Data Driven Credit Risk Management Process: A Machine Learning Approach

Mingrui Chen
Southern Methodist University
6425 Boaz Lane
Dallas, Texas, USA 75205
mingruic@smu.edu

Yann Dautais
GDS Link, LLC
5307 East Mockingbird Lane
Dallas, Texas, USA 75275
yann.dautais@gdslink.com

LiGuo Huang
Southern Methodist University
6425 Boaz Lane
Dallas, Texas, USA 75205
lghuang@smu.edu

Jidong Ge
State Key Laboratory for Novel Software Technology
Software Institute, Nanjing University
Nanjing, China 210093
gjd@nju.edu.cn

ABSTRACT

Credit scoring process, the most important part in credit risk management, aims at estimating the probability that an applicant will perform bad credit behaviors (e.g., loan default). Managing and developing effective and reliable risk assessment procedures in order to mitigate potential loss caused by new applicants heavily relies on the performance of scoring process. Traditionally this process is manually developed, which is time-consuming. In this paper, we propose an automated credit risk management process based on machine learning to ease the scoring process in order to reduce the human effort. This process is data driven: it leverages machine learning to automatically analyze vast amounts of historical data and build predictive model. We evaluate our process with a real-world proprietary dataset and achieved good performance, which shows the feasibility of using machine learning to facilitate the credit risk management process.

CCS CONCEPTS

•Computing methodologies → Machine learning; •Information systems → *Decision support systems; Data analytics;*

KEYWORDS

Credit risk management, Data driven, Machine learning, Process improvement

ACM Reference format:
Mingrui Chen, Yann Dautais, LiGuo Huang, and Jidong Ge. 2017. Data Driven Credit Risk Management Process: A Machine Learning Approach. In *Proceedings of 2017 International Conference on Software and Systems Process, Paris, France, July 2017 (ICSSP'17),* 5 pages.
DOI: 10.1145/3084100.3084113

ICSSP'17, Paris, France

DOI: 10.1145/3084100.3084113

1 INTRODUCTION

The scoring process is an essential part of the credit risk management system used in financial institutions to predict the risk of loan applications. This process uses a statistical model that takes into account the application data and performance data of a credit or loan applicant and estimates the defaulting probability, which is the most important factor used by the lender to rank applicants for decision making.

Developing an effective scoring process depends on two factors: (1) appropriate selection of key attributes from historical data, and (2) accurate estimation of defaulting probability produced by predictive model. The traditional way of developing this process is known as scorecard, which is a manual approach that subjectively select attributes and creating rules based on personal experience of analysts. However, when the dimension of data is increasing, manual analysis becomes more and more time consuming and error-prone, and the results are often subjective or even biased depending on experience. To meet these challenges, we propose an automated scoring process that simplifies the task of estimating the defaulting probability.

In this study, we employ Support Vector Machine (SVM) as the core algorithm to automate the scoring process due to two reasons: (1) SVM can solve both linear and non-linear classification problems and yield promising classification performance on multi-dimensional data with the capability to avoid the curse of dimensionality problem [14], (2) SVM has been well developed for probability estimation task [17]. The SVM algorithm was first proposed by Vladimir N. Vapnik [15] and we use LibSVM library [2].

The advantages of our approach are twofold: (1) all attributes are taken into account by machine learning algorithm so that the predictive model is less subjective (2) it eliminates the need to pick attributes and create rules manually so that it can be easily and effectively adapted to different datasets with different attributes.

We organize the paper as follows: Section 2 presents some related work of credit scoring. Section 3 introduces the basic idea of the traditional scoring process. Section 4 introduces our automated process based on machine learning algorithm. Section 5 describes

the experiment setup and evaluation results. Section 6 summarizes our work and proposes future work.

