**Vysoká škola ekonomická v Praze**

Fakulta financí a účetnictví

katedra bankovnictví a pojišťovnictví

Finanční inženýrství

**DIPLOMOVá práce**

2024 Stevan Vujčić

**Vysoká škola ekonomická v Praze**

Fakulta financí a účetnictví

katedra bankovnictví a pojišťovnictví

Finanční inženýrství

**Komparativní analýza modelů pro predikci pravděpodobnosti defaultu**

Autor diplomové práce: Bc. Stevan Vujčić

Vedoucí diplomové práce: Ing. Milan Fičura, Ph.D.

Rok obhajoby: 2024

**Prague University of Economics and Business**

Faculty of Finance and Accounting

Department of Banking and Insurance

Financial Engineering

**Comparative Analysis of Probability of Default Models**

Master’s thesis author: Bc. Stevan Vujčić

Master’s thesis supervisor: Ing. Milan Fičura, Ph.D.

Year of defence: 2024

Čestné prohlášení – Declaration of Authorship

Prohlašuji, že jsem diplomovou práci na téma „Komparativní analýza modelů pro predikci pravděpodobnosti defaultu“ vypracoval samostatně a veškerou použitou literaturu a další prameny jsem řádně označil a uvedl v přiloženém seznamu.

I hereby declare that I wrote the Master’s thesis “Comparative Analysis of Probability of Default Models” independently and that I adequately referenced all the used literature and other sources listed in references.

V Praze dne 26. 5. 2024 – 26th May, 2024, Prague

.................................................................................

Bc. Stevan Vujčić

Abstrakt

Série standardně používaných modelů strojového učení je odhadnuta na behaviorálních datech o úvěrech na bydlení a hypotéky významné české bankovní instituce. Za účelem ověření robustnosti výsledků jsou modely odhadnuty za různých přístupů přeprocesování dat. Uvažuje se několik kombinací transformací vysvětlujících proměnných, balancování poměru tříd závislé proměnné, odstranění multikolinearity a selekce vysvětlujících proměnných. Zjištěním je, že v případě portfolia úvěrů na bydlení a hypotéky není klasicky používaná logistická regrese nijak překonána modernějšími metodami, jako jsou neuronové sítě či boosting.

Klíčová slova

Kreditní riziko · Strojové učení · Pravděpodobnost defaultu

Abstract

A series of common machine learning models is estimated using a behavioral home loans data of a major Czech banking institution. To address robustness, the algorithms are estimated using different approaches to data preprocessing. Namely, several takes at feature transformations, class balancing, multicollinearity removal and feature selection are accounted for. It is found that in the case of the Czech home loans data the classical logistic regression approach is not outperformed by the more modern methods such as artificial neural networks or boosting.

Key words

Credit Risk · Machine Learning · Probability of Default

Table of Contents

[Introduction 7](#_Toc167570541)

[1 Institutional Framework and Current Status 8](#_Toc167570542)

[1.1 Regulation 8](#_Toc167570543)

[1.1.1 EBA Note 8](#_Toc167570544)

[1.2 Literature Review 13](#_Toc167570545)

[2 Overview of Modeling Methodology 17](#_Toc167570546)

[2.1 Model Estimation 17](#_Toc167570547)

[2.1.1 Logistic regression 17](#_Toc167570548)

[2.1.2 Neural Network 20](#_Toc167570549)

[2.1.3 Support Vector Machines 23](#_Toc167570550)

[2.1.4 Tree-based Methods 25](#_Toc167570551)

[2.1.5 KNN 29](#_Toc167570552)

[2.2 Model Testing 29](#_Toc167570553)

[2.2.1 Accuracy Ratio and CAP 29](#_Toc167570554)

[2.2.2 AUC and ROC 31](#_Toc167570555)

[2.2.3 Rates of Error 32](#_Toc167570556)

[2.3 Preprocessing 32](#_Toc167570557)

[2.3.1 Data Transformations 32](#_Toc167570558)

[2.3.2 Feature selection criteria 34](#_Toc167570559)

[2.3.3 Class imbalance and Reject Inference 35](#_Toc167570560)

[3 Application to Czech Home Loans Portfolio 37](#_Toc167570561)

[3.1 Data 37](#_Toc167570562)

[3.2 Programming Solution 38](#_Toc167570563)

[3.2.1 Data Ingestion 38](#_Toc167570564)

[3.2.2 Preprocessing: transformations and shortlisting 39](#_Toc167570565)

[3.2.3 Modeling 39](#_Toc167570566)

[3.2.4 Validation 41](#_Toc167570567)

[3.3 Application 41](#_Toc167570568)

[3.3.1 Researched Questions 41](#_Toc167570569)

[3.3.2 Estimated Models 41](#_Toc167570570)

[3.3.3 Results 45](#_Toc167570571)

[3.3.4 Comparison & Answer to Researched Questions 54](#_Toc167570572)

[Final Remarks 57](#_Toc167570573)

[References 58](#_Toc167570574)

[Graphics and Tables 60](#_Toc167570575)

# Introduction

As a result of the ever-growing array of machine learning algorithms that can be used to address credit risk management, discussions around the necessity, appropriateness and legal compliance of these models unfolded during this century.

Models such as artificial neural networks are a part of the broader artificial intelligence (AI) picture. Recent developments in this field such as generative AI accelerated discussions about regulation. Under this wide umbrella of discussion, creditworthiness assessment in fact finds a significant spot due to its potential impact on the life of a person. At the forefront of this discussion is the European Union whose Council passed an *AI Act 2024* in May 2024 (Council of the EU, 2024). As for institutions practicing credit risk management, they have to ensure that they are compliant with the stipulations of this Act when using neural networks and similar solutions.

To answer whether artificial neural networks, tree methods and such newly used models bring added value, financial institutions and academic researchers delved into the topic. It came out that determining whether the currently new methods are better is not a simple exercise. This depends on the quantity of data at disposal, whether there are non-linearities to be examined, how the data is being preprocessed or whether it is transformed at all, and many more questions.

In the first section of this thesis, discussed is the European Banking Authority-led communication with financial institutions and their representations in the context of machine learning modeling. A reflection on the AI Act is made as well. Given the sheer size of literature, only several publications were carefully picked and used as a reference throughout the text. In the second section, a turn is taken at trying to explain modeling approaches. In the third section, a comparative study of the performance of different machine learning models for the probability of default estimation is performed. A sample of “real-world” home loan behavioral data is obtained from a major Czech banking institution. The choice of the tested models is limited to those that are normally considered at banking institutions. Also, the modeling procedure itself when it comes to data transformations is done in such a way that it is in line with what is being done at banks.

# Institutional Framework and Current Status

In this section, the global regulatory framework by which the PD modeling practice is driven is briefly summarized. A more specific commentary is provided about the latest discussions on the use of machine learning in credit risk modeling. Finally, a literature review is provided.

## Regulation

Under the Internal ratings-based (IRB) approach banks are able to create their own estimations of the PD. This comes as a contrast to the Standardized approach (SA) where such possibility does not exist. On top of the Basel Framework (described by e.g. Barfield et al. (2011)) that provides a general set of rules on risk management, the EBA modeling guidelines (EBA, 2017) provide more specific instructions on how the modeling part is to be performed.

### EBA Note

In August 2023, EBA published EBA/REP/2923/28 – “Follow-up report from the consultation on the discussion paper on machine learning for IRB models” (EBA, 2023). The document addresses the use of ML techniques by the industry, their complexity and their interaction with the *GDPR Act 2016* (EU) and the EU *AI Act 2024* (Council of the EU, 2024)[[1]](#footnote-1). The ML techniques are understood as “*models characterized by a high number of parameters, that require a large volume of (potentially unstructured) data for their estimation and that are able to reflect non-linear relations between the variables.*” (EBA, 2023, p. 5). When the term is used in this chapter, the same understanding is meant. The note itself is a reflection to a 2021 questionnaire EBA/DP/2021/04 issued by EBA (EBA, 2021). It contained 17 questions and 14 respondents answered them. The EBA notes that 8 of the answers were provided by associations. It remains unclear whether these eight answers provided by associations represent a combined view of multiple financial institutions that are members of these associations. In the rest of this chapter, individual sections of the analyzed note are being paraphrased and reviewed. The scope of the review is limited to uses and formulations relevant for PD modeling under IRB.

#### Use of ML in IRB

The note addresses the following use areas:

1. Firstly, the terms risk differentiation and risk quantification as outlined in the EBA modeling guidelines (EBA, 2017) are pinpointed. The former refers to identifying the risk drivers that have satisfactory discriminatory power. The latter refers to calibrating the risk parameters in such a way that they reflect the long-run averages. It is reported that financial institutions use ML techniques for risk differentiation but not for calibration. The latter is namely not possible due to shorter time series of new data sources. This might imply that when ML techniques are used for modeling, a larger scale of features is considered, i.e. it is not just the estimation algorithm that changes.
2. It is noted that ML techniques are not always outright used for model estimation. They may be used for some supportive and model-enhancing operations such as text mining, missing data imputation and tackling of unstructured data.
3. It is noted that the added value of ML is especially noted in cases when risk differentiation was not successful. ML techniques are able to exploit non-linearities and thus find relationships that cannot be easily identified by more traditional estimation approaches.
4. It is noted that challenger models are often ML-based, whilst the tested models can remain traditional. In this way, the end model that is going to be used is still the interpretable and fully IRB-compliant traditional model. Nonetheless, using ML as the benchmark will always provide input into either how better the tested model might be or whether it should be replaced by an ML model. It is, however, also mentioned that challenging a traditional model with an ML model is not so straightforward. For example, if a validator wants to test the significance of individual features, this is easily done in a logistic regression-based model. The contribution of individual features in the case of a neural network is, however, much harder to pinpoint.

#### Complexity of ML

The note documents that the industry practitioners reported issues in the areas of statistical inference, know-how and model interpretation.

1. The industry respondents reported overfitting of the training sample as the main issue of using ML techniques. One case is highlighted by which this is more pronounced in the case of low default portfolios. It is not, however, clear to what extent the low-default portfolio characteristic causes issues. For example, both home loans and municipal portfolios are low default portfolios in relative terms. The difference is although in the sizes of these 2 portfolios. Whilst there are only several hundred municipalities per mid-sized country and defaults are quite rare, there are hundreds of thousands of observations on home loan portfolios.
2. It is reported that one respondent had difficulties implementing standardized solutions to model development and model validation when ML techniques are used. The model development procedure should generally be streamlined and algorithmic. In the case of neural networks and other ML techniques, this is all that more important due to the variety of options to tune hyperparameters. Although an initial validation might require case-by-case individual tests, annual validations can be standardized with relative ease and ML implementations should not pose an issue. It is therefore concluded that the fact that only one respondent reported this as an issue is rather an outlier.
3. Financial institutions responded that additional know-how must be acquired in order to enable the usage of ML models. This implies that modeling departments in individual banks do not necessarily have an established group of experts in the area. In addition, once ML expertise is acquired in the form of new hires, it takes time to get them aligned with the understanding of the credit risk management ecosystem.
4. It is reported that the explainability of ML models is challenging. In its note, EBA (2023) collected responses about what banking practitioners use to explain an ML model:
   1. Shapley values (40%),
   2. model documentation (28%),
   3. graphical tools (20%), and
   4. sensitivity analysis (8%).

Although not explicitly said in the note, here it is understood that, for example, 40% of respondents use Shapley values. Conversely, 60% do not or they did not report it. It is also understood that the respondents use more than one metric outlined above.

1. EBA notes that respondents reported traceability as an issue. Nonetheless, a specific commentary on how financial institutions approach this is not provided.

#### Tracking Model Changes

The view of EBA (2023) is that the model change framework applies to ML models as well. In particular, they repeat that a model change that results in a significant change in the rank ordering is a material model change. In addition, it is highlighted that a single material model change should not be artificially split into several immaterial model changes. The last point warrants a discussion about whether the current understanding of what makes a single model change fits into the ML workflow. It is noted that financial institutions require further clarification.

#### GDPR and AI Acts

EBA (2023) notes that the respondents did not provide much specific commentary w.r.t. IRB-GDPR interaction. In the text it is therefore stated that *GDPR Act 2016* (EU) explicitly prohibits the use of certain kinds of personal data for creditworthiness estimation. It is also prohibited to track information from social networks and use it as modeling input. Finally, the so-called *minimization principle* is to be followed. They cite the understanding by *GDPR Act 2016* (EU) as “*adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed.*” (EBA, 2023, p. 14). This means that banking institutions have to, if already not, tighten their management of such sensitive data. For example, sensitive information about retail portfolio individuals cannot be easily accessible by employees. Rather, a time-limited request for access can be reviewed upon sufficient use case argumentation and granted for example for 24h.

At the time of writing of this thesis the *AI Act 2024* was approved, but its final formulation was not yet available on the official pages. Prior to its finalization and approval, it was expected that the act would define the creditworthiness assessment as a “high-risk” system. More specifically because it determines a “*persons’ access to financial resources*” (EBA, 2023, p. 15). The act was expected to be in power for AI models that underwent a substantial change or were newly designed after a set date of the act stepping in power. In effect this means that if an AI IRB model was designed historically, it would not be subject to the *AI Act 2024* until it has a significant model change. However, it is not clear whether the *material model change* will meet the definition of a *significant model change* once the act is in place. EBA (2023) argues that a distinction between creditworthiness assessment for the purpose of loan granting should be decoupled from the same kind of assessment for the purpose of capital calculation. They argue that the latter does not have an impact on natural persons.

#### EBA Recommendations

EBA (2023) defines the following principle-based recommendations:

1. “*All relevant should have an appropriate level of knowledge of the model’s functioning*”. Some of these stakeholders are: model developers & model owner, model validators, CRO and business representatives.
2. “*Avoid unnecessary complexity*”. Financial institutions should avoid introducing an excessive number of variables into their models. It is not explained what *excessive* is, i.e. how many variables this could be. It is further recommended that structured data should be preferred to unstructured data, when possible. Finally, it is argued that a simpler model should be chosen over a more complex one in the case that the performance is similar. This point can be addressed through automated model validation. For example, if a neural network is tested against an automated challenger logistic regression-based model and the performance difference is negligible, then the neural network should be abolished or re-designed.
3. Ensure that the “*model is correctly interpreted and understood*”. Financial institutions are recommended to analyze the impact of individual features, their contribution w.r.t. other features, have an economic understanding of the relationships in the model and provide a documentation of the points mentioned in this sentence. Attention should be paid to removing bias and overfitting. Although this requirement re-iterates the general process of modeling regardless of the kind of algorithm used, it explicitly strengthens the requirement around junction points that in practice might be avoided due to their complicated tackling.
4. Human judgement should be corroborated with appropriate understanding of the ML model that is being affected by this judgement.
5. It is recommended that a close eye is kept on regular model calibrations. The rationale is that credit risk should not be subject to frequent changes in the data generating process. At times, a structural shift in some key drivers would naturally lead to the necessity of re-calibrating the model, but this is not expected often.

