A Neural Network Approach for the Estimation of Mortgage Prepayment Rates

Roberto Baccaglini [*1], Federico Cozzi[*1], Vittorio Malacchia[*1], and Luca Sitzia[*1]

[1] *UniCredit SpA, Piazza Gae Aulenti 3, 20154 Milano, Italy*

April 17, 2021

Abstract

The owner of a mortgage can decide to repay his loan before the contractual maturity, completely or in part, in most cases without suffering any penalty. This might happen for a number of reasons which can go even beyond market driven incentives - ie. a more favourable general level of interest rates; as a direct consequence, the client behavior can modify in a material way the future cashflows that the mortgage issuer would expect to receive according to a purely contractual view. Thus, the estimation of such *behavioral* effects is a key factor in order to guarantee an accurate representation of the future asset profile, which in turn is at the base of core activities of a bank such as asset and liability management, interest rate risk assessment and mitigation, as well as liquidity management. We present a behavioral model for prepayment risk based on a Deep Learning approach in which the rate of early repayment on the mortgage portfolio is estimated by means of a Neural Network. The Network is calibrated using the past history of the observed repayment events; the explanatory variables (ie. features) are selected via a Random Forest approach among both contractual elements and market factors. The model is applied to a sample portfolio of contracts, considering both floating-rate as well as fixed-rate mortgages. The resulting predictions are compared with the actual repayment rates observed in the past, proving that the Neural Network approach is suitable for this kind of behavioral modeling. Also, the model shows that the inclusion of the market rate levels as explanatory factors might be beneficial in estimating the repayment phenomenon also in the case of floating rate mortgages, and in particular when a fixed-rate mortgage that would be attainable for an equivalent loan become favourable, as in an environment of extremely low interest rates.

Keywords: Behavioral Models; Mortgages; Prepayment; Neural Network; Deep Learning; Conditional Prepayment Rate (CPR); Asset and Liability Management (ALM), Interest Rate Risk in the Banking Book (IRRBB); Liquidity Risk.

Key messages:
Estimation of mortgage prepayment rates via Neural Network approach.

[*]The views, thoughts and opinions expressed in this paper are those of the authors in their individual capacity and should not be attributed to UniCredit group or to the authors as representatives or employees of UniCredit group.

Contents

1 Introduction

Loans and mortgages are a key component of the balance sheet of a Commercial Bank. Typically they represent a material portion of the Bank's assets, spanning mostly the medium and long term investment horizons. In fact, while products such as personal loans are in general granted for medium term maturities, mortgages are typically long terms assets with maturities that can extend up to 25 or even 30 years in the future. Due to their intrinsic nature and features, mortgages essentially bear two types of risk into a Bank balance sheet: besides the well known *credit risk* - which is the risk that the mortgagee will be not able to repay his debt, and which is not in scope for the present paper - a material source of uncertainty is given by the so-called *prepayment risk*. Indeed, despite the actual terms agreed when the contract is signed, a significant portion of clients tends to pay the mortgage back earlier than the original maturity. This prepayment option is legally allowed in many Countries, and the client usually has the right to proceed without any fee - possibly apart from some administrative cost, anyway not material in comparison with the mortgage outstanding amounts. As an example, in Italy the reference law is the Bersani Decree, dating back to 2007, which provides customers with a range of opportunities - such as the so called "portability of mortgages". Essentially, the law allows customers to transfer a mortgage to another Bank without any penalty, and allows the new Bank to take over the mortgage guarantee issued for the previously-granted loan. The new Bank can also offer better conditions compared to the previous one in terms of interest required, or give the borrowers relevant advantages such as the modification of the interest rate type or duration of the mortgage. Recent developments of online technologies and free mortgage comparison services further ease the process and provide an additional incentive to borrowers.

The goal of a *prepayment behavioral model* is indeed to predict the behavior of the clients in terms of prepayment rate, defined as the portion of the outstanding amount of the mortgage portfolio that is expected to be repaid in advance in a give time period (a precise definition is given in Section 2). It is evident that the capability of making such estimation in a reliable way is of primary importance in performing the Asset and Liability Management in the Bank. First of all, mortgages that are repaid in advance obviously affect the Bank's asset horizon, and estimating the actual behavioral maturity is important in terms of liquidity management. Also, in the case of fixed rate loans, the loan maturity influences the interest rate risk profile of the portfolio, and thus potentially the interest rate hedging strategy of the Bank. Indeed, Basel framework for the management of the Interest Rate Risk in the Banking Book (IRRBB) [1] recognizes the importance of such behavioral aspects by explicitly requiring a dedicated modeling in its IRR Principles:

*"**Principle 5**: In measuring IRRBB, key behavioral and modeling assumptions should be fully understood, conceptually sound and documented. Such assumptions should be rigorously tested and aligned with the Bank's business strategies.*
[...]
45 (i): Banks should understand the nature of prepayment risk for their portfolios and make reasonable and prudent estimates of the expected prepayments. [...]"

While the importance of the topic at many levels of the Bank's activities is therefore apparent, it turns out that building up a reliable behavioral model for prepayment poses a number of challenges, and making prepayment predictions may be not a straightforward task. First of all, it is common experience that each client might consider several aspects while deciding to repay his mortgage, and such aspects can be both financial - ie. the mortgage market has moved and now more favorable conditions are available - and non-financial, in the sense that client assesses that exercising the early repayment options is beneficial for him in a way which cannot be explained by market financial variables only. This is indeed the intrinsic feature that distinguishes a behavioral model from a pure market-driven model.
Another reason is that the link between any meaningful explanatory variable and the client's behavior is likely to be highly non-linear. Explicit evidences of such non-linearity are available in literature, for example in [2], where the relationship between the empirical prepayment rate actually observed and the theoretical prepayment incentive - as given by the initial mortgage rate minus the market rate- is analyzed. It is indeed shown that the sensitivity of the prepayment to the financial incentive can vary quite remarkably, even changing sign; this supports the intuition that a naive linear modeling could be prone to substantial inconsistency and estimation error. A third element to be considered is

that many of the most widely used statistical estimators, even if allowing for a certain degree of non-linearity, anyway rely on pre-determined functional forms. For example, a quite common approach for the repayment estimation is the use of a logistic regression: here the analytic expression of the model relies on the logit function, and the goal of the optimization routine is to find the best possible parameters that maximize the associated likelihood function.

The Deep Learning approach that is proposed in this paper aims to circumvent the mentioned limitations. In fact, a Neural Network can be seen as a generalization of the logistic regression in which no explicit functional relationship between explanatory variables and the probability of pre-payment events is superimposed. The Network is made of multiple layers of nodes that combine the independent variables together in a highly non linear way, while an optimization procedure is used to let the Network learn the input-output dependency directly from data. Furthermore, a key step in the Network set up is the decision of the explanatory variables (ie. the *features* of the Neural Network) to be used: the proposed model leverages on the Random Forest approach to identify the variables that may show a high explanatory power.
Recent literature has started exploring the application of similar Deep Learning approaches to the problem of estimating the loan prepayment rate. An example can be found in the already cited [2], where a multi-period model is presented. [3] presents an approach in which the estimation is split in different sub-components and the prepayment phenomenon is modeled via a pure classification problem (the loan is early repaid or not) while the actual amount subject to repayment is modeled separately via a correction factor. A comprehensive and detailed description of the theoretical Neural Network framework applied to the behavioral field can be found for example in [4], while general references for Machine Learning approaches can be found for example in [5], [6], [7].

