Modelling Probability of Defaults with Machine Learning Algorithms for practical implementation

Maximilian Arrich Florian Benkhalifa Hendrik Halsband

Maximilian Tragl

June 14, 2019

Code available at: https://github.com/ArrichM/Machine-Learning

Abstract

In this paper we fit several machine learning algorithms to classify defaults of single loan payments. Precisely, we consider Logistic Regressions, LogitBoost, Neural Networks as well as Support Vector Machines and employ a repeated k-fold cross validation. For this purpose, we use panel data on more than 600,000 loan payments that include loan-, borrower as well as macro-characteristics over 60 periods. After cross-sectioning the data, we employ the mentioned techniques to obtain inferences about the prediction power. To choose the best algorithm we apply an extensive model selection process, capable to deal with heavy imbalances in the target first, and to eventually filter out the highest performing model. We attest the highest predictive power towards averaged Neural Networks as compared to logistic regression, while LogitBoost and Support Vector Machines show a rather dismal performance. Particularly the Neural Network achieves to increase both the classification of true positives and true negatives. We conclude that the standard approach in the credit industry can be clearly outperformed by suitable machine learning algorithms.

Contents

List of Tables

List of Figures

1 Introduction

Risk management is the core business of credit institutions, since it prevents financial frictions and aids banks in carefully exercising their role as intermediaries in financial markets. Among the variety of different threats a bank has to deal with, such as market-, liquidity-, model- or operational risk, especially credit risk poses one of the largest perils. Sound modeling ensures not only solvency but also profitability and thus delivers the basis for financial success. Besides these economic reasons, also regulation has become more and more sophisticated since the financial crisis and banks need to comply with more complex laws, with Basel III being the most prominent example. In order to comply with these requirements and to assess risk as accurately as possible financial institutions are allowed to either rely on external or internal rating systems (Hallblad, 2014). Unsurprisingly, many banks have developed internal models, commonly denoted as the "internal ratings-based" (IRB) approach. Despite the aforementioned significance and complexity of credit risk modelling, these IRB models, to our knowledge, usually rely on standard logistic regressions (LOGIT) (Jagric et al., 2011, p. 384). Nonetheless, this procedure is highly inflexible when it comes to modelling decision boundaries other than linear and thus different algorithms might pose more powerful alternatives in some situations.

In this context, the topic of credit risk modelling has been discussed in several previous studies. For example, Jagric et al. (2011) find that neural networks outperform LOGITs in default predictions. In a later publication, also Bagherpur argues that non-linear and non-parametric algorithms might be better suited for default prediction models 2017.

But while most scholars usually focus either on a single machine learning algorithm or compare two standard approaches, we use a broad set of techniques to select the most suitable solution. We consider this novel approach vital for practical implementation as each algorithm performs best under different circumstances. To achieve this goal, we apply several machine learning methods such as Neural Networks and Logistic regression in order to model the credit risk for single loan payments. Since each loan payment can only adopt one of two states - default and paid/non-default - we model a binary outcome variable.

In credit risk management, simple logistic regressions are predominantly employed to classify credit risk data. However, due to the inflexibility of logistic regressions when it comes to modeling decision boundaries other than linear, prediction power might be limited (James et al., 2013, p. 184). Therefore, the machine learning algo-

rithms used in this paper seek to model more complex dependencies and do not rely on strict assumptions about the underlying distributions.

To extract the best performing machine learning algorithm, we employ an extensive model selection procedure tailored to the characteristics of credit risk data.

Credit risk data typically exhibits imbalances in the target variables, meaning that defaults occur significantly less often that non-defaults. This needs to be considered, otherwise the algorithm will not be able to recognize minor classes. To overcome this challenge we employ several subsampling techniques, such as down- and upsampling to correct for such imbalances. To choose the most accurate technique we train all our models to each of these approaches and eventually choose the approach yielding the best overall performances.

After having selected the subsampling method, we proceed with choosing the best machine learning algorithm. After tuning each model to its highest performance through repeated k-fold cross-validation, we test for significant differences between the performances and we finally conclude the support vector machines to yield the most accurate results.

The economic rationale behind identifying the best machine learning classifier is obvious: The better one manages to model whether a credit defaults or not, the lower is the risk stemming from future unexpected defaults. Further, since banks are required to keep certain cash reserves, a more accurate modelling of default risk reduces the amount of excess cash required since the uncertainty of credit defaults may be reduced. More advanced modelling techniques thus lead to faster, cheaper and more reliable credit granting processes and allow for economic success of the credit institution.

The remainder of this paper is organized as follows: Section 2 provides an overview over various machine learning algorithms used in this paper to model credit default probabilities. Section 3 presents the data set used in this paper and also discusses selected descriptive statistics. Section 4 presents the methodology used to test and compare the aforementioned machine learning algorithms. Finally, in section 5, we summarize the results and conclude.

2 Machine Learning Algorithms

In the following, we present a selection of machine learning methods to predict individual mortgage loan defaults. Since the outcome variable is binary, the employed algorithms are bi-variate classifiers in a supervised learning framework (Khandani et al., 2010, p. 5). Such machine learning techniques are, for instance, random forests and neural networks which enjoy great popularity among machine learning

applications in general as well as in credit risk frameworks in particular (Petropoulos et al., 2018, p.7). We also consider boosted logistic regression, a promising method in the banking industry due to its parametric structure allowing for more transparency (Kirori and Ogutu, 2013, p. 34).

2.1 Logistic Regression

For discrete binary variables, such as predictions on credit defaults, modelling based on linear regression might result in PD-predictors $\notin (0, 1)$. These predictions are not sensible, since of course any default probability must fall within this range (James et al., 2013, p. 131). A logistic regression, as used by Frame, Gerardi, and Willen (2015) to forecast loan defaults, is one of the most widely used techniques to overcome this type of classification problem and is, to our knowledge, also the standard approach used by most financial institutions (Jagric et al., 2011, p. 384).
Let π be a binary random variable that equals one if the loan is defaulted and zero otherwise. Suppose $X = (x_1, ..., x_p)$ are exogenous variables. Then the conditional probability of a default follows a function of the form:

$$\pi(x) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}} = \frac{1}{1 + e^{-(\beta_0+\beta_1 X)}} \tag{1}$$