## Literature Review

In this section, previously published research on the comparative analysis of different estimation techniques is reviewed. Since the literature on the matter is broad, the choice of the cited literature is limited to renowned publications as well as publications that are focused on applications on mortgage portfolios. To ensure reproducibility of the literature search, the following table summarizes the search criteria and the obtained results.

Table 1. Literature search results

| **Search engine** | **Criteria** | **# of results** | **# of referenced results** | **Query link** |
| --- | --- | --- | --- | --- |
| Web of Science | 1. keywords: machine learning credit scoring; 2. filter: “4.61.1820 Credit Scoring” 3. filter: from April 2014 to April 2024 | 340 | 2 | https://www.webofscience.com/wos/woscc/summary/0cf5c555-09d9-4136-89f3-139c8f876da5-da7518dd/relevance/1 |
| Web of Science | 1. keywords: machine learning mortgage market; 2. filter: “4.61.1820 Credit Scoring” 3. filter: from April 2014 to April 2024 | 3 | 0 | https://www.webofscience.com/wos/woscc/summary/c8bf1a64-c3f2-4119-98bb-7c9991d4002c-da8dbe21/sort-group-background-citingcount/1 |
| Web of Science | 1. keywords: machine learning home loan; 2. filter: “4.61.1820 Credit Scoring” 3. filter: from April 2014 to April 2024 | 4 | 0 | https://www.webofscience.com/wos/woscc/summary/0748a1c4-abd4-4b06-b71b-6e67020c1984-da8ddb8a/sort-group-background-citingcount/1 |

Source: author’s search

The first search results in a large number of suggested publications. This is due to the fact that the used keywords can pick up on just anything that mentions ML w.r.t credit risk. Out of the four chosen articles, three represent frequently cited comparative studies of PD classification models and the last one is an extensive literature review. Finally, the search was repeated with keywords held specific to mortgage portfolios. The results were very narrow, and they did not particularly match the portfolio scope of this thesis. On top of the results obtained above, the first paper that is cited below was obtained rather randomly whilst looking for a most recent and reliable literature review kind of study.

A general view is provided by Shi et al. (2022) where they identify 76 published studies that have contributed to the field over the last years. The major findings that they report are that modern machine learning algorithms outperform the traditional statistical approaches. Secondly, they find that most papers reported ensemble methods as outperforming the machine learning algorithms. According to portfolio-type scoping, they split the reviewed studies into consumer and corporate finance portfolios. When researching the abstracts of the articles reported as consumer portfolios, it could not be confirmed that any of the studies tested models on mortgage portfolios. Rather, these would either be publicly available datasets or in several cases, undisclosed datasets procured from banks in Europe or elsewhere in the world, e.g. China.

Lessmann et al. (2015) perform extensive research testing 41 different approaches and 8 different datasets. In their literature review, they provide an overview of approximately 50 researches that were done by the time they wrote their article. They find that the datasets were on average relatively small. Many of them had around several hundred or thousand observations, usually less than 10,000. Out of the 8 datasets they use are from the retail segment, 3 exceed 10,000 observations where the largest dataset has 150,000 entries and a default rate of 6.7%. They refrain from using oversampling methods such as SMOTE as they argue that this is not of importance when algorithms are compared in relative terms. In addition, they are of the view that oversampling would mask potential sensitivities of some algorithms to class imbalance. They use 6 metrics to compare the estimated models – (1) the percentage correctly predicted (PCC) and the KS statistic for classification testing, (2) the area under the curve (AUC), the Gini index and the H-measure, and (3) the Brier score for calibration. Following the 41 approaches, they estimate and test 1,141 different models. They test a variety of individual classifiers which includes but is not limited to logistic regression, vanilla neural networks and support vector machines. They also look at classification models that stack underlying simpler models – random forests or boosted trees. Finally, they look at heterogenous ensemble learning. They find that various ensemble methods provide the best results. When looking at the specific individual classifiers, the best performance is found in artificial neural networks followed by the logistic regression. Interestingly, these 2 are the only individual learning models that tightly enter the top 20 (out of 41). Different bagging and boosting methods perform better than the previous 2. Nonetheless, the top spots are reserved for heterogenous ensemble learning methods, i.e. the methods that combine various types of models into one.

Barboza, Kimura and Altman (2017) published a study that is a 21st-century extension of the groundbreaking Altman (1968) paper. Unlike Lessmann et al. (2015), they investigate a narrower set of estimation methods and they have one dataset at their disposal. The techniques they consider are logistic regression, discriminant analysis, support vector machines, artificial neural networks, bagging, boosting and random forests. The modeling scope concerns the business segment. They use 10,000 observations about North American companies for the period of 1985-2005, 449 of which defaulted. They maintain class balance by randomly choosing 449 non-default companies. The compiled train dataset therefore has less than 1,000 observations. Balancing of classes via simple reducing the dominant class was performed in the 1968 study as well. There the split is 33 default and 33 non-default observations. The decision to create an artificially balanced sample set removes the ability of this study to assess how different methods deal with unbalanced sets. In addition, 449 default observations is substantially more than 33. This in fact should enable most estimation algorithms to recognize patterns even though the observed default rate is just slightly lower than 4.5%. In the study, they further note that no standardization or data-transformative steps were made. Although they are aware that lacking data-transformative steps might reduce the power of some models, they opt to perform the comparison on data without this preprocessing. Removing major preprocessing steps such as data normalization or binning from the research makes the reported result on the performance of models less useful for the industry. A credit risk practitioner should be interested in a comparison of these models that was conducted in such a way that it reflects how the models would have been built in practice. They report that random forests, bagging, boosting and neural network models outperform the logistic regression approach. Versions of SVM as well as the linear discriminant analysis did not yield powerful models overall.

# Overview of Modeling Methodology

In this section, the basics of estimation techniques for default event classification problem and the performance tests are commented. Also, testing metrics as well as data preprocessing approaches are reviewed.

Prior to delving into the technicalities, the terms discrimination and calibration should be explained. It can be learned from Witzany (2017) that discrimination refers to the ability of a model to separate between classes (e.g. good and bad debtors). In the case of calibration, the goal is for the model to assign probabilities that are as correct as possible. Witzany (2017) explains that a simple train sample preprocessing wrangle such as undersampling would distort the ability of the model to provide correct probabilities of default. In this thesis, the point of concern is discrimination. Further, the credit risk practice differentiates between through-the-cycle (TTC) and point-in-time (PIT) calibrations. Whilst TTC is intended to represent a longer term average over at least one whole economic cycle, the PIT estimate focuses on being more up to date. Witzany (2017) comments that PIT is more useful for business.

## Model Estimation

In this subsection, the estimation techniques which are used in the application section are reviewed.

### Logistic regression

Logistic regression is used as the industry standard due to its sound performance and good interpretability of its results. Consider the probability of default of the *i*-th observation *pi*, then the probability of default conditioned by right-hand side variables **x**i is:

|  |  |
| --- | --- |
| , |  |

where **β** is the vector of coefficients, ui is the error term for which a 0 mean and a distribution are being assumed. Since can take upon values from minus infinity to plus infinity, Fi is used as a link function in order to confine that result between 0 and 1. In that way, the result is converted into a probability (Witzany, 2017). Witzany (2017) further argues that the deterministic part of the equation can be interpreted as the currently observable debtor’s credit capacity, whilst the error term reflects the unknown development in the future.

The choice of the link function can vary. The two basic ones are Probit and Logit. The two link functions give very similar results in econometric applications. Nevertheless, Witzany (2017) highlights that the logistic distribution has heavier tails, which can be seen as being more conservative since it assigns a larger weight to extreme events. The choice of the link function ultimately marks what assumption about the distribution of the error term is being made. In the case of Logit, the link function is the sigmoid curve:

|  |  |
| --- | --- |
| , |  |

where z equals the model specification introduced above: . The sigmoid curve is depicted below.

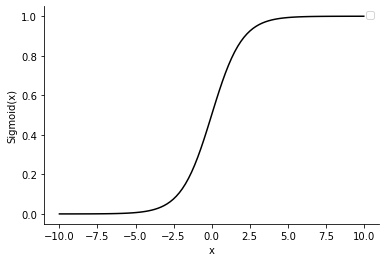


Figure 1: sigmoid function, source: author

To solve the equation defined above, the maximum likelihood estimator is used. In its log-transformed form:

|  |  |
| --- | --- |
| , |  |

where L is the likelihood function and **b** is the estimation of **β**. The equation can simply be extended by L1 (Lasso), L2 (Ridge) penalty terms or their combination (elastic net) via addition. These are, for L1 (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where ||b||1 is the L1 norm equal to . The L2 penalty term transforms the basic estimator to (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where ||b||2 is the L2 norm equal to . Finally, combining L1 and L2 would yield something called the elastic net penalization.

Since the first equation holds, it also holds that the good-bad odds are:

|  |  |
| --- | --- |
| , |  |

where pi equates . Taking logarithms of both sides of the equation above yields the log good-bad odds:

|  |  |
| --- | --- |
| . |  |

Now, in the case that the sigmoid function is used, then the partial derivatives of the log-likelihood function are found. Here, lecture notes of Piech (2017) are adjusted toward the notation used throughout this section and slightly more operational detail is provided. Starting from (7) above, the derivative of the likelihood function w.r.t. the parameter *bj* from **b** is:

|  |  |
| --- | --- |
| . |  |

First, derive the logarithmic terms using the derivative of log rule and isolate the underived residuum:

|  |  |
| --- | --- |
| . |  |

Using the derivative of as the outer function and the derivative of as the inner:

|  |  |
| --- | --- |
| , |  |

where xi,j is a scalar. The above can be manipulated by:

|  |  |
| --- | --- |
| , |  |

further into,

|  |  |
| --- | --- |
| , |  |

finally, the denominator and the numerator below can partially cancel out,

|  |  |
| --- | --- |
| , |  |

and result in:

|  |  |
| --- | --- |
| . |  |

Witzany (2017) further states that given the negatively definite Hessian matrix the solution exists and it is unique. There are a variety of algorithms to solve the equation, one of them is the Newton-Raphson’s according to Witzany (2017).

### Neural Network

Consider a vector of input data **x** of *m* rows and 1 column. The vector is multiplied with a weights vector **w** that has 1 row and *m* columns (**wx**). The resulting linear combination is a scalar. The scalar result is then used as an input for the so-called activation function *f*. This function is chosen to be non-linear, e.g. sigmoid. The resulting calculation from the non-linear function is an activation – *a*.

The operations described in the previous paragraph create one neuron, which is the basic structure in this model. A neuron is understood as (1) a node characterized by an activation function that (2) uses linearly combined input from the previous layer to produce an output. It follows that one neuron is defined as:

|  |  |
| --- | --- |
|  |  |

where **w** is (*1,m*) and **x** is (*m,1*) and *f* is an arbitrary non-linear function. Schematically this is depicted in Figure 2, where each line represents a multiplication and the scalars subscripted with *i* are entries of matrices **x** and **w** in the definition above. The rectangle represents the neuron *a*.

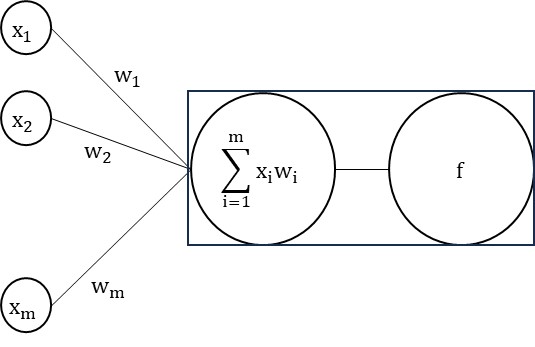


Figure 2: structure of a neuron, source: author, Witzany (2017)

Now, one neuron as defined above is part of a larger model structure. Consider that the neuron *a* defined above is just one of many – *ak* neurons. Together they form a vector of activations **a**. The vector **x**, being the vector of input data, remains unchanged. However, for every *ak* in the vector **a** to be treated separately, each *ak* is multiplied by a separate set of weights – **wk**. Hence, the vector **w** from the initial example is replaced by a matrix **W**. The dimensions of this matrix are (*k,m*) – the number of rows corresponding to the number of activations they are being fed towards and the number of columns corresponding to the number of rows of the vector **x**. It follows that this augmentation results in:

|  |  |
| --- | --- |
| , |  |

where the notation and dimensions are explained in the paragraph above. Extending the somewhat general form written above to multiple sets of activations, i.e. hidden layers boils down to simple stacking. For example, a neural network of 3 hidden layers would have:

|  |  |
| --- | --- |
| , |  |

where is **a1** – the first layer, or more clearly is **a2** – the second layer and so on. Finally, y is the output. The example developed above works only with a vector of **x**, i.e. one observation. For practical reasons it is more realistic to expect multiple observations:

|  |  |
| --- | --- |
| , |  |

where the change boils down to **x** and y becoming **X** and **y**, respectively. Figure 3 shows a neural network of 3 input features, 3 activation nodes and one output. The matrix **W1** contains the weights connecting the input and the hidden layer whereas the vector **w2** leads to the output.

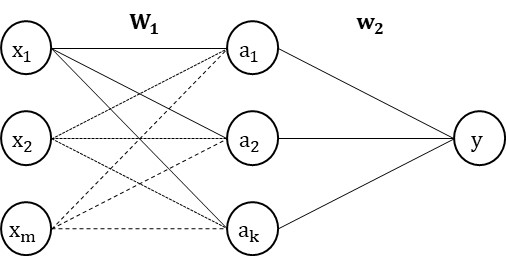


Figure 3: neural network with multiple neurons in hidden layer, source: author, Witzany (2017)

If the response variable y is continuous, the loss function used for optimization of the weights and biases would traditionally be the sum of squared residuals. Since in this thesis the point of concern is a binary dependent variable (event of default), further discussion goes straight into addressing the binary dependent variable problem.