Current paper presents the application of a Deep Learning approach on the two main relevant examples of mortgage types, namely

1. floating rate mortgages,
2. fixed rate mortgages.

The nature of the financial products involves obviously repayment incentives of different nature, as the owner of a fixed-rate mortgage should be in principle more interested in exploiting a decrease in market rates. Nevertheless, the Neural Network model proves to provide good performance on both application, at the small cost of minor adaptations of the considered features. On top of this flexibility, the approach presented in this papers has some innovative features with respect to the available models. First, literature models usually tend to rely on a wide range of variables. While having undoubtedly a benefit in terms of statistical estimation, in practice this may pose problems when the model is used for forecasting purposes in a Bank day-to-day activity (Section 2 elaborate more on this point). The proposed model is based on a parsimonious set of variables that can be forecasted in a straightforward way, and which show to provide anyway a satisfactory statistical performance. Second, it explores the inclusion of float-to-fix as well as fix-to-float incentives, which prove to be relevant in a regime of very low market rates.

The paper is organized as follows: Section 2 sets the problem in a formal way and introduces the quantity to be estimated, namely the conditional prepayment rate; Section 3 presents the Neural Network model that has been developed and all its features. The model is tested on a sample portfolio of mortgages[1], via application to both cases of floating rate and fixed rate contracts; results are presented in Section 4. Conclusions are summarized in Section 5, while technical details of the Network implementations can be found in Appendix.

2 Modeling Early Repayments for Mortgages

2.1 The Prepayment phenomenon and factors for estimation

Mortgages can be split into two broad categories, according to how the interest to be payed by the client is determined: in a *fixed rate* mortgages the coupon is determined at the time of contract signing,

[1]The model has been applied on an artificial sample portfolio for the only purpose of discussion and presentation of the performance of the neural network approach. The portfolio is not representative of UniCredit mortgages and associated prepayment levels in any way.

while in a *floating rate* mortgage the interest is linked to a market rate (eg. Euribor) and it is re-fixed on a periodic basis.

The main types of events considered in the modeling are real prepayments and extra-contractual renegotiations. The first are relevant for forecasting the evolution of the bank loans portfolio from both a liquidity and an interest rate risk perspective, and in general include:

- *direct prepayments*: in this case the client simply extinguishes the loan by repaying itself, in a bullet cashflow, the principal amount which is still outstanding;
- *substitution prepayments* or *surrogas*: in this case the client opens a new loan with another bank (typically at a lower nominal rate) with amount equal to the residual outstanding of the original one and then uses the borrowed amount to prepay the first loan.

For what concerns extra-contractual renegotiations, these are relevant only from an interest rate perspective since they do not involve unexpected repayments from the client, but only a change in the conditions of the contract (again, typically a lower nominal rate), whose regular repayments continue with the original bank after such event.

Each client may show some sensitivity to the incentives to repay it's mortgage earlier than contractual maturity, which might depend on various reasons. Among them we can identify

- non-financial reasons: a client might be more or less sensitive to incentives to prepay according to idiosyncratic elements and contractual features, such as for instance
  - the outstanding mortgage amount; in fact, clients borrowing a large amount of money could be more reactive in taking advantageous alternative opportunities;
  - the remaining time before expiry, as obviously a longer loans entails higher interest costs;
  - time effects as seasonality, ie. periodic oscillation undergone by prepayment rates during certain periods of the year;
- financial reasons, as the interest required by Banks in general tracks the level of rates in the markets (specifically the IRS curve) and thus more beneficial conditions may arise in the loan market when rates go down.

Typically this financial incentive is mostly associated to fixed-rate mortgages, as in principle floating-rate products automatically adjust the coupon levels according to rate movements; anyway recent market conditions - where the EUR IRS curve can go negative up to medium-long term - somehow challenge this assumption. Indeed in such a market it shall be considered if a floating-rate borrower might prefer to switch its loan into a fixed-rate product to exploit the possibility of locking a very low fixed interest, thus protecting himself from future rate increase, and not paying much in terms of additional spread applied as the final client rate must indeed be non-negative. Therefore, adding this kind of effect also in the modeling of prepayment for floating rate mortgages seems reasonable; the present model is developed according to these considerations, and the results obtained in fact support this approach.
Clearly the set of repayment drivers can be further widened to include additional factors such as macroeconomic variables (eg. GDP, unemployment rate,...) or additional borrower-specific information, such as income of the mortgagee, marital status and potentially many other ingredients. Examples of models considering such variables can be found in [2] and [3]. Whilst the inclusion of additional features might potentially further enhance the performance of the direct statistical estimation, a practical drawback shall be kept in mind: the final goal of the behavioral model is predict the future clients behavior. In practice, this is usually done in two steps

1. *statistical estimation step*, in which a prepayment rate in a given time interval (eg. one month) is estimated; this is the step on which this paper and most related literature- focuses on;
2. *forecasting step*, in which a prediction for multiple time periods in the future is obtained.

In other words, after a dependency is assessed between the prepayment attitude and a set of variables, and a statistical relationship is estimated (via Neural Network or whatever other statistical method), one has to forecast a future evolution of the driving factors from which to infer the future expected prepayment reactions. Considering the typical duration of mortgages, predictions spanning up to 30 years are required. In this perspective, it becomes thus clear that enlarging the pool of independent variables comes at the cost of being able to get reliable projections of these factors in the future; uncertainties in this step might overcome the potential benefit that could arise in the pure statistical

estimation step. One of the advantages of the approach proposed in this paper is to focus on variables that can be forecasted in a straightforward way, while obtaining a satisfactory statistical model performance.

2.2 The Conditional Prepayment Rate

As outlined in Section 1, the reference quantity which is analysed in order to describe the prepayment phenomenon is the conditional prepayment rate (sometimes also referred to as *hazard rate*), which is defined as the ratio between the amount of principal cashflows which is early repaid in a given time interval t, and the total outstanding amount at the beginning of t:

$$CPR_t = \frac{V_{\text{prep},t}}{V_{\text{out},t}} \tag{2.1}$$

The term "conditional" stems from the fact that, considering from a modeling point of view the subsequent time intervals $\{t_j\}$ of future instalment, CPR_{t_j} represents also the probability that a given contract is prepaid during interval t_j, provided that such contract has not been already prepaid before; "early" repayments indicate reimbursements that are anticipated with respect to their contractual schedule.

When t corresponds to one month (as in this case), the quantity above Equation 2.1 is also referred to as *single month mortality rate*, while the corresponding quantity for a different time period t' is calculated as:

$$CPR_{t'} = 1 - (1 - CPR_t)^{\frac{t'}{t}} \tag{2.2}$$

For instance, in case t'=12 months, the above formula gives the total fraction of initial volume which is prepaid within one year, under the hypothesis that during each month a fraction CPR_t is prepaid respect to the outstanding volume at the beginning of the month.

