing at random. Note that missing values are not common in most features in the data(see table 3.1).

hyperparameters.

Hyperparameters cannot be optimized through gradient descent, they are non-trainabele. This means that the hyperparameters are a choice of the model builder. For the activation function there exists a large body of research that finds that ReLU is superior to other activation functions such as sigmoid in virtually all settings. Loss functions are designed for a specific purpose with binary cross entropy being the most popular choice for binary classification. So we do not optimize these two hyperparameters but simply follow general machine learning practice.

The optimal value of the other hyperparameters, most importantly the number of layers and the number of neurons in each layer, depend on the dataset that is used. There exist, to my knowledge, no standard practice or heuristic when it comes down to this choice. Therefore, a grid search (or something similar) needs to be performed. For this we have to define a range in each hyperparameter over which we want to search for a optimal solution. Furthermore, a step size needs to be defined to determine the increment between each search. In this research three hyperparameters are optimized. Firstly, the number of layers, where we search between 1 and 5 layers with a step size of 1. Secondly, the number of neurons in each individual layer where we search between 32 and 256 neurons with a step size of 32. Third, the learning rate where we choose between 0.001 and 0.0001[13].

A calculation of the options for this relatively basic architecture shows that there exist a total of $(8^5 + 8 \times 4) \times 2 = 65,600$[14] different configurations of the model. In a grid search, for each combination a scoring metric is calculated and the optimal combination is chosen. In our case we would have to find the optimal point in a 7-dimensional[15] grid containing 65,600 datapoints. It can be quickly seen that this is unfeasible due to processing limitations. To circumvent this problem we use KerasTuner[16] (Python module).

In this research KerasTuner is used to apply the Hyperband methodology of Li, Jamieson,

[13] Note that the choice of these ranges and step sizes are essentially still arbitrary.
[14] 8 options for each layer times the 2 options for the learning rate. The addition of 32 comes from the options of no more layers. Once the option zero neurons occurs in a layer all further layers are necessarily also zero. The first layer cannot have zero neurons, the minimum is 32.
[15] This hyperspace consist of a dimension for each layer, one dimension for the learning rate and one dimension for the out of sample AUC.
[16] See: https://keras.io/keras_tuner/

DeSalvo, Rostamizadeh and Talwalkar (2017). These researchers define the optimization problem as something called a "pure-exploration non-stochastic infinite-armed bandit problem" where a defined number of resources like iterations, data or features are allocated to randomly sampled configurations. The search narrows down on promising configurations. The details of this algorithm are beyond the scope of this research but suffice to say that it finds optimal combinations of hyperparameters with orders of magnitude less processing time than a grid search.

KerasTuner also searches for the optimal number of iterations. Depending on the data and the structure of the model, the AUC can be increased by minimizing the loss functions through repeated iterations over the complete dataset. The problem with this process is that the model will overfit, meaning that the weights are fine-tuned to the dataset to such an extent that it essentially starts remembering the data instead of learning general relationships. The neural network will effectively model random noise in the dataset. This loss of generality becomes apparent when the AUC is calculated on data outside of the training set. If this out of sample AUC is lower than the in sample AUC, the model is overfitting.

In general, machine learning methods are better than statistical methods in the sense that they have a lower bias. They do not underfit. However, a disadvantage of machine learning is that it has high variance. It tends to overfit. In (hyper)parameter optimization and machine learning in general, the fundamental problem is this bias-variance tradeoff (Kohavi and Wolpert, 1996). This tradeoff can only be assessed by computing the out of sample AUC, which is the metric along which the tuning of the hyperparameters is scored and the choice of the number of iterations is made. It is also a key metric that is used to asses model performance, which will be discussed next.

4.5. Model Performance

4.5.1. Area Under Curve

To asses the model performance we use an out-of-sample forecasting metric. Specifically, a metric called Area Under Curve (AUC) is used (DeLong, DeLong and Clarke-Pearson, 1988). The metric is not only used to asses the performance of the final model it also functions

as an optimization objective during the hyperparameter tuning. The metric calculates the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve plots all possible points on a two dimensional grid with the true positive rate on the y-axis and the false positive rate on the x-axis and connects these points. Each point corresponds to a unique confusion matrix that can be calculated by shifting the decision boundary between default and non-default. The area under this curve is the AUC which can be used to compare different ROC's. The true positive rate is given:

$$\text{True Positive Rate} = \frac{\text{True Positives}}{\text{False Negatives} + \text{True Positives}} = \frac{\text{True Positives}}{\text{Positives}} \tag{4.5}$$

Equation 4.5 shows which percentage of the defaults were correctly classified by the model. The false positive rate is given:

$$\text{False Positive Rate} = \frac{\text{False Positives}}{\text{True Negatives} + \text{False Positives}} = \frac{\text{False Positives}}{\text{Negatives}} \tag{4.6}$$

Equation 4.6 shows which percentage of the non-defaults where incorrectly classified.

Lets say that 10% of loans default and we want to predict defaults. For any given configuration of the model, a simple solution would be to shift the decision boundary so that all predictions are zero. The forecasting accuracy would be 90%, which is pretty good. The false positive rate would be zero which is nice but the problem is that the true positive rate is zero as well, which renders the model completely useless. From this point we can shift the decision boundary so that the model start to actually predicts defaults. As a consequence the true positive rate increases which is what we want. However, it is intuitive that the false positive rate also increases. There is a tradeoff here, which point to chose depends on the users of the model. All possibilities are summarized by the ROC.

Now lets say we change the configuration of the model, we get a new ROC. How do we decide which model is better? This is where the AUC comes in, it simply allows us to compare the two sets of options without having to make the tradeoff between true and false positives. The higher the AUC, the better options we have.

4.5.2. Value Added

After the AUC is optimized and the final model is complete, the value added can be calculated as follows:

$$\text{VA(TN,FN,r,PIW)} = \text{TN} \times \text{r} - \text{FN} \times \text{PIW} \tag{4.7}$$

Where, r is the average of the interest rate multiplied by the loan amount (€736.57) and PIW the sum of the average principal, interest and penalty write offs in case of a default (€965.99). The value added reflects the revenue generated by loans correctly classified as non-default (TN) minus the costs of loans that were predicted as non-default but did in fact default (FN). The amount of TN and FN depends on the decision boundary. When comparing models, in this research, the decision boundary will be set as to optimize equation 4.7.