Consider a binary response variable, the neural network for that optimization problem would have 2 output nodes, y1 and y2, together forming a vector – **y**T = [y1, y2]T. Assume that y1 would contain the raw output for the event occurring (i.e. attain value 1) and, conversely, that y2 would contain the raw output value for the event not occurring (i.e. represented by value 0). The first question is (1) how to interpret the raw output values and the second question is (2) how to optimize the weights and biases of the neural network. The interpretation problem can be resolved by using the so called argmax function. It simply states that the chosen response corresponds to the yp node that has the highest raw value. However, although the argmax function provides a simple interpretation of the raw output values, it cannot be used for the mentioned optimization. The reason for this is that the slope of this function is always 0 at any point. To tackle the optimization problem, one can use the softmax function, which has the following form (James et al., 2017).

|  |  |
| --- | --- |
| , |  |

where is the exponent of i-th raw output divided by the denominator which contains the sum of exponents of all p raw outputs. In a binary classification problem, p equals 2. This provides a differentiable function and its p softmax calculations sum to 1. They are therefore interpreted as probabilities. Now, the softmax-transformed raw output values of the neural network are passed onto the cross-entropy function (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where the subscript *n* refers to the cross-entropy calculation of the *n*-th observation of the matrix of input features **X**. The *N* calculations are summed which generalizes the expression above into (James et al., 2017):

|  |  |
| --- | --- |
| . |  |

To optimize the neural network, the operational goal is to minimize the cross-entropy it outputs. The minimization problem is not solved analytically but rather an algorithm is chosen to solve the task. In this text, gradient descent is explained, and its steps are the following (James et al., 2017):

1. Randomly initialize the parameters (weights and biases) of the neural network.
2. Analytically express the derivatives of the loss function (cross-entropy) w.r.t. the parameters (weights and biases), this is the gradient of the loss function (Witzany, 2017):

|  |  |
| --- | --- |
| . |  |

Note that the negative of the gradient is taken in order to move in the opposite direction. The derivatives w.r.t. the respective weight parameters can be expressed using the chain rule.

1. Use the parameters generated at point (1) as inputs for the analytical forms in step (2), save the result.
2. Multiply the result from (3) with a learning rate and save the result.
3. Subtract the result from (4) from the initialized values of weights and biases.
4. Use (5) as the new weights and biases and repeat the process (steps 3-6) until some arbitrary criterion is met. The criterion is set so the gradient would be approximately 0.

### Support Vector Machines

The basic idea of the Support Vector Machines (SVM) model starts from the Maximal Margin Classifier (MMC). The latter fits a hyperplane to data which is linearly separable. The MMCs optimization problem is defined as (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

subject to:

|  |  |
| --- | --- |
| , |  |

and,

|  |  |
| --- | --- |
|  |  |

where M is the margin area around the hyperplane. In this basic MMC model, no data points are allowed to be in the marginal area. This is commonly referred to as the hard margin. The goal of the maximization problem is to fit a hyperplane such that it correctly classifies the observed data points and maintains the M-distance of all data points from the hyperplane itself. Correctly classifying the data with a linear hyperplane (including a margin) strictly means that no errors can be made. In practice this ends up narrowing the margin. However, what is more frequent in practice is also that observations cannot be separated with a hyperplane and some error tolerance should exits. The Support Vector Classifier (SVC) relaxes the hard margin condition and introduces a soft margin, the optimization problem is re-defined as (James et al., 2017):

|  |  |
| --- | --- |
| max(M), |  |

subject to:

|  |  |
| --- | --- |
| , |  |

and,

|  |  |
| --- | --- |
|  |  |

where:

|  |  |
| --- | --- |
| and . |  |

The variable is known as the slack variable and, depending on its value, it will allow its incorrect classification. The tuning parameter C introduces the possibility to control the slack variable. The larger the tuning parameter, the larger the error tolerance of the classifier (James et al., 2017).

The decision boundaries that are created with the hard and soft margin algorithms described above are linear. Such a classifier cannot handle non-linear relationships between the features and the response variable. This is in some cases crucial for having an algorithm that can serve its purpose. To handle nonlinearities, it may suffice to scale the data, e.g. extend the hyperplane above by adding second powers of features. This would ensure a non-linear shape of the data-dividing hyperplane. A non-linearity is simply introduced into the margin condition, for example second powers addition (James et al., 2017):

|  |  |
| --- | --- |
|  |  |

The SVM approach is to handle a non-linear condition such as the one written above through a so-called kernel function *K*, which can be specified in different ways. For example, the polynomial kernel is (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where d stands for degree. Another popular choice is the radial kernel (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where γ is larger than 0. Whichever *K* is chosen, it is always about the evaluation of dot products of pairs of observations. This kernel function is then embedded into the support vector classifier (James et al., 2017):

|  |  |
| --- | --- |
| . |  |

### Tree-based Methods

The building block of the random forest algorithm is a single decision tree. In the case of this thesis, the classification tree is relevant. The classification tree passes observations onto a sequence of questions. It is designed in such way that the evaluation of a metric *i* (a binary “yes” or “no”) determines what the *i+1* metric will be. This results in branching – for example, at initialization, there is only 1 classification metric (node) with 2 evaluations; This results in 2 new nodes with 4 evaluations, and so on until the *leaf*/*terminal* nodes. In practice classification trees do not need to grow symmetrically. Stopping criteria, “pruning” techniques or simply achieving “purity” of the terminal node might result in an asymmetric tree (James et al., 2017). An part of a decision tree is depicted schematically on the following figure, following from Witzany (2017). To maintain simplicity of the diagram, the thresholds are not documented. Nonetheless, arbitrary thresholds can be assumed in order to understand the scheme. For example, the length of employment is less than 7 years, the observation is sent to the left part of the tree. Conversely, if the length of employment is 7 or more years, the observation is sent to the right part of the tree for further decision making.

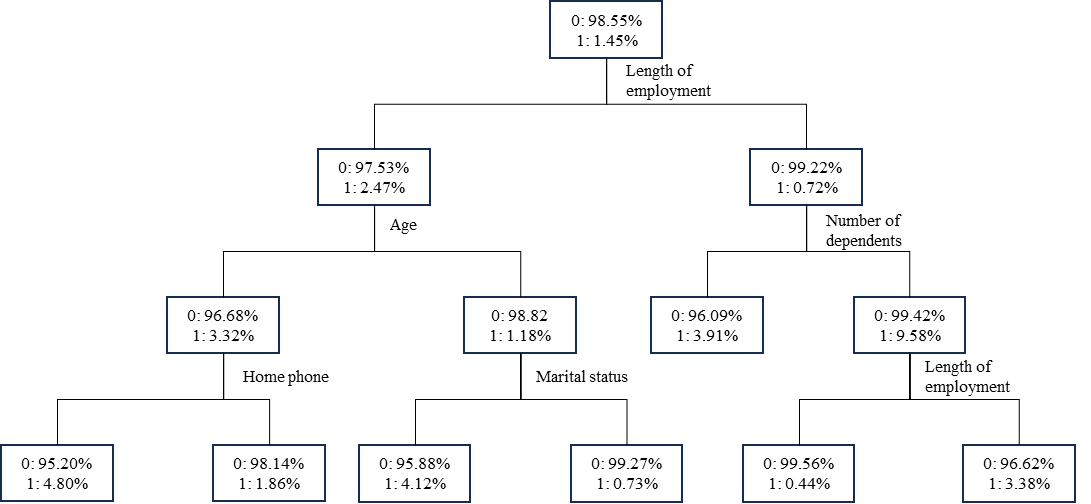


Figure 4: example of a decision tree, source: Witzany (2017)

To optimize for an adequate tree, one looks at (1) the optimal sequence of binary classifications based on the discriminatory power of individual features (**X**) and (2) the termination criteria for the generation of new nodes. The discriminatory power of individual features can be evaluated using different criteria. For example, the Gini coefficient of discrete variables is (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

where is the probability that the class k will be obtained in the m-th region (node). If the probability of observing k is high, then G will be relatively low. This indicates high discriminatory power. Another popular metric is entropy (James et al., 2017):

|  |  |
| --- | --- |
| , |  |

which also takes upon relatively low values for highly discriminating regions of observations. For example, a dataset may contain information about whether credit card applicants had a permanent employment contract at the moment of applying. Also, assume that this dataset has information about default events. The more often the “no permanent job contract” classification coincides with the “defaulted” classification in the dependent variable as well as the opposite holding for the applicants with a permanent job contract, the lower the G and D will be. Calculating G and D is more involved in the case of continuous explanatory variables. There, a continuous variable is first sorted. Then, averages of 2 nearest observations are calculated. These points are candidates for the discretization of the continuous variable. The best performing threshold as calculated by Gini or entropy is then used as a data transformation for the classification tree. A single classification tree is not robust and it typically overfits the training data. This issue is tackled by bagging, random forests and boosting methods (James et al., 2017).

#### Bagging

In classification problems, bagging simply refers to (James et al., 2017):

1. estimating multiple decisions trees on subsets of the training data observations (this data can be bootstrapped in the case there is not enough data),
2. collect the predicted classification for each observation and take the most frequent classification of each observation as the final classification.

Once many outcomes are averaged in this way, the simple interpretability of a decision tree is lost. The *variable importance* measure is a way of reconciling the contribution of individual features to the overall performance of the model. In the case of classification problems, tracked are contributions of individual variables to the overall Gini (James et al., 2017).

#### Random Forests

Random forests extend the idea of bagging by not only limiting the training data to subsets of observations, but also to subsets of explanatory variables. Consider a number of features *m*, then the random forest algorithm randomly chooses *p* features such that *p* ≤ *m*. The smaller the *p*, the more different the approach is from bagging. Conversely, if *p* = *m*, the random forest would practically yield the same result as bagging. The goal of the random forest approach is to have less correlated decision trees (James et al., 2017).

#### Boosting

The basic idea behind boosting methods is to build consecutive trees. Each tree learns from the errors made by the previous ones. Popular are AdaBoost and gradient boosting, there are also many other variations.

AdaBoost creates simple decisions trees with one metric and 2 possible results. Such trees are called *stumps* amongst practitioners. James et al., (2017) provide a pseudo-algorithm of boosting.

1. At initialization, each observation is assigned a weight of 1/*N*.
2. The errors of a former stump are stored and are used as input for the next one.
3. The observations on which the former stump predicts incorrectly are assigned with a new weight which is larger relative to the correctly predicted ones.
4. A new dataset is generated where weights are used as inputs. This results in a synthetic repetition of the incorrectly predicted observations. The next stump is therefore more sensitive to these.

Steps 2-4 are repeated until some stopping criteria is met.

Gradient boosting has its name given the small steps approach based on the learning rate, the steps are written based on an elaboration of Joshua Starmer (“Video Index”, n.d.).

1. At initialization of a classification gradient boost algorithm, each observation is assigned with the overall log-odds of the dataset. These are then transformed to probabilities using the sigmoid function.
2. The pseudo-residuals of the former step are obtained as differences between the predicted probabilities and the *observed probabilities* (which are 0 and 1). They are used as inputs for the next step, which is the creation of a new decision tree.
3. The next decision tree forecasts the residuals of the previous step. Their terminal leaves contain values of those residuals. Now, their output values have to be transformed, for example for one leaf by:

|  |  |
| --- | --- |
| , |  |

where *i* is the current decision tree, *i-1* is the previous step and *K* is the number of observations in one leaf. This is repeated for every leaf of the decision tree *i*.

1. The raw output value from the previous step is multiplied by the learning rate and added to the previous steps to obtain the log-odds *i*.
2. The log-odds *i* is transformed into a probability for example using the sigmoid function.

Steps 2-5 are repeated until some stopping criteria is met.

### KNN

Consider a dataset of known outcomes **y** and corresponding explanatory features **X**. For a new observation yi and **xi**, the nearest neighbor approach calculates the fitted value as the average of K nearest observations (Witzany, 2017):

|  |  |
| --- | --- |
| . |  |

The formulation above does not contain the right-hand side features. In fact, the explanatory variables are used only to identify the nearest observations. Hence, the main problematic of this method surrounds the question of how to identify the most similar ones. The most straightforward way to identify similar observations is to use the Euclidian distance (Witzany, 2017):

|  |  |
| --- | --- |
| . |  |

Categorical variables have to be converted into numerical in order to enable this computation (Witzany, 2017). This can be achieved by, for example, introducing a dummy variable transformation or calculating the Weight of Evidence.

## Performance Metrics

In this section, various model accuracy testing approaches are reviewed.

### Accuracy Ratio and CAP

The accuracy ratio is a summary statistic of the cumulative accuracy profile (CAP) (Engelmann et al., 2003). Consider 3 random variables – ST, SD and SND which denote score distributions of all, defaulted and non-defaulted debtors respectively. Engelmann et al. (2003) then define that the probability that a debtor *j* will receive a score value of *i* is:

|  |  |
| --- | --- |
| , |  |

where π is the probability of default and the total score of the debtor at hand is the sum of the defaulted and non-defaulted score probabilities – pD and pND. The cumulative probabilities CDD, CDND and CDT are simply defined as sums of probabilities of different scores *i*. Then, the CAP plot represents combinations of points CDTi and CDDi for every *i*. A random model would have all those points equal, which results in a straight line. A theoretical example of the CAP plot is outlined below, sourced from Witzany (2017).

A diagram of a model

Description automatically generated

Figure 5: example of CAP, source: Witzany (2017)

The larger the area between the CAP line and the random model line, the better the discriminatory power of the model. Conversely, the surface between the CAP line and the random model line can be divided by the surface between the perfect model line and the random model line. In that way, the accuracy ratio is obtained (Engelmann et al., 2003):

|  |  |
| --- | --- |
| . |  |

Witzany (2017) cites Engelmann et al. (2003) and further provides the following formulation. Consider a non-defaulted *good* debtor G and a defaulted *bad* debtor B. It is then considered that a rating tool performs well w.r.t these 2 debtors if the rating of the debtor G is higher than the rating of the debtor B. In total, there are three possible scenarios and each can be assigned a probability:

|  |  |
| --- | --- |
| , |  |
| , and |  |
| . |  |

The accuracy ratio can also be expressed in terms of these probabilities (Witzany, 2017):

|  |  |
| --- | --- |
|  |  |

### AUC and ROC

The Area Under the Curve (AUC) is a summary statistic developed by using the Receiver Operating Characteristic (ROC) curve (Engelmann et al., 2003). As outlined in the section above, good and bad debtors have their associated distributions of possible scores that they can receive. Since rating models are imperfect in practice, one can expect an overlap between these 2 distributions. Given the overlap, a cut-off value has to be introduced to rate a debtor as a good or bad. The optimization problem can be summarized in a confusion matrix.

