

COMPARING A GENETIC FUZZY AND A NEUROFUZZY CLASSIFIER FOR CREDIT SCORING

F. HOFFMANN

*Royal Institute of Technology, Center for Autonomous Systems,
SE-10044 Stockholm, Sweden
E-mail: Hoffmann@nada.kth.se*

B. BAESENS, J. MARTENS, F. PUT, J. VANTHIENEN

*K.U.Leuven, Dept. of Applied Economic Sciences,
Naamsestraat 69, B-3000 Leuven, Belgium
E-mail: {Bart.Baesens; Jurgen.Martens; Ferdi.Put;
Jan.Vanthienen}@econ.kuleuven.ac.be*

In this paper, we evaluate and contrast two fuzzy classifiers for credit scoring. The first classifier uses evolutionary optimisation and boosting whereas the second classifier is based on a fuzzy neural network. We show that, for the case at hand, the boosted genetic fuzzy classifier performs better than both the neurofuzzy classifier and the well-known C4.5 algorithm that we included as a reference classifier. However, the better performance of the genetic fuzzy classifier is offset by the fact that it infers approximate fuzzy rules which are less comprehensible than the descriptive fuzzy rules inferred by the neurofuzzy classifier.

1 Introduction

Numerous methods have been proposed to develop classification models for credit scoring¹. However, most of these methods focus at developing models with high predictive accuracy without trying to explain how the classifications are made. Recent developments in fuzzy rule extraction algorithms allow to infer rules that, besides being accurate, also provide the expert with an explanation. Fuzzy rules are believed to be more comprehensible than crisp rules because they are expressed in terms of linguistic concepts. In this paper, we investigate if fuzzy rule extraction techniques can generate meaningful and accurate fuzzy rule sets for a real-life credit scoring case. We include two learning paradigms in our study: genetic algorithms and neural networks. While the former uses concepts from evolutionary optimisation, the latter uses neural network based optimisation to learn fuzzy rules. The genetic fuzzy classification algorithm infers approximate fuzzy rules whereby each rule has its own definition of membership functions. On the other hand, the neurofuzzy algorithm infers descriptive fuzzy rules where all fuzzy rules share a common, linguistically interpretable, definition of membership functions.

2 Boosted Genetic Fuzzy Classification

The past decade witnessed an increasing interest in learning algorithms that automate the knowledge acquisition step in fuzzy rule base design. The general term *genetic fuzzy rule based systems* (GFRBS) encompasses methods in which an evolutionary algorithm adapts the entire knowledge base or its individual components². Genetic fuzzy systems have been successfully applied to fuzzy control system design, fuzzy modelling and classification problems.

Our method follows the iterative rule learning approach in GFRBS augmented by a boosting mechanism to generate a set of fuzzy classification rules. Boosting algorithms for fuzzy classification systems have been previously proposed in^{3,4}. The key idea of boosting is to aggregate multiple hypotheses generated by the same learning algorithm invoked over different distributions of training data into a single composite classifier⁵. After each invocation of the rule generation algorithm the distribution of training instances is changed based on the error the newly generated rule exhibits on the training set. The evolutionary algorithm extracts a fuzzy rule that describes a subset of the current training examples. The boosting mechanism reduces the relative importance of correctly classified examples covered by the new rule. Therefore, the next rule generation step focuses on rules that capture previously uncovered or misclassified instances.

Fuzzy classification rules are of the form

$$R_i : \text{if } X_1 \text{ is } A_{1i} \text{ and } \dots X_N \text{ is } A_{Ni} \text{ then } Y = c_i \quad (1)$$

with $\{X_1, \dots, X_N\}$ a set of attributes, $\{A_{1i}, \dots, A_{Ni}\}$ a family of fuzzy sets in X_1, \dots, X_N and $\{c_1, \dots, c_K\}$ a set of classes. The class label of an unseen instance x_k is obtained by weighting and aggregating the votes c_i casted by the individual fuzzy classification rules R_i . The instance is classified by the class label that accumulates the majority of rule activations $\mu_{R_i}(x^k)$.

Evolutionary algorithms constitute a universal optimization method that imitates the type of genetic adaptation that occurs in natural evolution. Our approach employs an evolution strategy, a variant of evolutionary algorithms distinguished by self-adaptation of additional strategy parameters⁶. The role of the evolution strategy is to identify fuzzy classification rules that correctly classify a sufficiently large subset of training examples. Our approach operates with approximate fuzzy rules that contain their own definition of underlying fuzzy sets, rather than descriptive rules based on linguistic labels that refer to a commonly defined set of membership functions. The chromosome describing a fuzzy rule R_i encodes the characteristic points of the trapezoidal fuzzy sets A_{1i}, \dots, A_{Ni} in the antecedent and the rule classification c_i . The genetic repre-

sentation of fuzzy sets makes no special provision for discrete integer attributes, but treats them as if they were continuous.

