Discovering Causes of Financial Distress by Combining Evolutionary Algorithms and Artificial Neural Networks

A.M. Mora, P.A. Castillo, J.J. Merelo
Departamento de Arquitectura y Tecnología de Computadores
Universidad de Granada, Spain
{amorag,pedro,jmerelo}@geneura.ugr.es

E. Alfaro-Cid, A.I. Esparcia-Alcázar, K. Sharman
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia, Spain
{evalfaro,anna,ken}@iti.upv.es

ABSTRACT

In this work we compare two soft-computing methods for producing models that are able to predict whether a company is going to have book losses: artificial neural networks (ANNs) and genetic programming (GP). In order to build prediction models that can be applied to an extensive number of practical cases, we need simple models which require a small amount of data. Kohonen's self-organizing map (SOM) is a non-supervised neural network that is usually used as a clustering tool. In our case a SOM has been used to reduce the dimensions of the prediction problem. Traditionally, ANNs have been considered able to produce better classifier structures than GP. In this work we merge the capability of GP for generating classification trees and the feature extraction abilities of SOM, obtaining a classification tool that beats the results yielded using an evolutionary ANN method.

Categories and Subject Descriptors

J.4 [**Computer Applications**]: Social and Behavioral Sciences; I.5.2 [**Computing Methodologies**]: Pattern Recognition—*Design Methodology*

General Terms

Economics

Keywords

Financial distress prediction, Artificial Neural Networks, Self-Organizing Maps, Genetic Programming

1. INTRODUCTION

Prediction of financial distress is one of the most interesting topics to research from a company manager point of view. To begin with, it is a difficult problem to solve because there are lots of variables to take into account and moreover, there are many relationships (visible or hidden) between them, which makes it hard to advance this situation. Looking at the history of this subject, the path of the studies moves from complex, accurate and difficult to implement approaches like univariate and statistical analysis [4, 5]; to more generic (less restrictive) tools, like the application of artificial neural networks (ANN) [11, 17, 18] and genetic programming (GP) [7, 20, 30].

A problem with using statistical techniques is the requirement for a functional relation among dependent and independent variables. In addition, these methods have a restricted range of application because they are very sensitive to exceptions. On the other hand, soft-computing techniques are more flexible.

In this paper we compare GP and ANNs for book losses prediction. The database used contains more than 400 companies and includes not only financial data from the companies, but also general information that can be relevant when predicting failure. The database is the same used in other studies for prediction of bankruptcy [1, 3]. One of the factors that was found to be the key in the successful application of GP to the bankruptcy prediction problem was the reduction of the number of variables. So, in [1, 3] the prediction was done in two steps. To start with, GP was run with all the available variables. Then, the resulting trees were analyzed to identify which data were used more frequently by the GP algorithm. This could be done since GP creates analytical models as a final result. Finally, the proper prediction models were evolved using only those variables that had been identified as important in the first stage. The results showed that reducing the number of variables not only simplifies the GP classifiers structures, but also improves the classification rates.

In [27], Mora et al. used a Kohonen's self-organizing map (SOM) [19] for surveying the financial status of Spanish companies. Using (visualizing) the map, the authors inferred which are the most relevant variables (in agreement with those identified using statistical techniques) so that, a fast diagnostic on the status of a company can be reached. Thus, in [2], Alfaro-Cid et al. considered some of the conclusions yielded in [27] to choose a set of variables which might be significant to predict the book losses for a company. The prediction itself was done using GP. The results showed in that article proved that SOM is a useful tool for reducing the dimensions of the prediction problem.

On the other hand, evolutionary neural networks are an efficient way of searching for the problem space of the neu-

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.

ral net [34]. Several authors have suggested evolving ANNs by coding the weights and learning parameters of the individuals of an evolutionary algorithm (EA), pre-establishing the number of neurons and the connectivity between them [21, 25]. However, these representations can lead to a lack of precision by restricting the search to just an area of the possible space. Leung et al. in [21] proposed the tuning of the parameters of an ANN by using an improved genetic algorithm (GA), but the number of hidden nodes must be chosen manually by increasing it from a small number until the learning performance is good enough. The method proposed in [23] tries to avoid overfitting by encoding the number of training epochs as a bit string in the individual chromosome. In principle, although the performance expected from this setup could be higher, without incurring longer runtimes, it would come at the cost of more difficulty in understanding the encoding.

That is the reason why in this work we want to assess the performance of the GP+SOM approach by comparing the results that it yields with those obtained with an evolutionary ANN method, GProp [12], which has proved its adequacy for solving classification problems, especially bankruptcy prediction [11].

Finally, it is important to mention that this work tackles the problem of prediction of book losses. This problem is interesting since there is a direct relationship between continued book losses and legal bankruptcy [29]. Moreover book losses usually happen at a stage prior to insolvency, so predicting book losses gives the management of a company more time to react and find a solution to the problem.

The remainder of this paper is organized as follows: section 2 describes the dataset used to make the study. The methods used to process the samples are introduced in section 3. Section 4 shows the results yielded by these methods, and the related conclusions as well as the future lines of work are reported in section 5.

2. DATA DESCRIPTION

The data used in this work were extracted from the Infotel database[1], and it is a set composed of data from 470 companies. 170 of these companies had continuous book losses during the years 2001 - 2003, while the remaining 300 companies presented a good financial health. There are available data of these companies from years 1998, 1999 and 2000 to perform the prediction.