2 RELATED WORK

Machine learning algorithms have been used by practitioners and researchers for several credit risk management tasks. Worrachartdatchai, U. and Sooraksa, P. applied Least Squares Support Vector Machine to classify credit applicants into four groups [16]. Gestel et al. proposed SVM-based approach to classify credit applicants into even more groups [7]. Zhang et al. investigated a hybrid approach in building credit scoring model by combining genetic programming with support vector machine [18]. Kiani, M. and Mahmoudi, F. proposed another hybrid method for credit scoring by combining clustering and support vector machine in order to increase the accuracy of classification [11]. Neural network have been used in credit scoring before SVM was introduced. Huang et al. demonstrate that SVM can outperform neural networks in building risk classification model [9].

However, to our best knowledge, existing works employing machine learning algorithms mainly study how to increase the accuracy of classification models that classify applicants into discrete risk levels, but none of them is focusing on automating the complete scoring process to estimate defaulting probability. In addition, based on our industrial experience, the scoring process is still manually developed in financial institutions nowadays.

3 TRADITIONAL PROCESS

First, we introduce some definitions to help understanding the process.

- Good applicants: existing customers that have been observed to have good credit behaviors and classified as good
- Bad applicants: existing customers that have been observed to have bad credit behaviors and classified as bad
- Accepted applicants: applicants that have been accepted. This population consists of both good and bad applicants mentioned above
- Rejected applicants: applicants that have been rejected. This population has no observation of credit behaviors thus cannot be classified as good or bad when collecting data

The traditional way to estimate defaulting probability is accomplished by building scorecard. The common procedures are:

(1) Collect data from existing customers and rejected applicants who are classified as Good, Bad and Reject
(2) Do reject inference to assign inferred probability to applicants who have no history
(3) Manually select a subset of attributes from the dataset based on analyst's experience
(4) Manually create rules based on selected attributes and estimate credit score for new applicants
(5) Estimate final defaulting probability using statistical models based on the distribution of Good/Bad samples and associated credit score

Reject inference (Step 2) is a widely applied technique[6][13] which is an essential step to determine whether the model is generalizable to the entire population of applicants. When building scorecard, samples need to be identified as Good or Bad. However, in practice, only the performance (Good or Bad) of existing customers who have been accepted is available in the modeling data. Thus the model built exclusively on accepted samples may incur critical bias when it is applied on the entire "through-the-door" population, which includes new applicants who may have significantly different behaviors than existing customers. To resolve this problem, we can leverage the information of rejected applicants in the model. However, their performance has never been observed, therefore it is not able to identify which category (Good or Bad) they belong to. Reject inference is a technique to infer the performance, typically the probability of being Bad of rejected applicants, so that the "performance" of rejected applicants can be included in the model construction.

4 PROPOSED PROCESS

Figure 1 shows the framework of our automated scoring process. It has four major components connected as a workflow. It is different from the traditional process in that attribute selection, model construction and probability estimation are all automated in order to eliminate intensive human effort.

4.1 Data Preparation

In the traditional process, analysts need to exam the attributes in the historical data and pick a subset of attributes based on their experience. However, in our automated process, we can keep all the attributes and feed the complete dataset to the learning algorithm. During preparation, each data sample shall be classified into one of the following categories, namely "Good", " Bad" and "Reject" which are mentioned in Section 3. In addition, since all the samples labeled as Good and Bad come from accepted applicants, they are also assigned a label "Accept".

4.2 Predictive Models

The predictive model aims at estimating defaulting probability using SVM, instead of using human-generated rules. First we introduce some notations:

- $P(B|A)$ denotes the probability of a sample being Bad in accepted samples
- $P(R)$ denotes the probability of a sample being Reject
- $P(B|R)$ denotes the inferred probability of a sample being Bad in rejected samples
- $P(B)$ denotes the probability of a sample being Bad in all samples

The predictive model comprises of four components:

(1) An initial Good/Bad model estimates the probability $P(B|A)$
(2) An Accept/Reject model estimates the probability $P(R)$
(3) A reject inference model infers $P(B|R)$ based on $P(B|A)$ and $P(R)$
(4) A final scoring model estimates the probability $P(B)$, which is equivalent to the defaulting probability

4.2.1 Initial Good/Bad Model. An initial Good/Bad model is first trained using SVM to estimate $P(B|A)$. Before training, we construct a training set using the samples labeled Good and Bad from collected data. SVM is a kernel-based algorithm, so selecting the effective