Table 2: confusion matrix

|  |  |  |
| --- | --- | --- |
|  | default | non-default |
| below *C* | hit | false alarm |
| above *C* | miss | correct rejection |

Source: Engelmann et al., (2017)

In the confusion matrix above, the cut-off value is denoted as C. If the score is below C then the debtor is assigned with a default rating. The opposite holds. The 4 decisions made above can be summarized into several rates, Engelmann et al., (2003) are followed to define the hit and false alarm rates:

|  |  |
| --- | --- |
| , and |  |
|  |  |

Where HR stands for hit rate and FAR stands for false alarm rate w.r.t. the defined cut-off value C, S stands for the assigned score. The metrics defined above are sufficient to construct the ROC curve – it is a plot of combinations of all HR(C*i*) and FAR(C*i*) for each *i*. The optimal cut-off value is the one that is the closest to the (0,1) point on the plot, i.e. the perfect model. Now, utilizing the whole ROC curve irrespective of the optimal cut-off value, the surface under it is calculated and the Area Under the Curve (AUC) is obtained. It is then divided by the surface of the perfect model. Engelmann et al., (2003) derive the probabilistic interpretation of the AUC. Witzany (2017) expresses the AUC in these terms:

|  |  |
| --- | --- |
|  |  |

where the assignment of equal ratings to G and B is taken as a half-success. Witzany (2017) as well as Engelmann et al., (2003) further note that the AUC is a linear transformation of AR:

|  |  |
| --- | --- |
| . |  |

### Rates of Error

Witzany (2017) describes the total weight of error (ER) as the weighted sum of Type I and Type II errors. In the confusion matrix outlined above, the Type I error corresponds to the false alarm, i.e. rejection of applicants that would have been performing. The Type II error corresponds to the miss rate, i.e. the approval of an applicant that subsequently defaults. The Type I and II errors are weighted by the proportions of good and bad applicants w.r.t. the total validation sample:

|  |  |
| --- | --- |
|  |  |

where HR(C) and FAR(C) are hit and false alarm rates as described in the section above given the cut-off C. πB and πG are probabilities of bad and good, respectively and they sum to 1.

The total rate of error introduced above is still somewhat impractical for business. Witzany (2017) notes that the impact of a single false positive is different than the impact of a single miss in financial terms. Therefore, the weighted cost of error (WCE) is:

|  |  |
| --- | --- |
|  |  |

where l is the loss realized from a bad debtor and q is the profit realized on a good debtor.

## Preprocessing

Thus far explained were different mathematical approaches to estimating a classification model as well as some general testing metrics of their performance. In this section, reviewed are major preprocessing operations. Here, data preprocessing is understood as any activity that comes before the actual mathematical calculation of the model and its subsequent testing.

### Data Transformations

In this section, several data-transformative operations are reviewed.

#### Binning

Consider an explanatory feature **x**, which can be both continuous or discrete. Binning is an operation in which the *N* scalars of **x** are grouped into *S* buckets. The observations in bucket *s* usually obtain some common data-transformative treatment. For example, all entries of the bucket *s* can be replaced by their within-group average, or they can be transformed to represent the average within-group default rate (see WoE below). In any case, prior to applying the transformative treatment to the grouped data, the binning itself has to be performed.

In solving this thesis, the OptBinning library published Navas-Palencia (2020) is used. In the OptBinning programming solution, the bins can be initialized via uniform/quantile discretization, decision tree or the minimum description length principle method (MDPL). The uniform/quantile discretization as well as the decision trees are implemented by Navas-Palencia (2020) using modules from the globally known Scikit-learn library (Pedregosa et al., 2011). The MDPL solution was programmed by Navas-Palencia (2020). When using the OptBinning solution, the default setting are decision trees. The splitting values as defined by the decision trees are used as the boundaries of buckets. In later sections of this thesis, the default setting will be followed. The algorithm can also take into account a range of constraints regarding monotonicity, reduction of bin sizes depending on their dominance over other bins (by setting minimum and maximum numbers bin sizes) and maximum p-value thresholds to ensure statistically significant differences between bins.

#### Weight of Evidence (WoE)

Witzany (2017) defines the weight of evidence (WoE) as:

|  |  |
| --- | --- |
| , |  |

where P[s|G] is the probability of a good debtor being observed in the bucket *s* and the same applies for bad debtors. In practice, the values obtained by the WoE transformation are often used to support the transformative treatment mentioned in the section above. Consider observations in a bucket *s* defined as per the section above. Then the WoE-transformation would be to replace all observations in the bucket *s* by the WoE value obtained for that bucket. If there are, for example, 5 bins, there would be a total of 5 WoE values. The WoE-transformed vectors are treated as continuous explanatory features.

#### Dummy Variable Transformation

The term dummy variable is widely used in econometrics whilst “one hot encoding” is popular in the data science community. Consider a discrete variable of *N* observations that attains *K* possible values. The dummy variable-transformation of this variable would result in a matrix of *K-1* variables (columns) and *N* rows. Each column represents one of the *K-1* retained categories whilst the *K-th* category is held as a reference. One category is always withheld as a reference category in order to avoid having a perfect linear reconciliation when, for example, the constant term is being used. This issue is commonly known as the “dummy variable trap”. In the case that multiple variables are transformed in such way, with an increasing number of features there is risk that they would linearly reconcile between themselves. This being despite the fact that an arbitrary category is dropped from each variable that is being encoded in this way. In that case, the feature selection procedure should be equipped to recognize such situation and make optimal selection decisions. Finally, when attempting to “dummify” a continuous variable, it first has to be categorized in some way. For example, binning as described in section 2.3.2.1 is a suitable transformative step.

### Feature selection criteria

The following metrics are used as industry-standard instances at which feature selection(exclusion) operations are performed. When appropriate, section 2.2 is referred to for technical detail. This is because the model testing metrics can be conceptually the same as the feature selection metrics during the model construction.

#### Information Value (IV)

The WoE calculation explained above is also funneled into the assessment of explanatory power of features. Witzany (2017) defines the information value (IV) as:

|  |  |
| --- | --- |
| , |  |

which can be interpreted as the average WoE of a variable, where the weighting is done using the probability distributions. The value is always positive and the rule of thumb is to set the minimal performance threshold between 0.04 and 0.1. This means that the features not satisfying the criterion are to be excluded from further modeling.

#### Gini

The Gini coefficient, or more commonly the AR as defined in section 2.2.1 is used as feature exclusion metric. As a rule of thumb, the Gini/AR value of 0.1 is considered a minimum for a variable to be used for modeling.

#### Multicollinearity Assessment

Groups of variables that exhibit high correlation can be identified using, for example, the Pearson correlation coefficient. For a given coefficient threshold T which ranges between 0 and 1 (including), the variable that in a univariate assessment provides the strongest discrimination of the target is retained. Conversely, the other variables of the highly correlated group are removed. To assess the strength of each variable, common metrics such as the AUC, the ANOVA F-value and so on can be used. Witzany (2017) suggest that 0.5 should be used as the threshold.

#### Forward and Backward Feature Selection

The forward feature selection algorithm is initialized by 0 features. Then, the first feature is determined as the feature that maximizes the chosen scoring criteria (for example, the AUC). The score is cross-validated. Then, the 2nd feature is looked for in the same way – the algorithm tries out all the remaining not yet-used features and selects the one that maximizes the cross-validated score (Pedregosa et al., 2011). The feature selection procedure continues until the desired number of features *N* is reached. Alternatively, more flexible stopping criteria can be defined.

The backward feature selection algorithm starts off with all the available features. Then, features are removed one-by-one until the desired number of features *N* is reached, or some other stopping criteria is met. Upon each removal of a feature, the remaining features are checked for their cross-validated score using the selection criteria. The weakest one is removed (Pedregosa et al., 2011).

### Class imbalance and Reject Inference

Class imbalance can pose an obstacle for a model to adequately discriminate between categories. The consequence of this is that the model always predicts the majority class. Firstly, it is necessary to maintain distinction between the relative proportion of the minority class to the majority class and the absolute count of minority class observations. For example, in a dataset that has 30 defaulted observations and 970 performing ones, it might be more challenging for a model to perform well than in the case when there are 300 defaulted observations and 9,700 performing ones. In any case, should class imbalance be treated proactively, oversampling and undersampling approaches offer themselves as possible ways to go.

Oversampling approaches are consisted of various synthetic generations of the minority class. They can range from simple duplicating of observations to synthetic generations based on KNN, such as SMOTE (Synthetic Minority Over-sampling Technique). An even more refined is the ADASYN (Adaptive Synthetic Sampling) method where the KNN combinations of initial minority observations are augmented by an error term. Both SMOTE and ADASYN have standardized programming solutions that were made by Guillaume et al. (2017). Undersampling is more aggressive in terms of data omissions. It simply refers to deleting a random number of majority class observations in such a way that the models are trained with a dataset that has a more balanced class ratio. As mentioned in section 1.2, Barboza, Kimura and Altman (2017) use the undersampling approach.

In the context of reject inference Witzany (2017) mentions that a financial institution can also consider defining a testing period during which applicants that would otherwise be rejected would be accepted. Since it is expected that this group of applicants would exhibit a higher default rate, this policy would provide new data for the purpose of modeling.

# Application to Czech Home Loans Portfolio

In this section, the methodology described in the previous chapters is put into an empirical review. Models are estimated using behavioral home loans portfolio data of a major Czech bank. In the remainder of this section, the first subsection provides an overview of the data, the second subsection describes the programming solution, the third section formulates the researched questions, estimated models, documents the results and finalizes the discussion.

## Data

The modeling dataset concerns a part of the home loans portfolio of a major Czech bank. The available dataset has observations for which at least 6 months elapsed since the moment of contract signing. The dataset therefore enables the creation of a behavioral scorecard. The dataset has 200,000 observations, out of which 726 facilities were market as defaulted. This amounts to an overall observed default rate of 0.36%, which is typical for mortgage portfolios. As it is implied from the table below, the dataset was formatted in a fixed cohort setting. Re-structuring the dataset into flexible cohorts would not be possible.

Table 3. Overview of modeling dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **cohort date** | **# observations** | **# defaults** | **default rate (%)** |
| 2011 | 15,182 | 103 | 0.68% |
| 2012 | 16,230 | 82 | 0.51% |
| 2013 | 18,936 | 104 | 0.55% |
| 2014 | 20,153 | 134 | 0.66% |
| 2015 | 21,923 | 73 | 0.33% |
| 2016 | 23,727 | 84 | 0.35% |
| 2017 | 25,930 | 70 | 0.27% |
| 2018 | 28,406 | 36 | 0.13% |
| 2019 | 29,513 | 40 | 0.14% |

Source: CSOB, author

It can be observed that whilst the number of facilities doubled by the end of the last decade, the portfolio experienced a decreasing number of defaults in absolute terms as well. The default rate is therefore 4 times lower than the one observable in the initial years.

## Programming Solution

The modeling exercise is performed in Python. Several Python classes are created and they are listed in the table below. The classes allow for a certain degree of variability in their settings. For example, the Modeler class can accommodate a forward feature selection or it can pass without using one.

Table 4: list of defined Python classes

|  |  |  |
| --- | --- | --- |
| **Class name** | **Defined functions** | **Purpose** |
| DataGetter | rename\_columns  bool\_to\_flg  wrangle\_martial\_status  exclude\_features | Data ingestion programmed specifically for used dataset. |
| Preprocessor | retain\_oot  split\_train\_test  undersample\_train  list\_categorical  bin\_and\_transform  remove\_multicollinearity | Data preprocessing procedures not dependent on used dataset. |
| Modeler | select\_features  apply\_one\_hot  model\_logit  model\_ann  model\_knn  model\_svm  model\_bagging  model\_rf  model\_boosting | Modeling procedures and post-selection feature engineering. |
| Validator | predict  plot\_roc  plot\_cap  plot\_ks | Validation procedures. |

Source: author

### Data Ingestion

The DataGetter class solves data manipulation issues that are specific to the dataset at hand. The longlist of all available variables along with their descriptions is included in Appendix A.

The treatment of missing values is performed depending on how the dataset is to be used upon having it passed to the Preprocessor class described in the next section. In the case that the intention is to use binning, minimal data manipulation operations are performed. These boil down to renaming via a mapping table, type changing, solving of minor encoding issues and feature exclusions based on a manually constructed mapping table. Missing entries are not treated proactively. They are left to the binning algorithm in the Preprocessor class. There, missing entries are assigned into a separate bin.

### Preprocessing: transformations and shortlisting

The Preprocessor class performs operations on data that is already tidied up with the DataGetter class. The only data deficiency that it can deal with are missing entries, and that provided that they are taken along into a binning procedure.

In this class, several functions are defined and they solve the following tasks:

1. train-test split.
2. binning and transformation, and
3. multicollinearity removal.

The train-test split function is straightforward. The binning and transformation function firstly performs the binning procedure as described in section 2.3.2. Then, features can be transformed into WoE based on the corresponding bins. The multicollinearity removal function identifies groups of correlated variables and retains the strongest feature of that correlated group.

It is upon the user of the class to specify which of the functions are to be ran. For example, one reasonable combination might be to use (1) and (2), where (2) is specified to end with a WoE transformation.

### Modeling

The Modeler class performs operations that are limited to feature selection, hyperparameter optimization and model estimation. The class has the following functionalities:

1. sequential feature selection,
2. dummy variable transformation,
3. hyperparameter grid search,
4. logistic regression estimation,
5. ANN estimation,
6. KNN estimation,
7. SVM estimation, and
8. RF estimation.

The dummy variable transformation takes place after the possible use of the forward feature selection. Should this encoding happen before the multicollinearity removal or the forward feature selection, one would risk distorted sets of dummy variables. For example, it would most probably happen that 3 out of 6 categories are taken for one variable and only 1 out of 10 categories are retained for the next variable. In order to avoid that, WoE-based features are used beforehand. Then, the WoE-transformed bins are sent into the dummy-based codification.

All of the functionalities above (except the dummy variable transformation) are built on primarily Scikit-learn classes (Pedregosa et al., 2011). In the case of feature selection, the selection metric, final number of features, forward/backward approach and the cross-validation parameters can be chosen. The hyperparameter grid search is always to be specified w.r.t. the relevant hyperparameters of the model that is being used. The output of the remaining functionalities is straightforward. For completeness, a list of adjustable hyperparameters/specific comments is provided:

1. Hyperparameter optimization in the case of logistic regression is enabled only for L1/L2 penalizations and their combinations (elastic net); In the case that multicollinearity is being removed when using the Preprocessor class, the logistic regression functionality will not have any such penalization.
2. The ANN hyperparameter optimization takes in various combinations of hidden layer number and sizes, activation functions, solvers and learning rates. Early stopping is always activated.
3. The KNN hyperparameter optimization tries out a range of values for K, tests uniform and distance-based weighting of neighbors, Manhattan and the Euclidean distances.
4. The SVM hyperparameter search tries out a range of kernels, degrees of polynomials and regularization parameters C.
5. The RF hyperparameter search iterates over a range of single decision tree depths.

### Validation

The Validator class takes in train and test data as well as the model fits outputted from the Modeler class. It has the following functionalities:

1. prediction,
2. ROC plot creation,
3. CAP plot creation, and
4. K-S plot creation.