In summary, the aim of the prepayment model is to produce a prediction of the future expected prepayment date, from which the volume prepaid will obtained through application on current outstanding portfolio; this is achieved via calibration on historically realized hazard rates. For the present model the calibration is done by means of a Neural Network approach, which is described in the next section.

3 Neural Network Approach

3.1 Building up the Neural Network

3.1.1 The sequential model

The kind of neural network employed for prepayment modelling corresponds to the simplest one, which is called *sequential model*: it consists of a stack of one or more elementary sub-structures, called *layers*, each one of which is composed of one or more *nodes* (also called *neurons*). In general, a neural network includes:

- an *input layer*, which receives as input the model independent variables (also called *features*),
- an *output layer*, which produces the final result of the model prediction,
- one or more *hidden layers*, each one of which receives the input from either the input or the immediately preceding hidden layer, and sends the output to either the output or the immediately following hidden layer.

In particular, when each node of each layer receives input from every single node of the previous layer, and sends output to each node of the following layer, the model is defined as *densely connected*. In Figure 1, for instance, is reported the scheme of a densely connected sequential model composed by:

- an input layer composed by 5 nodes,
- a hidden layer consisting of 3 nodes,
- an output layer with a single node

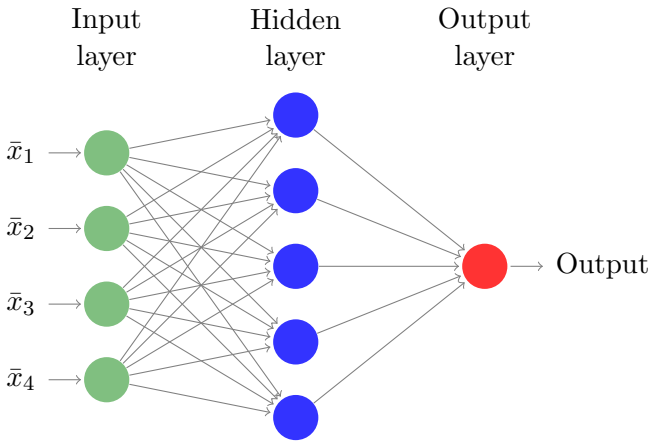


Figure 1: Example of Dense Neural Network architecture with one hidden layer.

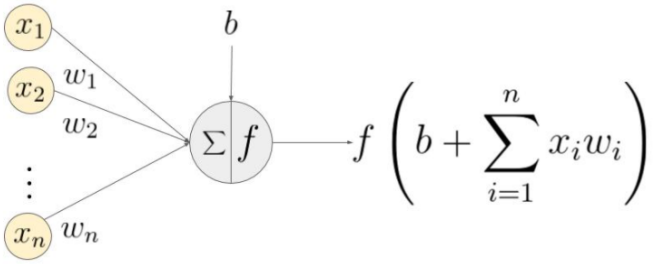


Figure 2: Schematic representation of how a single neuron works.

The nodes represent the most elementary structure of the neural network, and from a mathematical point of view they consist of a linear part - in which they simply aggregate all the N inputs $\{x_n\}_1^N$ by means of a set of weights $\{w_n\}_1^N$ and a bias b - on which a non-linear transformation is applied by a means of the so-called *activation function* f. This structure is represented in Figure 2.
In formulas, the output of the $j-$th neuron of the $\ell-$th layer is given by:

$$
\begin{aligned}
z_j^{[\ell]} &= f^{[\ell]}(y_j^{[\ell]}), && \forall j \in \{1, \dots, J^{[\ell]}\}^2 \\
y_j^{[\ell]} &= \sum_n w_{j,n}^{[\ell]} \cdot x_n^{[\ell]} + b_j^{[\ell]}, && \forall j \in \{1, \dots, J^{[\ell]}\} \\
x_n^{[\ell]} &= \begin{cases} z_n^{[\ell-1]}, & \ell > 1 \\ \bar{x}_n, & \ell = 1 \end{cases}
\end{aligned}
$$

where the last equality simply means that the input corresponds to the raw data features in the case of the input (i.e the input vector $\bar{x}$) layer, or to the output of the previous layer for the subsequent ones ($z_n^{[\ell-1]}$). Here, the activation function $f^{[\ell]}$ has the same form for all nodes of the same layer, and thus does not depend on j.
It can also be noted that, as well-known, the neural network is in fact a generalization of a logistic regression approach; more precisely, the logistic regression can be seen as a trivial example of neural network, where input, hidden and output layers are collapsed in a single layer, composed of a single node with activation given by the sigmoid function (see Equation 3.1), as represented in Figure 3, where $L = 1$.

$$
\phi(y) = \frac{1}{1 + e^{-y}} \tag{3.1}
$$

[2]The number of Neuron J in each layers can be different, we us the apex $[\ell]$ to identify that we are referring to the $\ell - th$ layer in a possible range of $\ell := \{1, \dots, L\}$

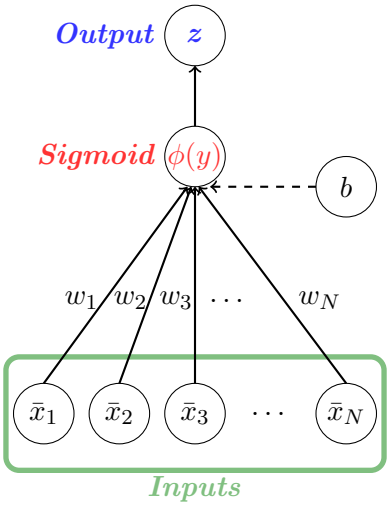


Figure 3: Schematic representation of logistic regression as trivial example of neural network.

3.1.2 Optimization through back-propagation

The aim of the calibration process is to optimize the loss function, i.e. to find the combination of model weights which minimize its value, in a similar way - in principle - to the maximization of the likelihood function in regression problems. The *training process*, i.e. the process during which the model is calibrated, essentially consists in two fundamental steps.

During the first step (the so-called *forward propagation*), the information simply flows through the sequential model, starting from the model inputs and, after being processed by the hidden layers, producing the final output which represents the model prediction.

Since the phenomenon which is being modelled is the prepayment, this means that the historical time series used for calibration also includes information about whether, for each past date, a prepayment event happened or not for each mortgage (the so-called *label*, $\bar{y}$). In other words a *supervised learning* process is being performed: in such a process the forward step ends with the computation of *loss* (also called *cost*) function $\mathcal{L}$, which is a quantification of the (average) degree of discrepancy between past model predictions (i.e. computed starting from historical realizations of the input features) and historical labels.

