Will machine learning and hyperparameter optimization become a game changer for credit scoring?

Authors: Knut Opdal[1,2], Rikard Bohm[1,2], Thomas Hill, PhD[3]

July 2017

Affiliations:

[1]StatSoft Norway AS (2006 – 2017).

[2]Precise Prediction UK Ltd (2017).

[3]TIBCO Software Inc.

Abstract: Machine learning algorithms and approaches have transformed analytic practice across various industries, and enabled more effective risk management in many domains ranging from manufacturing to insurance, health-care, or warranty analytics. Machine learning methods such as decision trees, neural nets, or deep learning (tensor flow) models apply algorithmic approaches to extract repeated (reliable) patterns from historical data, enabling more accurate predictions without depending on statistical models and assumptions as is the case with traditional logistic regression.

However, the most powerful machine learning techniques such as deep learning networks or gradient boosting are complex, and require analysts to "experiment" with many parameters to successfully steer the respective algorithms ("hyperparameter optimization"). The purpose of this presentation is to discuss the analysis of a real dataset describing consumer loans portfolios in the UK. Results from logistic regression models are compared against results derived from advanced machine learning methods, including gradient boosting and multilayer ("deep") neural nets. Model tuning using recent approaches to automatic hyperparameter optimization are also applied. The results are discussed in terms of the quality and interpretability of different models and approaches, and the effort required. Ultimately, we aim to address the question whether machine learning and automatic hyperparameter optimization represent disruptive technologies for risk management.

Keywords— machine learning, hyperparameter optimization

I. SCOPE

We want to demonstrate that it is possible to build score models with higher performance using machine learning techniques compared to industry standard logistic regression on Weight-Of-Evidence (WoE) transformed predictors (Siddiqi, 2016).

We will evaluate how much model performance (accuracy) can be increased using hyperparameter optimization to find the best parameter settings for the chosen machine learning technique. As an example, recently popular methods described as Stochastic Gradient Boosting (Hastie, Tibshirani, Friedman, 2009) and "deep learning" neural networks (see Schmidhuber, 2014) are applied to the example data problem as well.

We assume that the training and test samples are representative for the out-of-time validation dataset. This implies that we assume there have not been any major changes in the application process and the way credit information has been collected and derived during the training period.

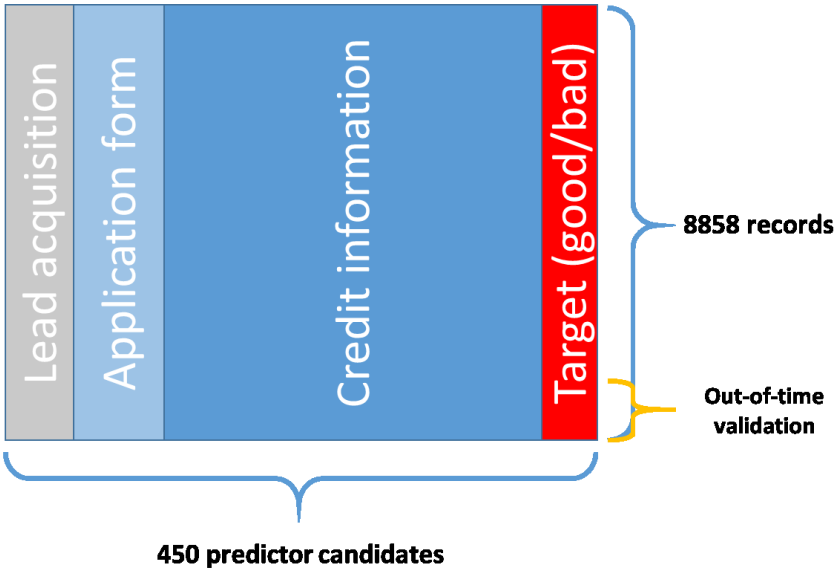
II. DATA

The example data discussed in this paper and presentation originated from customer data as typically collected by financial institutions. The data contain information on:

- lead acquisition (channel, lead price, other lead info)
- data provided on the application form (demographic information, employment status and salary, etc.)
- credit information (Experian, Equifax, CallCredit)
- other data (the method/device used for application, date and time of application)

Figure 1 summarizes the nature of the data and the available total number of records.

Fig. 1. Data Sources and Total Number of Records



III.DATA PREPROCESSING/DATA CLEANING

A. Data Preparation for Logistic regression:

A standard default risk model was created first, based on the typical analytic flow that includes data pre-processing and binning, followed by logistic regression with weight-of-evidence (WoE) coded inputs (Siddiqi, 2006). Specifically, the following data preparation steps were applied:

- Categorization of all variables as continuous or discrete, and calculation of the corresponding optimal WoE values (see below) to be used as predictor candidates for logistic regression
- Transformation of missing data values and outliers by assigning WoE values

The specific automated optimal WoE binning method is briefly illustrated below.

