Improving Logistic Regression Classification of Credit Approval with Features Constructed by Kaizen Programming

Vinícius Veloso de Melo
Institute of Science and Technology (ICT)
Federal University of São Paulo (UNIFESP)
São José dos Campos, SP, Brazil
vinicius.melo@unifesp.br

Wolfgang Banzhaf
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, A1B 3X5, Canada
banzhaf@mun.ca

ABSTRACT

In this contribution, we employ the recently proposed Kaizen Programming (KP) approach to find high-quality nonlinear combinations of the original features in a dataset. KP constructs many complementary features at the same time, which are selected by their importance, not by model quality. We investigated our approach in a well-known real-world credit scoring dataset. When compared to related approaches, KP reaches similar or better results, but evaluates fewer models.

CCS Concepts

•Computing methodologies → Supervised learning by classification; Genetic programming; •Software and its engineering → Automatic programming;

Keywords

Credit approval, Logistic regression, Classification, Kaizen Programming

1. INTRODUCTION

Accurate credit scoring prediction is an extremely important task for financial companies. Credit scoring through classification categorizes credit applicants into good or bad risk classes, aiming to reduce the risk of losing money. Even small accuracy improvements may save financial companies huge amounts of money, making these methods interesting to financial companies.

Several researchers have been using Evolutionary Computation (EC) methods to solve this task [1, 2, 3, 4, 5, 6]. Here, we investigate Kaizen Programming (KP [7]) for constructing high-level features to improve the classification.

Our contribution has three aspects: (i) We use, for the first time, KP with Logistic Regression (LR), a traditional and very popular statistical tool for classification; (ii) we use a hybrid method of GP and LR for credit scoring; (iii) we evaluate KP performance on a publicly available dataset and compare the results with state-of-the-art EC methods from the literature.

2. KAIZEN PROGRAMMING

Kaizen Programming [7] is a computational method based on the concepts of the Kaizen methodology [8], i.e., a computational implementation of a Kaizen event with Plan-Do-Check-Act (PDCA) methodology to guide a continuous improvement process. KP iteratively improves complementary partial solutions to solve a particular problem. Both the partial solutions and the complete solution are evaluated after building a model, which is the actual solution to the problem.

In this contribution, we use Logistic Regression (LR) as the model building technique, the p-value of each covariate as feature importance, and AIC as model (complete solution) quality for selection. KP employs GP crossover and mutation operators as experts.

3. EXPERIMENTAL RESULTS

We investigate the performance of KP on the well-known Australian Credit Approval dataset. It has 690 examples with 14 features, and two classes, 383 positive examples and 307 negative. As some features are categorical but our implementation does not work with mixed-type variables, we discretize the continuous features.

The discretized dataset is transformed using the one-hot procedure, where each discrete value becomes a binary column. KP is configured to use only Boolean functions (AND, OR, NOT) to construct features. The number of desired features varies from $D = 2$ to 15 and is run for a maximum of 100 iterations, resulting in 201 models. Before building models, duplicate and highly correlated features are dropped (a feature that appears first is kept). We implement KP in Python and run the experimental analysis on Weka 3.6.11 [9].

3.1 Evaluation

We executed ten independent runs of KP to minimize AIC. After constructing the features, all ten new datasets (only the new features, not the original ones) were loaded into WEKA Experimenter and evaluated on 10 distinct 10-fold cross-validation runs (100 runs) using the Logistic Regression classifier with default configuration.

GECCO '16 Companion, July 20-24, 2016, Denver, CO, USA

ACM ISBN 978-1-4503-4323-7/16/07.
DOI: http://dx.doi.org/10.1145/2908961.2908963

For each value of D, we selected the new dataset (out of ten) that presented the best accuracy. We compare the results to the original dataset. Accuracy was chosen because it is the measure used in the related works and the dataset is just slightly unbalanced.

3.2 Results and Comparison with the Literature

The best results obtained here are compared to state-of-the-art evolutionary algorithms from the literature. Those methods generate either features from the dataset or actual classifiers. Our method employs only discrete features, while other methods were able to cope with mixed attributes. Not all of these methods used *10*-fold cross-validation, and a direct comparison is not absolutely fair, just an approximation.

Table 1 shows that KP produces competitive results. It generates several features while some of the other methods generate a single feature. However, KP works by decomposing the problem into partial solutions; thus, it is not appropriate for constructing a single feature. Even though KP generates several features, they are short (no such information available from related work). For $D = 2$, for instance, the results were F_1= or(A3_(25.2-inf), A11_(33.5-40.2]) and F_2= or(A9_t, A11_(33.5-40.2]), which presented $Correlation = 0.06$, $TP = 0.9277$, $FP = 0.2259$, $TN = 0.7905$, and $FN = 0.067$.