The implicit assumption in calculating the value added is that positives, whether true or false, are not eligible for a loan. Why positives are not included in equation equation 4.7 can be further illustrated as follows. let N, denote the total number of non-defaults and P the total number of defaults. We know that $N = \text{TN} + \text{FP}$ and $P = \text{TP} + \text{FN}$, this implies;

$$\text{TN} \uparrow \iff \text{FP} \downarrow \tag{4.8}$$

$$\text{FN} \downarrow \iff \text{TP} \uparrow \tag{4.9}$$

Let's assume for simplicity that $r = \text{PIW}$, now equation 4.7 is maximized when $\text{TN} - \text{FN}$ is optimal. From 4.8 and 4.9 it can be concluded that;

$$\text{TN} - \text{FN} \uparrow \iff \text{TP} - \text{FP} \uparrow \tag{4.10}$$

Hence, the value added reflects the amount of true positives and false positives indirectly. When $r > \text{PIW}$ the optimal decision boundary will shift leftwards relative to the point where $\text{TN} - \text{FN}$ is optimal. However, the illustration that positives are indirectly reflected still holds.

The costs of a default are difficult to estimate. When using default_2, the subset of loans has complete information on write offs and PIW is calculated as the average write offs over all defaults. However, when using default_1 there is no complete information on PIW because most loans have not yet matured. So, when using default_1, the PIW and r are calculated based on the subset of default_2, with the side-note that the relative proportions of

revenue and costs might have changed slightly over time. When the costs are overestimated the optimal decision boundary has a downwards bias and vice versa when underestimated.

4.5.3. Comparison to Other Techniques

Another way to asses model performance is to compare it to alternative techniques. The selected features will be the same for all methods. In this subsection all alternative techniques will be discussed briefly.

Logistic Regression

Logistic regression (LR) is a specific form of a binary response model which estimates the following conditional probability (Wooldridge, 2018, p. 560):

$$\text{P(Default = 1|X)} = G(\beta_0 + \beta_1 x_1 + ... + \beta_k x_k) = G(\beta_0 + X\beta) \tag{4.11}$$

The β's are estimated using linear regression and put through a normalization function $G(z)$. X equals the vector of selected features. The function $G(z)$, in logistic regression, is given:

$$G(z) = \frac{e^x}{e^x + 1} \tag{4.12}$$

This logistic function normalizes all inputs to range from zero to one. Furthermore, equation 4.12 is an example of a sigmoid function which is often applied as an activation function in neural networks. This research will apply the logistic regression algorithm of sklearn[17].

Decision Tree

A decision tree (DT) classifier is a non-parametric learning method used for classification. The method creates a decision tree which splits the data into nodes that contain thresholds for feature values. Information flows from the top node (root) down the decision tree. Each node (branch) is connected to two other nodes. Through which of both nodes the information flows depends on whether the defined threshold is met. The final node (leaf) determines how the input vector is classified. The tree is build until all nodes are "pure", meaning that all final nodes (leafs) exclusively contains defaults or non-defaults, or until some prior defined maximum number of leafs is reached.

[17] See: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

The thresholds are defined by calculating all Gini impurity metrics (or some similar metric) which reflects how well the possible node, on its own, is able to discriminate between defaults and non-defaults. It sorts all continues features based on their feature value from smallest to largest and puts a threshold between adjacent cells. The Gini impurity can then be calculated as follows:

$$I_G(p) = 1 - \sum_{i=1}^{2} p_i^2 \tag{4.13}$$

i reflects the target; one for default and zero for non-default. The equation is summing the probabilities p_i of all loans with target i being correctly classified times the probability of a mistake in the classification of those loans.

The node with the lowest Gini impurity will be the root node. Once the root node is defined, the gini impurity metrics are calculated again and the lowest values will form the next branches of the decision tree and so forth[18]. The disadvantage of this method is that it tends to overfit if the number of leafs is not limited. We will apply the decision tree algorithm of sklearn[19] and perform a grid search based on the maximum amount of leafs in the tree. The range is set from 10 to 500 with a stepsize of 10. The data point with the highest validation AUC will be selected.

Random Forest

The random forest (RF) algorithm, introduced by Breiman (2001), creates many decision trees based on randomly selected features (bootstraps). The features are selected with replacement, meaning that individual features can be utilized multiple times. This method is known as feature bagging. Information flows through all trees and classification takes place through a majority vote among all trees in the "forest".

Single decision trees have a low bias but very high variance. Low bias means high relevance between features which means there will be no underfitting, high variance means modelling random noise which causes overfitting. The multitude of de-correlated trees in the forest mitigate this problem of overfitting but increase the bias i.e., the problem of underfitting. Underfitting is, broadly speaking, less of a problem in machine learning so random forest generally outperforms single decision trees. Note that this research applies

[18]The decision tree of the subset of matured loans is depicted in appendix A.2.
[19]See: https://scikit-learn.org/stable/modules/tree.html.

recursive feature elimination to mitigate random noise which already prevents overfitting to some degree.

This research applies the random forest algorithm of sklearn[20]. The following hyperparameters will be optimized: the maximum amount of leafs in the trees (50 to 500 with step size 50), the number of trees in the forest (200 to 2000 with step size 200) and the maximum amount of features in each tree (choice between all features and $\sqrt{\text{features}}$). For this optimization the randomized search[21] algorithm of sklearn is applied. It selects a predefined number of random values for each hyperparameter and validates it using 5-fold cross validation, where the AUC metric is monitored. The set of hyperparameters with the highest validation AUC is selected.

k-Nearest Neighbors

k-Nearest Neighbors (KNN) is another non-parametric machine learning algorithm, introduced by Fix and Hodges (1989), that simply uses a majority vote among "nearest neighbors" i.e., input vectors with similar feature values. The most important hyperparameter of this model is the number of neighbours (k). In this research the KNN algorithm of sklearn[22] is applied. A grid search is performed to find the optimal number of neighbours on a range from 5 to 100 with step size 5. The distance (D) between two input vectors (X =$(x_1 + x_2 + ... + x_n)$) and (Y =$(y_1 + y_2 + ... + y_n)$) is calculated as follows:

$$\text{D(X,Y)} = \sum_{i=1}^{n} |x_i - y_i| \tag{4.14}$$

Equation 4.14 is a special case (p=2) of the Minkowski distance that is equivalent to simple Euclidean distance.

Support Vector Machine

A support vector machine (SVM) is a non-probabilistic binary classifier introduced by Boser, Guyon and Vapnik (1992). The algorithm separates the feature space by a straight line (a decision boundary). The decision boundary is set as to optimize the margin around it. The

[20] See: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
[21] See: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html#.
[22] See: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html.

only feature vectors that influence the optimal decision boundary are the ones that determine where the edges of this margin are. These are the support vectors. Input vectors are simply mapped on the feature space and classified based on their location relative to the decision boundary[23].