The fitness of a classification rule depends on a number of optimality criteria, such as rule support, class coverage degree over positive examples and rule consistency. *Class coverage* compares the number of training instances covered by the rule R_i with the overall number of training instances that possess the same class label c_i . The second criterion *rule support* computes the overall fraction of instances covered by the rule independent of their class label. *Rule consistency* compares the number of correctly n_c^+ and incorrectly n_c^- classified instances covered by the rule. The individual criteria *class coverage*, *rule support* and *rule consistency* are aggregated into a scalar fitness value to be maximized by the evolution strategy. For details of the genetic representation and the exact definition of the fitness function the reader is referred to³.

The boosting algorithm repeatedly invokes the genetic fuzzy rule generation method on the current distribution of training examples and adjusts the weights w^1, \dots, w^M of the instances x^1, \dots, x^M according to the error $E(R_t)$ of the new fuzzy rule R_t . Each fuzzy classification rule constitutes an incomplete, weak classifier, in the sense that it only classifies those instances matched by its antecedent but makes no prediction about training examples outside its support. Therefore, the classification error $E(R_t)$ of a fuzzy rule R_t is determined by the degree of matching $\mu_{R_t}(x^k)$ between the training instance (x^k, c^k) and the rule antecedent as well as its weight w^k

$$E(R_t) = \frac{\sum_{k|c^k \neq c_t} w^k \mu_{R_t}(x^k)}{\sum_k w^k \mu_{R_t}(x^k)}. \quad (2)$$

The distribution of training instances is updated such that the weight $w^k(t)$ of instances correctly classified by R_t is reduced by a factor $\beta^k = \left(\frac{E(R_t)}{1-E(R_t)}\right)^{\mu_{R_t}(x^k)}$. The amount of weight reduction depends on the error $E(R_t)$ of the fuzzy rule as well as the degree of matching $\mu_{R_t}(x^k)$ between the fuzzy rule and the training example. Effectively, those examples that match the rule antecedent and are classified correctly are down-weighted, whereas misclassified or uncovered examples maintain their original weights. The optimal number of rules is determined by a simple heuristic, in that the boosting algorithm generates up to 20 rules and ultimately only retains the number of rules that show the smallest training error.

In order to classify unseen instances, the votes of the fuzzy rules R_t are aggregated into a final classification. The vote c_t casted by a rule R_t is weighted by the factor $\log(1/\beta_t)$ with $\beta_t = \frac{E(R_t)}{1-E(R_t)}$ such that more accurate rules have a larger impact on the overall classification. An unseen instance x is

classified by the class label that accumulates the highest weighted votes or $\operatorname{argmax}_c \{ \sum_{R_t | c_t=c} \log(1/\beta_t) \mu_{R_t}(x) \}$. Weighting has the drawback that the interpretation of the classifier becomes less intuitive, as the contribution of a rule to the overall classification not only depends on its activation but on its weight as well.

3 Neurofuzzy Classification using NEFCLASS

In this section, we provide only a brief walkthrough of NEFCLASS. A detailed explanation of NEFCLASS may be found in ⁷. NEFCLASS is a so-called three-layer *fuzzy Perceptron* network. It uses fuzzy sets as weights between the input and the hidden layer, and 0/1 weights between the hidden and the output layer. The neurons on the input layer correspond with the features in the attribute space, the neurons on the hidden layer represent the fuzzy rules, and the neurons on the output layer stand for the different classes. A fuzzy *if then* rule is derived from a hidden neuron by assembling all input to hidden layer connection weights in an antecedent part and by setting the conclusion part equal to the class of the output neuron to which the hidden neuron is connected.

Prior to the learning algorithm, each feature is equipped with a number of fuzzy sets. These fuzzy sets are associated with linguistic terms such as *small, medium, large*, etc. These terms form the universe of input to hidden layer connection weights, and thus make up the granularity of description for each feature in the antecedent part of a rule. The fuzzy sets representing these terms may be shifted or their core and support widened or shortened during learning, but their connection with the linguistic terms remains fixed. To maintain a level of consistency, NEFCLASS enforces that for each attribute, a linguistic term is modelled by the *same* fuzzy set in *each* rule. NEFCLASS thus implements a *descriptive* fuzzy rule base as apposed to the *approximate* rule base of the genetic algorithm in section 2. In addition, since weights on hidden to output layer connections are 0/1 weights, rules are not weighted by NEFCLASS contrary to the genetic fuzzy classifier.