The second step (so-called *back-propagation*) consists of adjusting the model weights in a sequential way, starting from the last layer and going backward to the first one in order to assimilate the first step into the parameters of the model. In formulas, the output of a sequential model with L layers[3] So we have for the $L-th$ layer $j := 1$ is generally given by:

$$
\begin{aligned}
z^{[L]} &= f^{[L]}\left(\sum_k w_k^{[L]} \cdot z_k^{[L]} + b^{[L]}\right) = \\
&= f^{[L]}\left(\sum_k w_k^{[L]} \cdot f^{[L-1]}\left(\cdots\left(f^{[1]}\left(\sum_{j,n} w_{j,n}^{[1]} \cdot \bar{x}_n + b_j^{[1]}\right)\cdots\right) + b_j^{[L-1]}\right) + b^{[L]}\right)
\end{aligned}
$$

Consequently, the application of the chain rule to the loss function $\mathcal{L}(z^{[L]}, \bar{y})$ - where $(\bar{y})$ is known and $z^{[L]}$ represents the prediction, ie. the output of the Neural Network- produces (see [8] for more details):

[3]The case described int he document is the one in which the final layer is composed of a single neuron, i.e. the model output is a number which corresponds to the value of the prepayment probability. This is the relevant case for modeling the prepayment phenomenon.

$$\frac{\partial \mathcal{L}}{\partial w_j^{[\ell]}} = \frac{\partial \mathcal{L}}{\partial z^{[L]}} \frac{\partial z^{[L]}}{\partial w_j^{[\ell]}} = \frac{\partial \mathcal{L}}{\partial z^{[L]}} \sum_k \frac{\partial z^{[L]}}{\partial z_k^{[L-1]}} \frac{\partial z_k^{[L-1]}}{\partial w_j^{[\ell]}} = \frac{\partial \mathcal{L}}{\partial z^{[L]}} \sum_k \frac{\partial z^{[L]}}{\partial z_k^{[L-1]}} \cdots \frac{\partial z_j^{[\ell]}}{\partial w_j^{[\ell]}}$$

The adjustment for each weight $w_j^{[\ell]}$ can be obtained, at each step of this recursive process, by simply computing the derivative $\frac{\partial z_j^{[\ell]}}{\partial w_j^{[\ell]}}$ in addition to the already computed quantity $\frac{\partial \mathcal{L}}{\partial z_j^{[\ell]}}$.

During the entire training process, the forward and backward propagation steps are performed multiple times: every macro step in which the entire dataset used for calibration is processed is usually referred to as *epoch*. Since Neural Network approaches are especially suited for model characterized by huge amount of data - and prepayment case is not an exception - performing forward/backward process at once on all observations is in fact challenging or even impossible due to technological constraints. In these cases, the dataset is generally divided into *mini batches*, so that each epoch actually consists in sub-steps (whose number is given by the ratio between the entire dataset size and a single mini batch size), during which the forward and backward pass are performed, in sequence, only on a single mini batch. The size of the mini batches represents one of the so-called *hyper-parameters*, i.e. not strictly a model parameter such as the weights $\{w_j^{[\ell]}\}$, but indeed a parameter which needs to be set. During the training i.e. the learning phase, standard practice is monitoring of the so called *metrics*. Several indicators of the training evolution can be defined, helping in the monitoring of the process. The most straightforward one is the loss function itself, but other can be added in relation to the problem. Such metrics can support the developer in identifying potential areas of improvement of the model.

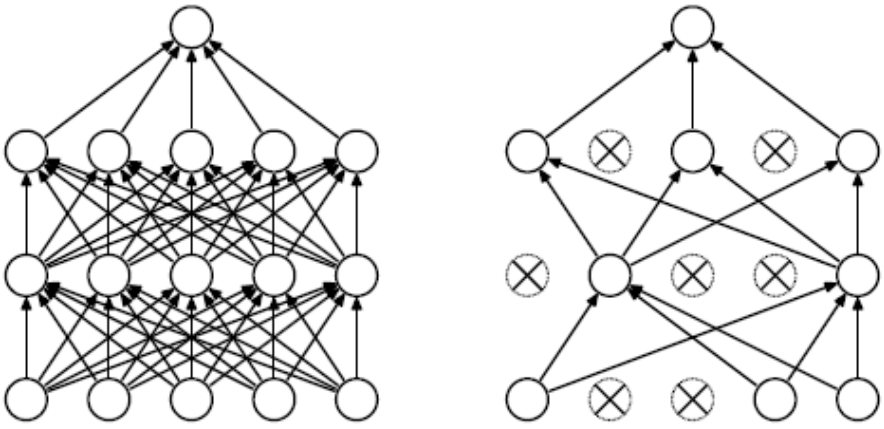
3.1.3 Overfitting and Regularization

Neural networks are characterized by a great flexibility in accurately describing the (unknown) function underlying the historical data concerning the phenomenon which is being modeled. On the other hand, it is important to avoid the risk that the network actually *over-fits* the data i.e. the risk that, as a result of training process, the network becomes in fact accurate in reproducing the historical data but at the cost of penalizing the capability to generalize the prediction power to data not originally included in calibration. In other words, it must be avoided that the network "memorizes" the training data without gaining prediction accuracy on new data. For this reason, good practice prescribes that the available data is generally divided in three distinct datasets:

- a *training* set, which is generally the largest one and is actually used in the forward/backward process
- a *validation* set, on which certain metrics (such as the loss) are computed at each step for monitoring purpose, in order to keep track of how the generalization power on new data decreases as the loss function is being optimized on the training data
- a *test* set, on which the model performance is evaluated at the end of the entire training process.

In addition, over-fitting is generally avoided, or at least limited, by means of so-called *regularization* techniques: this model adopts one among those used most, which is the *dropout* ([9]). This procedure simply consists in randomly dropping nodes from a given layer (see Figure 4), where each node can be dropped with a certain probability, which represents the *dropout rate* and, in general, is an hyper-parameter set externally by the user.

The main reason for the introduction of dropout is to prevent the so-called *co-adaptation*, namely the tendency of a neural network to rely, in order to improve its performance, on a restricted part of its neurons, the weights of which are adjusted in a more accurate way. The rest of the neurons, in this way, is co-adapted in the sense that their performance is conditioned to the presence of the best performing ones: by randomly removing some of them at each forward pass, each neuron is instead brought to tune its weights in order to better perform and rely more on its own. A different perspective also enables to view dropout in terms of *ensemble* of models: in brief, each distinct way of dropping the neurons represents a new network configuration, so that during the training procedure what is being calibrated is actually an entire set of different models. In consequence, the whole calibrated network represents an average of multitude of different sub-networks and, in this way, the chance to better perform when tested in out of sample data are increased because it is more likely that at least some of these sub-models do not over-fit on the training set respect to the single initial one.



(a) Standard Neural Net (b) After applying dropout.

Figure 4: Example of dropout procedure.

In the application to prepayment prediction two types of splitting of the database can be adopted, in order to capture different aspects of the time series:

- *vertical splitting*, where the database is split according to the underlying *time-evolution* of the data, ie. each subset contains mortgages between two determined dates
- *horizontal splitting*, where the spit is cross-time and, for each month, the mortgage portfolio is split into the three subsets.