B. Data Preparation for Machine learning:

Continuous variables with text labels were replaced by -999998, -999997, and so on. This includes text labels like “Missing values”, “Not Derived”, “NA” and so on.

IV. WEIGHT-OF-EVIDENCE (WOE) TRANSFORMATIONS

A specific analytic tool (Statistica Version 13.2, 2017) was used to automatically bin all inputs (predictor candidates for the logistic regression analysis) to maximize the respective Information Values, subject to the constraint that the resulting WoE function over the binned levels of continuous inputs was monotone. This is illustrated in the screenshot of the tool shown in Figure 2, where the *Monotone* WoE graph is highlighted by a frame for the predictor candidate *Age*.

Fig. 2 Example of the Weight-of-Evidence (WoE) Coding Tool

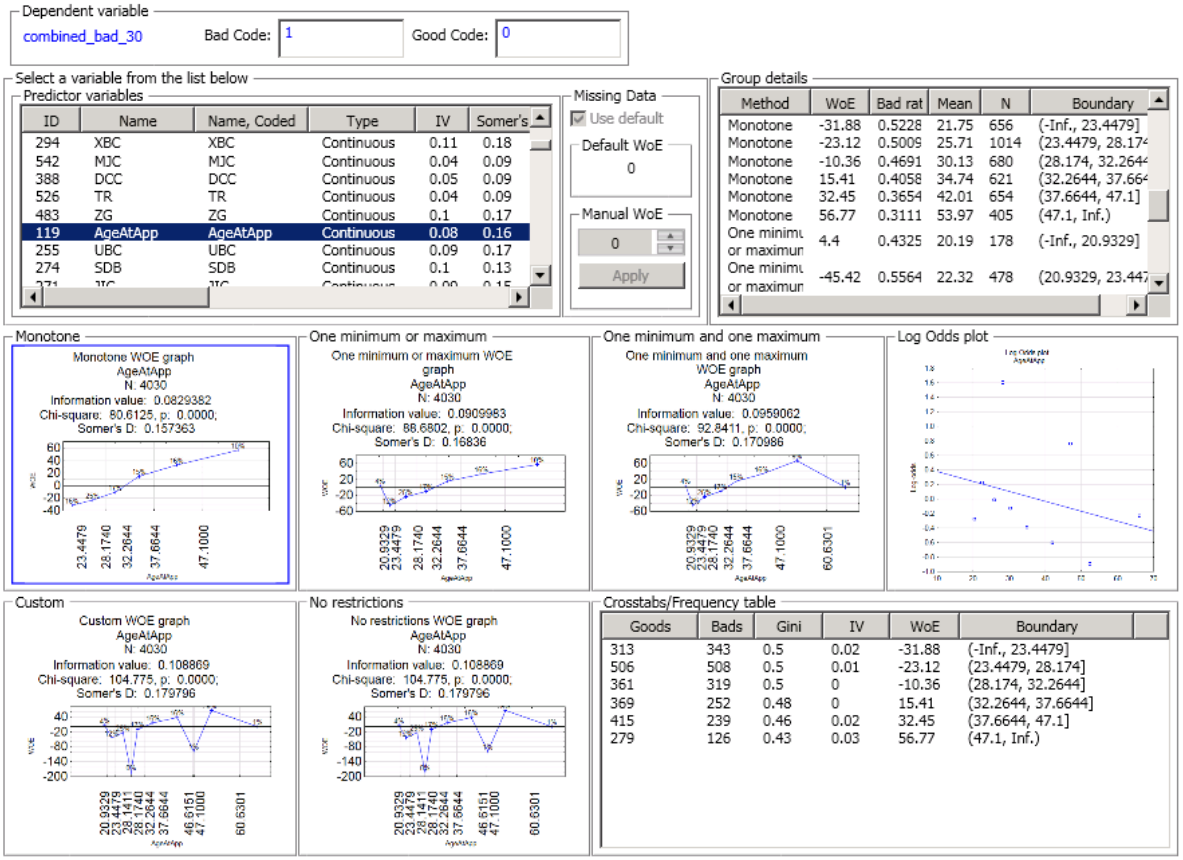


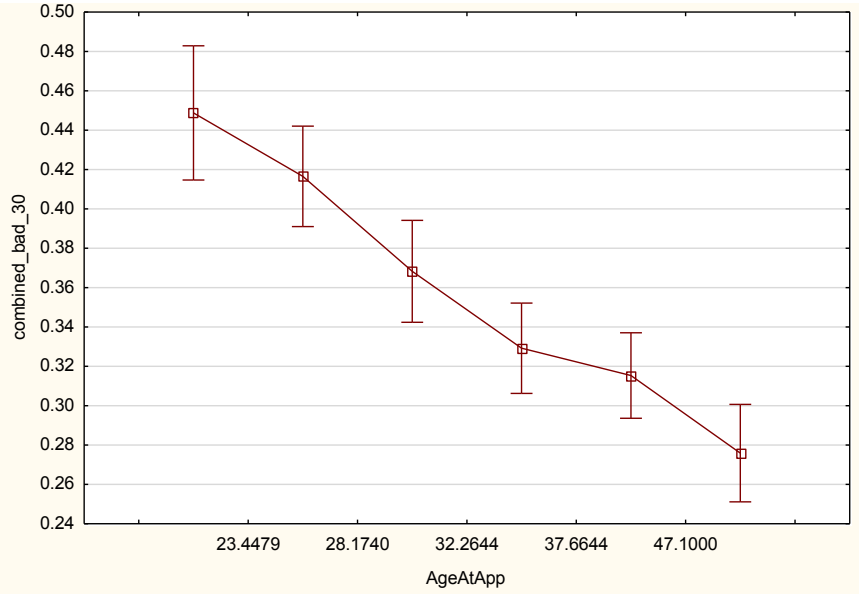
Table 1 shows the resulting coding for variable *Age*.

Tab. 1. Weight-of-Evidence Coded Predictor Age

Age group	AgeAtApp	Bad Rate	WOE Value	Number of loans
AgeAtApp <= 23.4479	21,7	0,45	- 31,9	885
23.4479 < AgeAtApp <= 28.174	25,8	0,42	- 23,1	1563
28.174 < AgeAtApp <= 32.2644	30,2	0,36	- 10,4	1460
32.2644 < AgeAtApp <= 37.6644	34,9	0,33	15,4	1765
37.6644 < AgeAtApp <= 47.1	42,0	0,32	32,5	1921
AgeAtApp > 47.1	53,2	0,28	56,8	1362

For example, given this coding applicants below 23 years are characterized by the same default risk. Further, the binning is chosen so that the WoE value and default risk decreases as a monotone function of age.

Fig. 3. Observed Default Rates vs. Age, Monotone Risk



V. TEST ENVIRONMENT

A. Definition of "Bad"

Installment/invoices not paid within 30 days after due date were defined as “Bad”. Installments/invoices before 30 days after due date were defined as “Good”. The target for loans that still had not reached 30 days after due date were defined to be missing.

B. Training, testing and validation data

The «last» 10% of the paid-out loans were used as out-of-time validation sample.