That has the attractive characteristics that $\pi(x)(0, 1)$ for all $x \in \mathbb{R}$. Rearranging this formula leads to:

$$G(x) = ln(\frac{\pi(x)}{1 - \pi(x)}) = \beta_0 + \beta_1 X \tag{2}$$

This transformation, G(x) is called the logit transformation and has many desirable characteristics of a linear regression model, such as linearity and continuity. This logistic regression model is fit to the training data by estimating β_0 and β_1 via maximum likelihood (instead of an OLS) approach (Bolton, p. 42).
Although we apply this procedure to calculate the likelihood of discrete variables, this approach resembles a simple method for binary classification of default events. This is the case since the values are as close as possible to either 0 or 1 as we predict one if the index exceeds a certain threshold (Horlemann, 2019). But as stated above, this procedure lacks flexibility in non-linear environments and might thus reduce the ultimate prediction accuracy. To emphasize that credit markets have in fact non-linear properties, Jagric et al. (2011) compare learning vector quantization (LVQ) neural networks with a LOGIT benchmark and find underperformance of the LOGIT strategy. Hence, we also expect the following algorithms to yield a better fit for predicting credit risk.

2.2 Neural Networks

Different tasks use different neural networks depending on input data types and prediction purposes. Consequently, before implementing a neural network approach one must clarify which approach one uses in particular since the term is commonly used as a container for a plethora of neural network algorithms. Commonly used neural networks in computer vision or speech recognition, for instance, are long/short term memory networks (LSTM) or recurrent neural networks (RNN) due to their ability to "remember" specific input data such as words or larger text blocks (Donahue et al., 2015, p. 2626). In credit risk assessing, however, we aim to model a binary classification problem. Hence, the required output of the neural network model to be implemented shall generate a binary value. This binary value may then be used as a classifier that will help banks to identify whether a specific borrower will default or not default her credit obligation. Since the data used in this paper is already correctly labelled, e.g. it is known whether an individual defaults her loan or not, we are in a supervised learning setting. For all these reasons we will rely on a classical feed-forward neural network as the workhorse model for the class of neural networks in this paper.

Neural network models are an attractive concept for credit risk modelling because they are both more flexible and capable of modeling complex non-linear functions compared to classical statistical models like linear discriminant analysis and logistic regression (Baesens et al., 2003, p. 312). For example, a logistic regression model is easy to interpret due to its additive linear combination of inputs and weights and is also adjusted by a learning algorithm. But it will give low accuracy results on complex non-linear relationships. However, for a neural network model using a logistic function, its higher number of hidden layers (H) allows it to learn complex non-linear relationships. Basically, a Neural Network with the value of H=0 and two output neurons breaks down to be a logistic regression (Bhoge, 2019). For our study, we apply a classifier named average neural network. As the name suggests, many neural networks are fitted and predictions are obtained by averaging. The corresponding abbreviation used in this paper is "avNNet".

2.3 LogitBoost

Boosting is one of the major recent developments in classification methodology. This algorithm has originally been proposed as ensemble method which relies on the principle of generating multiple predictions and majority voting (averaging) among the individual classifiers (Bühlmann et al., 2007, p. 1): A classification algorithm is sequentially applied to reweighted versions of the training data to create a weighted

majority vote of the sequence of classifiers thus produced (Bauer and Kohavi, 1999, p. 106).
For many classification algorithms, this innovative strategy results in dramatic improvements in performance (Agresti, 2007, p. 137). Both LogitBoost and MART (multiple additive regression trees) can thus be viewed as generalizations to the classical logistic regression. If one considers AdaBoost as the underlying generalized additive model and then applies the cost function of logistic regression, one can derive the LogitBoost algorithm.

2.4 Support Vector Machines

The support vector machine (SVM) is a generalization of the so-called maximal margin classifier. In contrast to the maximal margin classifier, the SVM has the twist that it can be applied to datasets whose classes are not separable by a linear boundary. Support vector machines seek to overcome this obstacle and can thus be used for both binary and multiclass classification. The basic idea is to separate the classes using a hyperplane. For instance, a hyperplane is a flat affine subspace of dimension p-1 in a p-dimensional space, a line in a 2D space and a plane in a 3D space. The maximal margin, or optimal separating hyperplane, is the separating hyperplane that is farthest from the observations. We thus calculate the perpendicular distance from each training observation given a hyperplane. The optimal separating hyperplane then is the one with the largest margin. The observations tangent to the hyperplane are the support vectors. The separating hyperplane only changes if these support vectors change. If other observations change, the separating hyperplane is not affected as long as these observations do no move closer to it than the support vectors (Friedman et al., 2001, p. 417). However, since hyperplanes cannot be "bent", one needs to work with kernel densities if the data is not linearly separable. The kernel trick implicitly maps the data inputs into high-dimensional feature spaces. Ideally, the data then becomes linearly separable in this higher dimensions. The derivation of a mapping into a higher dimensional space is quite technical and would go beyond the scope of this study. The corresponding abbreviation used in this paper is "svmRadial".

3 Data

3.1 Sample Selection

The dataset for this analysis is a randomized selection of mortgage-loan-level data and consists of the portfolios underlying U.S. residential mortgage-backed security

(RMBS) portfolios. The information was provided by International Financial Research and derived from Bart Baesens' website[1], which has been made available in 2016 (Baesens et al., 2016). The data contains comprehensive panel data on macroeconomic factors as well as holding information, origination and performance observations for 50,000 residential U.S. mortgage borrowers with 622,489 individual loan payments over a 60-year period. Naturally, one loan thus consists of several loan payments. If a borrower defaults his payment, he basically defaults a single loan payment, not necessarily the entire loan. That means that we consider some individual loans multiple times in the sample until they default or reach maturity. This might imply default dependency as loan- and borrower characteristics for one individual loan can be expected to remain fairly similar through the years. However, since the maximum number of payments per borrower amounts to only 60, we assume i.i.d variables. As in the real world, loans may originate before the start of the observation period. The loan observations may thus be censored as the loans mature or borrowers refinance.

3.2 Summary Statistics

In total, we observe 15,158 defaulted loans, which represents around 2.4% of the whole sample of more than 600,000 observations. The summary statistics for the defaulting and non-defaulting credits are displayed in table 1.