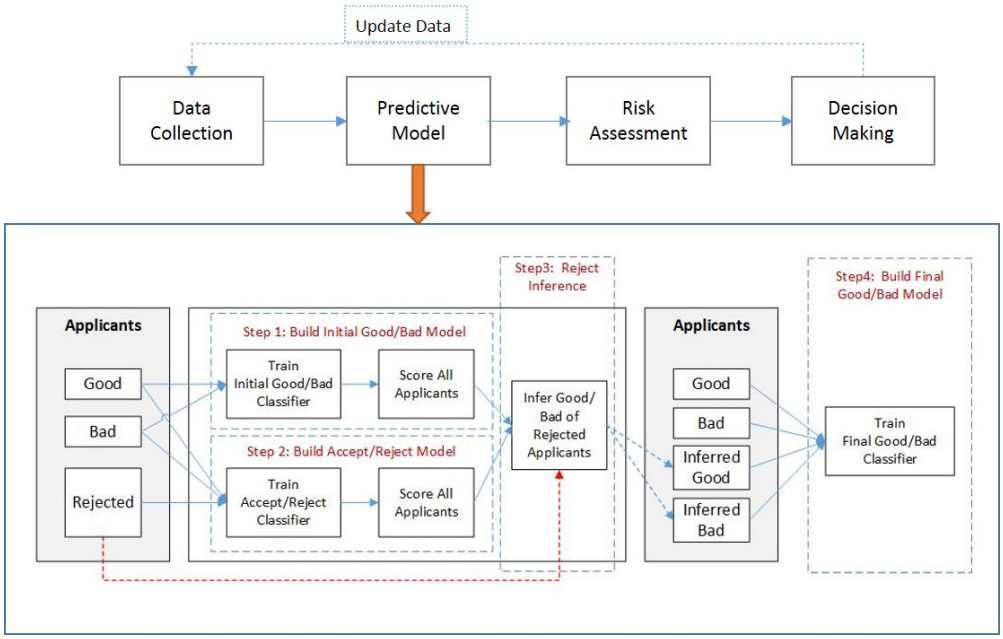

Figure 1: Automated Scoring Process

kernel is important to achieve good results. We choose the RBF kernel in our experiments because RBF kernel has been proven to achieve very promising classification performance [10, 12]. We tune two parameters for RBF kernel, namely cost (c) and gamma (g). During the training process, the algorithm selects the (c, g) parameters using an exhaustive searching algorithm implemented in LibSVM [3]. Suppose there is a search space p = {c, g}, in which c is cost and g is gamma. The search algorithm is to select the pair of (c, g) which results in the lowest rate of erroneous classifications in the training data set by conducting five-fold cross validation internally. The resulting classifier is then used to estimate $P(B|A)$ of each sample by enabling the probability output option in LibSVM.

4.2.2 Accept/Reject Model. An Accept/Reject model is then trained using SVM as well to estimate $P(R)$. Before training, we construct a training set using the following way: if the sample is labeled Good or Bad, it is assigned Accept, otherwise it is Reject.

The Accept/Reject model is used not only to differentiate Accept and Reject samples, but also to rank Reject samples in such a way that those who are likely to have been bad had they been accepted get a high probability of being rejected. This implies we need to emphasize the attributes that are important to classify samples into Accept or Reject, as well as to differentiate the Bad samples from Good ones in the Accept/Reject model. Thus, when training the Accept/Reject model, we take into account the importance of attributes from the initial Good/Bad model. We use a F-score based feature selection approach implemented in [4] to find the most significant attributes in the initial Good/Bad model and increase the weights of the these attributes when training the Accept/Reject model. The resulting classifier is then used to estimate $P(R)$ by enabling probability output in LibSVM.