For all analyses 75% of the remaining observations were used for training and 25% for testing.

The training and testing samples were stratified by target and a generic score to make sure the training set was representative.

C. Metrics

The Area-Under-Curve or AUC statistic (Siddiqi, 2006) in the validation data sample was used as performance metric.

VI. FINDING THE BEST PARAMETER SETTINGS THROUGH HYPERPARAMETER OPTIMIZATION

Most machine learning methods can best be understood as pattern recognition algorithms, to identify repeated patterns in historical predictor data (the *X* variables) which allow for more accurate predictions of the outcomes of interest (the *Y* variables; Nisbet, Elder, Miner, 2009). Neural networks, deep neural networks, or recursive partitioning algorithms (tree-algorithms) are examples of such methods (Schmidhuber, 2014, Hastie, Tibshirani, Friedman, 2009). We will not discuss these here in detail, but focus on describing their implementations to a typical credit default modeling task and data set.

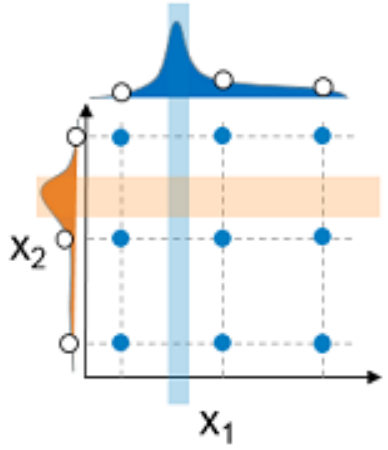
In order to successfully apply machine learning methods, it is almost always necessary to fine-tune the respective parameter estimation algorithms for optimal performance. This fine tuning is required in order to control overlearning, i.e., to prevent the respective algorithm to represent the training data "too perfectly", thus yielding reduced accuracy in hold-out or validation samples. Unfortunately, such fine-tuning requires experimentation and experience with *hyperparameters* that are *not* estimated as part of the model-estimation procedure itself, but must be explored and set by the analyst. Hyperparameter optimization requires experience or systematic exploration of the hyperparameter space, and significant computing resources (Claesen & De Moor, 2015). We will illustrate this with the example data and the popular stochastic-gradient-boosting-trees and deep-learning multi-layer feedforward neural network algorithms, optimizing the hyperparameters that affect overall model complexities.