We compare the means of the credits across relevant credit- and macro-characteristics and examine the differences between the means of defaulting and non-defaulting loans. To our surprise, credit balance and LTV-ratio are negatively related to credit default. While, GDP-growth is on average 0.5% lower in years of defaults, (indicating that the macroeconomic situation may indeed be somehow related to credit quality) employment ratios do not seem to influence the observed defaults. On the other hand, it should come as no surprise that the FICO-score is negatively related to credit default. Figure 1 shows Gaussian kernel estimates of the densities of the separate explanatory variables conditional on defaulting or surviving. The plots allow for a first, narrow assessment of the predictive information held in the data. The more the two conditional densities differ, the more predictive information can be suspected.

Fore specific plots above we thus expect GDP and unemployment rate (ue r_time) at observation time to carry strong predictive impact for default classification while the time to maturity or account balance seem to be almost independent from defaulting. These plots are to be handled with care since only conditioning on observing a default or not is controlled for. In a multivariate model, many other dependencies are

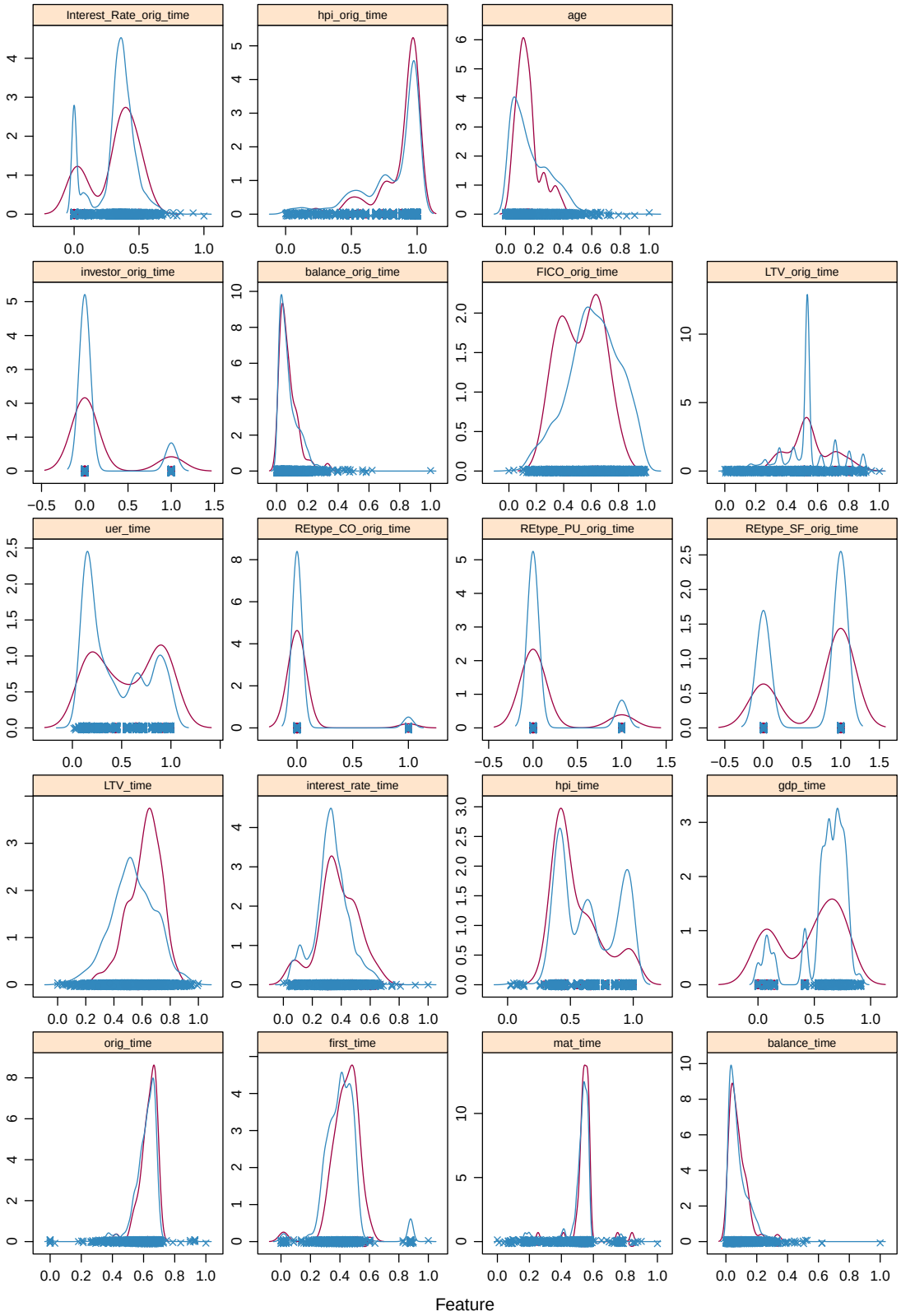
[1] Accessible through http://www.creditriskanalytics.net/datasets-private.html

controlled for such that variables which from the plotted densities seem not to carry interesting information can turn out to be of pivotal importance.

Table 1: Characteristics of Defaulting and Non-Defaulting Credits. This table reports the summary statistics for loan-level observations of U.S. residential mortgage-backed security (RMBS) portfolios in the sample. It displays average credit and macroeconomic characteristics comparing defaulting credits with those that do not default. The variable balance_time represents the outstanding balance at observation time; LTV_time is the loan-to-value ratio at observation time, in %; REtype_CO_orig_time takes a value of 1 if the real estate type is condominium, otherwise 0; REtype_SF_orig_time takes a value of 1 if the real estate type is a single-family home, otherwise 0; REtype_PU_orig_time takes a value of 1 if the real estate type is planned urban development, otherwise 0; investor_orig_time takes a value of 1 if the real estate was acquired as a financial investment, otherwise 0; interest_rate_time is the interest rate on the loan at observation time, in %; gdp_time is the growth of the gross domestic product (GDP) at observation time, in %; uer_time is the Unemployment rate at observation time, in %; and FICO_orig_time is the FICO-Score at the time of credit origination.

	Default	Non-Default
id	24,825.310	25,148.900
time	35.706	35.802
orig_time	22.902	20.518
first_time	26.147	24.568
mat_time	142.862	137.073
balance_time	252,550.700	245,890.100
LTV_time	96.946	82.732
interest_rate_time	7.546	6.680
hpi_time	176.659	184.288
gdp_time	0.553	1.402
uer_time	6.855	6.509
REtype_CO_orig_time	0.068	0.068
REtype_PU_orig_time	0.122	0.125
REtype_SF_orig_time	0.623	0.612
investor_orig_time	0.129	0.139
balance_orig_time	253,540.600	256,435.600
FICO_orig_time	649.016	674.284
LTV_orig_time	81.026	78.925
Interest_Rate_orig_time	5.881	5.645
hpi_orig_time	209.456	197.866
default_time	1	0
payoff_time	0	0.044
status_time	1	0.088
age	12.804	15.283

Figure 1: Variable Exploration: Densities The plots below show the densities of the explaining variables conditional on defaulting or surviving. The densities were estimated using a Gaussian kernel.