## Application

The first part of this section formulates the researched questions, the second part summarizes all the approaches to model estimation. Finally, the last two parts provides an overview of the results and their discussion.

### Researched Questions

The aim is to formulate answers to the following questions. Note that the first question is the main aim of this thesis whilst the remaining questions are additional. Nonetheless, they provide better insight for the purpose of discussion.

1. Whether non-traditional ML modeling techniques (ANN, KNN, SVM and tree methods) outperform the traditionally used LR.
2. Whether there is a difference between same model estimation techniques depending on whether they are trained on WoE-based features or dummy-based features.
3. Whether there is a difference between same model estimation techniques depending on whether they are trained on more class-balanced or highly class-imbalanced data.

### Estimated Models

The following combinations of data preprocessing and model estimations are considered. The data is split into train, test and out-of-time samples (OOT). The OOT sample contains all observations registered as at January EOM, 2019. The pre-2019 snapshot data is split into train and test following a 70:30 ratio. The IV & Gini-based shortlisting, undersampling, binning, WoE/dummy transformations, multicollinearity removal, forward feature selection and finally the model estimation are all performed on the train sample. The conclusions are made on a narrower and more common range of ML models, this is consistent with Altman et al. (2017).

Table 5: estimated models

| **Batch** | **IV & Gini shortlisting** | **Undersample** | **Binning & WoE/dummy** | **Multicoll. removal** | **Forward feature selection** | **Estimation techniques** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Yes | No | WoE | Yes | Yes | LR  ANN  KNN  SVM  bagging  RF  boosting |
| 2 | Yes | No | WoE | No | No | LR  ANN  KNN  SVM  bagging  RF  boosting |
| 3 | Yes | No | Dummy | Yes | Yes | LR  ANN  KNN  SVM  bagging  RF  boosting |
| 4 | Yes | Yes | WoE | Yes | Yes | LR  ANN  KNN  SVM  bagging  RF  boosting |
| 5 | Yes | Yes | WoE | No | No | LR  ANN  KNN  SVM  bagging  RF  boosting |
| 6 | Yes | Yes | Dummy | Yes | Yes | LR  ANN  KNN  SVM  bagging  RF  boosting |

Source: author

The initial shortlisting based on minimum Gini and IV criteria is always performed. The first three batches of models is trained on datasets that are not undersampled. Another three batches of models are trained on an undersampled dataset where the majority class is reduced in the number of observations. The ratio between defaults and non-defaults is 1:4. Then a regular non-undersampled test set where the majority class maintains its proportion is used for testing. The same holds for OOT. Either WoE or dummy-based data transformations are used for model training. Multicollinearity is removed in two sets of WoE-based models. Not removing multicollinearity in the case of dummy-based models is not considered due to the fact that the number of features would approach the number of defaults, if not exceed it. In addition, the multicollinearity removal is in practice a good method to prevent perfect collinearity between features. As mentioned, the forward feature selection is always WoE-based.

The models listed in the last column are subject to hyperparameter optimization via a grid search.

In the case of logistic regression (LR), L1, L2 and the elastic net penalization are tried out when multicollinearity is not removed during data preprocessing.

The optimal hyperparameters of the ANN model are searched over the following grid.

Table 6: hyperparameter grid for ANN model

| **Hyperparameter** | **Combination** |
| --- | --- |
| Hidden layer sizes *(the number of integers in the bracket implies the number of layers, n integers means n layers)* | [7] [5, 5] [7, 7] [10, 5]  [10, 10]  [20, 20] [40, 40, 20]  [40, 20, 20]  [20, 40, 20] [100, 200, 100]  [200, 100, 100]  [200, 200, 100]  [200, 400, 200] |
| Activation function | Logistic Tanh Relu |
| Solver | Adam |
| Alpha | 0.001 |
| Learning rate | Constant Invscaling Adaptive |

Source: author

The optimal hyperparameters of the KNN model are searched over the following grid.

Table 7: hyperparameter grid for KNN model

| **Hyperparameter** | **Combination** |
| --- | --- |
| # of neighbors | 1, 5, 10, 30, 50, 100 |
| Weights | Uniform  Distance |
| Solver | Manhattan  Euclidean |

Source: author

The optimal hyperparameters of the SVM model are searched over the following grid.

Table 8: hyperparameter grid for SVM model

| **Hyperparameter** | **Combination** |
| --- | --- |
| Regularization parameter | From 0 to 1, by 0.25 |
| Kernel | Linear Polynomial RBF Sigmoid |
| Polynomial degree | From 1 to 3, by 1 |

Source: author

The optimal hyperparameters of the bagging model are searched over the following grid.

Table 9: hyperparameter grid for bagging model

| **Hyperparameter** | **Combination** |
| --- | --- |
| # of estimators | 10, 20, 30, 50, 100, 150 |

Source: author

The optimal hyperparameters of the RF model are searched over the following grid.

Table 10: hyperparameter grid for RF model

| **Hyperparameter** | **Combination** |
| --- | --- |
| Maximum tree depth (# of branches) | 1, 2, 3, 4, 5, 10, 15 |
| # of estimators | 10, 20, 30, 50, 100, 150 |

Source: author

The optimal hyperparameters of the boosting model are searched over the following grid.

Table 11: hyperparameter grid for boosting model

| **Hyperparameter** | **Combination** |
| --- | --- |
| # of estimators | 10, 20, 30, 50, 100, 150 |

Source: author

### Results

The results section is split into two parts. The first part showcases step-by-step details about one of the estimated models. The purpose of the section is to provide a tangible insight example of how each modeling step contributes to the final result. The second part provides a summary of all estimated models.

#### Example of One Run

Reporting the list of selected features at each step, bin specifications, resulting WoE values, multicollinearity removal outputs etc. would be a cumbersome exercise for every estimation. Since the main task of this thesis is not to compare the results of preprocessing procedures, only one example is shown in order to provide a general orientation of how the modeling is being done step-by-step. These operations are performed on a 70:30 train-test split of the pre-2019 data, where the 2019 data is held as OOT, with run seed set to 130816. Firstly, several examples of the binning procedure outputs are observed. Starting with the behavioral score.

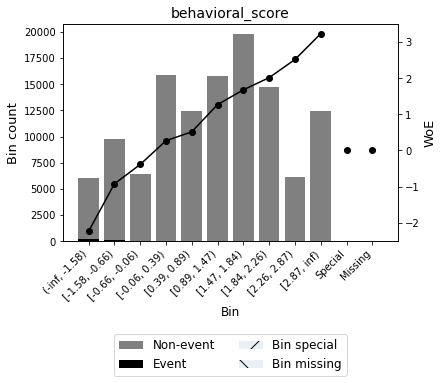


Figure 6: binning and WoE values of behavioral score, source: author

A large number of clients that apply for a home loan already have some history with the issuing bank. The data about their previous behavior, such as overdraft accounts can serve as a good indicator of how they will manage their home loan. Here, WoE grows monotonically from the lowest scores to the highest one. Although there are no missing or special values, the binning algorithm provided values for those cases as well.

The larger the debt service-to-income (DSTI) ratio, the higher the occurrence of defaults. The rightmost bin corresponds to clients that have a relatively high DSTI, 30,000 of them. On the left side of the plot, it can be seen that about 35,000 clients have relatively high income (w.r.t. debt). As a result of that, their default rates are lower.

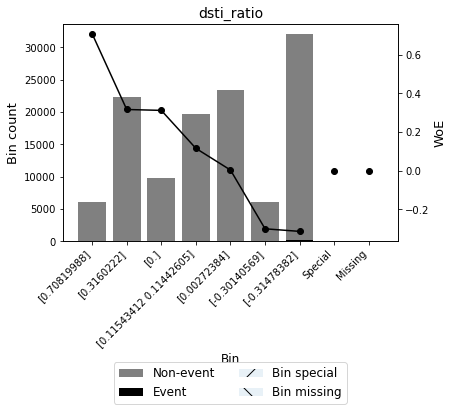


Figure 7: binning and WoE values of DSTI ratio, source: author

Although the LTV ratio has a much more direct relationship with LGD modeling, it is also reasonable to expect that clients that have more cash w.r.t. the loan amount to start with are less likely to default.

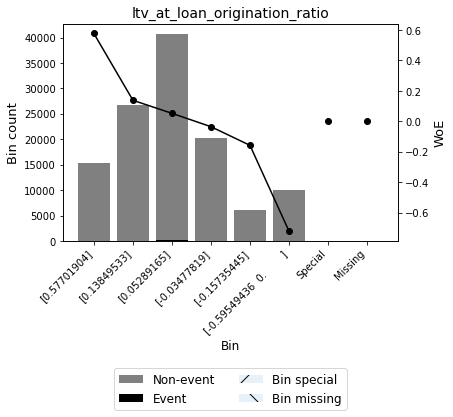


Figure 8: binning and WoE values of LTV ratio, source: aut hor

Upon defining the bins, a feature exclusion is performed by excluding any column that has IV lower than 0.05 or Gini lower than 0.1. This operation reduced the number of features from the initial 263 outputted from the DataGetter class to 120. These features are then passed to the multicollinearity removal procedure, where the maximum tolerated correlation between two variables is up to 0.5, as per Witzany (2017). Finally, the number of features in the shortlist is 40. They are summarized in the table below and sorted by Gini.

Table 12: feature shortlist obtained after data preprocessing

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **# of bins** | **IV** | **Gini** |
| behavioral\_score | 10 | 2.080 | 0.702 |
| debt\_summary\_2qs\_max\_amt | 2 | 0.779 | 0.432 |
| account\_balance\_2qs\_avg\_amt | 6 | 0.858 | 0.430 |
| days\_in\_deliquency\_6m\_avg\_count | 2 | 1.035 | 0.384 |
| interest\_paid\_to\_next\_installment\_6m\_min\_ratio | 9 | 0.504 | 0.377 |
| fee\_paid\_mtd\_amt | 6 | 0.416 | 0.325 |
| application\_score | 9 | 0.373 | 0.322 |
| collateral\_required\_amt | 8 | 0.359 | 0.315 |
| penalty\_interest\_paid\_mtd\_amt | 2 | 0.694 | 0.304 |
| interest\_paid\_6m\_max\_to\_next\_payment\_cat | 3 | 0.487 | 0.288 |
| product\_type\_cd | 4 | 0.346 | 0.288 |
| fee\_mtd\_to\_installment\_ratio | 6 | 0.297 | 0.285 |
| debit\_turnover\_mtd\_amt | 5 | 0.272 | 0.276 |
| brki\_installment\_amt | 3 | 0.318 | 0.265 |
| income\_to\_expense\_all\_applicants\_ratio | 10 | 0.214 | 0.249 |
| principal\_paid\_6m\_avg\_amt | 9 | 0.185 | 0.229 |
| paid\_to\_limit\_ratio | 7 | 0.223 | 0.214 |
| interest\_paid\_3m\_min\_amt | 7 | 0.217 | 0.205 |
| principal\_paid\_to\_outstanding\_6m\_max\_ratio | 7 | 0.129 | 0.202 |
| no\_feeflg | 2 | 0.214 | 0.193 |
| expense\_all\_aplicants\_to\_next\_installment\_ratio | 8 | 0.124 | 0.192 |
| age | 10 | 0.114 | 0.184 |
| main\_obj\_value\_amt | 5 | 0.151 | 0.183 |
| od\_limit\_utilization\_amt | 2 | 0.234 | 0.181 |
| interest\_paid\_6m\_max\_amt | 5 | 0.182 | 0.179 |
| client\_capital\_to\_paid\_mtd\_ratio | 5 | 0.166 | 0.172 |
| since\_live\_acc\_opening\_mths\_count | 5 | 0.120 | 0.172 |
| client\_income\_amt | 8 | 0.104 | 0.161 |
| principal\_paid\_6m\_max\_amt | 8 | 0.084 | 0.159 |
| ltv\_at\_loan\_origination\_ratio | 6 | 0.139 | 0.156 |
| dsti\_ratio | 7 | 0.080 | 0.154 |
| main\_applicant\_expense\_amt | 6 | 0.080 | 0.150 |
| installments\_count | 6 | 0.086 | 0.148 |
| collateral\_value\_to\_outstanding\_ratio | 6 | 0.067 | 0.134 |
| fixation\_to\_installments\_ratio | 6 | 0.068 | 0.132 |
| all\_applicants\_expense\_amt | 7 | 0.082 | 0.128 |
| principal\_paid\_to\_outstanding\_3m\_avg\_ratio | 8 | 0.050 | 0.123 |
| debt\_summary\_mtd\_amt | 1 | 0.053 | 0.111 |
| fixation\_period\_mths\_count | 3 | 0.064 | 0.109 |
| limit\_pct | 3 | 0.056 | 0.105 |

Source: author

Upon running the forward feature selection, an LR model is estimated as summarized in the table below.

Table 13: logistic regression summary

| **Variable** | **Coefficient** | **Std. error** | **[0.025]** | **[0.975]** |
| --- | --- | --- | --- | --- |
| fee\_paid\_mtd\_amt | -0.458 | (0.015)\*\*\* | -0.486 | -0.429 |
| principal\_paid\_6m\_max\_amt | 0.015 | -0.03 | -0.044 | 0.074 |
| interest\_paid\_6m\_max\_amt | -1.012 | (0.025)\*\*\* | -1.061 | -0.962 |
| days\_in\_deliquency\_6m\_avg\_count | -0.673 | (0.015)\*\*\* | -0.703 | -0.643 |
| installments\_count | -0.877 | (0.031)\*\*\* | -0.937 | -0.816 |
| age | -0.580 | (0.026)\*\*\* | -0.630 | -0.529 |
| main\_applicant\_expense\_amt | -1.436 | (0.036)\*\*\* | -1.507 | -1.365 |
| limit\_pct | 1.741 | (0.050)\*\*\* | 1.643 | 1.839 |
| main\_obj\_value\_amt | -1.652 | (0.030)\*\*\* | -1.712 | -1.593 |
| account\_balance\_2qs\_avg\_amt | -0.553 | (0.012)\*\*\* | -0.575 | -0.530 |
| product\_type\_cd | -0.384 | (0.020)\*\*\* | -0.423 | -0.344 |
| paid\_to\_limit\_ratio | -0.446 | (0.019)\*\*\* | -0.482 | -0.410 |
| collateral\_value\_to\_outstanding\_ratio | 0.073 | (0.040)\* | -0.005 | 0.151 |
| principal\_paid\_to\_outstanding\_6m\_max\_ratio | -0.454 | (0.028)\*\*\* | -0.509 | -0.399 |
| ltv\_at\_loan\_origination\_ratio | -1.414 | (0.039)\*\*\* | -1.491 | -1.337 |
| income\_to\_expense\_all\_applicants\_ratio | -0.184 | (0.020)\*\*\* | -0.223 | -0.145 |
| expense\_all\_aplicants\_to\_next\_installment\_ratio | -0.291 | (0.026)\*\*\* | -0.341 | -0.241 |
| interest\_paid\_to\_next\_installment\_6m\_min\_ratio | -0.652 | (0.015)\*\*\* | -0.682 | -0.622 |
| behavioral\_score | -0.385 | (0.009)\*\*\* | -0.404 | -0.367 |
| application\_score | -0.057 | (0.015)\*\*\* | -0.087 | -0.027 |

\*\*\* represents significance at 0.01, \*\* at 0.01 and \* at 0.05, source: author

The model performance decreases in the case of the test and more notably of the OOT sample.