VII. SEARCHING THE HYPERPARAMETER SPACE

Grid search is the simplest and most common way of searching a parameter space for a global optimum. However, if there are several parameters, and the parameters are on a continuous scale, grid searches can result in many model building runs and will therefore be time-consuming. For instance, if there are 8 hyperparameters, then to test (only) three levels for each parameter will require 3^8=6561 complete model estimation runs with different hyperparameter settings; and even after completing all runs there is no guarantee that the global optimum set of hyperparameter values will be identified.

To illustrate this, see for example the simplified 2-hyperparameter problem for parameter x_1 and x_2 depicted in Figure 4. The two curves shown at the top and on the left side of the figure show the respective narrow bands with respect to hyperparameters x_1 and x_2 where the response function (e.g., AUC) peaks, i.e., where the actual optimal and most accurate prediction models can be found. However, as indicated by the parameter grid, the specific parameter values for *x1* and *x2* tested during the grid search will miss those optimal values because they are located between grid points for both x_1 and x_2.

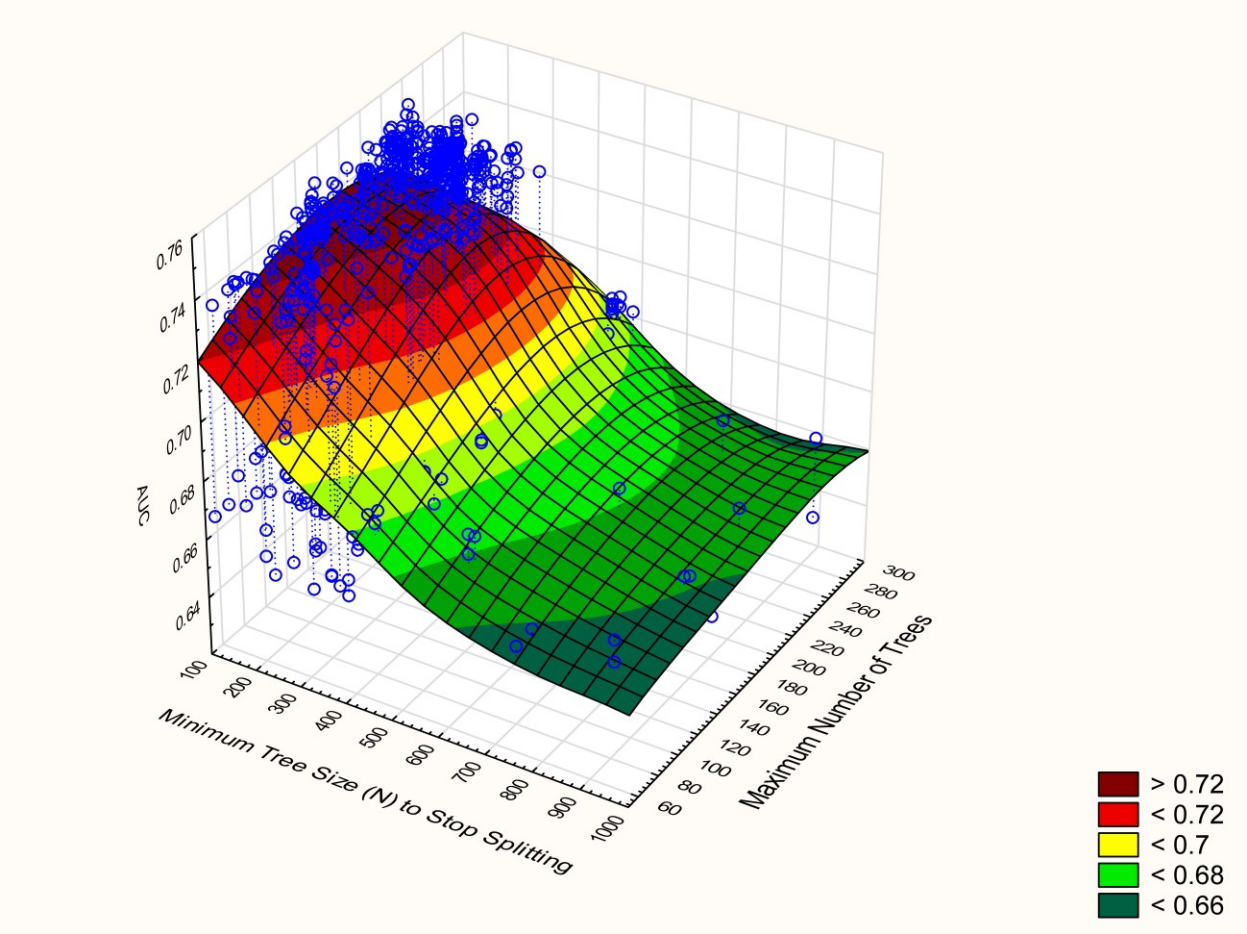
Fig. 4. Illustration of a Simple Standard Grid Search



There are other and sometimes more efficient methods for general function optimization that systematically search for the specific best combinations of hyperparameters which yield the most accurate results (best models). For example, general Genetic Algorithm optimization can be used to perform this task, as well as random searches and other specialized procedures (Claesen & De Moor, 2015). These procedures may identify the best set of hyperparameters in fewer complete model estimation runs, provided that there is an easily identifiable optimum of the response function.