4 Methodology

We evaluate several machine learning algorithms which we find to be the most promising. In a first step, we control for the heavy imbalance inherent to the data evaluation resampling and data synthesizing techniques. After settling for one methodology, the long-listed algorithms are fitted using 2-times repeated 5-fold cross validation. Finally, we apply a number of performance measures to detect the best performing one.

4.1 Resolving Imbalance

Data from credit portfolio observations often suffer from an imbalance in the observed outcome variable. Due to the nature of the data, defaults are underrepresented and the majority of the observations are non-default observations. This can lead to a serious bias in the estimation process in favour of the majority. An algorithm blindly classifying every observation as a non-default observation would still get a very high accuracy of 97.6% but would never predict a default, besides degenerating to a highly trivial prediction method. To resolve problems of minority class representations, there exists a series of different subsample techniques. Examples of such sampling methods are enumerated below:

1. Oversampling defaults: This approach involves explicitly surveying the minority group more frequently. This is not an option in our case since we do not sample the data ourselves. Furthermore, since in general we observe the whole default portfolio, oversampling of defaults is not possible since it would imply not recording non-defaults. Besides, oversampling the minority, e.g. defaults can lead to model overfitting, since it will introduce duplicate instances, drawing from a pool of instances that is already small (Kim et al., 2015, p. 1075).

2. Undersampling non-defaults: When we split the data into training and testing data, we do it in an i.i.d. way, meaning that every observation has the same probability to enter into either set. This results in the same default-/non-default ratio in the training set as we have in the total data set. In order to train our algorithms on a balanced data set, we designate 50 percent of the training data set to consist of defaults and sample at random from the defaulted observations until half of the training set is made up of defaulting positions. The remainder is then filled up using non-default observations. This results in a balanced training set. This process is preferable to the creation of new data points described in the following section, however it introduces a large loss of data if the imbalance

is large.

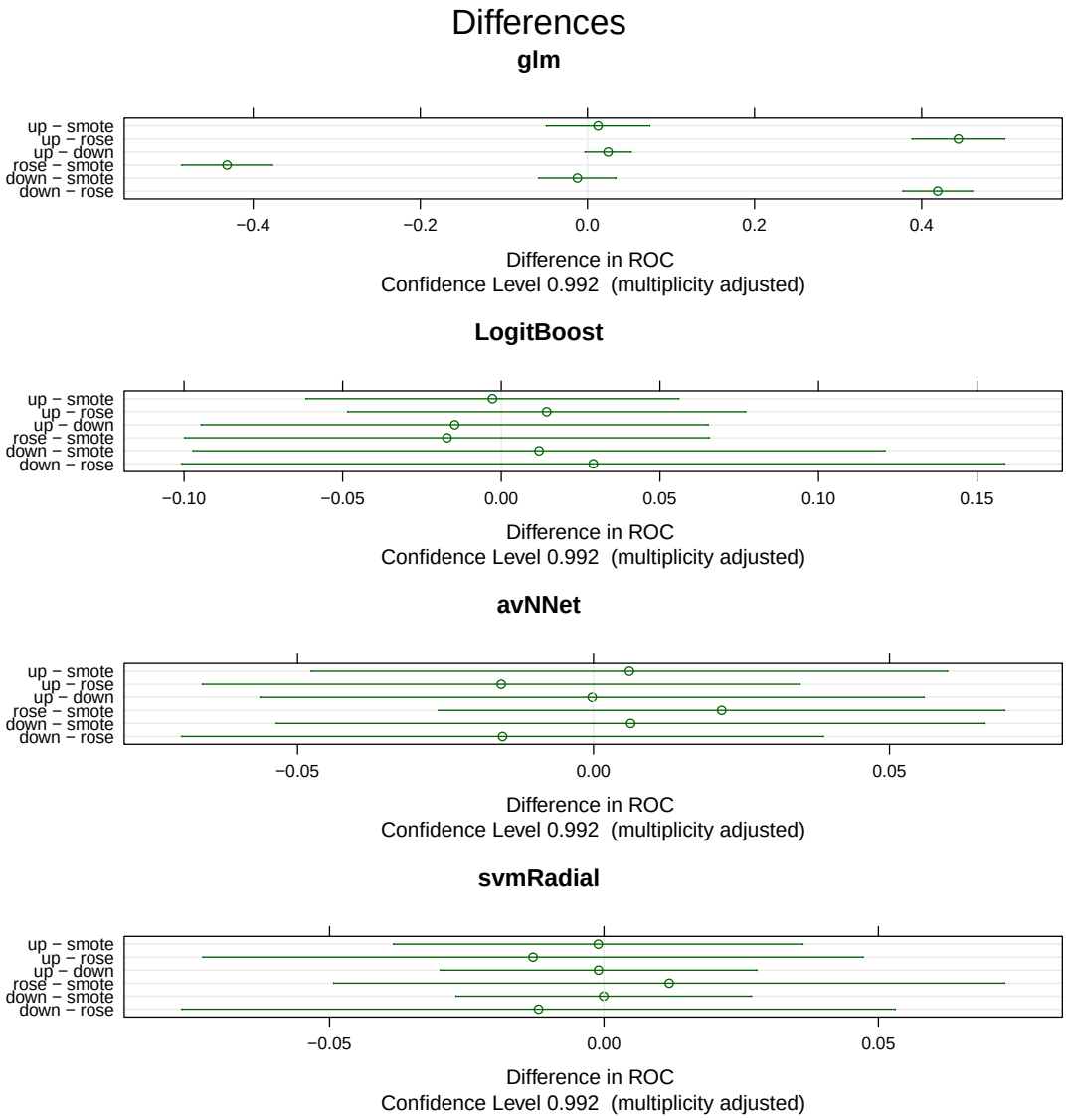
3. Synthesize new data points: Known as hybrid method (CITE), techniques as SMOTE and ROSE synthesize new data points in the underrepresented class. SMOTE, in particular, creates new instances of the minority class by forming convex combinations of neighboring instances. Subsequently, it draws lines between minority points in the feature space, and creates new data points along these lines. The advantage of this procedure is that it allows to balance the data-set without as much overfitting as in the previous approaches, as new synthetic examples rather instead of duplicates are used. This, however, does not prevent all overfitting, as the synthetic points are still created from existing data points. (Chawla et al., 2002, p. 322).