Table 1: Comparison with works from the literature (averages). '-' means not available.

Method	Evaluation	Accuracy	# Models
LR w/ Original data	10 x 10-fold cv	83.83	1
KP (D=2)	10 x 10-fold cv	85.54	201
KP (D=3)	10 x 10-fold cv	86.38	201
KP (D=5)	10 x 10-fold cv	86.48	201
KP (D=10)	10 x 10-fold cv	87.59	201
KP (D=15)	10 x 10-fold cv	87.59	201
GP [1]	5 x 2-fold cv on test set	82.40	50,000
GP [2]	5-fold cv	88.27	72,000
kNN-GP [3]	10-fold cv	83.1	2000
MOGP [4]	5 x 2-fold cv	87.4	350,000
GBMGP [5] (*resampling*)	10-fold cv	87.00	160,000
GP [6]	10-fold cv	87.00	700,000

The second comparison criterion is the number of models that were tested in the process. For KP, this is calculated as *Number of initial random models* + *2* × *Number of cycles* = *1* + *2* × *100* = *201*. Here, a single random model is created to be the *Initial Standard*; then, at every cycle one model is generated using all partial ideas and another model is generated using only the most important ideas.

We estimated the number of models for the related work as *Population size* × *Number of generations* × *2 children* × *Crossover probability*. As individuals in evolutionary algorithms normally encode a complete solution, each individual is a model to solve the classification problem, ignoring cross-validation steps during the training.

With respect to the number of models, KP largely outperformed all other methods, showing that its search procedure can be very efficient. The model building technique is very effective and implicitly selects features generated by KP experts, which only need to keep improving them as they are not responsible for actually solving the problem.

4. SUMMARY AND CONCLUSIONS

Kaizen Programming (KP) is a hybrid collaborative problem-solving approach. Here, KP was coupled with Logistic Regression (LR) to extract useful features from a widely studied credit scoring dataset, aiming at improving the prediction performance of LR.

Our comparison with related work shows results using KP to be similar or better than those reported in the literature; however, much fewer models were evaluated, i.e., 201 by KP versus many thousands by other methods.

We will test KP on other credit scoring datasets reported in the literature, with or without the application of a discretization procedure.

5. ACKNOWLEDGMENTS

This paper was supported by grant #2016/04946-4, São Paulo Research Foundation (FAPESP), grant #486950/2013-1, the Brazilian Government CNPq (Universal), and grant #12180-13-0, CAPES (Science without Borders program) to V.V.M., and Canada's NSERC Discovery grant RGPIN 283304-2012 to W.B.

6. REFERENCES

[1] K. Krawiec, "Genetic programming-based construction of features for machine learning and knowledge discovery tasks," *Genetic Programming and Evolvable Machines*, vol. 3, pp. 329–343, 2002.

[2] C.-S. Ong, J.-J. Huang, and G.-H. Tzeng, "Building credit scoring models using genetic programming," *Expert Systems with Applications*, vol. 29, no. 1, pp. 41–47, 2005.

[3] M. C. Bot, "Feature extraction for the k-nearest neighbour classifier with genetic programming," in *Genetic Programming: 4th European Conference, EuroGP 2001 Lake Como, Italy, April 18–20, 2001 Proceedings*, J. Miller, M. Tomassini, P. L. Lanzi, C. Ryan, A. G. B. Tettamanzi, and W. B. Langdon, Eds. Springer, 2001, pp. 256–267.

[4] Y. Zhang and P. I. Rockett, "A generic optimising feature extraction method using multiobjective genetic programming," *Applied Soft Computing*, vol. 11, no. 1, pp. 1087–1097, 2011.

[5] H. Li and M.-L. Wong, "Financial fraud detection by using grammar-based multi-objective genetic programming with ensemble learning," in *Evolutionary Computation (CEC), 2015 IEEE Congress on.* IEEE, 2015, pp. 1113–1120.

[6] C.-L. Huang, M.-C. Chen, and C.-J. Wang, "Credit scoring with a data mining approach based on support vector machines," *Expert Systems with Applications*, vol. 33, no. 4, pp. 847–856, 2007.

[7] V. V. De Melo, "Kaizen programming," in *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, ser. GECCO '14. New York, NY, USA: ACM, 2014, pp. 895–902.

[8] M. Imai, *Kaizen, the key to Japan's competitive success.* McGraw-Hill, 1986.

[9] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.