4.2.3 Reject Inference. The purpose of reject inference has been explained in section 3. In this paper, we use a standard reject inference method which is called extrapolation. The extrapolation is performed by adjusting the probabilities $P(B|A)$ produced by initial Good/Bad model based on the probabilities $P(R)$ produced by Accept/Reject model such that $P(B|A)$ are extrapolated onto Reject samples. The lower $P(R)$ is, the more similar is the rejected sample to a good one, so that it has a lower inferred probability $P(B|R)$. The extrapolated probability is factored in the following way:

$$P(B|R) = f(P(B|A) \cdot (P(R))) \quad (1)$$

where $P(B|R)$ is the inferred probability of being Bad for rejected samples, $P(B|A)$ and $P(R)$ are estimated probabilities mentioned in section 4.2.1 and 4.2.2., and $f()$ is a scaling function that scales $P(B|A)$ to the entire rejected samples. As this is a standard method used in credit scoring industry, we don't explain it in every detail and more details can be found in these two papers [6][13]. When $P(B|R)$ is available for rejected samples, we can assign Good or Bad labels based on a cutoff percentage p. For instance, we can rank rejected samples by their inferred probability $P(B|R)$ in ascending order, and set p to 20%. Then we label the top 20% samples in the ranked list as Good and the remaining samples as Bad. Note that p can be different, and should be smaller than the percentage of original Good samples in Accept category.

4.2.4 Final Scoring Model. Now that we have a dataset which consists of both original and inferred Good/Bad samples, we can train a final scoring model on this less biased dataset to estimate the defaulting probability. We use the same setting of SVM mentioned in 4.2.1 to train the final scoring model.

Table 1: Distribution of data samples

Label	Good	Bad	Reject	Total
# of Samples	1019	379	4226	5624

4.3 Risk Assessment & Decision Making

After the final scoring model is built, we can use it to estimate the credit risk of new applicants. Decision can be made according to the outcome of the final scoring model and specific guidelines of financial institutions (i.e. to which level of defaulting probability will an applicant be accepted or rejected).

4.4 Update Data

Capturing the change of new and past applicants' performance data is essential to keep the predictive model effective and reliable. Our approach leverages automatic feature selection which provides the capabilities of handling attributes that are frequently changing and new data samples that are added from new applicants. For example, if some rejected applicants are accepted and credit observations become available, or some Bad applicants become Good, we can update the data and retrain the predictive model.

5 EVALUATION RESULTS

This section evaluates our approach. The evaluation is based on a statistical model using accuracy and Gini coefficient as the primary evaluation metrics.

5.1 Experiment Setup

The dataset used in our study contains 5624 data samples which are provided by a bank in Bulgaria in collaboration with GDS Link, LLC . Each sample has 256 attributes (e.g. age, income, number of on-time payments). Originally, data is annotated by three categories, namely Good, Bad and Reject respectively. Table 1 shows the distribution of data samples in our dataset. Among the 5624 samples, there are 1019 labeled as Good, 379 labeled as Bad and 4226 labeled as Reject.

Recall that when training the initial Good/Bad model and final scoring model, we use data samples that are labeled as Good and Bad. Thus when preparing the dataset, we split the samples into 10 folds, each of which contains 10% Good samples and 10% Bad samples. If the sample belongs to Good, its class value is +1; otherwise, its class value is -1. We perform the jack knife by training the classifier based on 9 out of the 10 folds and test the classifier on the remaining one to get the probability $P(B|A)$ of test fold. We performed 10-fold cross validation in order to get a complete set of probability $P(B|A)$ on the whole dataset.

When training the Accept/Reject model, on the other hand, we use data samples that are labeled as Accept and Reject. If the training instance belongs to Accept, its class value is +1; otherwise, its class value is -1. Again we use the 10-fold cross validation aforementioned to get a complete set of Probability $P(R)$ on the whole dataset.

5.2 Classification Accuracy

In order to estimate reasonable probabilities, we should first ensure a good classification accuracy of initial Good/Bad mode and Accept/Reject model before doing reject inference and building the final scoring model. The classification accuracy of these three models are listed below. The accuracy is obtained by combining the prediction results of 10 test folds.