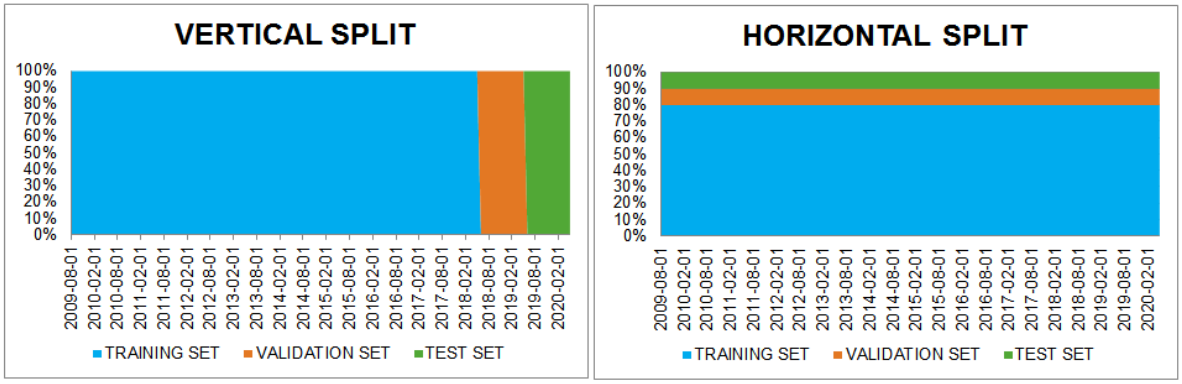


Figure 5: Example of Vertical and Horizontal splitting.

In the splitting phase, a specific role is played by the nature of the modeled phenomenon itself. As the prepayment events are usually quite rare, the sampling of the 3 sets must be done carefully in order do not introduce an artificial bias in the calibration engine, especially in the horizontal splitting case. In order to tackle this point, a random shuffling of the dataset before extracting the sample for each month helps to grant sufficient equiprobability for the events and to avoid the introduction of biases in the sampling.

3.2 Model 's explanatory variables

An important aspect in the building process of a Neural Network consists in the selection of the drivers of the model, usually referred to as *feature selection*. This is analogous to the case of simpler approaches, eg. a regression, where the *regressors* themselves need to be identified. This choice is obviously dependent on the phenomenon to be modeled, and indeed justified and benchmarked against the final model accuracy; nevertheless, some preliminary statistical considerations may be of help.
It is worth to note that in the machine learning context it is not uncommon to delegate the choice of

meaningful variables to the learning algorithm itself; the capability of such algorithms to manage large datasets allows in principle such an approach. Nevertheless, in the present applications the feature selection has been performed in two steps. First, a pool of potentially meaningful variables has been identified in relation to the nature of the problem; then, a statistical assessment has been performed in order to fine tune the choice and guide the selection. This approach has two main advantages: it allows to build the model in a parsimonious way by focusing on variables with strong economic meaning; moreover, this helps any *ex post* analysis and interpretation of the results, which is a key aspect for the model to be successfully applied and contribute to the understanding and management of the prepayment phenomenon.

In the current model formulation three typologies of features have been identified as main driver of the clients behavior and thus used in the set up of the model:

- *contractual features*, capturing the conditions which are specific to each individual contract;
- *market rate features*, in general independent from the contracts and representing the market environment condition;
- *time features*: capturing the time evolution in the portfolio and in the model in general.

Usual techniques to select the most relevant drivers for the model aim to analyze somehow the levels of statistical correlation. In the current case the assessment has been performed by means of a Random Forest algorithm, that tests several configurations in order to identify the more promising ones. This provides a starting point for setting up the neural network configuration; nevertheless, as already stated the final assessment of adequacy is indeed defined in terms of satisfactory model performance.

3.2.1 Floating rate mortgage model

Contractual features The contractual features which are taken into account in the model are:

- *current outstanding amount*: the sum of principal cashflows which - at the time of model calibration - the borrower is expected to pay in the future, in line with the contract amortization profile;
- *initial outstanding amount*: the total amount originally borrowed by the client;
- *time from contract origination*: the time interval, measured in months, elapsed from the stipulation of the contract or, to be more precise, from the starting of the amortization plan;
- *contract full rate*: the nominal rate which determines the accrued interest during current period. Contractually, the nominal rate is defined by the sum of a fixed rate (ie. the contract spread rate) and a market rate (see interest rate variables below);
- *contract spread rate*: the component which, added on top of the market rate, determines the full rate of the contract;
- *original maturity*: the total length of the contract as originally agreed at stipulation. For sake of simplicity, the actual maturity is cast into a predefined grid of maturities such as 10, 15, 20, 25, 30 years; in other words, an original maturity equal to 12 years is represented as 15 years.

Market rate features In order to capture the effects that the interest rate levels might have on the customer behavior, a set of variables linked to market evolution are used as input in the network:

- *spread on new floating rate contracts*: volume-weighted average of contract spread rates from floating rate mortgages originated during the last month. In order to allow for some time smoothing, then the rolling average on the past 12 months of such quantity is applied;
- *difference between fixed and floating rate on new contracts*: difference between volume-weighted average rate of fixed-rate and floating-rate mortgages originated in the last month. Again, a rolling average on the past 12 months is considered.

The spread on new floating rate contracts, compared against the original contract spread, aims to capture the potential incentive for the client to substitute the current mortgage with an alternative, more favorable floating-rate contract when the market presents the possibility to obtain a lower spread condition.

The difference between fixed and floating rate on new contracts, instead, represents the potential incentive to move from a floating to a fixed rate mortgage: this can represent a profitable change when, as happened for example at end of 2019, the difference between fixed and floating rate on new contracts in the market shrinks, also in consideration of the persistent negative Euribor levels. In such conditions, in fact, the borrower might be able to fix a low interest rate for the rest of the residual maturity, at the price of paying only a few basis points above the current floating rate.

Time features An significant feature dealing with time effects input to the network is the *seasonality*, which is represented as a boolean variable equal to *true* if the current month corresponds to August and *false* otherwise. In this way it is possible to capture, and then predict, the characteristic drop which is empirically evident in the prepayment phenomenon of the portfolio under consideration during summer period, and which is attributed to a general decrease in the bank activity due to holidays.

3.2.2 Fixed rate mortgage model

In this case *contractual* and *time* features coincide with the floating rate model specification, with the only exception that the contract spread rate is not used because the client is expected to pay attention only to the single full rate instead of deep diving into its breakdown between market and spread rate. This is also supported by the fact that the full rate is often the only parameter reported by banks in advertising their mortgage products.

Market rate features In order to capture the effects that the interest rate levels might have on the customer behavior, a set of variables linked to market evolution are used as input in the network:

- *full rate on new fixed rate contracts*: volume-weighted average of contract rates from fixed rate mortgages originated during the last month;
- *full rate on new floating rate contracts*: volume-weighted average of contract rates from floating rate mortgages originated during the last month;
- *3-month Euribor rate*: average of 3-month Euribor rate daily fixings during the last month;
- *IRS rate*: average of IRS rate daily fixings (for the tenors 10, 15, 20, 25, 30 years) during the last month;
- *Difference of current IRS rate versus historical minimum*: the current (monthly average) IRS rate is compared with the historical minimum reached in the past, but neglecting the history of the last 12 months;

In this case, both full and market rates are included as explanatory variables in order to take into account possible interaction effects between interbank and mortgage market. In addition, the last variable represents how much the interest rates have moved in a favorable way for the borrower respect to the previous year: this means that it is assumed a time lag up to 12 months before the client notices the presence of a financial incentive and exercises the prepayment option. Finally, for the first four variables it is taken both the value corresponding to current month, and the one realized 3 month before: this is done in order to take into account possible trends in rates evolution; for the last variable, instead, only the series lagged by 3 months is taken.