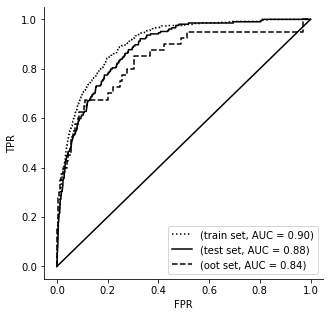


Figure: ROC curves of logistic regression model from batch (1), source: author

#### Overview of Results

The batch (1) models are estimated using the full sample of train data (no undersampling), WoE-transformed features as well as a multicollinearity removal have the specifications summarized in the following table. The LR, ANN and boosting models exhibit solid performance with a gradual decrease in AUC across different samples. KNN and bagging overfit the train data. Whilst KNN exhibits weaker performance on the test and OOT sample, bagging is almost on par with LR, ANN and RF. SVM also has a tendency to overfit, although not perfectly. Its test and OOT performance is, however, alike the one of KNN. Overall, it can be concluded that all models estimated under these settings have a solid performance.

Table 14: optimal hyperparameters of batch (1) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | n.a. | 0.90 | 0.85 | 0.86 |
| ANN | Activation: sigmoid  Hidden layers: [7]  Learning rate: constant  Solver: adam | 0.90 | 0.85 | 0.86 |
| KNN | # of neighbors: 10  Weights: uniform  Distance: Manhattan | 1.00 | 0.78 | 0.77 |
| SVM | C: 0.25  Polynomial degree: 0.25  Kernel: linear | 0.93 | 0.82 | 0.79 |
| Bagging | # of estimators: 50 | 1.00 | 0.83 | 0.84 |
| RF | Maximum tree depth: 1  Minimum samples per leaf: 10  # of estimators: 10 | 0.91 | 0.86 | 0.85 |
| Boosting | # of estimators: 10 | 0.90 | 0.85 | 0.85 |

Source: author

The batch (2) models are estimated using the full sample of train data (no undersampling), WoE-transformed features whilst multicollinearity is not removed prior to model estimation. The LR model is estimated using the elastic net approach, where the L1 ratio of 0.5 points that both Ridge and Lasso are used. The result of that estimation is similar to the one above. The ANN has a slightly lower performance than the previously estimated one. In addition, the algorithm retained the same simple structure of the neural network – one layer with seven neurons. It appears that the dataset with correlated features further complicates the performance of KNN and SVM. A milder drop is noticed in the case of RF and bagging as well. Boosting is largely consistent with the previous estimate.

Table 15: optimal hyperparameters of batch (2) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | L1 ratio: 0.5 | 0.91 | 0.88 | 0.83 |
| ANN | Activation: sigmoid  Hidden layers: [7]  Learning rate: constant  Solver: adam | 0.88 | 0.87 | 0.82 |
| KNN | # of neighbors: 10  Weights: uniform  Distance: Manhattan | 0.99 | 0.62 | 0.60 |
| SVM | C: 0.25  Polynomial degree: 1  Kernel: Polynomial | 0.56 | 0.55 | 0.47 |
| Bagging | # of estimators: 10 | 1.00 | 0.80 | 0.78 |
| RF | Maximum tree depth: 1  Minimum samples per leaf: 10  # of estimators: 10 | 0.80 | 0.78 | 0.78 |
| Boosting | # of estimators: 10 | 0.88 | 0.85 | 0.81 |

Source: author

The batch (3) models are estimated using the full sample of train data (no undersampling), dummy-transformed features and a multicollinearity removal. No feature selection is performed. Again, the LR estimation’s performance is overlapping with the previous two models. Along with boosting, LR is the only model that maintains decent performance as measured by AUC. The remaining estimates – ANN, KNN, SVM, bagging and RF do not perform well.

Table 16: optimal hyperparameters of batch (3) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | n.a. | 0.91 | 0.88 | 0.82 |
| ANN | Activation: sigmoid  Hidden layers: [7]  Learning rate: constant  Solver: adam | 0.57 | 0.60 | 0.52 |
| KNN | # of neighbors: 10  Weights: uniform  Distance: Euclidean | 0.99 | 0.59 | 0.59 |
| SVM | C: 0.25  Polynomial degree: 1  Kernel: linear | 0.60 | 0.55 | 0.60 |
| Bagging | # of estimators: 50 | 1.00 | 0.69 | 0.71 |
| RF | Maximum tree depth: 1  Minimum samples per leaf: 10  # of estimators: 10 | 0.77 | 0.77 | 0.69 |
| Boosting | # of estimators: 20 | 0.89 | 0.87 | 0.84 |

Source: author

The batch (4) models have the same specifications as batch (1) with the only difference being that undersampling is used in the case of batch (4). All models have test performances between 0.86 and 0.88. OOT is slightly lower and in the range of 0.83 to 0.86. Overall, it is concluded that no single model stands out from the rest. What makes batch (4) different from batch (1) is that KNN and SVM perform substantially better. Hence, it can be concluded that class balance is a necessary prerequisite to achieve good performance of these models.

Table 16: optimal hyperparameters of batch (4) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | n.a. | 0.91 | 0.87 | 0.86 |
| ANN | Activation: tahn  [200, 100, 100]  Learning rate: constant  Solver: adam | 0.91 | 0.87 | 0.85 |
| KNN | # of neighbors: 30  Weights: distance-based  Distance: Euclidean | 1.00 | 0.86 | 0.83 |
| SVM | C: 0.5  Polynomial degree: 1  Kernel: polynomial | 0.90 | 0.87 | 0.85 |
| Bagging | # of estimators: 100 | 1.00 | 0.86 | 0.85 |
| RF | Maximum tree depth: 10  Minimum samples per leaf: 10  # of estimators: 30 | 0.94 | 0.88 | 0.84 |
| Boosting | # of estimators: 100 | 0.92 | 0.86 | 0.84 |

Source: author

The batch (5) models have the same specifications as the batch (2) with the only difference being that undersampling is used in the case of batch (5). What comes to attention is that class imbalance is a much larger issue for SVM and KNN than multicollinearity. Although the two models do not perform as well as the remaining ones, their performance is much better when compared to batch (2). Nonetheless, it can still be concluded that multicollinearity poses an issue due to the fact that these 2 estimates still do not outperform the SVM and KNN from batch (4). The remaining models have a solid performance.

Table 17: optimal hyperparameters of batch (5) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | L1 ratio: 0.1 | 0.93 | 0.87 | 0.84 |
| ANN | Activation: logistic  [200, 100, 100]  Learning rate: constant  Solver: adam | 0.91 | 0.88 | 0.85 |
| KNN | # of neighbors: 10  Weights: uniform  Distance: Euclidean | 0.92 | 0.84 | 0.77 |
| SVM | C: 1  Polynomial degree: 1  Kernel: rbf | 0.95 | 0.86 | 0.79 |
| Bagging | # of estimators: 150 | 1.00 | 0.87 | 0.86 |
| RF | Maximum tree depth: 10  Minimum samples per leaf: 10  # of estimators:  20 | 0.95 | 0.87 | 0.85 |
| Boosting | # of estimators: 100 | 0.94 | 0.86 | 0.82 |

Source: author

The batch (6) models have the same specifications as the batch (3) with the only difference being that undersampling is used in the case of batch (6). At first, an attempt was made to remove multicollinearity at a 0.5 threshold. However, at 0.5 there still are perfect correlations between various dummy variables. This is driven by the fact that the undersampled dataset is relatively small. In order to remedy this problem, the optimal multicollinearity tolerance level is found to be at 0.2. The LR model improved slightly. Nevertheless, it is not strictly the case that the underbalanced datasets contributed better performance of the LR model. When compared to batch (3), it can be seen that the performance of all the remaining models improved significantly.

Table 19: optimal hyperparameters of batch (6) models

| **Model type** | **Hyperparameter specification** | **AUC train** | **AUC test** | **AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | n.a. | 0.91 | 0.87 | 0.85 |
| ANN | Activation: tahn  [100, 200, 100]  Learning rate: constant  Solver: adam | 0.90 | 0.87 | 0.83 |
| KNN | # of neighbors: 10  Weights: distance-based  Distance: Euclidean | 1.00 | 0.86 | 0.79 |
| SVM | C: 0.75  Polynomial degree: 2  Kernel: polynomial | 0.91 | 0.88 | 0.85 |
| Bagging | # of estimators: 150 | 1.00 | 0.87 | 0.83 |
| RF | Maximum tree depth: 15  Minimum samples per leaf: 10  # of estimators: 50 | 0.95 | 0.88 | 0.84 |
| Boosting | # of estimators: 50 | 0.93 | 0.86 | 0.85 |

Source: author

### Comparison & Answer to Researched Questions

The following table summarizes the average performance metrics of the models estimated above. The LR and boosting methods are the most robust approaches when it comes to the tested data preprocessing approaches.

Table 20: average AUCs on train, test and OOT samples

| **Model type** | **Average AUC train** | **Average AUC test** | **Average AUC OOT** |
| --- | --- | --- | --- |
| LR | 0.91 | 0.87 | 0.84 |
| ANN | 0.85 | 0.82 | 0.79 |
| KNN | 0.98 | 0.76 | 0.73 |
| SVM | 0.80 | 0.76 | 0.78 |
| Bagging | 1.00 | 0.82 | 0.81 |
| RF | 0.89 | 0.84 | 0.81 |
| Boosting | 0.91 | 0.86 | 0.84 |

Source: author

The LR performs most consistently across all 6 batches. Should batch 3 be taken out of the picture, it can be seen that most of the other models are closer to the LR one in terms of AUC. This is shown in the following table. It is observed that, on average, LR and ANN provide the best performance. These are closely followed by tree methods and SVM. KNN is the least robust to different types of data preprocessing.

Table 21: average AUCs on train, test and OOT samples without batch 3 estimates

| **Model type** | **Average AUC train** | **Average AUC test** | **Average AUC OOT** |
| --- | --- | --- | --- |
| LR | 0.91 | 0.87 | 0.85 |
| ANN | 0.90 | 0.87 | 0.84 |
| KNN | 0.98 | 0.79 | 0.75 |
| SVM | 0.85 | 0.80 | 0.82 |
| Bagging | 1.00 | 0.85 | 0.83 |
| RF | 0.91 | 0.85 | 0.83 |
| Boosting | 0.91 | 0.86 | 0.83 |

Source: author

The following table summarizes the best performance of each model. The best performing model estimation is identified as the one that has the highest OOT AUC score.

Table 22: best AUCs on train, test and OOT samples

| **Model type** | **Batch number** | **Best\* AUC train** | **Best\* AUC test** | **Best AUC OOT** |
| --- | --- | --- | --- | --- |
| LR | 4 | 0.93 | 0.87 | 0.86 |
| ANN | 1 | 0.90 | 0.85 | 0.86 |
| KNN | 4 | 1.00 | 0.86 | 0.83 |
| SVM | 6 | 0.91 | 0.88 | 0.85 |
| Bagging | 5 | 1.00 | 0.87 | 0.86 |
| RF | 5 | 0.95 | 0.87 | 0.85 |
| Boosting | 6 | 0.93 | 0.86 | 0.85 |

\* means that the reported AUC is not necessarily the best, it is the AUC that corresponds to the OOT AUC which is the key to determination of the best model by this metric, source: author

In connection to the Literature review in section 1.2, the following comparisons are made. Since it is not suggested that the modern ML techniques outperform the traditional LR, the results are inconsistent with Shi et al. (2022) who report otherwise. Since it is found that SVM and KNN perform substantially better on class-balanced datasets, the result of this research does not fully corroborate the claim of Lessmann et al. (2022) that class balancing is not important for algorithm comparisons. The result of this research can only partially support that claim in the sense that SVM and KNN can be seen as more sensitive to class imbalance and therefore less preferred. Conversely, the fact that Altman et al. (2017) use only undersampled data, it can be concluded that potential class imbalance sensitivities were masked in that study. Since Altman et al. (2017) find that SVM underperforms on undersampled data whilst this is not the case in this research, a difference in data preprocessing should be highlighted once again. It is noted that data-transformative steps were not introduced in that research and that the authors are aware of its potential impact. The fact that SVM underperforms on non-undersampled data cannot be directly checked against Altman et al. (2017). Nonetheless, since the SVM there underperforms on the undersampled data as it is, it should be expected that the usage of imbalanced data would bring no better results. The results obtained here are consistent with Ernst & Young (2022), who find that the predictive power of LR is around the same as of the other methods. The reported data-transformative steps they use are similar to the ones followed in this thesis.

To answer the researched questions listed in section 3.3.1:

1. ML modeling techniques do not outperform the traditionally used LR in the case of the Czech home loans data.
2. Certain ML modeling approaches are sensitive to data preprocessing types. LR is the most robust in this sense.
3. SVM and KNN are the most sensitive to class imbalance.

In addition to the conclusions made about model performance, it is noted that the time needed to train these models varies highly. Whilst for a logistic regression to be estimated it takes up to 10 minutes, the ANN model requires 1-2 hours. The most demanding algorithm is the SVM. There the estimation times take anywhere between 3 and 8 hours for the data of this size.

# Final Remarks

In this thesis, a commentary on the current status of the use of ML in credit risk modeling and a reference to the relevant literature is made. It was found that the use of ML is currently not at a mature stage. Rather, financial institutions are still exploring the optimal ways for its use and concerns about model explainability, necessary expertise and their complexity persists.

Turning to the primary goal of this work, which is to determine whether the industry-standard logistic regression model is outperformed by newer approaches (ANN, KNN, SVM and tree ensembles), models are estimated using Czech home loans data. In addition, several data preprocessing approaches are tested in order to strengthen the reliability of the results. In the case of a Czech home loans data sample, it is found that they do not. This result is inconsistent with the findings of the cited academic publications throughout the thesis. On the other hand, it is compliant with the findings of an industry practitioner located in the Czech Republic. Until a substantial improvement in performance cannot be guaranteed by non-traditional modeling approaches, financial institutions should be cautious about having them as their first choice. Rather, they could be used to support the traditional logistic regression models by, for example, challenging them during a validation.