Defaults and non-defaults are obviously not linearly separable in the feature space. So SVM applies a kernel trick which adds an extra dimension to each feature vector where the feature space is transformed in such a way that the decision boundary can be straight. The mathematics of these kernels are beyond the scope of this research, but they allow classification of classes that are not linearly separable. The type of kernel that is used is a hyperparameter which is optimized using a grid search (choice of; radial basis function, polynomial function or sigmoid function)

This research applies the SVM algorithm of sklearn[24]. In addition to the kernel, two other hyperparameters are optimized: C, which adds a penalty for misclassified data (controls for bias/underfitting) and γ, which is a coefficient in the kernel that controls the distance of influence of a single feature vector. For high values of γ feature vectors need to be relatively close together to be recognized as a distinct class which increases overfitting. With lower values of γ, the regions separating classes get more generalized and overfitting is reduced. For both values the right order of magnitude needs to be found, we will search in the range 10^k for $k \in \{-2, ..., 2\}$.

[23] SVM is non-probabilistic, meaning that feature vectors are mapped and classified as either default or non-default with a probability of 100%. As a result, the validation ROC curve consists of a single confusion matrix. It does not realy make sense to use the AUC metric but for comparison it is calculated as follows:

$$\text{AUC} = (\text{TPR} \times \text{FPR} \times 0.5) + ((1 - \text{FPR}) \times \text{TPR}) + ((1 - \text{FPR}) \times (1 - \text{TPR}) \times 0.5)$$

[24] See: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

5

Results

5.1. Recursive Feature Elimination

The results of the recursive feature elimination are depicted in figure 5.1. RFE selected 49 out of the total of 173 features. Features with high absolute weights are, generally, more often included than features with relatively low absolute weights. Only nominal variables are eliminated all ordinal and continuous variables are included. All occupation areas, application signed hours and application signed weekdays are eliminated. However, not all nominal variables were excluded. For example, home ownership type 0 is included and increases the risk of default significantly (high weight), this nominal variable denotes homeless people. Most other home ownership types are eliminated.

Total income has the highest negative weight followed by monthly payment day 30 and 29, also relatively high we find monthly payment day 31. This is probably due to the very low amount of positives. These three days added up, only account for 81 out of the 217,693 observations. Use of loan is an interesting nominal variable as some types have a negative weight (accounts receivable, acquisition of stocks, working capital financing) while others have positive weights (home improvement).

Some final things to note; Liabilities and bids increase the risk of default and are all included. All gender dummies are eliminated. Martial status of widow, cohabitant and married have positive weights and are included. The countries of Spain and Slovakia increase default risk and are included, the countries of Estonia and Finland reduce risk but are eliminated. Furthermore, new credit credit customer is eliminated. Income from leave

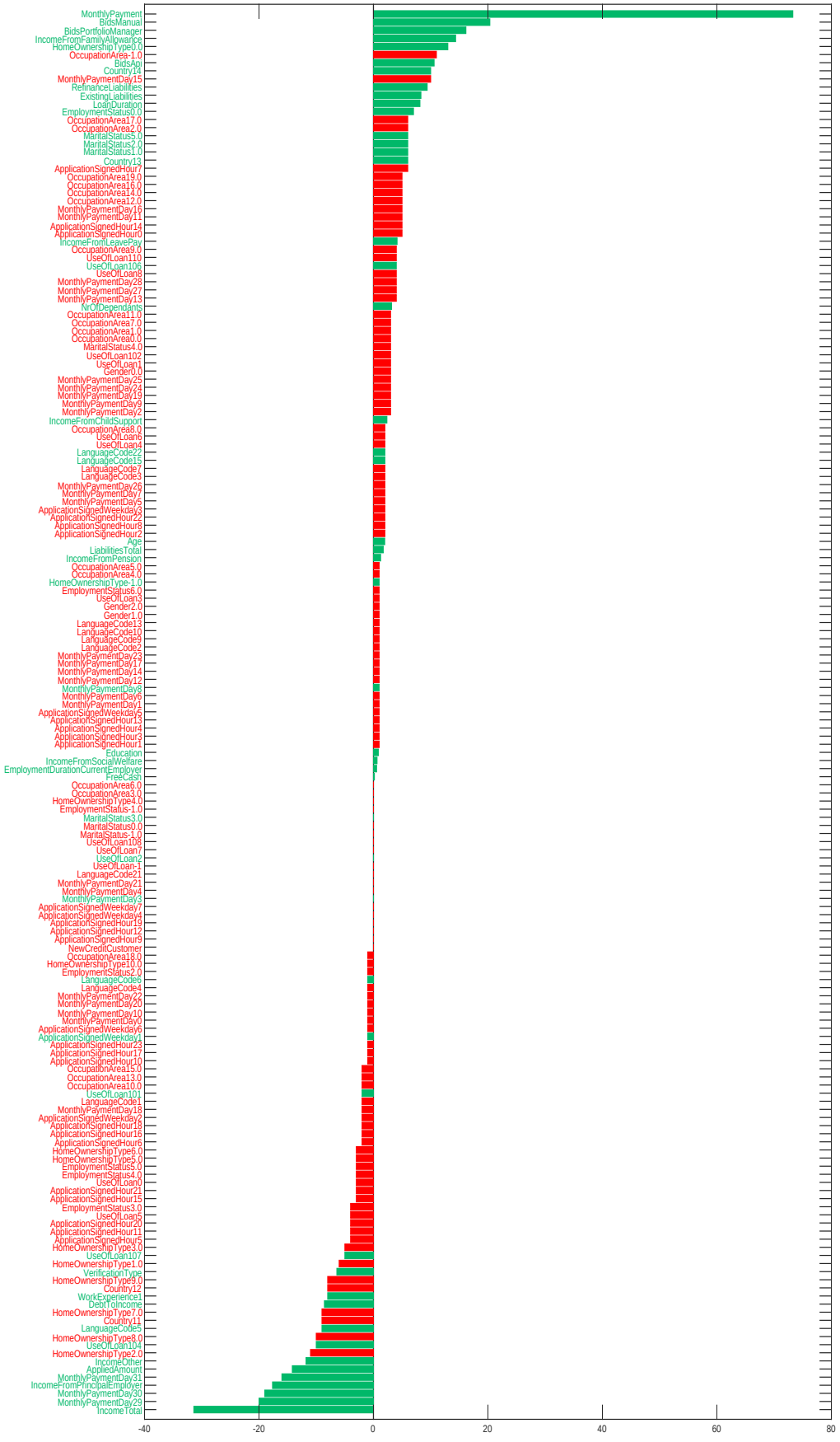


Figure 5.1: Recursive Feature Elimination by 5-fold cross validation. Features in red are eliminated, green features are included. Feature weights are graphed on the x-axis.

pay, child support and pension increase default risk, income from employer and other income decrease default risk, all are included. The better the verification type of the income the lower the default risk. Remarkably, debt to income decreases default risk as well.

5.2. Hyperparameter Optimization

Now that it is determined which features to include and which to eliminate, the model can be optimized. Two different architectures are defined, one for the complete dataset ($default_1$) and one for the subset of matured loans ($default_2$). The architectures are summarized in table 5.1. Note that the optimal number of layers for both architectures is four.

