Available online at www.sciencedirect.com

ScienceDirect

Procedia Economics and Finance 10 (2014) 4 – 9

Procedia Economics and Finance

www.elsevier.com/locate/procedia

7th International Conference on Applied Statistics

Applying fuzzy logic and machine learning techniques in financial performance predictions

Adrian Costea[a,b,*]

[a]University of Economic Studies, Statistics and Econometrics Department, Virgil Madgearu Building, Calea Dorobantilor No.15-17, 6th floor, Sector 1, Bucharest 010552, Romania

[b]National Bank of Romania, Bucharest Regional Branch, Statistical Reporting and Business Surveys Unit, Lipscani st. No. 16, 5th floor, room 5090, Sector 3, Bucharest 030035, Romania

Abstract

In this article we apply a fuzzy logic technique, namely Fuzzy C-Means clustering, and artificial intelligence algorithms for evaluating comparatively the financial performance of non-banking financial institutions (NFIs) in Romania. The NFIs' performance dataset consists of indicators that define the capital adequacy, assets' quality and profitability performance dimensions. The class performance variable is obtained by applying on the performance dataset the Fuzzy C-Means algorithm and obtaining clusters with similar performance. We attach to each input dataset observation a performance class depending on which cluster contains the observation given the characterization and hierarchy of the clusters in "good", "medium" and "poor" performance clusters. Finally, we apply artificial neural networks (ANNs) trained with genetic algorithms in order to find a function that maps the input performance space on the newly constructed performance class variable. The classification model obtained can be used by different beneficiaries (e.g.: the Supervision Department of National Bank of Romania) to classify new NFIs as having a "good" or "poor" performance so that the limited resources of the supervision authority to be better allocated.

Selection and peer-review under responsibility of the Department of Statistics and Econometrics, Bucharest University of Economic Studies.

Keywords: classification models; artificial neural networks; genetic algorithms; non-banking financial institutions

* Corresponding author. Tel.: +4-031-132-6207; fax: +4-021-319-1793.
E-mail address: adrian.costea@csie.ase.ro

Selection and peer-review under responsibility of the Department of Statistics and Econometrics, Bucharest University of Economic Studies.
doi:10.1016/S2212-5671(14)00271-8

1. Introduction

The non-banking financial institutions' (NFIs') sector has been recently regulated in Romania. The National Bank of Romania (NBR) proposed and the Parliament passed a series of provisions regarding the activities carried out by these entities in order to strengthen the stability of the financial sector as a whole. According to these provisions, the NBR is obliged to prudentially supervise the most important NFIs: those register in the NBR' Special register. The other NFIs are monitored and only in special cases are further scrutinized. However, all NFIs have to report on their activity to the Supervision Department. One such reporting consists of periodic financial statements (PFSs) that a NFI has to sent to NBR quarterly. Currently, these PFSs are analysed manually by the inspectors from the department. Based on their assessment, NFIs that present difficulties are further scrutinized and, eventually, an on-site inspection is organized. The scarce time and personnel resources of the Supervision Department and the need to balance the subjective interpretation of the inspectors motivate the use of some sorts of techniques that would classify the NFIs as having a "good" or "poor" performance. Based on these techniques we can build so-called classification models that might provide additional help in taking the decision for an on-site inspection.

Data Mining techniques (Han & Kamber, 2006) can help with building the clustering/classification models. Clustering techniques can be used to find performance clusters within the NFIs' performance dataset and classification techniques can be used to place a new NFI into a predefined cluster as data become available. In other words, clustering techniques have descriptive properties and classification techniques have prescriptive ones.

In this article we apply the Fuzzy C-Means algorithm (Bezdek, 1981) and obtaining clusters with similar performance. We attach to each input dataset observation a performance class depending on which cluster contains the observation given the characterization and hierarchy of the clusters in "good", "medium" and "poor" performance clusters. Finally, we apply artificial neural networks (ANNs) trained with genetic algorithms in order to find a function that maps the input performance space on the newly constructed performance class variable. In this way we overcome the problem of accommodating new NFIs on the performance map as data become available.

Next, we present in more detail our methodology. Then, we present the NFIs' performance dataset and our experiment on applying the Data Mining techniques on these data. Finally, we draw our conclusions.