(1) The initial Good/Bad model and final scoring model yields overall accuracy of 77.8% and 87.1% respectively
(2) The Accept/Reject model yields overall accuracy of 93.1%

From the results we can see that SVM provides satisfactory performance for us to build predictive models.

Table 2: Cumulative probabilities of goods and bads

Levels	Goods	Bads	Cumulative Goods	Cumulative Bads
			0	0
0	575.796	238.204	33.23%	6.12%
0.1	267.358	113.642	48.66%	9.04%
0.2	147.798	77.202	57.19%	11.03%
0.3	93.542	67.458	62.58%	12.76%
0.4	71.336	64.664	66.70%	14.42%
0.5	75.396	78.604	71.05%	16.44%
0.6	55.442	84.558	74.25%	18.61%
0.7	67.898	106.102	78.17%	21.34%
0.8	52.011	152.989	81.17%	25.27%
0.9	79.331	320.67	85.75%	33.51%
1	246.952	2587.05	100%	100%
	1732.86	3891.14		

5.3 Gini Coefficient

Gini coefficient [8] is a statistical method which is commonly used to measure the ability of a predictive model [5]. The higher Gini coefficient is, the better the model is. Formally, it is defined as:

$$Gini = 1 - \sum_{i} (G_{i+1} - G_i) \cdot (B_{i+1} - B_i) \quad (2)$$

where i is a set of cut-off values to select Good and Bad applicants, G_i is the proportion of Good applicants passing cut-off value i , and B_i is the proportion of Bad applicants passing cut-off value i. More specifically, the Gini calculation is done with each rejected applicant assigned to both the Bad and Good categories with a probability $P(B)$ and 1 - $P(B)$ respectively and plotted based on 10 levels ranging from 0 to 1, with an interval of 0.1. For example, for a rejected applicant, if the probability of being Bad is 0.65, the probability of being Good will be 1 - 0.65 = 0.35. Hence on the level of 0.6, which is the 9th row of Table 2, 0.65 will be added to the Bad category and 0.35 will be added to the Good category. Table 2 shows the cumulative values used to plot the Gini curve.

Fig. 2 visualizes the Gini coefficient. The red dotted line represents the ability of discrimination of an actual predictive model, the diagonal represents no discrimination. In the graph, the larger the area between the red line and the diagonal, the better the predictive model is at discriminating Good and Bad. As a rule of thumb, predictive models that have 50% or more Gini coefficient are considered to have good predictive power [1]. Our approach achieves 64% Gini coefficient, which is acceptable.