# References

Altman, E. (1968). FINANCIAL RATIOS, DISCRIMINANT ANALYSIS AND THE PREDICTION OF CORPORATE BANKRUPTCY. *The Journal of Finance*. *23*(4), 589–609. doi: 10.1111/j.1540-6261.1968.tb00843.x

Council of the EU. (May 21, 2024). *Artificial intelligence (AI) act: Council gives final green light to the first worldwide rules on AI* [Press release].

Barboza, F., Kimura, H. & Altman, E. (2017). Machine learning models and bankruptcy prediction. *Expert Systems with Applications*, *83*, 405–417. doi: 10.1016/j.eswa.2017.04.006

Barfield, R., Barling, M., Beach, C., Cadiou, C., De la Mora, F., De Vry, I., Du Plessis, S., Everett, A., Fell, P., Forrest, N., Gealy, S., Gosling, T., Kibble, R., Mars, M., Matten, C., McNab, D., Mitra, P., Pontiki, A., Rieger, C., Stomeo, A. & Worsnip, J. (2011). *A Practitioner’s Guide to BASEL III AND BEYOND*. Thompson Reuters (Professional) UK Limited.

European Banking Authority. (2017). *Guidelines: on PD estimation, LGD estimation and the treatment of defaulted exposures* (EBA/GL/2017/16).

European Banking Authority (2021). *EBA discussion paper on machine learning for IRB models* (EBA/DP/2021/04).

European Banking Authority. (2023). *Machine learning for IRB models: follow-up report from the consultation on the discussion paper on machine learning for IRB models* (EBA/REP/2023/28).

European Parliament. (2023). *EU AI Act: first regulation on artificial intelligence*. Retrieved April 1, 2024, from https://www.europarl.europa.eu/topics/en/article/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence

Engelmann, B., Hayden, E. & Tasche, D. (2003). Measuring the Discriminative Power of Rating Systems. *Deutche Bundesbank, Frankfurt a. M., Discussion Paper Series 2, 2003(01)*. Working paper.

Ernst & Young. (2022). *2EK614 Applied Operational Research in Business Consulting: Credit risk – Predictive Modelling*. Prague University of Economics and Business.

Lemaitre, G., Nogueira, F. & Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. Journal of Machine Learning Research. doi: 10.48550/arXiv.1609.06570

James, G., Witten, D., Hastie, T., Tibshirani, R. & Taylor, J. (2023). An Introduction to Statistical Learning: with Applications in Python. Springer Cham. doi: 10.1007/978-3-031-38747-0

Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of Research. *European Journal of Operational Research*, *247*(1), 124–136. doi: 10.1016/j.ejor.2015.05.030

Pech, C. (2016). *CS109 Intro to Probability for Computer Scientists: Handout #40*. Stanford University.

Pedragosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*. doi: 10.48550/arXiv.1201.0490

*Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL LAYING DOWN HARMONISED RULES ON ARTIFICIAL INTELLIGENCE (ARTIFICIAL INTELLIGENCE ACT) AND AMENDING CERTAIN UNION LEGISLATIVE ACTS* *2021*. (EU)

*Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) 2016*. (EU)

Navas-Palencia G. (2022). Optimal binning: mathematical programming formulation. *ArXiv*, *2001.08025*. doi: 10.48550/arXiv.2001.08025

Shi, S., Tse, R., Luo, W., D’Addona, S. & Pau, G. (2022). Machine learning-driven credit risk: a systemic review. *Neural Computing and Applications*, *34*, 14327–14339. doi: 10.1007/s00521-022-07472-2

*Video Index* (n.d.). Retrieved May 25, 2024, from https://statquest.org/video-index/

Witzany, J. (2017). *Credit Risk Management*. Springer Cham. doi: 10.1007/978-3-319-49800-3

# Graphics and Tables

Graphics

[Figure 1: sigmoid function, source: author 17](#_Toc167471742)

[Figure 2: structure of a neuron, source: author, Witzany (2017) 19](#_Toc167471743)

[Figure 3: neural network with multiple neurons in hidden layer, source: author, Witzany (2017) 20](#_Toc167471744)

[Figure 4: binning and WoE values of behavioral score, source: author 44](#_Toc167471745)

[Figure 5: binning and WoE values of DSTI ratio, source: author 45](#_Toc167471746)

[Figure 6: binning and WoE values of LTV ratio, source: aut hor 45](#_Toc167471747)

Tables

[Table 1. Literature search results 12](#_Toc167471748)

[Table 2: confusion matrix 29](#_Toc167471749)

[Table 3. Overview of modeling dataset 35](#_Toc167471750)

[Table 4: list of defined Python classes 36](#_Toc167471751)

[Table 5: estimated models 40](#_Toc167471752)

[Table 6: hyperparameter grid for ANN model 41](#_Toc167471753)

[Table 7: hyperparameter grid for KNN model 42](#_Toc167471754)

[Table 8: hyperparameter grid for SVM model 42](#_Toc167471755)

[Table 9: hyperparameter grid for bagging model 42](#_Toc167471756)

[Table 10: hyperparameter grid for RF model 42](#_Toc167471757)

[Table 11: hyperparameter grid for boosting model 42](#_Toc167471758)

[Table 12: feature shortlist obtained after data preprocessing 46](#_Toc167471759)

[Table 13: logistic regression summary 47](#_Toc167471760)

[Table 14: optimal hyperparameters of batch (1) models 48](#_Toc167471761)

[Table 15: optimal hyperparameters of batch (2) models 49](#_Toc167471762)

[Table 16: optimal hyperparameters of batch (3) models 50](#_Toc167471763)

[Table 16: optimal hyperparameters of batch (4) models 51](#_Toc167471764)

[Table 17: optimal hyperparameters of batch (5) models 51](#_Toc167471765)

[Table 19: optimal hyperparameters of batch (6) models 52](#_Toc167471766)

[Table 20: average AUCs on train, test and OOT samples 53](#_Toc167471767)

[Table 21: average AUCs on train, test and OOT samples without batch 3 estimates 54](#_Toc167471768)

[Table 22: best AUCs on train, test and OOT samples 54](#_Toc167471769)

[Table 21: longlist of Czech Home Loans portfolio dataset 60](#_Toc167471770)