4. Manipulate the machine learning algorithm itself: Make the optimization algorithm weigh false classifications of the minority class more severe. This approach, called Cost-sensitive Learning uses a cost function C(non-default, default) that specifies the cost of misclassifying an instance of class default as class non-default. This allows to penalize misclassifications of the minority class more heavily than we do with misclassifications of the majority class, in hopes that this increases the true positive rate. A common scheme for this is to have the cost equal to the inverse of the proportion of the data-set that the class makes up. This increases the penalization as the class size decreases. While this technique might improve performance, it hinders convergence and is in general not applied if one of the alternatives above can be used (Elkan, 2001, p. 974).

Due to the heavy imbalance of the classes, we spend much effort to handle the disparity. Although, manipulation the machine learning algorithm might seem a valid alternative, academic evidence suggest that these methods often perform poorly (CITE). Therefore, we focus on extracting the best subsampling technique of the above presented. Our approach consists on fitting the selected machine learning algorithms on the data for each of the subsampling methods. By comparing for significant differences in the ROC performances between the same model under different subsampling methods we are capable to the detect the best performing method. Due to very large memory usage of the fitted objects, we restrict the analysis to a subsample of 200000 observations of the original data. To compare their performances, we can apply a simple t-test to evaluate the null hypothesis that there is no difference between each of the selected models. More specifically, we employ a t-test to check whether the collection of hypothesis differs significantly from 0. In this case we chose a confidence interval of 95% which after applying a Bonferroni correction

corresponded to an interval of 99.2%. The resulting dotplot is displayed in figure 2.

Figure 2: Sampling Performance Differences: We fit each model for all sampling methods using the described cross validation.



The multiplicity adjusted confidence level reflects the chance of getting this differences within a collection of hypotheses when the null hypothesis was true [Wright, 1006]. In this case, the adjusted confidence level is 99.2% and the null, that there is a difference, cannot be rejected, when the dotplots include the value 0.It can be seen that, although the differences in performance are not significant at the chosen level, upsampling mostly dominates the other techniques. As a consequence, we choose upsampling for the fitting of our final model. Note, that due to computational restriction, we were only able to perform the tests for the best subsampling on relatively

small training sets. Results might turn out different if we employed our subsampling model selection method on the complete dataset.

4.2 Repeated k-fold Cross Validation

Cross validation evaluates the performance of machine learning programs through resampling, the process of refitting a model based on repeatedly drawn samples from a training set. As it allows to evaluate a model's performance from a limited data sample it is especially useful in practical applications, such as this analysis. Besides delivering better estimates for the test error associated with a model it can also be used to pick the best parameters for a model (James et al., 2013, p. 175). In this paper, however, the main purpose of employing cross-validation will be model selection. Several strategies of cross-validation exist, but we restrict our focus on repeated k-fold cross-validation. Simple k-fold cross-validation generally consists of choosing a positive integer for k, which denotes the number of groups the underlying data is splitted in. The procedure can then be described as the following:

1. The dataset is shuffled randomly
2. The dataset is separated into k groups
3. For each unique group:

   One group is selected as test data set

   The remaining groups are chosen as training data sets

   A model is fitted on the training sets and evaluated on the test set

   The evaluation score is retained and the model discarded
4. The skill of the model is summarized using the sample of evaluation scores

Repeated k-fold cross-validation simply repeats the whole procedure, while reshuffling the data sample prior to each repetition. Further, each parameter can be "tuned" by applying the repeated cross-validation for a set of possible values of the adjustable parameters and computing the average cross-validation for each parameter. Eventually, the parameter delivering the lowest error is chosen and the corresponding model selected. As we want to improve the predictive power of the model, we target to minimize the average Receiver Operator Characteristic (ROC), which will be explained in the following chapter.

4.3 Model selection

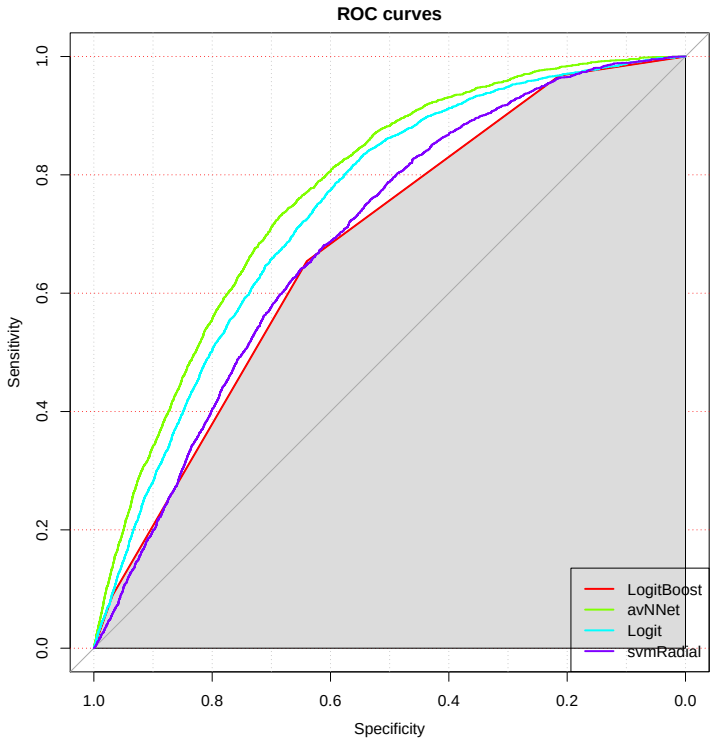
Performance measures to assess forecast accuracy are predominantly obtained by confusion matrices which depict the predicted classes against the true classes. Based

on the confusion matrix a series of easy-to-interpret evaluation metrics can be derived, Another metric of importance is which gives the rate of, how often is it correct?