Figure 5 illustrates this with the current data problem and the Stochastic Gradient Boosting modeling method, with respect to two hyperparameters: The *Minimum Tree Size (N) to Stop Splitting,* and the *Maximum Number of Trees* to build. The *Y* axis indicates the AUC values for the respective model runs. In this case, the Genetic Algorithm Search quickly directs the search to the optimum of a convex response surface (fit to these data for illustration).

Fig. 5. Result of a Genetic Algorithm Search of Parameter Space

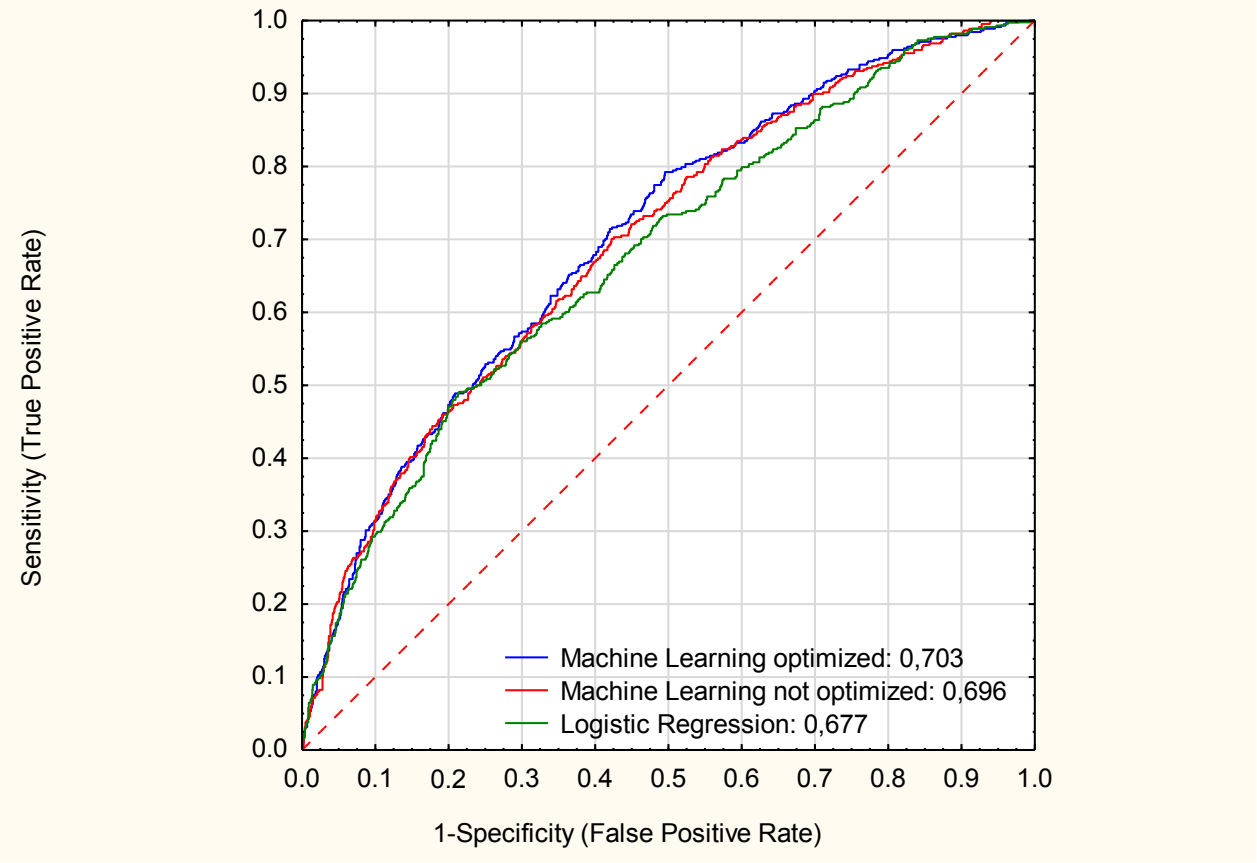


This figure illustrates that most of the runs are completed in the most "promising" best-model area. Other parameters are changed in the different runs, so the performance will vary for run to run even if the two parameters in the plot are the same.

VIII. AUC FOR THE DIFFERENT APPROACHES

Taking the optimized model from the Genetic Algorithm search, as expected we found that machine learning techniques perform better than logistic regression. By optimizing the parameter settings in the machine learning algorithm the accuracy of prediction models can be increased.

Fig. 6. ROC Curves for Different Models.