3.3 The loss function and the network hyperparameters

Being a Neural Network essentially based on an optimization procedure, defining the form of the loss functions itself is a fundamental step. Several examples of loss functions are presented in literature; maybe the most famous and widely applied in classification problems is the so called *binary cross-entropy* function:

$$\mathcal{L}_{CE} = -\frac{1}{N_C}\sum_{c=1}^{N_C} \bar{y_c} \cdot \log(z_c^{[L]}) \tag{3.2}$$

This type of loss counts every contract equally, thus essentially implementing a number-weighted computation; in other words, the model tries to maximize the classification of contracts without any specific indication about the materiality of the contracts themselves. Nevertheless, from a practical perspective, the final aim of the prepayment model is to assess how much of the outstanding amount is

subject to prepayment risk and, as such, a higher precision is desirable especially on loans presenting a large residual notional. In order to enforce this mechanism, the loss function is modified by introducing an additional weight to the expression above, given by the relative magnitude of the $c-th$ contract in the overall portfolio:

$$v_c = N_C \cdot \frac{V_{out,c}}{\sum_{c=1}^{N_C} V_{out,c}} \tag{3.3}$$

so that the final function being optimized is

$$\mathcal{L} = -\sum_{c=1}^{N_C} \bar{y_c} \cdot \log(z_c^{[L]}) \cdot u_c, \qquad u_c = \frac{V_{out,c}}{\sum_{c=1}^{N_C} V_{out,c}} \tag{3.4}$$

As a result, instead of a pure arithmetic average of the individual terms, the loss function is changed into a volume-weighted average, where the weights are given by the current outstanding amount of each contract, and possible volume/concentration effects in the prepayment phenomenon are taken into account.

The other fundamental ingredients of the technical structure of the Neural Network are the so called *hyperparameters*, ie. all those parameters that are needed to give a precise and details picture of the Neural Network in its detailed implementation. Examples of such parameters are the number of layers in the network, or the number of nodes for each layer, or the learning rate - essentially setting the size of the step at each iteration during the optimization. In the current approach, hyperparameters are subject to a tuning procedure in order to identify an optimal configuration to be used; a complete description and list of the actual parameters applied is reported in the Appendix.

4 Results

The Neural Network approach described in Section 3 has been applied on a sample portfolio of Italian retail mortgages[4]. The analysis has been performed by considering the time window from 2009 up to 2020; prepayment data before 2009 cannot be compared to more recent ones due to the already mentioned evolution in the legal framework that has materially affected the mortgage market (ie. Bersani law, as mentioned in Section 1).
Nevertheless, such time window is large enough to include periods in which market conditions are very different. As an example, one can see (Figure 6) that interest rate levels have been dramatically changing during these last 11 yeas: the 10Y Interest Rate Swap, initially around 3.5%, has fallen down starting from second half of 2011 and finally reaching unprecedented low levels of about -30bps.

Indeed this is expected to be reflected in the levels of prepayments, specifically in terms of renegotiations or surrogas, especially for fixed rate contracts. The sample portfolio consists of a mix of mortgages of all maturities, issuance dates and contractual rates; it is made of approximately 5.8 millions of monthly observations of floating rate contracts and 5.4 millions of monthly observations of fixed rate contracts, distributed along a time period starting from August 2009 till December 2020.
In order to benchmark the performance of the Neural Network model, the prepayment estimation has been tested on the same portfolios by applying also two alternative approaches, quite standard and widespread:

- a Kaplan-Mayer estimator, ie. a non parametric approach essentially evaluating the historical average trend of prepayment rates,
- a logistic regression estimator, applied by considering the same explanatory variables presented in Subsection 3.2.

It quite evident that the two mentioned approaches are characterized by different levels of sophistication which is expected to translate in different levels of accuracy in prepayment predictions; it is also well known that the approach 2 is equivalent to a *degenerate* neural network with a single layer as described in Subsubsection 3.1.1. Finally, data used for the model calibration have been normalized, for each feature, by subtracting the mean and dividing by the standard deviation as calculated from the training set.

[4]The model has been applied on an artificial sample portfolio for the only purpose of discussion and presentation of the performance of the neural network approach. The portfolio is not representative of UniCredit mortgages and associated prepayment levels in any way.

Figure 6: Market rates time series.

4.1 Floating rate mortgage model

The historically observed prepayment rates for the sample floating rate loan portfolio under consideration are presented in Figure 7 [5]. The figure reports also the comparison against model estimations. As expected, a non-parametric model provides a rough estimation of repayment levels which is less and less precise when a material inversion in the trend of events appears, such as during the 2012-2014 decrease followed by an increase until 2018, and again a second upward trend from late 2019 on. Conversely, both parametric approaches are able to follow such evolution; nevertheless the Neural Network proves to follow the data with a higher degree of accuracy.

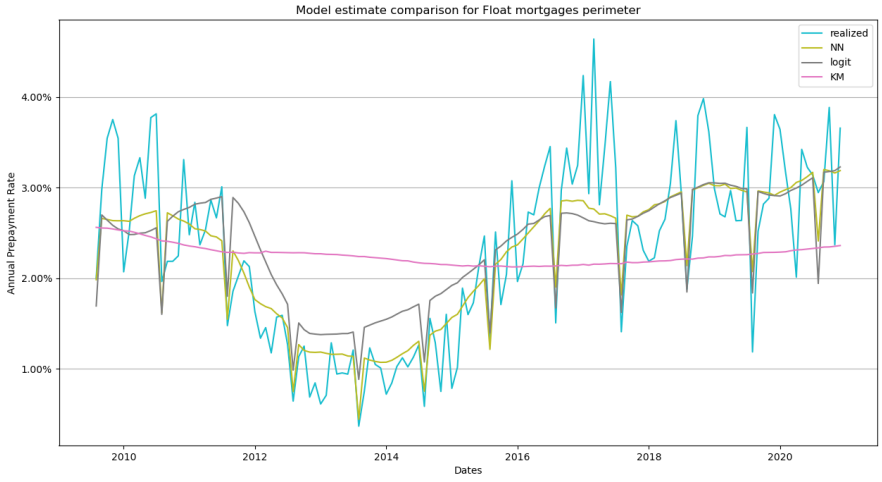


Figure 7: Comparison of model prediction respect to historical prepayment trend (floating rate mortgages)

The superior performance of the Neural Network can be also seen from Figure 8 and Figure 9, where the *residuals* of the estimation (ie. the difference between model predictions and realizations per each estimation point) is reported. The Neural Network presents a more reliable outcome in terms

[5]The data and levels displayed in this and the following charts have been subject to normalization, for complete anonymization purposes.