Epochs are the optimal number of iterations and batch size is the number of loans that are passed through the network for one calculation of the gradient of the loss function[1]. If the batch size is one the gradient of the loss function is calculated for each individual loan which is computationally demanding and does not provide better performance. The batch size is set so that the total number of batches in one epoch is roughly 500.

Table 5.1: Architecture summary.

Dataset	$default_1$	$default_2$
Neurons input layer	49	49
Neurons first layer	224	64
Neurons second layer	128	160
Neurons third layer	224	224
Neurons fourth layer	224	128
Neurons output layer	1	1
Trainable parameters	119,521	78,401
Learning rate	0.001	0.001
Epochs	20	11
Batch size	256	64

5.3. Model Performance

The model performance is summarized in figure 5.2. Figure 5.2a and 5.2b depict the Receiver Operator Characteristics (ROC) curves and accompanying area's (AUC) of the complete dataset (left panel) and the subset of matured loans (right panel). The validation AUC

[1] One complete backpropagation.

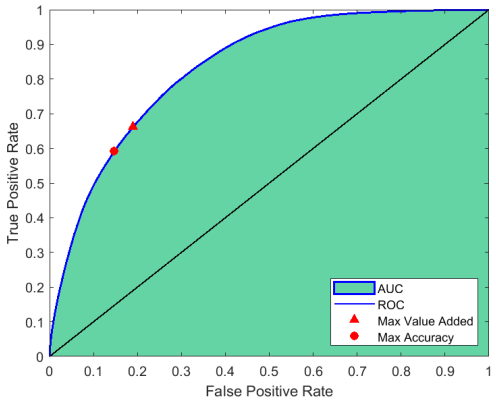
of the complete dataset is 0.8284 and that of the subset of matured loans is 0.800. Compared to other researches, this seems decent. Baesens, Van Gestel, Viaene, Stepanova, Suykens and Vanthienen (2003) summarize credit scoring results over nine different datasets and 17 different techniques, including neural networks, and find AUC's ranging from 0.50 to 0.94. The neural network had the highest AUC in five of these databases with an average of 0.77. Brown and Mues (2012) review six different databases. These researchers also look at neural networks in a subset of the data with a default rate of 30% and find an average AUC of 0.79. Djeundje, Crook, Calabrese and Hamid (2021) use data from a mexian and nigerian bank and find AUC's ranging from 0.53 to 0.63

As mentioned in section 2.1, the variation in scoring metrics across datasets is generally larger than across different techniques. So, ideally, we should compare our results to researchers that use the Bondora dataset. Byanjankar, Heikkilä and Mezei (2015) use the Bondora dataset but, unfortunately, do not report the AUC. The researchers do report a validation accuracy of 64.51%, this research has a maximum validation accuracy of 75.05% for the complete dataset and 74.22% for the subset of matured loans. The accuracy of 64.51% seems low but the default rate in their dataset is 56.97% which deflates accuracy statistics relative to our datasets which have a higher default rate. That is one reason why AUC is more appropriate[2], however, as mentioned, the AUC is not reported.

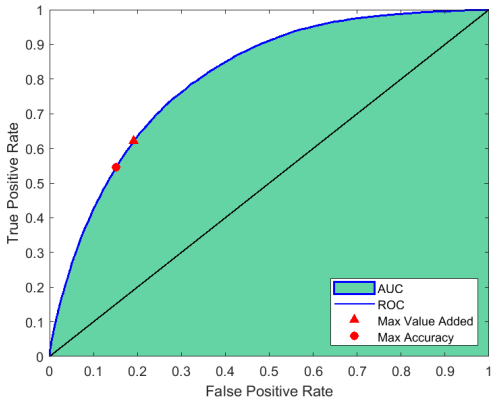
Figure 5.2a and 5.2b also depict the confusion matrices that yield the highest accuracy (red circle)[3] and the highest value added (red triangle). These two points are quite close to each other which is due to the estimation of the revenue generated by true negatives, which is €736.56, and the average cost of false negatives, which is €965.99. If the costs are underestimated the optimal decision boundary will shift to the left due to the higher actual cost of false negatives. As a result of this leftward shift more loans will be classified as default. Consequently the false positive rate and true positive rate would both increase and the triangles in figure 5.2a and 5.2b would shift to the right. This would also happen if the

[2]The AUC is much less sensitive to the underlying distribution of binary classes compared to plain accuracy metrics. However, it is not completely insensitive to the distribution. For example see Brown and Mues (2012, p. 3451).

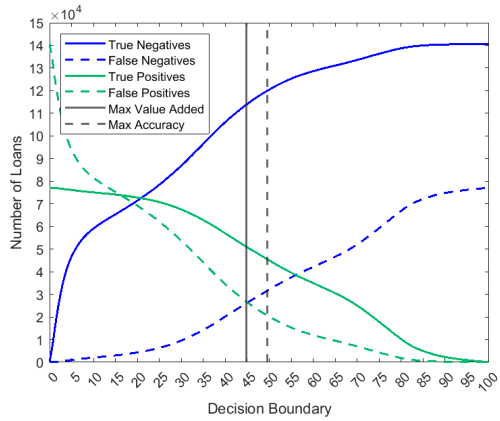
[3]Accuracy is simply calculated as $\frac{TP+TN}{TP+FP+TN+FN}$.



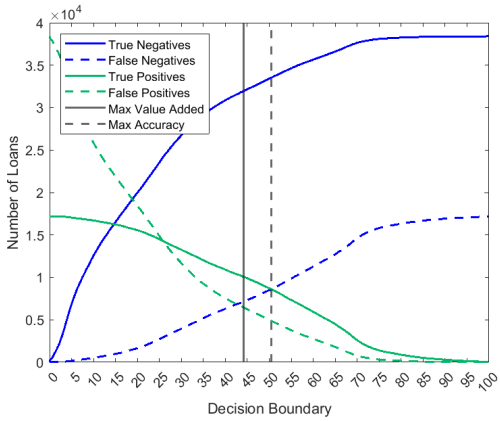
(a) AUC and ROC of predictions on the complete dataset (default$_1$).



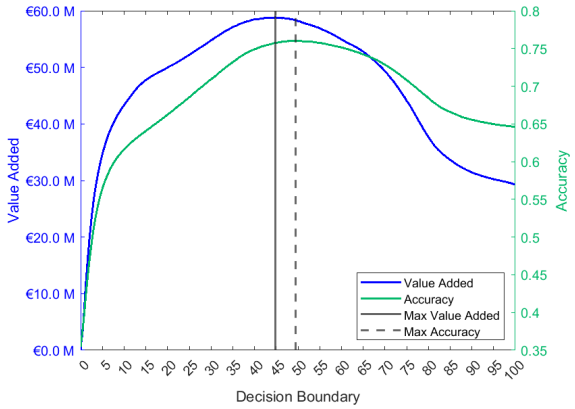
(b) AUC and ROC of predictions on a subset of the data (default$_2$).