2. Methodological framework

We use Fuzzy C-Means (FCM) algorithm (Bezdek, 1981) to group the NFIs with similar characteristics. The FCM algorithm minimizes the following objective function, $J_m(U, v)$:

$$J_m(U,v) = \sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m (d_{ik})^2 \tag{1}$$

where c is the number of clusters, n is the number of observations, $U \in M_{fc}$ is a fuzzy c-partition of the data set X, $u_{ik} \in [0,1]$ is the membership degree of observation x_k in cluster i,

$$d_{ik} = \|x_k - v_i\| = \left[\sum_{j=1}^{p}(x_{kj} - v_{ij})^2\right]^{1/2} \tag{2}$$

is the Euclidean distance between the cluster center v_i and observation x_k for p attributes (financial ratios in our case), $m \in [1,\infty)$ is the weighting exponent, and the following constraint holds

$$\sum_{i=1}^{c} u_{ik} = 1. \tag{3}$$

The solution for the FCM clustering consist of the final cluster centers and membership degrees of the observations. Once we obtain the final cluster centers and membership degrees we construct the performance class

variable by attaching to each observation a performance class depending on which cluster contains the observation. The number of clusters *c* and the weighting exponent *m* are parameters for FCM algorithm.

Once we constructed the performance class variable we apply artificial neural networks (ANNs) trained using genetic algorithms to build the classification model. Firstly, we apply ANNs trained using normal gradient descent algorithms in order to find the best ANN architecture. The general procedure for determining the ANN architecture consists of the following steps:

- split the dataset in training (TR) and test (TS) sets;
- perform 3 experiments and for each experiment:
  - split the training set in effective training (TRe) and validation (VAL) sets;
  - vary the number of neurons in the first and second hidden layers (NH_1, NH_2) between 5 and 8 and for each combination perform 4 trainings follows:
    - initialize the ANN weights;
    - train the ANN;
    - calculate the mean squared errors for each set (TR, VAL, TR and TS);
- we save the ANN architecture if the mean squared error for VAL is less than the one for TRe multiplied by a factor of 6/5. This condition has been imposed in order to avoid saving ANN architectures for which the effective training and validation mean squared error are too far from each other.
- at the end, we obtain the best ANN architecture in terms of mean squared error for training (TR).

Once we obtain the architecture, we can use genetic algorithms (GA) to train the network. Unlike the traditional gradient-descent training mechanisms, GAs are provided with a population of solutions, and by initialisation, selection and reproduction mechanisms, achieve potentially good solutions. All solutions (chromosomes) compete with each other to enter the new population.

In the case of GA-based ANN training, the GA's chromosome (solution) is the set of ANN weights after training represented as a vector. Next, we describe the GA steps performed to train the ANN.

2.1. Initialisation and fitness evaluation

The population size is a parameter of our models. It was set to $PS = 20$. Dorsey & Mayer (1995) suggest that this value is good enough for any grade of problem complexity. The first chromosome of the population is the set of weights obtained when determining the ANN architecture. The other 19 chromosomes are generated by training the ANN with the previously obtained architecture. Afterwards, the first generation of the algorithm may begin. The number of generations is related to the empirical formula suggested in Ankenbrandt (1991). Each chromosome is evaluated using the accuracy rate for the training set (ACR_{TR}).

2.2. Selection

Firstly, the elitism technique is applied in the sense that the best N_{elite} chromosomes in terms of ACR_{TR} are inserted into the new population. The rest of the chromosomes (20-N_{elite}) are selected based on the probability of selection (*roulette wheel* procedure) for each chromosome:

$$P_i = ACR_{TR-i} : SUM\left(ACR_{TR-i}\right)$$

The higher the probability P_i for a chromosome is, the higher its chance of being drawn into the new population. We decided to employ *elitist* selection in our algorithms as a consequence of what was reported in the literature. Rudolph (1994), Miller & Thomson (1998), Shimodaira (1996), Fogel *et al*. (2004) are a few papers that prove the usefulness of using elitist selection.

Next, 80 per cent (probability of crossover: $P_c = 0.80$) of the chromosomes obtained previously are randomly selected for mating. The probability of crossover is not essential for the performance of our algorithm as long as it has a high value. This is because after reproduction we increase the population to include both the parents and their offspring.