1. Accuracy: The ratio of correct predictions compared to all predictions: $ACC = \frac{TP+TN}{TP+TN+FP+FN}$.
2. Sensitivity (or True Positive Rate): A measure of how many true positives were identified as compared to all identified positives $TPR = \frac{TP}{TP+FN}$.
3. Specificity (or True Negative Rate): A measure of how many true negatives were identified as compared to all identified negatives $TNR = \frac{TN}{TN+FP}$
4. Cohen's Kappa: A measure of how well the classifier performed as compared to how well it would have performed by chance

Beyond these metrics there exist more sophisticated methods to evaluate the performance. For this paper, we primarily employ the ROC, a widely used metric in model evaluation, produced by the fitted models to evaluate performance. The ROC represents the ratio of the probability mass of correct and incorrect predictions. The curve depicted in 3 shows the ratio of the true positive rate (Sensitivity) and the false positive rate as a function of a separating probability border. If there are no misclassifications, we observe a ROC point in the ROC space at (1,0). If there are only misclassifications, we observe a point at (0,1). The printed ROC curve is thus not plotting the ROC directly, but instead the two entities from which the ratio is defining the ROC. The diagonal in the ROC space is describing random classification. Note that the two corners on the diagonal of the plot can always be reached by classifying all observations as positive or negative respectively. Thus, a model for which all points in ROC space lie above the diagonal is superior to randomly assigning positive or negative attributes to the observation. Note that a notoriously bad predictor can be inverted to obtain a good one. Since points on the ROC curve that lie to the north of the ROC space are always superior to those below them, we can concisely concentrate the information held in the ROC curve plot by looking at the area under the curve (AUC). The larger this area is, the better model performs. This area is then used to rate the models performance with higher AUC values being superior to lower ones (James et al., 2013, p. 148).

Figure 3: ROC curves for all tested algorithms. The grey are corresponds to the AUC of the Boosted Logit model.



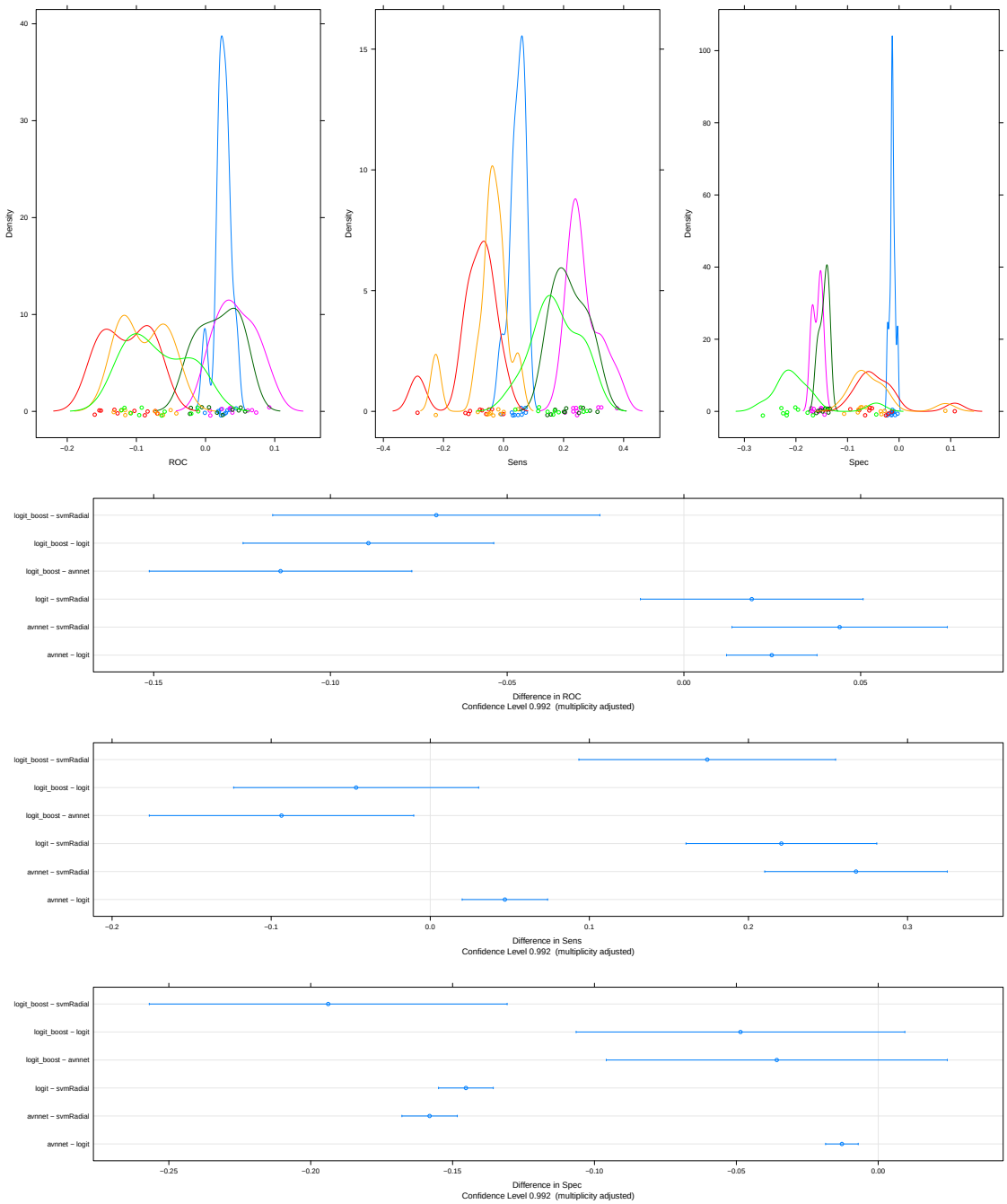
From figure 3 we can read the ROC performance of the fitted models. The more the plotted curves are to the northwest of the ROC space, the higher we expect the performance of the model. Already by eyeballing the plot, we conclude that the averaged Neural Networks (avNNet) perform best. In order to concentrate the information hold in the ROC curve, we calculate the area under the curve:

Table 2: AUC of the fitted models

	Logit_Boost	avNNet	Logit	SVM
AUC	0.679	0.772	0.742	0.692

As assumed, the avNNet algorithm is rated best. To investigate the significance of the superiority, we perform a series of t-tests, again choosing a 95% significance level and applying a Bonferroni correction.

Figure 4: Evaluation of the ROC Differences. The colors correspond to the differences depicted in the dotplots from top in the following order: Green, Orange, Red, Olive, Pink, Blue.