The rule base induction algorithm of NEFCLASS consists of three parts: (1) creation of an initial set of rules, (2) selection of the best rules according to some criterion, and (3) fine tuning of the fuzzy sets that model the linguistic terms. Typical of NEFCLASS is the third step where it invokes a fuzzy heuristic variant of the gradient descent method that is used to adapt the connection weights in a classic three-layer neural network. This algorithm is known as *fuzzy backpropagation* (see ⁷ for further details).

One of the shortcomings of NEFCLASS is that, when the algorithm is

blindly applied to a particular classification problem, it may easily lead to a very large rule base, scoring high on instances from the training set, but having little generalisation capability to correctly classify new, unseen cases. This problem of *overfitting* is inherent to the rule creation algorithm of NEFCLASS and is directly linked to the granularity of the fuzzy partitioning. Previous work⁸ revealed that the performance of NEFCLASS significantly improves when some sort of *feature selection* or *input space reduction* (e.g. factor analysis) is performed. In this paper, we apply a number of pruning strategies that were recently added to NEFCLASS. In contrast to feature selection or factor analysis, the pruning algorithm is wrapped by the NEFCLASS learning algorithm. The NEFCLASS pruning strategies are covered in⁷.

4 Empirical Evaluation

4.1 Data set and experimental setup

We obtained credit scoring data from a major Benelux financial institution. The purpose is to classify applicants as creditworthy or not using both socio-demographic inputs and loan specific information. The data set consists of 3123 observations each described by 28 inputs. A bad loan is defined as a loan whereby the customer has missed three consecutive months of payments¹. We split the data set randomly into two-third training set and one-third test set. We include C4.5 and C4.5rules as a benchmark to compare the results of the fuzzy rule extraction algorithms⁹. All algorithms are evaluated by their classification accuracy and their complexity. For the C4.5 tree, the complexity is measured by the number of leaf nodes and the total (leaf and internal) number of nodes. We compare the classification accuracy of the classifiers using McNemar’s test¹⁰.

4.2 Results

Table 1 presents the classification accuracy and the complexity of both fuzzy classifiers and C4.5(rules). For the genetic fuzzy classifier, the evolution strategy automatically adapts the mutation rates instead of working with fixed mutation rates. Furthermore, it uses a (μ, λ) -selection scheme in which the 20 best individuals out of 100 offspring, become the parents for the next generation. The single best rule after 40 generations is added to the rulebase. The cycle of reweighting the training instances and invoking the evolution strategy to identify the next fuzzy rule is repeated until 40 fuzzy rules are generated. Figure 1 shows the evolution of training and test set accuracy with the number of rules. The low classification rate for a small number of rules is due to the

| | Training set | Test set | Complexity |
|------------------------------|--------------|----------|------------------------------|
| C4.5 | 89.67 | 69.07 | 269 leaves and 429 nodes |
| C4.5rules | 76.61 | 69.64 | 17 propositional rules |
| Genetic Fuzzy | 75.94 | 72.05 | 20 fuzzy rules |
| Genetic Fuzzy | 75.32 | 71.89 | average of 11-20 fuzzy rules |
| NEFCLASS no list pessimistic | 67.73 | 67.07 | 3 rules |
| NEFCLASS no list optimistic | 69.40 | 69.55 | 3 rules |
| NEFCLASS list pessimistic | 64.51 | 62.92 | 29 rules |
| NEFCLASS list optimistic | 67.10 | 65.03 | 29 rules |

Table 1: Classification accuracy of the fuzzy classifiers versus C4.5(rules).

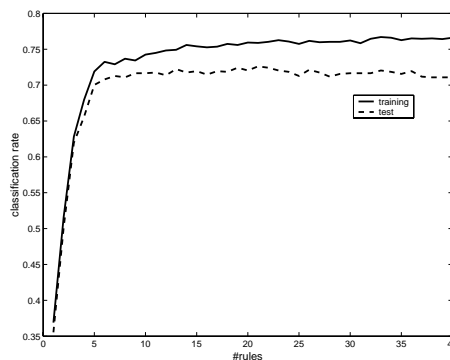


Figure 1: Evolution of the training and test set classification rate of the genetic fuzzy classifier.

fact that a substantial number of instances are initially not covered by any rule and are therefore counted as misclassified.