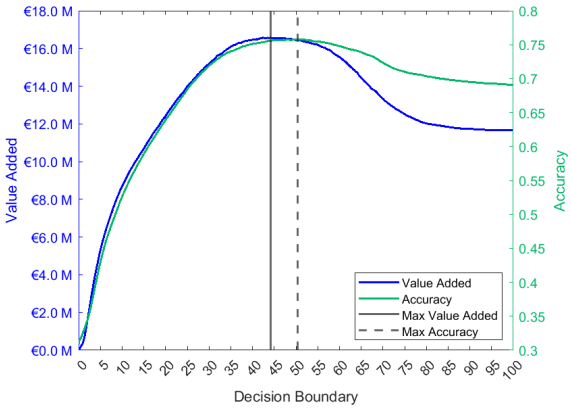
(c) Developments of the True Negatives, False Negatives, True Positives and False Positives (y-axis) as the Decision Boundary is shifted (x-axis), based on predictions of the complete dataset (default$_1$).



(d) Developments of the True Negatives, False Negatives, True Positives and False Positives (y-axis) as the Decision Boundary is shifted (x-axis), based on predictions of a subset of the data (default$_2$).



(e) The Value Added (left y-axis) and the accuracy (right y-axis) as the Decision Boundary (x-axis) is shifted, based on predictions of the complete dataset (default$_1$).



(f) The Value Added (left y-axis) and the accuracy (right y-axis) as the Decision Boundary (x-axis) is shifted, based on predictions of a subset of the data (default$_2$).

Figure 5.2: Model performance graphs, the left panels depict the model based on default$_1$, the right panels depict the model based on default$_2$.

revenue of true negatives is overestimated because then false negatives become relatively more costly.

The development of true negatives, false negatives, true positives and false positives are depicted in figure 5.2b (complete dataset) and 5.2c (subset of matured loans). As can be seen, when the decision boundary is zero, all loans will be classified as default. The true positives equal the total amount of defaults (77,099 left panel, 17,210 right panel) and the false positives equal all non-defaults (140,593 left panel, 38,408 right panel). The true and false negatives are obviously zero. When the decision boundary is shifted to the right the amount of false positives decreases steeply and the amount of true negatives increase steeply, in the complete dataset more steeply than in the subset of matured loans. The true positives decrease gradually and the false negatives increase gradually. Again in the complete dataset more gradually than in the subset of matured loans. The optimal accuracy is always around the decision boundary of 50, at this point the true positives and negatives relative to the false positives and negatives are maximized. When the decision boundary is 100 all loans are classified as non-defaults and the true negatives equal non-defaults and the false negatives equal defaults, positives are zero.

Figure 5.2e and 5.2f depict the development of the value added (left y-axis) and the accuracy (right y-axis) as the decision boundary (x-axis) is shifted. As can be seen, when all loans are classified as default (decision boundary is 0) the value added is zero. The accuracy equals the default rate. Now we shift the decision boundary to the right and we start to make a profit on the loans. At a decision boundary of roughly 45 this profit is maximized, beyond this point the true negatives do not weigh up to the costs of the additional false negatives (see figure 5.2c and 5.2d) and the value added becomes suboptimal.

When the decision boundary is 100 the accuracy is the non-default rate and the value added is the actual profit of bondora. Remember that all loans in the dataset were granted. Meaning that in reality the decision boundary was at 100. When one accepts the estimations at face value it can be claimed that the credit scoring methodology of this research could add additional value up and above Bondora's current profit. This value is realized by not providing loans to any applicants that score above the decision boundary of 45. This would eliminate 80,017 loans (about 36,76%) in the case of the complete dataset and 16,705

loans (about 30.04%) in the subset of matured loans.

The above estimations are based on predictions of the complete datasets, not just the test set. We have optimized the dataset based on the test set but this does not mean that the model is not overfitting on the training data. As a consequence the value added depicted in figure 5.2e and 5.2f might be misleading. However, when value added is calculated on the test data the same pattern emerges with the decision boundary in roughly the same spot. The fact that there is virtually no difference between the optimal decision boundary, for the value added, between test and training data is evidence that the model is not overfitting. This is confirmed by the small difference between the training and validation AUC (see table 5.2).

5.4. Comparison to Other Techniques

Table 5.2 summarizes the results of the comparison to other techniques. The neural network has the highest validation AUC for the complete dataset ($default_1$). Random forest has the highest validation accuracy. Both methods score roughly equal on value added[4]. The difference between train AUC and test AUC is higher in the random forest model which shows that it overfits slightly more.

Note that overfitting in decision tree algorithms is related to the maximum allowed leaf nodes and does not necessarily come at a cost in terms of validation AUC. The reason is that overfitting happens in the lowest part of the decision tree which can memorize all input vectors. Out-of-sample input vectors rarely make it that far down the tree. However, this redundant part should always be removed as it can lead to misclassification in rare cases and adds nothing to the validation. Neural nets are parametric models, where the parameters change when the model starts to memorize input vectors. As a result, test AUC diminishes as the train AUC surpasses the test AUC i.e., when the model starts to overfit.

In the complete dataset, the support vector machine scores lowest on AUC. This is because the AUC is not a valid metric for non-probabilistic models where the decision boundary is defined by the model itself. The support vector machine can best be compared on

[4]Note that the value added in table 5.2 reflects the value added based on the test data, which is only 20% of the complete dataset which is depicted in figures 5.2e and 5.2f.

Table 5.2: Comparison to Other Techniques

	$default_1$				$default_2$			
	Train AUC	Test AUC	VA	Accuracy	Train AUC	Test AUC	VA	Accuracy
NN	0.8331	0.8284	11.4	0.7505	0.8117	0.8000	3.2	0.7422
LR	0.7645	0.7631	10.1	0.7223	0.7581	0.7531	3.0	0.7261
DT	0.8191	0.8021	10.8	0.7357	0.8051	0.7678	3.1	0.7330
RF	0.8437	0.8203	11.4	0.7532	0.8777	0.8026	3.3	0.7567
KNN	0.8119	0.7857	9.2	0.7201	0.8179	0.7700	2.4	0.7170
SVM	0.6980	0.6960	11.0	0.7491	0.6956	0.6656	3.2	0.7433

Note: NN = neural network, LR = logistic regression, DT = decision tree, RF = random forest, KNN = k-nearest neighbors, SVM = support vector machine, VA = value added (in millions, in euro).

the basis of its accuracy and scores reasonably well, outperforming k-nearest neighbors, logistic regression and the decision tree classifier[5]. Ignoring the support vector machine validation AUC, logistic regression scores lowest followed by k-nearest neighbors and decision trees.