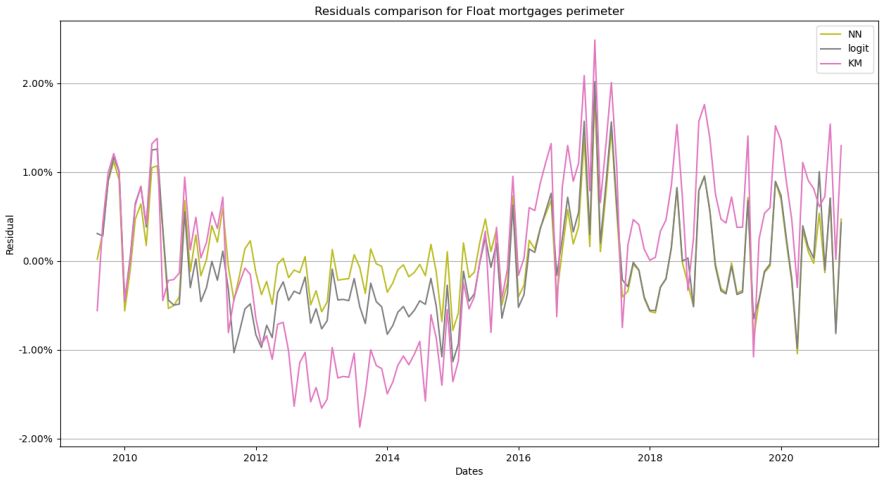


Figure 8: Comparison of model residuals time series (floating rate mortgages)

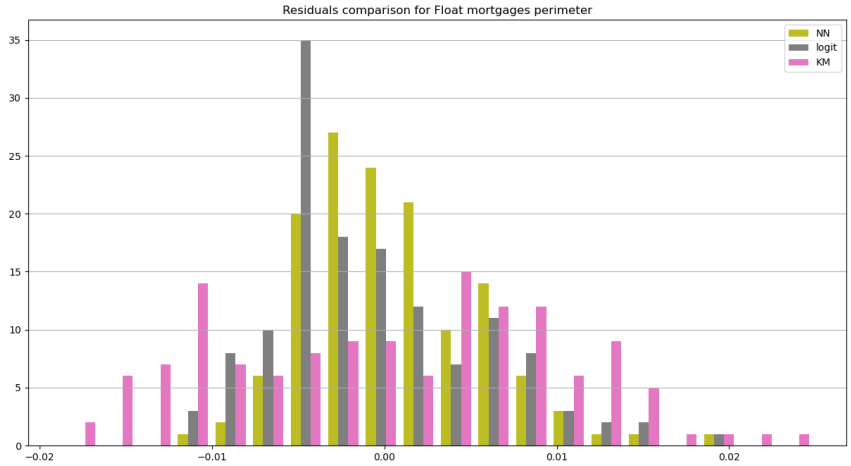


Figure 9: Comparison of model residuals distributions (floating rate mortgages)

of pure distance from the actual rates, as measured for instance by a Root Mean Square Error, and also shows a distribution of residuals which is less skewed.

4.2 Fixed rate mortgage model

The analysis of the evolution of prepayment rates for fixed rate mortgages in the sample portfolio (see Figure 10) is even more interesting, and intrinsically more challenging for a prediction model. As expected, prepayment rates are much more volatile for fixed rates contracts, and the effect of the evolution of the market interest rates giving rise to financial incentives is quite evident. Nevertheless, the interplay between the evolution of market rates and the behavior of the clients is highly non-linear. In fact, from a purely theoretical standpoint it is expected that a decrease in market rates provides an incentive for prepayment rates to grow, but the way in which this effect materializes in the mortgage portfolio can be not straightforward. It is empirically observed that the reaction of mortgagees may have different speeds, showing material delays in some occasions or associating sharp steepenings of

the prepayment curve vis-a-vis apparently smooth changes in the market rates - as for example in late 2019. Several reasons might be deemed relevant for such effects (including for example an overall status of economy, both in terms of actual conditions and perception of the future), and are beyond the scope of this paper. What is relevant here is that this intrinsic strong non-linearity highlights the limits of the logistic approach, which shows a weak performance especially in correspondence of peaks; on the other side, the neural network model is flexible enough to track repayment evolution in a much accurate way.
This is evident by observing the period of 2015-2018 when, after capturing the initial peak, the logistic approach tends to overestimate the decrease in the prepayment decrease, or - in other terms - to underestimate the expected repayment levels. Instead the Neural Network reacts in a more accurate way, both in this period and also in correspondence of the subsequent 2019 peak, where it is able to react in a faster way and to reach the actual high levels of prepayment quite soon.

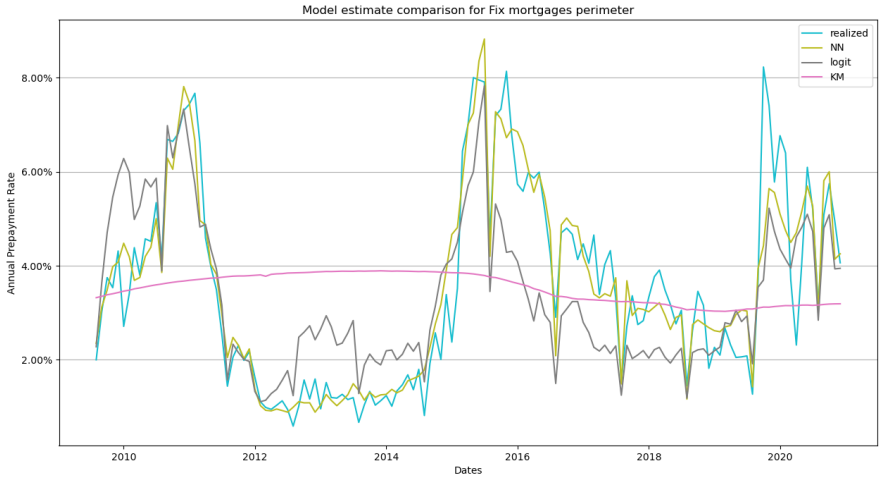


Figure 10: Comparison of model prediction respect to historical prepayment trend (fixed rate mortgages)

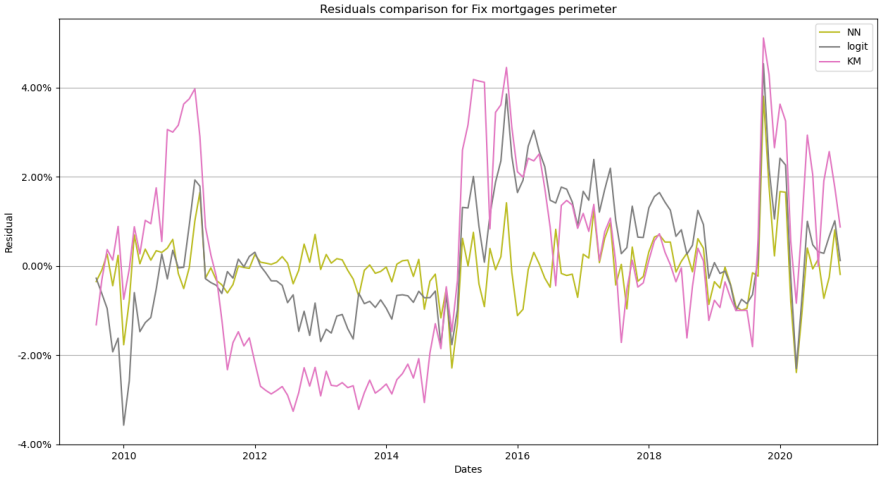


Figure 11: Comparison of model residuals time series (fixed rate mortgages)