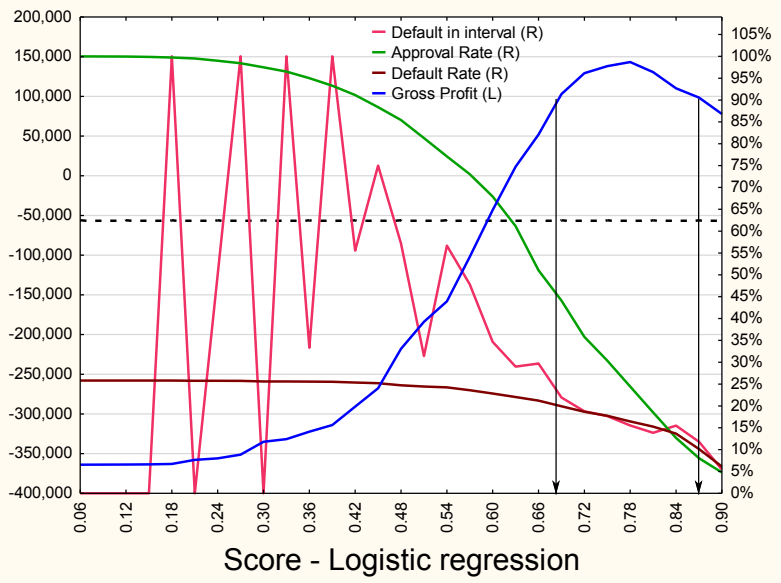
We will look into how much these differences are worth in increased profit below.

IX. PROFIT, APPROVAL RATE AND DEFAULT RATE

How much an increase in AUC is worth in profit depends on a number of things like approval rate, default rate, loan size, interests and the expected losses on defaulting loans to name a few. In this example the loans are small loans typically between 300 – 1000 GBP and the repayment times is varying between 1 – 12 month. The average profit on a loan (that is repaid) is 200 £ and the average loss on a defaulted loan is 700 £.

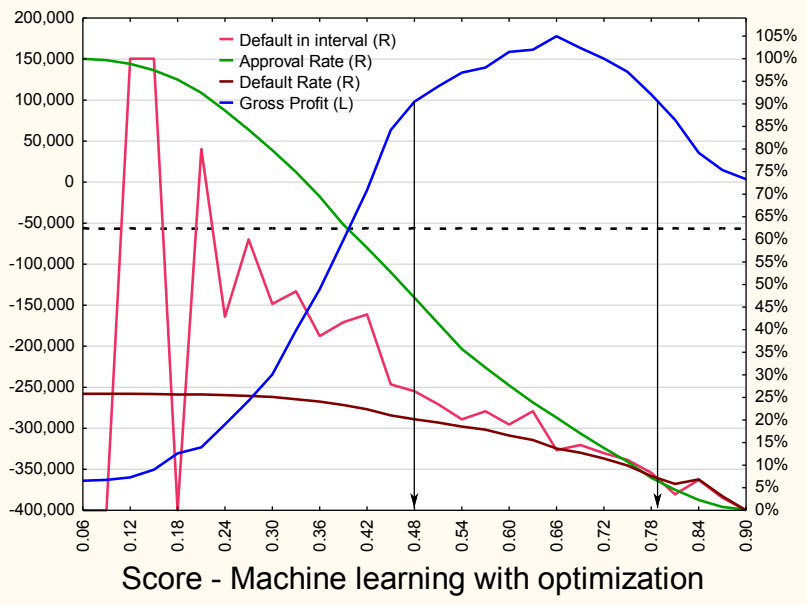
Under these assumptions, we can estimate the profit for different choices of cut-off for the 3 different models:

Fig. 7. Logistic regression



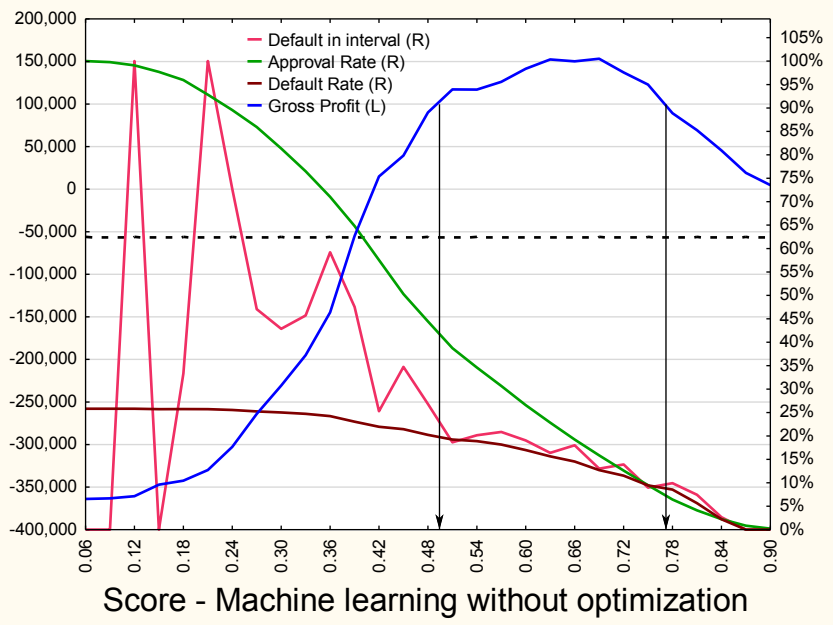
For logistic regression, the best cut-off is 0.78 giving 143.498 GBP gross profit per month (the blue line in fig. 7. at cut-off 0.78).