In the subset of matured loans ($default_2$), random forest outperforms the neural network, in terms of validation AUC, value added and accuracy. The random forest model is overfitting slightly but, as mentioned, this is no matter of great consequence. The neural network comes in second on merit of validation AUC, the support vector machine comes second in terms of accuracy. The worse performing models are, again; logistic regression, decision tree, and k-nearest neighbors. In machine learning, neural networks are the most popular method for large datasets (N > 100,000). In general; the larger the dataset, the better neural nets perform relative to alternative methods. So it is not that surprising that random forest outperforms the neural network in the subset of matured loans (N = 55,618) but not in the complete dataset (N = 217,692).

[5]The decision tree of the subset of matured loans is depicted in appendix A.2.

6

Conclusion

This research proposes a deep hybrid learning approach to improve credit default risk assessment in peer-to-peer lending markets. The method consist of a two step procedure where, in the first step, redundant features are removed by recursive feature elimination. Perceptron weights are applied as a metric for feature importance. This results in the selection of 49 informative features from a total of 173. This step removes noisy features from the dataset which helps prevent the neural network, applied in the second step, from overfitting. The hyperparameters of the neural network are optimized with the Hyperband method.

Two models are defined, one for the complete dataset and one for a subset of matured loans. The validation AUC of the complete dataset is 0.8284, the subset of matured loans has a validation AUC of 0.8000. These scores look promising, although it is difficult to compare metrics across different datasets and researches. An estimation of the average cost of false negatives and the average revenue of true negatives shows that the model could potentially ad value to the current risk assessment. However, the actual added value is difficult to estimate because the costs of default are not observed in the dataset.

Compared to other methods the deep hybrid learning approach performs best in the complete dataset. However, random forest outperforms the deep hybrid learning model on the subset of matured loans. This is probably due to the smaller size of that dataset which benefits the random forest methodology as compared to the neural network. In both datasets the deep hybrid learning models outperform logistic regression, decision tree, k-nearest neighbors and the support vector machine algorithms based on the validation AUC.

In conclusion, this research shows that deep hybrid learning has the potential to improve risk assessment in peer-to-peer lending markets. Moreover, it shows that as the number of loans in the dataset increases, the relative performance of the neural network improves. Taking into consideration the rapidly growing peer-to-peer lending market, it can be expected that the gain from the suggested method will increase over time. The model can possibly help mitigate the expected adverse effects of the growth in this type of lending. In particular, the improved risks assessment may prevent peer-to-peer markets from being used as a vehicle for offloading risk to counterparties.

Regulatory arbitrage is often mentioned as one of the driving forces behind the popularity of FinTech developments such as peer-to-peer lending. The possibility to circumvent credit regulation poses a threat to economic stability. On the flip side, the digitization of financial markets creates large information flows (big data) which can be used to improve risk assessment as compared to assessment in a centralized economy. Deep hybrid learning helps filter the information and yield superior default risk forecasts. Mitigating the information asymmetries in decentralized financial markets can ultimately increase economic stability and decrease systemic risk.

A

Appendix

A.1. Data

Table A.1: Feature Description

Feature	Pre-process Type	Description
Bids Portfolio Manager	continuous	The amount of investment offers made by portfolio managers.
Bids Api	continuous	The amount of investment offers made via Api.
Bids Manual	continuous	The amount of investment offers made manually.
New Credit Customer	dummy	Zero if the customer had at least three months of credit history in Bondora, one otherwise.
Application Signed hour	nominal	N/A
Application Signed Weekday	nominal	N/A
Monthly Payment Day	nominal	The day of the month on which the one payment is scheduled. Weekend days are shifted to monday.
Language Code	nominal	1: Estonian, 2: English, 3: Russian, 4: Finnish, 5: German, 6: Spanish, 9: Slovakian.
Age	continuous	Age of the borrower.
Gender	nominal	0: male, 1: female, 2: other.
Country	nominal	1: Estonia, 2: Finland, 3: Spain 4: Slovakia. Residence of borrower.