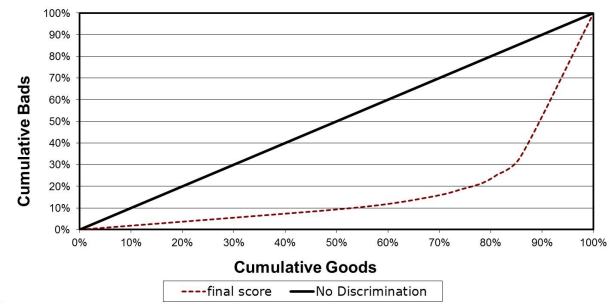

Figure 2: Gini Coefficient Curve

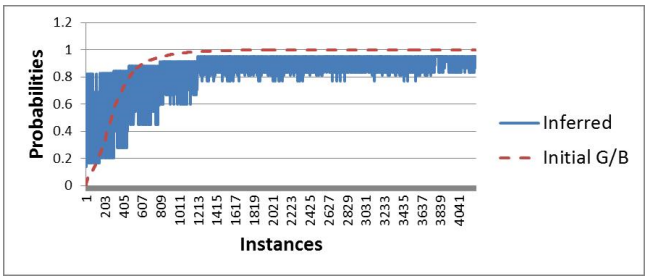

Figure 3: Initial vs. Inferred

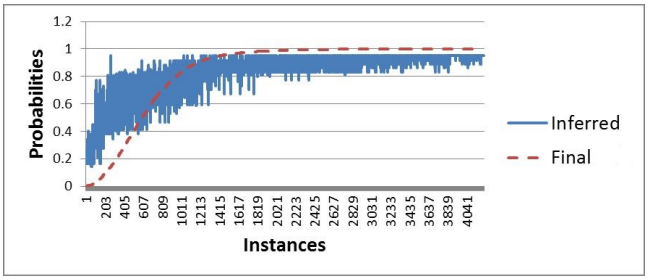

Figure 4: Final vs. Inferred

5.4 Effect of Reject Inference

This section does not evaluate the accuracy of predictive model, but shows a general idea about how reject inference can affect the predictive model on rejected samples. In Fig. 3 and Fig. 4, the X-axis represents all the rejected samples and the Y-axis represents the probabilities of being Bad for each sample. They reflect the trend of probabilities of being Bad on the rejected samples by applying the initial Good/Bad model before doing reject inference and the final scoring model after doing reject inference. The red dotted lines represent probability estimated by SVM and the blue lines represent the probability directly inferred by reject inference model. The probabilities are sorted in ascending order based on the ones predicted by SVM and keep the inferred one tied with each sample. We can see that both models have similar trend, whereas after employing reject inference, the final scoring model generates a more flat curve, which indicates more rejected applicants are predicted to have lower defaulting probability.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose an automated credit scoring process based on SVM. Our approach is fully automated so that it can largely reduce the human effort in developing the scoring process and effectively adapt to different datasets and financial institutions. We evaluate and validate our SVM-based predictive model on a real-world data set from a bank in Bulgaria and it achieves high accuracy and 64% Gini coefficient. Future work includes testing our approach on other credit datasets, exploring ways to further increase the performance and comparing with other machine learning algorithms.

REFERENCES

[1] Raymond Anderson. 2007. *The credit scoring toolkit: theory and practice for retail credit risk management and decision automation.* Oxford University Press.
[2] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27. Issue 3. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[3] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27. Issue 3.
[4] Yi-Wei Chen and Chih-Jen Lin. 2006. Combining SVMs with various feature selection strategies. In *Feature extraction.* Springer, 315–324.
[5] A. G. Christodoulakis and S.Satchell. 2007. *The analytics of risk model validation (Quantitative Finance).*
[6] AJ Feelders. 2003. An overview of model based reject inference for credit scoring. (2003).
[7] T. Gestel, B. Baesens, I. Garcia, and P. Dijcke. 2003. A support vector machine approach to credit scoring. In *Forum Financier-Revue Bancaire et Financiaire Bank en Financiewezen.* 73–82.
[8] C. Gini. 1912. Variabilit e mutabilit. Memorie di metodologica. (1912).
[9] Z. Huang, H. Chen, C. Hsu, W. Chen, and S. Wu. 2004. Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision Support System* (2004), 543–558.
[10] S. Keerthi and C. Lin. 2003. Asymptotic behaviors of support vector machine with gaussian kernel. *Neural Computation* 7 (2003), 1667–1689.
[11] M. Kiani and F. Mahmoudi. 2010. A new hybrid method for credit scoring based on clustering and support vector machine (ClsSVM). In *Processings of the Second International Conference on Information and Financial Engineering.* 585–589.
[12] Hsuan-Tien Lin and Chih-Jen Lin. 2003. A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods. *submitted to Neural Computation* (2003), 1–32.
[13] D. Montrichard. 2008. Reject Inference Methodologies in Credit Risk Modeling. In *SESUG Processings.*
[14] P. Tan, M. Steinbach, and V. Kumar. 2006. *Introduction to data mining.*
[15] V. Vapnik. 1995. *The natural of statistical learning.*
[16] U. Worrachartdatchai and P. Sooraksa. 2007. Credit scoring using least squares support vector machine based on data of Thai Financial Institutions. In *Processings of the Ninth International Conference on Advanced Communication Technology.*
[17] Ting-Fan Wu, Chih-Jen Lin, and Ruby C Weng. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* 5, Aug (2004), 975–1005.
[18] D. Zhang, M. Hifi, Q. Chen, and W. Ye. 2008. A hybirb credit scoring model based on generic programming and support vector machine. In *Processings of the Fourth International Conference on Natural Computation.*