Fig. 8. Machine Learning with Optimal Parameter Settings



For the model generated by Machine learning with hyperparameter optimization the best cut-off is 0.66 giving 177.937 GBP gross profit per month. This is 24% more than logistic regression.

Fig. 9. Machine Learning with Default Parameter Settings



For the model generated by Machine learning with default parameter settings the best cut-off is 0.66 giving 154.583 GBP gross profit per month. This is 8% more than logistic regression.

Tab. 2. AUC and Profit for the 3 Models

Method	AUC	Profit increase
Machine Learning with optimized hyperparameters	0,703	24 %
Machine Learning without optimized hyperparameters	0,696	8 %
Industry standard Logistic Regression	0,677	0 %

X. DEEP LEARNING NEURAL NETWORKS

The machine learning approach described above is based on boosted recursive partitioning (*stochastic gradient boosted trees*). This algorithm repeatedly partitions the training data -- based on the *X* variables -- into increasingly "pure" segments, i.e., homogeneous groups with respect to the *Y* variable of interest (default risk in this case). Our experience has shown that this approach is particularly efficient and effective at extracting from historical behavioral data the diagnostic information for predicting default risk.

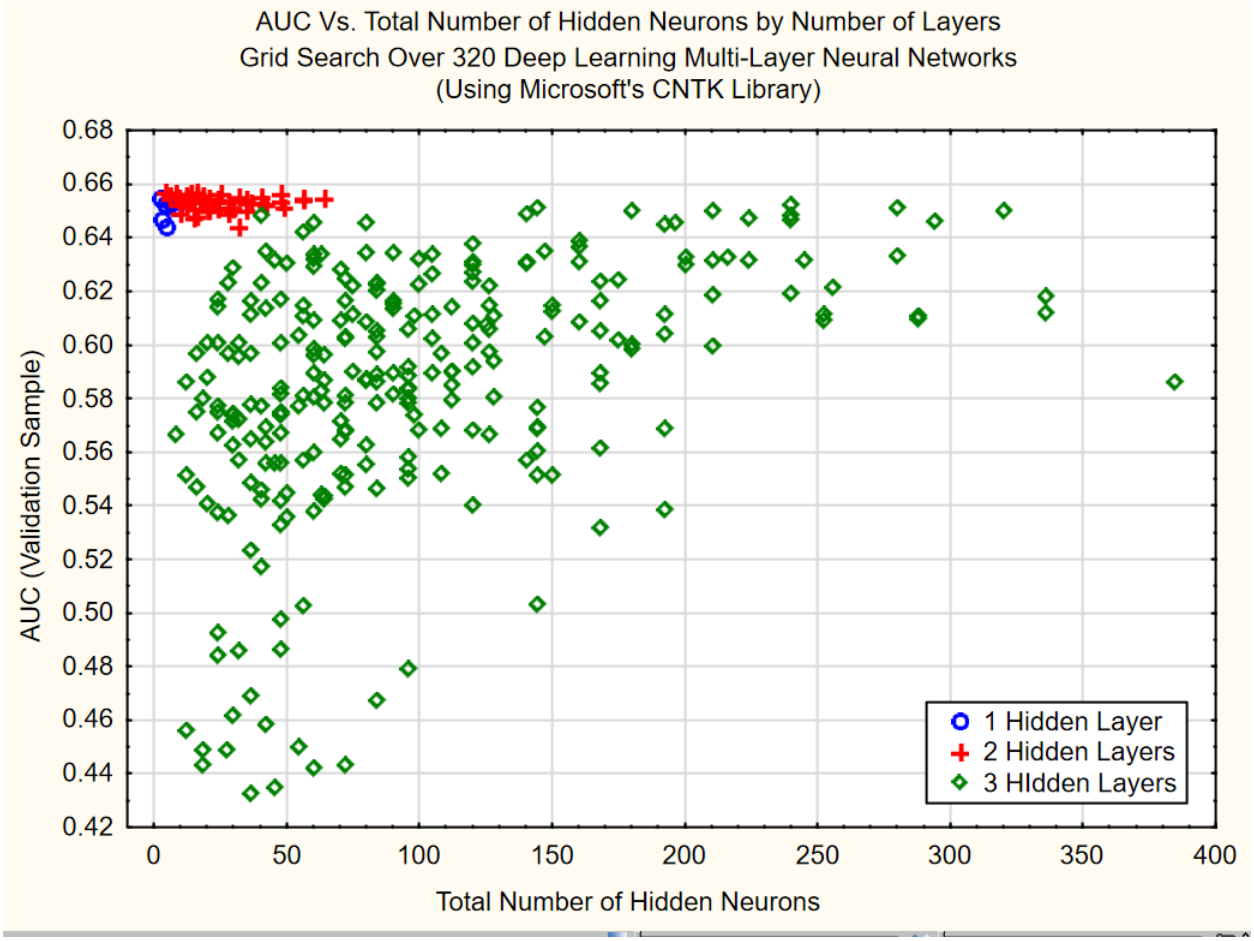
Recently, Deep Learning Neural Nets have become popular for solving predictive analytics problems with complex interactions and nonlinear relationships. Deep network architectures have been shown to be particularly effective at classification of bitmaps, or natural language processing, and similar complex prediction tasks.