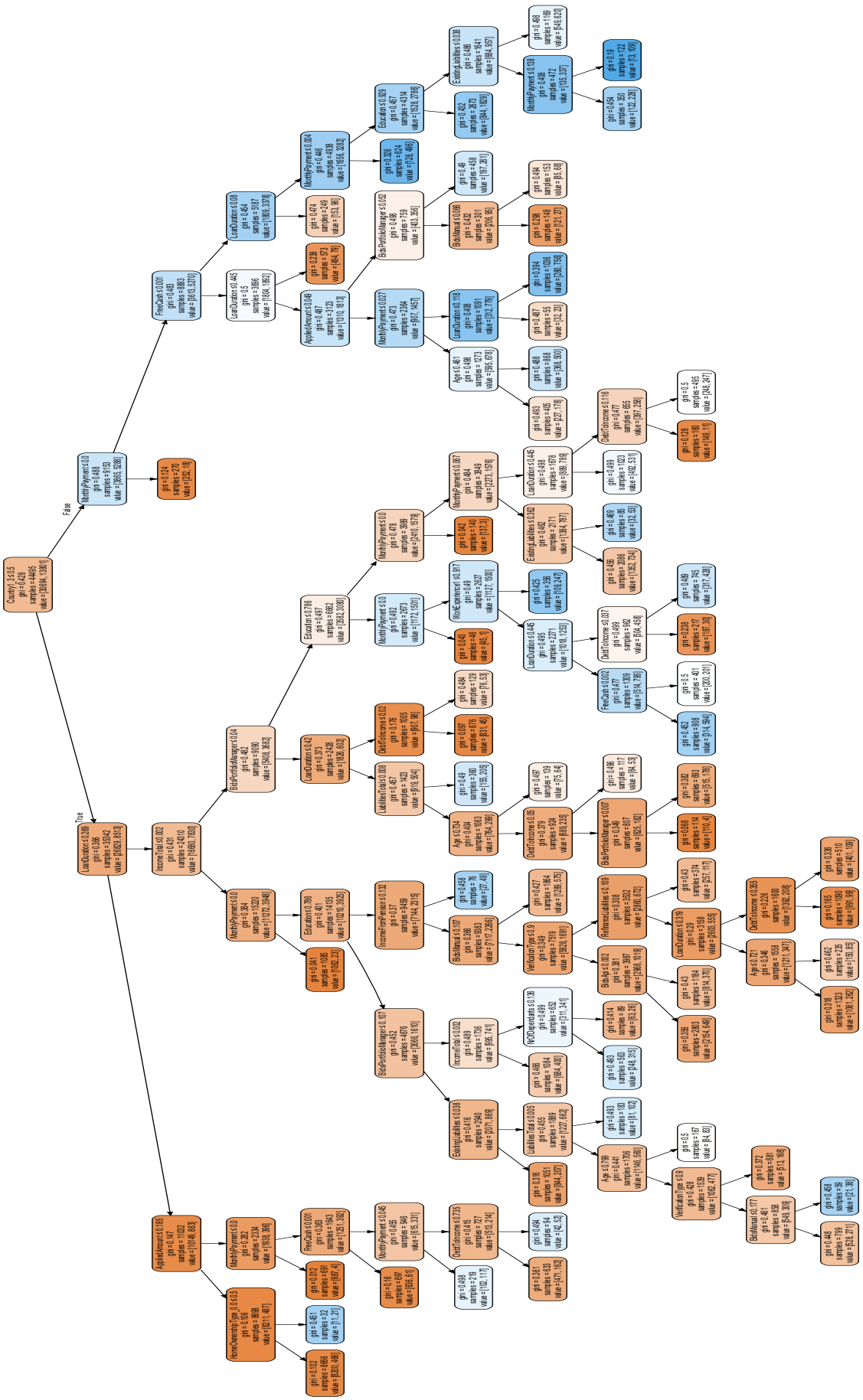
Education	ordinal	1: primary education, 2: basic education, 3: vocational education, 4: secondary education, 5: Higher Education.
Marital Status	nominal	1: married, 2: cohabitant, 3: single, 4: divorced, 5: widow.
Number of Dependants	continues	Number of children or other dependants.
Home Ownership Type	nominal	0: homeless, 1: owner, 2: living with parents, 3: tenant, pre-furnished property, 4: tenant, unfurnished property, 5: council house, 6: joint tenant, 7: joint ownership, 8: mortgage, 9: owner with encumbrance, 10: other.
Applied Amount	continuous	Originally applied amount.
Loan Duration	continuous	Loan duration in months.
Monthly Payment	nominal	Estimated amount the borrower has to pay every month.
Employment Status	nominal	1: unemployed, 2: partially employed, 3: fully employed, 4: self-employed, 5: entrepreneur 6: retiree.
Employment Duration Current Employer	ordinal	0: other, 1: trial period, 2: up to 1 year, 3: up to 2 years, 4: up to 3 years, 5: up to 4 years, 6: up to 5 years, 7: more than 5 years, 8: retiree.
Work Experience	ordinal	0: less than 2 years, 1: 2-5 years, 2: 5-10 years, 3: 10-15 years, 4: 15-25 years, 5: more than 25 years.
Occupation Area	nominal	1: other, 2: mining, 3: processing, 4: energy, 5: utilities, 6: construction, 7: retail, 8: transport and warehousing, 9: hospitality and catering, 10: info and telecom, 11: finance and insurance, 12: real-estate, 13: research, 14: administrative, 15: civil service and military, 16: education, 17: healthcare and social help, 18: art and entertainment, 19: agriculture, forestry and fishing.
Income From Principal Employer	continuous	N/A
Income From Pension	continuous	N/A
Income From Family Allowance	continuous	Income from child support.
Income From Social Welfare	continuous	N/A

Income From Leave Pay	continuous	Income from paternity leave.
Income From Child Support	continuous	Income from alimony payments.
Income Other	continuous	N/A
Income Total	continuous	N/A
Debt to Income	continuous	Ratio of the borrowers monthly gross income that goes towards paying loans.
Free Cash	continuous	Discretionary income after monthly liabilities.
Verification Type	ordinal	Method used for loan application data. 0: not set, 1: income unverified, 2: income unverified, cross-referenced by phone, 3: income verified, 4: income and expenses verified.
Use of Loan	nominal	0: loan consolidation, 1: real estate, 2: home improvement, 3: business, 4: education, 5: travel, 6: vehicle, 7: other, 8: health, 101: working capital financing, 102: purchase of machinery equipment, 103: renovation of real estate, 104: accounts receivable financing, 105: acquisition of means of transport, 106: construction finance, 107: acquisition of stocks 108: acquisition of real estate, 109: guaranteeing obligation, 110: other business. All code in format 1xx are for business loans that are not supported since oct 2012.
Existing Liabilities	continuous	Number of existing liabilities.
Liabilities Total	continuous	Total monthly liabilities.
Refinance Liabilities	continuous	Total amount of liabilities after refinancing.

A.2. Results

The figure on the following page depicts the decision tree of the subset of matured loans. The colors indicate targets, red is non-default, blue is default. The color intensity reflects the probability. Each node contains a threshold feature value, a Gini impurity metric and the number of defaults and non-defaults in that node.

The optimal number of leaf nodes in the depicted tree is 60, this is a relatively small tree. The optimal number of leaf nodes for the complete dataset is 160 which increases the size of the tree substantially. The trees in the random forest model have an optimal amount of leaf nodes of 500 and are even larger still.



Bibliography

Abdou, H. A., & Pointon, J. (2011). Credit scoring, statistical techniques and evaluation criteria: A review of the literature. *Intelligent systems in accounting, finance and management, 18*(2), 59–88.

Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., & Vanthienen, J. (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the operational research society, 54*(6), 627–635.

Balyuk, T., & Davydenko, S. (2019). Peer-to-peer lenders versus banks: Substitutes or complements? *Michael J. Brennan Irish Finance Working Paper Series Research Paper, 18*.

Basha, S. A., Elgammal, M. M., & Abuzayed, B. M. (2021). Online peer-to-peer lending: A review of the literature. *Electronic Commerce Research and Applications, 48*(1), 463–482.

Basle Committee. (2000). *Principles for the management of credit risk*. Bank for International Settlements.

Belleflamme, P., Omrani, N., & Peitz, M. (2015). The economics of crowdfunding platforms. *Information Economics and Policy, 33*(1), 11–28.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory*, 144–152.

Boyle, M., Crook, J. N., Hamilton, R., & Thomas, L. C. (1992). Methods for credit scoring applied to slow payers. In L. C. Thomas, J. N. Crook, & D. B. Edelman (Eds.), *Credit scoring and credit control* (pp. 75–90). Oxford University Press.

Braggion, F., Manconi, A., & Zhu, H. (2018). Can technology undermine macroprudential regulation? evidence from peer-to-peer credit in china.

Bre, F., Gimenez, J. M., & Fachinotti, V. D. (2018). Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, *158*, 1429–1441.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–3.

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, *39*(36), 3446–3453.

Buchak, G., Matvos, G., Piskorski, T., & Seru, A. (2018). Fintech, regulatory arbitrage, and the rise of shadow banks. *Journal of financial economics*, *130*(3), 453–483.

Byanjankar, A., Heikkilä, M., & Mezei, J. (2015). Predicting credit risk in peer-to-peer lending: A neural network approach. *2015 IEEE symposium series on computational intelligence*, 719–725.

Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, *8*(1), 1–12.

Chollet, F. (2021). *Deep learning with python*. Simon & Schuster.