The densities depicted in figure 4 already hint to the avNNet algorithm being the best performing one. The dotplots show the statistical significance of the superior performance of the avNNet according to the ROC statistic. The avNNet also dominates when it comes to Sensitivity. When it comes to differences in Specificity, the

picture is quite different with the Support Vector Machine algorithm showing the best values. Overall however, the avNNet clearly is superior to the other applied algorithms and is chosen as the final model.

Table 3: Confusion Matrix for the Boosted Logit Classifier

	default	non.default
default	1571	35127
non.default	831	62471

Table 4: Confusion Matrix for the avNNet Classifier

	default	non.default
default	1867	35787
non.default	535	61811

Table 5: Confusion Matrix for the Logit Classifier

	default	non.default
default	1718	34097
non.default	684	63501

Table 6: Confusion Matrix for the svmRadial Classifier

	default	non.default
default	5	198
non.default	2397	97400

Average Neural Network was selected as best performing model in our model selection procedure. The confusion matrix with all the corresponding metrics are presented below. It can be seen, that avNNet captures a good amount of true positives but also exhibits numerous false positives. Especially compared to the classical logistic regression classifier, the average Neural Network performs significantly better predicting both defaults and non-defaults more accurately. Interestingly, boosted Logistic Regression and the Radial Support Vector Machine performs worse than

logistic regression.

5 Results and conclusion

The analysis showed that Neural Networks, as the best performing algorithm, could significantly increase the predictive power as compared to the logistic regression. However, from an economic point of view, this should be handled with care, since the model suggests an overly aggressive lending strategy where many defaulting obligors receive a loan. Nonetheless, compared to the industry standards of logistic regression, credit institutions would clearly benefit from including a neural network approach instead of simple logistic regressions since on an ordinal scale the avNNet could improve precision of obligor ratings.

From a technical point of view, our approach exhibits several caveats. First, we applied both our subsampling and model selection approaches only on a subset of the data. This is due to the restricted computationally power we had access to. Applying the algorithm to the whole dataset would clearly lead to more accurate results.

Another limitation of our procedure is the cross-sectioning of the whole data. By that, we loose valuable information. To make use of the panel data structure, alternative machine learning techniques capable to specifically handle time-series could have been employed, as for example recurrent neural networks. Again, computational power was one reason why we did not use recurrent networks, although this would have been an interesting alternative.

Third, although we try to control for the heavy imbalance of the data by testing different subsampling techniques, more advanced methods exist to handle this issue. Another approach, which could have been employed on top of subsampling, is to specify different probability thresholds. By k-cross validation this method allows not only to specify the optimal tuning parameter, but to also determine possible probability cutoffs. By that, both the specificity and the sensitivity can be modified until the area under the ROC curve achieves a maximum. This method, however, increases the number of fitted models drastically and the need for computational power. For further information see (Kuhn and Johnson, 2013, p. 262 ff.). For future research we expect more complex, time-series-considering machine learning algorithms to be of particular interest.