Appendices

Appendix A – Longlist

Appendix A – Longlist

Table 21: longlist of Czech Home Loans portfolio dataset

| **variable\_name** | **variable\_description** |
| --- | --- |
| obs\_yyyymm | Observation date |
| limit\_amt | Limit of a loan |
| limit\_no-purpose\_amt | Supplementary part limit of a loan |
| drawn\_amt | Drawn amount up to the date |
| drawn\_no-purpose\_amt | Drawn amount of the supplementary part up to date |
| participation\_amt | Own capital of a client on the loan |
| refinancing\_flg | Flag if a loan was refinanced |
| mths\_since\_sign\_count | Months since contract sign date |
| next\_repayment\_amt | Next instalment amount |
| principal\_amt | Principal balance |
| principal\_after\_maturity\_amt | Principal after maturity |
| principal\_before\_maturity\_amt | Principal before maturity |
| principal\_paid\_mtd\_amt | Paid Principal current month |
| interest\_after\_maturity\_amt | Interest after maturity |
| interest\_paid\_mtd\_amt | Paid interest current month |
| penalty\_interest\_past\_due\_amt | Penalty interests past due in current month |
| penalty\_interest\_paid\_mtd\_amt | Penalty interests paid in current month |
| fee\_after\_maturity\_amt | Fee after maturity |
| fee\_paid\_mtd\_amt | Paid fee up current month |
| reservation\_fee\_accued\_amt | Reservation fee accrued since last repayment |
| outstanding\_amt | Outstanding amount |
| paid\_amt | Amount paid since sign date |
| deliquency\_status\_flg | Delinquency status |
| deliquency\_category\_cd | Deliquency status |
| days\_in\_deliquency\_count | Delinquency days |
| days\_in\_deliquency\_count\_2 | Delinquency days according to NDoD |
| principal\_past\_due\_3m\_max\_amt | Maximal principal past due within past 3 months |
| principal\_paid\_3m\_max\_amt | Maximal principle paid with past 3 months |
| interest\_past\_due\_3m\_max\_amt | Maximal interests past due within past 3 months |
| interest\_paid\_3m\_max\_amt | Maximal interests paid with past 3 months |
| days\_in\_deliquency\_3m\_max\_count | Maximum of delinquency days within past 3 months |
| days\_in\_deliquency\_3m\_max\_count\_2 | Maximum of delinquency days according to NDoD within past 3 months |
| principal\_past\_due\_3m\_min\_amt | Minimal principal past due within past 3 months |
| principal\_paid\_3m\_min\_amt | Minimal principle paid with past 3 months |
| interest\_past\_due\_3m\_min\_amt | Minimal interests past due within past 3 months |
| interest\_paid\_3m\_min\_amt | Minimal interests paid within past 3 months |
| days\_in\_deliquency\_3m\_min\_count | Minimum of delinquency days within past 3 months |
| days\_in\_deliquency\_3m\_min\_count\_2 | Minimum of delinquency days according to NDoD within past 3 months |
| principal\_past\_due\_3m\_avg\_amt | Average principal past due within past 3 months |
| principal\_paid\_3m\_avg\_amt | Average principle paid with past 3 months |
| interest\_past\_due\_3m\_avg\_amt | Average interests past due within past 3 months |
| interest\_paid\_3m\_avg\_amt | Average interests paid with past 3 months |
| days\_in\_deliquency\_3m\_avg\_count | Average of delinquency days within past 3 months |
| days\_in\_deliquency\_3m\_avg\_count\_2 | Average of delinquency days according to NDoD within past 3 months |
| principal\_past\_due\_6m\_max\_amt | Maximal principal past due within past 6 months |
| principal\_paid\_6m\_max\_amt | Maximal principle paid with past 6 months |
| interest\_past\_due\_6m\_max\_amt | Maximal interests past due within past 6 months |
| interest\_paid\_6m\_max\_amt | Maximal interests paid within past 6 months |
| days\_in\_deliquency\_6m\_max\_count | Maximum of delinquency days within past 6 months |
| days\_in\_deliquency\_6m\_max\_count\_2 | Maximum of delinquency days according to NDoD within past 6 months |
| principal\_past\_due\_6m\_min\_amt | Minimal principal past due within past 6 months |
| principal\_paid\_6m\_min\_amt | Minimal principle paid with past 6 months |
| interest\_past\_due\_6m\_min\_amt | Minimal interests past due within past 6 months |
| interest\_paid\_6m\_min\_amt | Minimal interests paid within past 6 months |
| days\_in\_deliquency\_6m\_min\_count | Minimum of delinquency days within past 6 months |
| days\_in\_deliquency\_6m\_min\_count\_2 | Minimum of delinquency days according to NDoD within past 6 months |
| principal\_past\_due\_6m\_avg\_amt | Average principal past due within past 6 months |
| principal\_paid\_6m\_avg\_amt | Average principle paid with past 6 months |
| interest\_past\_due\_6m\_avg\_amt | Average interests past due within past 6 months |
| interest\_paid\_6m\_avg\_amt | Average interests paid with past 6 months |
| days\_in\_deliquency\_6m\_avg\_count | Average of delinquency days within past 6 months |
| days\_in\_deliquency\_6m\_avg\_count\_2 | Average of delinquency days according to NDoD within past 6 months |
| participants\_count | Number of participants of a loan |
| installments\_count | Number of instalments of a loan |
| collateral\_registered\_amt | Total collateral value registered at the cadastre |
| collateral\_required\_amt | Total collateral value required for a loan |
| non-purpose\_existence\_flg | Flag if a loan includes also a non-purpose part besides the purpose one |
| main\_borrower\_bool | Main borrower indicator |
| cz\_resident\_flg | Flag if a client is a CZ resident |
| marital\_status\_cd | Marital status of a client |
| adult\_person\_count | Number of adults in the client's household |
| children\_below\_6\_count | Number of children below 6 years in the client's household |
| children\_below\_10\_count | Number of children below 10 years but older than 6 years in the client's household |
| children\_below\_15\_count | Number of children below 15 years but older than 10 years in the client's household |
| children\_below\_26\_count | Number of children below 26 years but older than 15 years in the client's household |
| client\_income\_amt | Client's income |
| all\_applicants\_expense\_amt | Aggregated expense of all the applicants |
| all\_applicants\_income\_amt | Sum of income of all the applicants |
| age | Age of the decisive applicant |
| main\_applicant\_expense\_amt | Expense of the decisive applicant |
| main\_applicant\_income\_amt | Income of the decisive applicant |
| loan\_length\_mths\_count | Length of a loan in months |
| fixation\_period\_mths\_count | Length of a current fixation period in months |
| current\_fixation\_sequential\_order\_int | Sequence number of a current interest rate assignment (fixation) |
| legal\_burden\_bool | Legal burden on a loan - distrait or sue |
| life\_insurance\_block\_bool | Flag if a life insurance payment indemnification is blocked in favour of the bank |
| standard\_approval\_exception\_bool | Exception from standard approval |
| limit\_pct | Limit of a loan |
| no\_payroll\_bool | Self-certified income flag |
| ownership\_share\_in\_coop\_bool | Share in a flat housing co-operative flag |
| pre-mortgage\_use\_bool | Pre-mortgage loan used |
| late\_purpose\_bool | Purpose will be defined later |
| supplementary\_loan\_bool | Loan with supplementary part |
| no\_fee\_bool | Fee included in the interest rate |
| non-resident\_bool | For non-residents |
| collateral\_insurance\_flg | Flag if the collateral is insured |
| collateral\_owners\_count | Number of owners of the main collateral |
| main\_object\_as\_coll\_flg | Flag if main object is collateral |
| main\_collateral\_value\_amt | Accepted value of a collateral |
| collateral\_value\_amt | Accepted value of all collaterals |
| collateral\_count | Number of collaterals (properties) |
| commercial\_real\_estate\_coll\_flg | Flag if a commercial real estate is a collateral of the loan |
| main\_object\_insurance\_flg | Flag if the main object is insured |
| main\_object\_owners\_count | Number of owners of the main object |
| main\_object\_as\_coll\_flg\_2 | Flag if main object is collateral |
| main\_obj\_value\_amt | Accepted value of a loan object |
| brki\_installment\_amt | Installment amount |
| pre-approval\_flg | Preapproved mortgage loan flag |
| live\_account\_mtd\_flg | Flag of at least one live account in actual month |
| live\_account\_prev\_mth\_flg | Flag of at least one live account in past month |
| account\_balance\_1q\_avg\_amt | Average of monthly summary balances during past 1 quarter |
| account\_balance\_2qs\_avg\_amt | Average of monthly summary balances during past 2 quarters |
| credit\_turnover\_mtd\_amt | Summary credit turnover in actual month |
| credit\_turnover\_prev\_mth\_amt | Summary credit turnover in past month |
| credit\_turnover\_1q\_avg\_amt | Average of monthly summary credit turnovers during past 1 quarter |
| credit\_turnover\_2qs\_avg\_amt | Average of monthly summary credit turnovers during past 2 quarters |
| debit\_turnover\_mtd\_amt | Summary debit turnover in actual month |
| debit\_turnover\_prev\_mth\_amt | Summary debit turnover in past month |
| debit\_turnover\_1q\_avg\_amt | Average of monthly summary debit turnovers during past 1 quarter |
| debit\_turnover\_2qs\_avg\_amt | Average of monthly summary debit turnovers during past 2 quarters |
| debt\_summary\_mtd\_amt | Summary debt in actual month |
| debt\_summary\_1q\_max\_amt | Maximum of monthly summary debts during past 1 quarter |
| debt\_summary\_2qs\_max\_amt | Maximum of monthly summary debts during past 2 quarters |
| debt\_mtd\_count | Number of debts in actual month |
| debt\_1q\_max\_count | Maximum of monthly count of debts during past 1 quarter |
| debt\_2qs\_max\_count | Maximum of monthly count of debts during past 2 quarters |
| since\_live\_acc\_opening\_mths\_count | Maximal number of months since any actual live account opening |
| since\_any\_acc\_opening\_mths\_count | Maximal number of months since any account opening |
| cl\_deliquency\_mtd\_status | Actual delinquency status |
| cl\_deliquency\_prev\_1q\_status | Maximal delinquency status during past 1 quarter |
| cl\_deliquency\_prev\_2qs\_status | Maximal delinquency status during past 2 quarters |
| cl\_debt\_1q\_max\_amt | Maximal debt during past 1 quarter |
| cl\_debt\_2qs\_max\_amt | Maximal debt during past 2 quarters |
| cl\_mths\_in\_deliquency\_prev\_1q\_count | Number of months account was delinquent during past 1 quarter |
| cl\_mths\_in\_deliquency\_prev\_2qs\_count | Number of months account was delinquent during past 2 quarters |
| cl\_limit\_utilization\_amt | Actual loan utilisation |
| cl\_limit\_amt | Actual loan limit |
| cl\_mths\_on\_book\_count | Months on book for CL |
| od\_deliquency\_mtd\_status | Actual delinquency status |
| od\_deliquency\_prev\_1q\_max\_status | Maximal delinquency status during past 1 quarter |
| od\_deliquency\_prev\_2qs\_max\_status | Maximal delinquency status during past 2 quarters |
| od\_debt\_prev\_1q\_max\_amt | Maximal debt during past 1 quarter |
| od\_debt\_prev\_2qs\_max\_amt | Maximal debt during past 2 quarters |
| od\_mths\_in\_deliquency\_prev\_1q\_count | Number of months account was delinquent during past 1 quarter |
| od\_mths\_in\_deliquency\_prev\_2qs\_count | Number of months account was delinquent during past 2 quarters |
| od\_limit\_utilization\_amt | Actual loan utilisation |
| od\_limit\_amt | Actual loan limit |
| ovd\_mths\_on\_book\_count | Months on book for ovd |
| cc\_deliquency\_mtd\_status | Actual delinquency status |
| cc\_deliquency\_prev\_1q\_max\_status | Maximal delinquency status during past 1 quarter |
| cc\_deliquency\_prev\_2qs\_max\_status | Maximal delinquency status during past 2 quarters |
| cc\_debt\_prev\_1q\_max\_amt | Maximal debt during past 1 quarter |
| cc\_debt\_prev\_2qs\_max\_amt | Maximal debt during past 2 quarters |
| cc\_mths\_in\_deliquency\_prev\_1q\_count | Number of months account was delinquent during past 1 quarter |
| cc\_mths\_in\_deliquency\_prev\_2qs\_count | Number of months account was delinquent during past 2 quarters |
| cc\_limit\_utilization\_amt | Actual loan utilisation |
| cc\_limit\_amt | Actual loan limit |
| cc\_mths\_on\_book\_count | Months on book for CC |
| live\_saving\_account\_mtd\_flg | At least one account is live in actual month - saving account |
| live\_saving\_account\_prev\_mth\_flg | At least one account is live in past month - saving account |
| saving\_account\_balance\_1q\_avg\_amt | Summary balance during past 1 quarter - saving account |
| saving\_account\_balance\_2qs\_avg\_amt | Summary balance during past 2 quarters - saving account |
| since\_live\_saving\_acc\_opening\_mths\_count | Maximal number of months since any actual live account opening - saving account |
| since\_any\_saving\_acc\_opening\_mths\_count | Minimal number of months since any actual live account opening - saving account |
| retail\_behavioral\_score | Consumer finance behavioural score from the previous month |
| dsti\_ratio | Debt service to income ratio |
| product\_type\_cd | Product type |
| mths\_to\_installments\_ratio | Months since contract sign date to total number of instalments |
| fixation\_to\_installments\_ratio | Length of current fixation period to total number of instalments |
| principal\_balance\_to\_limit\_ratio | Principal balance to Limit |
| principal\_after\_maturity\_to\_limit\_ratio | Principal after maturity to Limit |
| principal\_before\_maturity\_to\_limit\_ratio | Principal balance before maturity to Limit |
| outstanding\_to\_limit\_ratio | Outstanding amount to Limit |
| paid\_to\_limit\_ratio | Total paid up to date to Limit |
| drawn\_to\_limit\_ratio | Drawn amount up to date to Limit |
| client\_capital\_to\_loan\_ratio | Own capital of a client on the loan to Limit |
| collateral\_value\_to\_limit\_ratio | Value of collaterals to Limit |
| principal\_after\_maturity\_to\_limit\_3m\_max\_ratio | Maximal principal after maturity within past 3 months to Limit |
| principal\_after\_maturity\_to\_limit\_3m\_min\_ratio | Minimal principal after maturity within past 3 months to Limit |
| principal\_after\_maturity\_to\_limit\_3m\_avg\_ratio | Average principal after maturity within past 3 months to Limit |
| principal\_after\_maturity\_to\_limit\_6m\_max\_ratio | Maximal principal after maturity within past 6 months to Limit |
| principal\_after\_maturity\_to\_limit\_6m\_min\_ratio | Minimal principal after maturity within past 6 months to Limit |
| principal\_after\_maturity\_to\_limit\_6m\_avg\_ratio | Average principal after maturity within past 6 months to Limit |
| principal\_after\_maturity\_to\_outstanding\_ratio | Principal after maturity to Outstanding |
| principal\_paid\_mtd\_to\_outstanding\_ratio | Principal paid month to date to Outstanding |
| client\_capital\_to\_outstanding\_ratio | Own capital of a client on the loan to Outstanding amount |
| collateral\_value\_to\_outstanding\_ratio | Total collateral value registered at the cadastre to Outstanding amount |
| principal\_after\_maturity\_to\_outstanding\_3m\_max\_ratio | Maximal principal after maturity within past 3 months to Outstanding |
| principal\_after\_maturity\_to\_outstanding\_3m\_min\_ratio | Minimal principal after maturity within past 3 months to Outstanding |
| principal\_after\_maturity\_to\_outstanding\_3m\_avg\_ratio | Average principal after maturity within past 3 months to Outstanding |
| principal\_paid\_to\_outstanding\_3m\_max\_ratio | Maximal principal paid month to date within past 3 months to Outstanding |
| principal\_paid\_to\_outstanding\_3m\_min\_ratio | Minimal principal paid month to date within past 3 months to Outstanding |
| principal\_paid\_to\_outstanding\_3m\_avg\_ratio | Average principal paid month to date within past 3 months to Outstanding |
| principal\_after\_maturity\_to\_outstanding\_6m\_max\_ratio | Maximal principal after maturity within past 6 months to Outstanding |
| principal\_after\_maturity\_to\_outstanding\_6m\_min\_ratio | Minimal principal after maturity within past 6 months to Outstanding |
| principal\_after\_maturity\_to\_outstanding\_6m\_avg\_ratio | Average principal after maturity within past 6 months to Outstanding |
| principal\_paid\_to\_outstanding\_6m\_max\_ratio | Maximal principal paid month to date within past 6 months to Outstanding |
| principal\_paid\_to\_outstanding\_6m\_min\_ratio | Minimal principal paid month to date within past 6 months to Outstanding |
| principal\_paid\_to\_outstanding\_6m\_avg\_ratio | Average principal paid month to date within past 6 months to Outstanding |
| collateral\_accepted\_value\_to\_outstanding\_ratio | Accepted value of a collateral to Outstanding amount |
| all\_collateral\_accepted\_value\_to\_outstanding\_ratio | Accepted value of all collaterals to Outstanding amount |
| principal\_balance\_to\_paid\_mtd\_ratio | Principal balance to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_ratio | Principal after maturity to Total paid up to date |
| principal\_before\_maturity\_to\_paid\_mtd\_ratio | Principal balance before maturity to Total paid up to date |
| outstanding\_to\_paid\_mtd\_ratio | Outstanding amount to Total paid up to date |
| client\_capital\_to\_paid\_mtd\_ratio | Own capital of a client on the loan to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_3m\_max\_ratio | Maximal principal paid month to date within past 3 months to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_3m\_min\_ratio | Minimal principal after maturity within past 3 months to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_3m\_avg\_ratio | Average principal paid month to date within past 3 months to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_6m\_max\_ratio | Maximal principal after maturity within past 6 months to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_6m\_min\_ratio | Minimal principal after maturity within past 6 months to Total paid up to date |
| principal\_after\_maturity\_to\_paid\_mtd\_6m\_avg\_ratio | Average principal after maturity within past 6 months to Total paid up to date |
| outstanding\_after\_to\_principal\_before\_maturity\_ratio | Principal after maturity + Interest after maturity + Fee after maturity) to Principal before maturity |
| ltv\_ratio | Actual loan to value ratio |
| ltv\_at\_loan\_origination\_ratio | Loan to value ratio at origination of the loan |
| income\_to\_expense\_all\_applicants\_ratio | (Income-Expense)/Income of all the applicants |
| income\_to\_expense\_main\_applicant\_ratio | (Income-Expense)/Income of an applicant |
| principal\_after\_maturity\_to\_next\_installment\_ratio | Principal after maturity to Next instalment amount |
| principal\_paid\_to\_next\_installment\_ratio | Principal paid month to date to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_ratio | Interest after maturity to Next instalment amount |
| interest\_paid\_to\_next\_installment\_ratio | Interest paid month to date to Next instalment amount |
| fee\_after\_maturity\_to\_installment\_ratio | Fee after maturity to Next instalment amount |
| fee\_mtd\_to\_installment\_ratio | Fee paid month to date to Next instalment amount |
| expense\_all\_aplicants\_to\_next\_installment\_ratio | Expense of all the applicants to Next instalment amount |
| income\_all\_aplicants\_to\_next\_installment\_ratio | Income of all the applicants to Next instalment amount |
| expense\_main\_applicant\_to\_next\_installment\_ratio | Expense of the decisive applicant to Next instalment amount |
| income\_main\_applicant\_to\_next\_installment\_ratio | Income of the decisive applicant to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_3m\_max\_ratio | Maximal principal after maturity within past 3 months to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_3m\_min\_ratio | Minimal principal after maturity within past 3 months to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_3m\_avg\_ratio | Average principal after maturity within past 3 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_3m\_max\_ratio | Maximal principal paid month to date within past 3 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_3m\_min\_ratio | Minimal principal paid month to date within past 3 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_3m\_avg\_ratio | Average principal paid month to date within past 3 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_3m\_max\_ratio | Maximal interest after maturity within past 3 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_3m\_min\_ratio | Minimal interest after maturity within past 3 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_3m\_avg\_ratio | Average interest after maturity within past 3 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_3m\_max\_ratio | Maximal interest paid month to date within past 3 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_3m\_min\_ratio | Minimal interest paid month to date within past 3 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_3m\_avg\_ratio | Average interest paid month to date within past 3 months to Next instalment amount |
| days\_in\_deliquency\_3m\_max\_to\_next\_installment\_ratio | Maximum of delinquency days within past 3 months to Next instalment amount |
| days\_in\_deliquency\_3m\_max\_2\_to\_next\_installment\_ratio | Maximum of delinquency days according to NDoD within past 3 months to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_6m\_max\_ratio | Maximal principal after maturity within past 6 months to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_6m\_min\_ratio | Minimal principal after maturity within past 6 months to Next instalment amount |
| principal\_after\_maturity\_to\_next\_installment\_6m\_avg\_ratio | Average principal after maturity within past 6 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_6m\_max\_ratio | Maximal principal paid month to date within past 6 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_6m\_min\_ratio | Minimal principal paid month to date within past 6 months to Next instalment amount |
| principal\_paid\_to\_next\_installment\_6m\_avg\_ratio | Average principal paid month to date within past 6 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_6m\_max\_ratio | Maximal interest after maturity within past 6 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_6m\_min\_ratio | Minimal interest after maturity within past 6 months to Next instalment amount |
| interest\_after\_maturity\_to\_next\_installment\_6m\_avg\_ratio | Average interest after maturity within past 6 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_6m\_max\_ratio | Maximal interest paid month to date within past 6 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_6m\_min\_ratio | Minimal interest paid month to date within past 6 months to Next instalment amount |
| interest\_paid\_to\_next\_installment\_6m\_avg\_ratio | Average interest paid month to date within past 6 months to Next instalment amount |
| days\_in\_deliquency\_6m\_max\_to\_next\_installment\_ratio | Maximum of delinquency days within past 6 months to Next instalment amount |
| days\_in\_deliquency\_6m\_max\_2\_to\_next\_installment\_ratio | Maximum of delinquency days according to NDoD within past 6 months to Next instalment amount |
| behavioral\_score | Behavioral score |
| application\_score | Application scole |
| days\_in\_deliquency\_6m\_categorical | Days in deliquency 6M average |
| product\_type\_categorical | Simplified category of a product (HUF, HUF 85, HUF 100, BP and NHU) |
| participation\_to\_limit\_ratio\_categorical | Loan participation to loan limit ratio |
| penalty\_and\_interest\_paid\_mtd\_cat | Penalties and interes paid month to date |
| interest\_paid\_6m\_max\_to\_next\_payment\_cat | Interest paid 6M maximum to next payment |
| retail\_behavioral\_score\_categorical | Retail behavioral score |
| education\_categorical | Education of a client |
| default\_event\_flg | Event of default |

Source: CSOB, author

1. At the time of writing of this text the approved version of the EU *AI Act 2024* was not available on the EUR-Lex pages. The reader is referred to the proposed formulation from 2021 (EU), available in the References section of this thesis. [↑](#footnote-ref-1)