2.3. Reproduction

The selected chromosomes are randomly paired and recombined to produce new solutions. There are two reproduction operators: *crossover* and *mutation*. With the first the mates are recombined and newborn solutions inherit information from both parents. With the second operator new parts of the search space are explored and, consequently, we expect that new information will be introduced into the population. We used one-point crossover. For each pair of chromosomes we generate a random integer X, $X \in \{1, L\}$ where L is the chromosome length. The two new born children are constructed as follows: $C_1 = g_{11}, g_{12}, \ldots, g_{1X}, g_{2,X+1}, \ldots, g_{2L}$ and $C_2 = g_{21}, g_{22}, \ldots, g_{2X}, g_{1,X+1}, \ldots, g_{1L}$

The children chromosomes are *added* to the population. The size of the population becomes $PS' > PS$. Next, we apply the mutation operator for all the chromosomes in PS'. We used only uniform mutation and add the new obtained chromosomes into the new population obtaining $PS'' > PS' > PS$. The probability of mutation is set to $P_m = 0.01$, which means that approximately one per cent of the genes will mutate for each chromosome.

The final step in constructing the new population is to reduce it in size to 20 chromosomes. We select from PS'' the best 20 chromosomes in terms of ACR_{TR} satisfying the condition that one chromosome can have no more than *max_lim* duplicates. We use the mutation operator to generate more chromosomes if the number of best chromosomes that satisfy the above condition is less than 20.

As a summary, excluding the crossover, the parameters of our GA model are as follows: number of generations (N_{gen}), population size (PS), number of elite chromosomes (N_{elite}), probability of crossover (P_c), probability of mutation (P_m), and maximum number of duplicates for the chromosomes (*max_lim*).

3. The dataset

Our NFIs' performance dataset consist of 11 indicators: 3 for the degree of capitalization, 4 for assets' quality and 4 for profitability. The data were collected quarterly from 2007 to 2012 for the NFIs registered in the Special Register that have been active since the introduction of the regulatory framework for these institutions in Romania. In total there were 68 NFIs that met the above criteria and 990 observations. Out of these 990 observations, 5 observations were discarded due to lack of data for certain financial indicators. In Table 1 we present the indicators for each performance dimension:

Table 1. The performance dimension and the corresponding financial ratios

Dimension	Indicators
Capital adequacy	Equity ratio (Leverage) = own capital / total assets (net value)
	Own capital / equity
	Indebtedness sources = borrowings / total assets (net value)
Assets' quality	Loans granted to clients (net value) / total assets (net value)
	Loan granted to clients (net value) / total borrowings
	Past due and doubtful loans (net value) / total loans portfolio (net value)
	Past due and doubtful claims (net value) / total assets (net value)
Profitability	Return on assets (ROA) = net income / total assets (net value)
	Return on equity (ROE) = net profit / equity
	The rate of profit = gross profit / total revenues
	Activity cost = total costs / total revenues

4. Experiment

Our dataset that consist of 11x985 observations has been transformed by levelling the extreme values for each variables in the [-20, 20] interval. We have done this in order to avoid the algorithms' results being affected by these extreme values.

In the next step, we apply FCM algorithm in order to build cluster with similar performance. We chose 4 clusters as we have done with a version of the same dataset in our previous work.

The other parameters of FCM were as follows: m = 1.5, no_of_iterations = 10000, the limit for the stopping criterion = 0.00001. After we run the FCM algorithm on the 11x985 dataset we obtained the following structure of the clusters: cluster 1 (95 observations), cluster 2 (770 observations), cluster 3 (59 observations), and cluster 4 (61 observations). Based on the clusterization we have constructed the class variable by associating to each observation the number of the cluster that the observation belongs to.

In order to have an uniform number of observations in each cluster to train the classification model we selected 59 observations (the number of observations in the smallest cluster) from each cluster, totalling 236 observations. Also, at this stage, we have split the data in training (*TR*) and testing (*TS*) sets by selecting one testing instance for every nine training instances. Thus, we obtained randomly 212 observations for training and the rest for testing (24 observations).

The next step of the methodology was to determine the proper ANN architecture based on the general procedure described in Section 2. The final ANN architecture consisted of 8 neurons on the first hidden layer and 5 neurons on the second hidden layer.

Next, we applied GAs to train the previously obtained network. The parameters used were number of generations (N_{gen} = 1000), population size (PS = 20), number of elite chromosomes (N_{elite} = 3), probability of crossover (P_c = 0.8), probability of mutation (P_m = 0.01), and maximum number of duplicates for the chromosomes (max_lim = 1).