DeLong, E. R., DeLong, D. M., & Clarke-Pearson, D. L. (1988). Comparing the areas under two or more correlated receiver operating characteristic curves: A nonparametric approach. *Biometrics*, *44*(1), 837–845.

Djeundje, V. B., Crook, J., Calabrese, R., & Hamid, M. (2021). Enhancing credit scoring with alternative data. *Expert Systems with Applications*, *163*(1).

Durand, D. (1941). Risk elements in consumer installment financing. *National Bureau of Economic Research*.

Dutta, S., & Shekhar, S. (1988). Bond rating: A non-conservative application of neural networks. *IEEE Int Conf on Neural Network*, 443–450.

Ederington, L. H. (1985). Classification models and bond ratings. *Financial review*, *20*(4), 237–262.

European Commission. (2021). Proposal for a directive of the european parliament and of the council on consumer credits [https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=COM:2021:347:FIN].

Fisher, L. (1959). Determinants of risk premiums on corporate bonds. *Journal of political economy*, *67*(3), 217–237.

Fix, E., & Hodges, J. L. (1989). Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review*, *57*(3), 238–247.

Gavurova, B., Dujcak, M., Kovac, V., & Kotásková, A. (2018). Determinants of successful loan application at peer-to-peer lending market. *Economics & Sociology*, *11*(1), 85–99.

Gentry, J. A., Whitford, D. T., & Newbold, P. (1988). Predicting industrial bond ratings with a probit model and funds flow components. *Financial review*, *23*(3), 269–286.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 315–323.

Goldstein, I., Jiang, W., & Karolyi, G. A. (2019). To fintech and beyond. *The Review of Financial Studies*, *32*(5), 1647–1661.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 1157–1182.

Han, L., Han, L., & Zhao, H. (2013). Orthogonal support vector machine for credit scoring. *Engineering Applications of Artificial Intelligence*, *26*(2), 848–862.

Hsieh, N. C. (2005). Hybrid mining approach in the design of credit scoring models. *Expert Systems with Applications*, *28*(4), 655–665.

Huang, Z., Chen, H., Hsu, C. J., Chen, W. H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: A market comparative study. *Decision support systems*, *37*(4), 543–558.

Ivakhnenko, A. G., & Lapa, V. G. (1965). *Cybernetic predicting devices*. CCM Information Corporation.

Käfer, B. (2018). Peer-to-peer lending–a (financial stability) risk perspective. *Review of Economics*, *69*(1), 1–25.

Khandani, A. E., Kim, A. J., & Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, *34*(11), 2767–2787.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, *97*(2), 273–324.

Kohavi, R., & Wolpert, D. H. (1996). Bias plus variance decomposition for zero-one loss functions. *ICML*, *96*(1), 275–283.

Kruppa, J., Schwarz, A., Arminger, G., & Ziegler, A. (2013). Consumer credit risk: Individual probability estimates using machine learning. *Expert Systems with Applications*, *40*(13), 5125–5131.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2017). Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, *18*(1), 6765–6816.

Little, R. J., & Rubin, D. B. (2019). *Statistical analysis with missing data*. John Wiley & Sons.

Magnuson, W. (2018). Regulating fintech. *Vand. L. Rev.*, *71*(1), 1167.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Morse, A. (2015). Peer-to-peer crowdfunding: Information and the potential for disruption in consumer lending. *Annual Review of Financial Economics*, *7*(1), 463–482.

Myers, J. H., & Forgy, E. W. (1963). The development of numerical credit evaluation systems. *Journal of the American Statistical association*, *58*(303), 799–806.

Namvar, E. (2014). An introduction to peer-to-peer loans as investments. *Journal of Investment Management First Quarter*.

Orgler, Y. E. (1970). A credit scoring model for commercial loans. *Journal of money, Credit and Banking*, *2*(4), 435–445.

Pinches, G. E., & Mingo, K. A. (1973). A multivariate analysis of industrial bond ratings. *The journal of Finance*, *28*(1), 1–18.

Plantin, G. (2015). Shadow banking and bank capital regulation. *The Review of Financial Studies*, *28*(1), 146–175.

Pogue, T. F., & Soldofsky, R. M. (1969). What's in a bond rating? *Journal of Financial and Quantitative Analysis*, *4*(2), 201–228.

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. *arXiv preprint*, *1710.05941*.

Rosenblatt, F. (1957). The perceptron. a perceiving and recognizing automation. *Cornell Aeronautical Laboratory Report*, 85–460.

Rothemund, M., & Gerhardt, M. (2011). *The european credit information landscape. an analysis of a survey of credit bureaus in europe*. ECRI Research Report.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533–536.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, *61*, 85–117.

Steenackers, A., & Goovaerts, M. (1989). A credit scoring model for personal loans. *Insurance: mathematics & economics*, *8*(1), 31–34.

Tang, H. (2019). Peer-to-peer lenders versus banks: Substitutes or complements? *The Review of Financial Studies*, *32*(5), 1900–1938.

Tarullo, D. K. (2019). Financial regulation: Still unsettled a decade after the crisis. *Journal of Economic Perspectives*, *33*(1), 61–80.

Thomas, L. C. (2000). A survey of credit and behavioural scoring: Forecasting financial risk of lending to consumers. *International journal of forecasting*, *16*(2), 149–172.

Tripathi, D., Shukla, A. K., Reddy, B. R., Bopche, G. S., & Chandramohan, D. (2022). Credit scoring models using ensemble learning and classification approaches: A comprehensive survey. *Wireless Personal Communications*, *123*, 785–812.

Tsai, C. F., & Chen, M. L. (2010). Credit rating by hybrid machine learning techniques. *Applied soft computing*, *10*(2), 374–380.

Tsai, M. C., Lin, S. P., Cheng, C. C., & Lin, Y. P. (2009). The consumer loan default predicting model–an application of dea–da and neural network. *Expert Systems with applications*, *36*(9), 11682–11690.

Wang, H., Xu, Q., & Zhou, L. (2015). Large unbalanced credit scoring using lasso-logistic regression ensemble. *PloS one*, *10*(2).

West, D. (2000). Neural network credit scoring models. *Computers & operations research*, *27*(11), 1131–1152.

Wilmarth, A. E. (2010). The dodd-frank act: A flawed and inadequate response to the too-big-to-fail problem. *Or. L. Rev.*, *89*(1), 951.

Wooldridge, J. M. (2018). *Introductory econometrics: A modern approach*. Cengage Learning.

Xia, Y., Liu, C., Li, Y., & Liu, N. (2017). A boosted decision tree approach using bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, *78*(1), 225–241.

Zhu, B., Yang, W., Wang, H., & Yuan, Y. (2018). Hybrid deep learning model for consumer credit scoring. *international conference on artificial intelligence and big data (ICAIBD)*, 205–208.