A Appendix

Figure 5: Variable Exploration: Boxplots

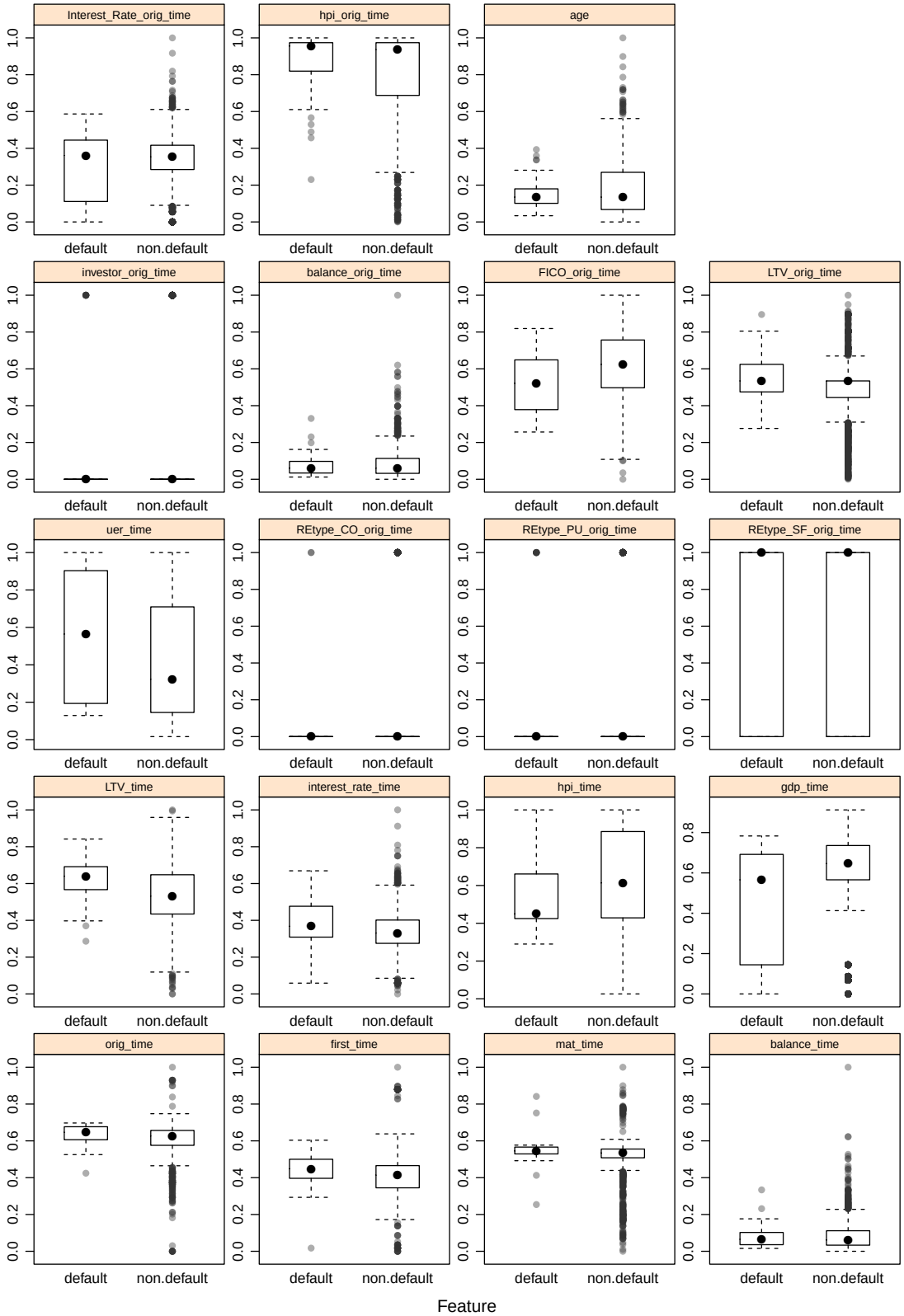
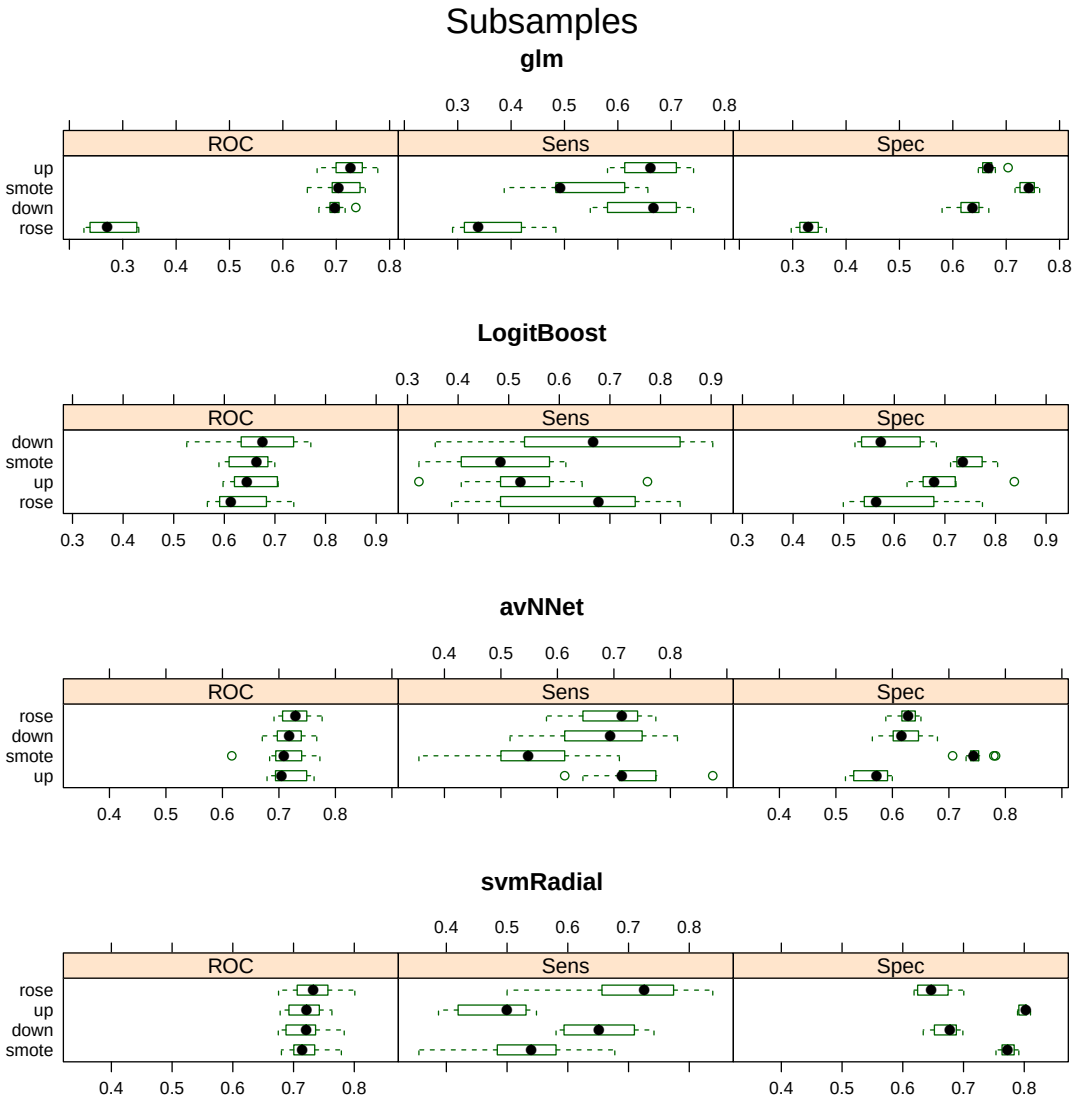


Figure 6: Within model comparisons.



References

Agresti, A. (2007). Building and applying logistic regression models. *An introduction to categorical data analysis*, pages 137–172.

Baesens, B., Roesch, D., and Scheule, H. (2016). *Credit Risk Analytics: Measurement Techniques, Applications and Examples in SAS*. Wiley, New Jersey.

Baesens, B., Setiono, R., Mues, C., and Vanthienen, J. (2003). Using neural network rule extraction and decision tables for credit-risk evaluation. *Management science*, 49(3):312–329.

Bagherpour, A. (2017). Predicting mortgage loan default with machine learning methods. Riverside.

Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139.

Bhoge, M. (2019). Using the artificial neural network for credit risk management.

Bolton, C.

Bühlmann, P., Hothorn, T., et al. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):477–505.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd.

Frame, W., Gerardi, K., and Willen, P. (2015). The failure of supervisory stress testing: Fannie mae, freddie mac, and ofheo. Federal Reserve Bank of Atlanta. Working paper series.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Hallblad, J. (2014). The multi-year through-the-cycle and point-in-time probability of default. Master's thesis, Umeå University, Sweden.

Horlemann, A.-L. (2019). Lecture notes in machine learning.

Jagric, V., Kracun, D., and Jagric, T. (2011). Does non-linearity matter in retail credit risk modeling? *Czech Journal of Economics and Finance*, 61(4):384–402.

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

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

Kim, M.-J., Kang, D.-K., and Kim, H. B. (2015). Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction. *Expert Systems with Applications*, 42(3):1074–1082.

Kirori, Z. and Ogutu, J. (2013). An application of the logitboost ensemble algorithm in loan appraisals.

Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*. Springer New York, New York, NY.

Petropoulos, A., Siakoulis, V., Stavroulakis, E., and Klamargias, A. (2018). A robust machine learning approach for credit risk analysis of large loan level datasets using deep learning and extreme gradient boosting. In *Ninth IFC Conference on "Are post-crisis statistical initiatives completed?"*. Bank of Greece.