The data problem described in this presentation represents a standard binary (Yes/No) prediction problem, based on the available predictors (*X* variables). Standard "deep" (multi-layers) feed-forward neural network models were explored to identify the accuracy of models as a function of model complexity, i.e., the numbers of hidden layers and numbers of hidden nodes (activation functions, parameters).

In most implementations of deep neural network models there are many hyperparameters that can be explored and manipulated to improve the performance of deep neural nets, including activation functions, error functions, and regularization penalties, to name only a few. This makes hyperparameter optimization a complex task, requiring experience with the respective domains and data types to be successful. In addition, significant computational resources and effort is typically required.

To illustrate, here is a summary of a grid search over 320 fully connected networks with 1, 2, or 3 layers, and up to 384 hidden neurons.

Fig. 10. Results of a Hyperparameter Space Grid Search, Multi-Layer "Deep" Neural Nets



A few things are immediately apparent: The larger the number of layers the greater is the chance of overfitting the data, resulting in poor prediction performance in the Validation data. Second and not surprisingly, the simplest models - in this case with the single layer -- produce results similar in accuracy to the logistic regression model.

In conclusion, and after many more experiments with convergence criteria, regularization constants, and activation functions, neural nets including deep-nets appear not be suitable to create the most accurate and parsimonious models for this data set and use case, which essentially is about partitioning observations into segments with similar characteristics (X variable values) and default risk (Y values). As is apparent by the many successful examples where deep neural networks have yielded excellent accuracy with very high-dimensional data and complex relationships (e.g., in the classification of bitmaps), this is not a conclusion that should be generalized to other domains or types of data.

XI. CONCLUSION

In our example, we showed that it was possible to increase the expected profit in a portfolio by 8% by using appropriate machine learning techniques instead of the industry standard logistic regression. Further, if the hyperparameters in the machine learning techniques (in this case Boosting Trees) were further fine-tuned (optimized), it was possible to increase the profit by 24 % compared to logistic regression.

Will machine learning and hyperparameter optimization become a game changer for credit scoring?

These techniques have potential to be a game changer, but we believe many banks will continue to use logistic regression as before, and therefore we do not believe that it will have a big enough impact to qualify to be called a game changer.

On the other side, we do believe that there will be banks that will embrace these methods and be able to produce scorecards with higher performance. These banks will be more competitive, and increase their profit. We therefore think these methods will play an important role for the banking sector in the time to come.

REFERENCES

[1] Breiman, L. (2001). Statistical modeling: The two cultures. Statistical Science, 16, 3, 199-231.

[2] Claesen, M. DeMoor, B. (2015). Hyperparameter search in machine learning. MIC 2015: The XI Metaheuristics International Conference

[3] Hastie, T., Tibshirani, R., Friedman, J. H. (2009). The elements of statistical learning : Data mining, inference, and prediction, 2nd Ed.. NY: Springer Verlag.

[4] Miner, G.; Elder, J., Hill, T., Nisbet, R., Delen, D., Fast, A. (2012) Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications. NY: Elsevier.

[5] [5] Nisbet, R., Elder, J., & Miner, G. (2009). Handbook of Statistical Analysis and Data Mining Applications.NY: Elsevier.

[6] Schmidhuber, J. (2014). Deep Learning in Neural Networks: An Overview. Technical Report IDSIA-03-13/arXiv:1404.7828V4.

[7] Siddiqi, N. (2006). Credit risk scorecards: Developing and implementing intelligent credit scoring. Wiley & Sons, NY.

[8] Statistica Version 13.2; 2017; see also Statistica Documentation: http://documentation.statsoft.com/STATISTICAHelp.aspx?path=WeightofEvidence/WeightofEvidenceWoEIntroductoryOverview; extracted July 12, 2017.