The choice of crossover probability as well as the other GA parameters (mutation probability, population size) is more art than science. Tuson & Ross (1998) suggested that the proper choice of the crossover in the case of non-adaptive GAs depends upon the population model, the problem to be solved, its representation and the performance criterion being used. DeJong (1975) considers mutation probability to be inversely proportional to population size. Hesser & Männer (1990) include in the calculation of mutation probability both population size and chromosome length. Hoehn (1998) introduced mutation at both parental and offspring levels and implemented four GAs based on the mutation probabilities for the two levels. The author finds that introducing parental mutation is generally advantageous when compared to the standard GA with only offspring mutation. In our experiments we used both parental and offspring mutation by applying mutation to both parents and their offsprings.

We obtained the following accuracy rates: effective training dataset accuracy rate (ACR_{Tre}) = 92.32 percent, validation dataset accuracy rate (ACR_{VAL}) = 91.34 percent, total training dataset accuracy rate (ACR_{TR}) = 92.11 percent and testing dataset accuracy rate (ACR_{TS}) = 89.55 percent. We obtained high accuracy rates and small differences between training and testing accuracy rates. However, the setting of the Gas parameters has to be done carefully in order for these high rates to be obtained in another context (another dataset).

5. Conclusions

In this study we train artificial neural networks with genetic algorithms in order to obtain performance classification models that might be used to find the future performance of non-banking financial institutions in Romania. As a first phase of our methodology we applied a clustering technique (FCM algorithm) in order to group NFIs with similar performances. Then, using initialization, selection and reproduction mechanisms we trained a neural network based on genetic learning. A brief rationale supported with references is given regarding the parameters' setting for the genetic algorithm.

We obtained high training and testing accuracy rates for the classifier and small differences between training and testing accuracy rates. The reproduction of the process of natural selection (which forms the base for the genetic learning) in addressing real problems is welcomed. However, the results have to be taken with some precaution, given the genetic algorithms take long time to learn compared to other computational techniques such as

backpropagation or other gradient-descent-like algorithms. The above classification model can be applied to discriminate between “poor” and “good” performing NFIs, hence helping the supervision authority in its activities.

Acknowledgements

This work was supported from the European Social Fund through Sectoral Operational Programme Human Resources Development 2007-2013, project number POSDRU/89/1.5/S/59184 „Performance and excellence in postdoctoral research in Romanian economics science domain”.

References

Ankenbrandt, C.A., 1991. An extension to the theory of convergence and a proof of the time complexity of genetic algorithms, Proceedings of 4th International Conference on Genetic Algorithm, pp. 53-68.

Bezdek, J.C., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.

DeJong, K.A., 1975. An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. dissertation, University of Michigan, Ann Arbor, MI.

Dorsey, R.E., Mayer, W.J., 1995. Genetic Algorithms for Estimation Problems with Multiple Optima, Non-differentiability, and other Irregular Features. Journal of Business and Economic Statistics 13(1), 53-66.

Fogel, G.B., D.G. Weekes, R. Sampath, Ecker, D.J., 2004. Parameter Optimization of an Evolutionary Algorithm for RNA Structure Discovery, Proceedings of 2004 Congress on Evolutionary Computation, IEEE Press, Piscataway, NJ, pp. 607-613.

Han, J., Kamber, M., 2006. Data Mining: Concepts and Techniques, second edition, Morgan Kaufmann, 2006.

Hesser J., Männer, R., 1990. Towards an Optimal Mutation Probability for Genetic Algorithms, Proceedings of the 1st Workshop on Parallel Problem Solving from Nature, Sringer-Verlag, October 01-03, pp.23-32.

Hoehn, P.T., 1998. Wolves in Sheep’s Clothing? The Effects of <<Hidden>> Parental Mutation on Genetic Algorithm Performances, Proceedings of ACM 36th annual Southeast regional conference, pp. 221-227.

Miller J.F., Thomson, P., 1998. Aspects of Digital Evolution: Geometry and Learning, Proceedings of the 2nd International Conference on Evolvable Systems - ICES98, September 23-25, 1998, EPFL, Lausanne, Switzerland.

Rudolph G., 1994. Convergence analysis of canonical genetic algorithms. IEEE Transactions on Neural Networks 5, 96-101.

Shimodaira H., 1996. A New Genetic Algorithm Using Large Mutation Rates and Population-Elitist Selection (GALME), Proceedings of the 8th International Conference on Tools with Artificial Intelligence (ICTAI '96), pp. 25-32.

Tuson A., Ross, P., 1998. Adapting Operator Settings in Genetic Algorithms. Evolutionary Computation 6(2), 161-184.