Figure 1 clearly illustrates that the classification accuracy on both the training and test set stagnates after approximately 11 rules and hence no significant overfitting occurs when more fuzzy rules are added to the knowledge base. According to the heuristic to select the classifier with the lowest training error, the final number of fuzzy rules equals 20. The average test set classification rate 71.89% of the ten classifiers with 11 to 20 rules is slightly lower than the rate 72.05% of the classifier with 20 rules. The latter is significantly better than both C4.5 and C4.5rules according to McNemar's test using a significance level of 5%.

For NEFCLASS, we experimented with both triangular and trapezoidal membership functions and used 2, 4 and 6 fuzzy sets per variable. We first treated the categoric variables as if they were continuous. We then also mod-

| |
|---|
| <p>Descriptive fuzzy rule If Percentage of Financial Burden is large and Code Regular Saver is large Then Applicant=bad</p> <p>Approximate fuzzy rule If Term is trapezoidal (19.19 31.90 70.25 81.42) and Property is trapezoidal(8.39 8.39 9.37 9.46) Then Applicant=bad Weight=0.71</p> |
|---|

Figure 2: Descriptive versus approximate fuzzy rules.

elled the categoric variables using list type membership functions which may be more interpretable. Furthermore, we also varied the upper limit on the number of rules in the knowledge base. Varying all these parameters of the NEFCLASS classifier, we found that the performance in terms of classification accuracy highly depends upon the specific parameter settings. Table 1 reports the best results we achieved using both list and non-list type membership functions for the categoric variables. Note that Table 1 reports both an optimistic and a pessimistic performance estimate for NEFCLASS. The former classifies all unclassified observations into the majority class (good applicant) whereas the latter treats them as being misclassified. Compared to the genetic fuzzy classifier, the best performance of NEFCLASS is clearly inferior yielding only 69.55% classification accuracy on the test set. This was a statistically significant difference according to McNemar’s test. However, the performance of NEFCLASS is not statistically different from C4.5(rules) hereby using only 3 descriptive fuzzy rules instead of 17 crisp propositional rules. Note also that the NEFCLASS configuration using no list type membership functions yields better performance than with list type membership functions. Figure 2 displays an example of a descriptive fuzzy rule inferred by NEFCLASS and an approximate fuzzy rule inferred by the boosted genetic fuzzy classifier.

In summary, the genetic fuzzy classifier yields the best classification accuracy on the test set. However, the fuzzy rules inferred are approximate fuzzy rules and may be less interpretable for the credit scoring expert as suggested by the rule examples in figure 2. Although the NEFCLASS classifier is clearly inferior in performance, its concise inferred rule set may be more comprehensible for the expert.

5 Conclusions

In this paper, we have evaluated and contrasted two types of fuzzy classifiers for credit scoring. Two learning paradigms were used to train the classifiers.

One learning paradigm used genetic optimisation and boosting whereas the other was based on a neural network representation. The genetic fuzzy classifier infers approximate fuzzy rules whereby each rule has its own definition of membership functions. The neurofuzzy algorithm infers descriptive fuzzy rules where all fuzzy rules share a common, linguistically interpretable, definition of membership functions. The experiments were carried out on a real life credit scoring data set. We showed that the genetic fuzzy classifier yields the best classification accuracy though the approximate fuzzy rules inferred are less intuitive and humanly understandable. An interesting topic for further research might therefore be to extend the genetic fuzzy classifier to infer descriptive fuzzy rules to enhance its interpretability.

1. L.C. Thomas. A survey of credit and behavioural scoring: forecasting financial risk of lending to customers. *International Journal of Forecasting*, 16:149–172, 2000.
2. O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena. *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. Advances in Fuzzy Systems. World Scientific, Singapore, 2001.
3. F. Hoffmann. Boosting a genetic fuzzy classifier. In *Proc. Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, pages 1564–1569, Vancouver, Canada, July 2001.
4. L. Junco and L. Sanchez. Using the adaboost algorithm to induce fuzzy rules in classification problems. In *Proc. Spanish Conference for Fuzzy Logic and Technologies (ESTYLF 2000)*, pages 297–301, Sevilla, Spain, September 2000.
5. Y. Freund and R.E. Schapire. Experiments with a new boosting algorithm. In *Proc. of the 13th Int. Conf. on Machine Learning ML-96*, pages 148–156, 1996.
6. T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, 1996.
7. D. Nauck. Data analysis with neuro-fuzzy methods. *Habilitation Thesis, University of Magdeburg*, 2000.
8. J. Martens, G. Wets, J. Vanthienen, and C. Mues. Improving a neuro-fuzzy classifier using exploratory factor analysis. *International Journal of Intelligent Systems*, 15:785–800, 2000.
9. J.R. Quinlan. *C4.5 programs for machine learning*. Morgan Kaufmann, 1993.
10. T.G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1924, 1998.