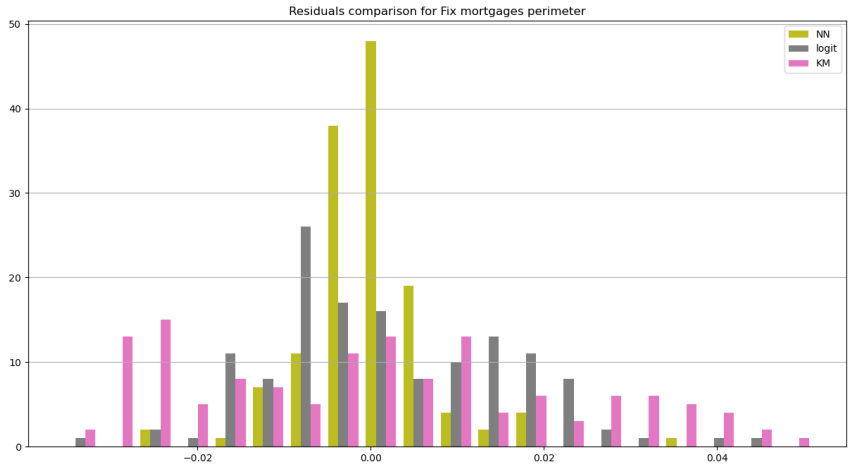


Figure 12: Comparison of model residuals distributions (fixed rate mortgages)

5 Conclusions

A Neural Network model for the estimation of the behavioral phenomenon of the early repayment of loans is presented. The Network is of a sequential type, and it is based on a parsimonious choice of features, performed via a Random Forest approach. The general Deep Learning set up is adapted to the specific nature of the problem under analysis by considering a modification to the loss function, in order to introduce a volume-weighting principle. Another key feature in the considered application is the implementation of a dropout procedure. The hyperparameters are fine-tuned in order to find the optimal network configuration. The model is tested via application to a sample of artificial mortgages on a 12 year time window; the tests show that the Neural Network approach outperforms alternative more standard approaches - such as a Kaplan-Mayer estimator or a Logistic estimator. The application of the model to floating rate loans leverages on the inclusion of a float-to-fix incentive feature, which proves to be effective in a market characterized by general low interest rate levels.

Appendices

Appendix A Neural Network Specifications

First of all, the model parameters are initialized by using the He normal scheme [10], which consists in drawing, for each layer, samples from a truncated normal distribution centered at 0 and with standard deviation equal to $\sqrt{\frac{2}{J^{[\ell]}}}$, where $J^{[\ell]}$ is the number of nodes of the layer. In short, the particular form of the drawing distribution helps to choose starting parameters which are not too small or too large, otherwise the calculation performed through each layer would rapidly make the result too tiny or too massive to be useful.

For what concerns the optimization procedure, for the model on the floating rate perimeter the *Adam* optimizer is adopted ([11]) which, in short, updates the network parameters according to the following steps:

$$
\begin{aligned}
m_t &\leftarrow \beta_1 m_{t-1} - (1-\beta_1)\nabla_w \mathcal{L}(w_{t-1}) \\
v_t &\leftarrow \beta_2 v_{t-1} - (1-\beta_2)\nabla_w \mathcal{L}(w_{t-1})^2 \\
\hat{m}_t &\leftarrow \frac{m_t}{1-\beta_1^T} \\
\hat{v}_t &\leftarrow \frac{v_t}{1-\beta_2^T} \\
w_t &\leftarrow w_{t-1} + \eta \frac{\hat{m}_t}{\sqrt{\hat{s}_t + \epsilon}}
\end{aligned}
$$

where:

- η is the *learning rate*, a factor which generally multiplies the vector which updates the model parameters,
- $\nabla_w \mathcal{L}(w)$ is the plain gradient of the loss function respect to the all model parameters w,
- m is the exponential moving average of the gradient (*biased first moment*), with decay rate given by β_1,
- v is the exponential moving average of the squared gradient (*biased second moment*), with decay rate given by β_2,
- $\hat{m}$ and $\hat{s}$ are bias-corrected versions of m and s respectively, i.e. they correct the bias which is introduced by in the initial update steps due to the value of the initialization (which consists of 0s) and which is persists when the decay is slow (i.e. for low β_1 and β_2)
- ϵ is a smoothing term.

In our case, we use the default values adopted by Keras for the parameters above, i.e.

$$\eta = 0.001 \qquad \beta_1 = 0.9 \qquad \beta_2 = 0.999 \qquad \epsilon = 1e-07$$

For the model on the fixed rate perimeter, instead, the *Adagrad* optimizer is adopted ([12]), which updates the network parameters in the following way:

$$w_t \leftarrow w_{t-1} + \frac{\eta}{\sqrt{G_{t-1} + \epsilon}} \odot \nabla_w \mathcal{L}(w_{t-1})$$

where the definitions of η and $\nabla_w \mathcal{L}(w)$ are the same as above, while:

- G_t is a diagonal matrix where each element $G_{t,ii}$ is given by the sum of the squares of the gradients respect to w_i up to step t,
- $\odot$ stands for the matrix-vector product.

In our case, the following initial values are adopted in Keras for η and the square gradients sum respectively:

$$\eta = 0.001 \qquad G_{0,ii} = 0.01$$

The following hyper parameters complete the details of the calibration process which are directly specified by the user in the current configuration:

- *number of epochs*: the process composed by forward and backward step is repeated across the entire database 30 times,
- *batch size*: each forward and backward step is performed on a mini batch containing 20000/10240 contract observations for the floating/fixed case respectively,
- *dropout rate*: the probability that each node is shut down, at each forward step, at the moment in which information pass through the dropout layer is set at 10% for the floating rate mortgages model, while dropout is not used on the fixed case,
- *patience*: at the end of each forward step, loss is computed on the validation set and if it stops decreasing for 10 consecutive epochs (and the total number of epochs has not been reached yet), then training finishes and model parameters are restored from the epoch with the best value of the validation loss.

References

[1] Basel Committee On Banking Supervision. Standards - interest rate risk in the banking book. *BIS*, 2016.

[2] Justin Sirignano, Apaar Sadhwani, and Kay Giesecke. Deep learning for mortgage risk. *SSRN Electronic Journal*, 07 2016.

[3] Taiyo Saito. Mortgage prepayment rate estimation with machine learning. *Master's thesis, Delft University of Technology*, 2018.

[4] Robben Riksen. Using artificial neural networks in the calculation of mortgage prepayment risk. *Master's thesis, University of Amsterdam*, 2017.

[5] Aurlien Gron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, Inc., 1st edition, 2017.

[6] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning.* 2020. `https://d2l.ai`.

[7] J. Brownlee and Machine Learning Mastery. *Deep Learning with Python: Develop Deep Learning Models on Theano and TensorFlow Using Keras.* Machine Learning Mastery, 2017.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016.

[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[11] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[12] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011.