Table 1 shows the independent variables, their description and type. As it can be seen, the variables can take values from different numerical ranges: real, integer and binary. Also, some of the non-financial data take categorical values; these are the type of company, the auditor's opinion and the size of the company; which usually is a numerical variable, but in this case means the category according to this size. Each categorical variable can take 3 different values, so to work with them they have been transformed into 3 binary variables each one. For example, the *size* variable has 3 possible values, '1', '2' or '3', thus we can create three new variables *size1*, *size2* and *size3*. Each one will have a value of '1' if the old value of *size*, was '1', '2' or '3', respectively, and a value of '0' otherwise. After this transformation, the available data set for each company includes 37 independent variables: 18 real, 7 integer and 12 binary variables.

The dependent variable takes a value of '1' if the company has suffered book losses 3 years in a row or '0', otherwise.

The available data have been divided into the training and testing sets. The training set comprises round 70% of the data and the remaining 30% has been used for testing.

3. METHODOLOGY

In the following subsections the general concepts related to our work are introduced and the approaches used in the study are presented.

3.1 GP Approach

In this section we briefly describe the GP framework that we have used to predict book losses. Basically, the GP algorithm must find a structure (a function) which can, once supplied with the relevant data from the company, decide if this company is going to have book losses or not. In short, it is a binary classification problem.

The classification process works as follows. Let $\boldsymbol{x}_i = \{x_{0i}, \ldots, x_{\mathrm{N}-1i}\}$ be the state of the *ith* company. Let $f(\boldsymbol{x}_i)$ be the function defined by a GP tree structure. We can apply $\boldsymbol{x}_i$ as the input to the GP tree and calculate the output $f(\boldsymbol{x}_i)$. Once the numerical value of $f(\boldsymbol{x}_i)$ is calculated, it will give us the classification result according to:

$$f(\boldsymbol{x}_i) > 0, \ \nabla i \in L \tag{1}$$

$$f(\boldsymbol{x}_i) \leq 0, \ \nabla i \in \overline{L} \tag{2}$$

where L represents the class to which companies with book losses belong and $\overline{L}$ represents the class to which healthy companies belong. The task of GP is to find the function $f(\boldsymbol{x}_i)$.

In order to define the fitness evaluation function, it is important to notice that there is an unbalance in the database in the sense that only 170 companies have book losses versus 300 healthy companies, we have modified the cost associated to misclassifying the positive and the negative classes to compensate for the imbalanced ratio of the two classes [16]. For example, if the imbalance ratio is 1:10 in favor of the negative class, the penalty for misclassifying a positive example should be 10 times greater. Basically, it rewards the correct classification of examples from the small class over the correct classification of examples from the oversized class. It is a simple but efficient solution.

Therefore, the fitness function to maximize is:

$$Fitness = \sum_{i=1}^{N} u_i \tag{3}$$

where

$$u_i = \begin{cases} 0 & : \ \text{incorrect classification of company } i \\ \frac{N_h}{N_l} & : \ \text{company } i \text{ with losses classified correctly} \\ 1 & : \ \text{healthy company } i \text{ classified correctly} \end{cases}$$

N_l is the number of companies with book losses and N_h is the number of healthy companies.

The GP implementation used is based on ECJ [2], a research evolutionary computation system developed at George Mason University's Evolutionary Computation Laboratory (ECLab). Table 2 shows the main parameters used during evolution.

[1] Bought from http://infotel.es

[2] http://cs.gmu.edu/~eclab/projects/ecj

Table 1: Independent Variables

Variable name (used in GP)	Financial Variables	Description	Type
x_8	Debt Structure	Long-Term Liabilities /Current Liabilities	Real
x_9	Debt Cost	Interest Cost/Total Liabilities	Real
x_{10}	Cash Ratio	Cash Equivalent /Current Liabilities	Real
x_{11}	Working Capital	Working Capital/ Total Assets	Real
x_{12}	Debt Ratio	Total Assets/Total Liabilities	Real
x_{13}	Operating Income Margin	Operating Income/Net Sales	Real
x_{14}	Leverage	Liabilities/Equity	Real
x_{15}	Debt Paying Ability	Operating Cash Flow/Total Liabilities	Real
x_{16}	Return on Operating Assets	Operating Income/Average Operating Assets	Real
x_{17}	Return on Equity	Net Income/Average Total Equity	Real
x_{18}	Return on Assets	Net Income/Average Total Assets	Real
x_{19}	Asset Turnover	Net Sales/Average Total Assets	Real
x_{20}	Receivable Turnover	Net Sales/Average Receivables	Real
x_{21}	Stock Turnover	Cost of Sales/Average Inventory	Real
x_{22}	Current Ratio	Current Assets/Current Liabilities	Real
x_{23}	Acid Test	(Cash Equivalent + Marketable Securities + Net receivables) /Current Liabilities	Real
Non-financial Variables (used in GP)	Description	Type	
x_0, x_1, x_2	Size	Small/Medium/Large	Categorical
x_3	Age of the company		Integer
x_4	Audited	If the company has been audited	Binary
x_5, x_6, x_7	Type of company	Public companies/Limited liability companies (Ltd))/Others	Categorical
x_{24}	Historic amount of money spent on judicial incidences	Since the company was created	Real
x_{25}	Amount of money spent on judicial incidences	Last year	Real
x_{26}	Number of changes of location		Integer
x_{27}	Number of employees		Integer
x_{28}	Historic number of serious incidences	Such as strikes, labour accidents...	Integer
x_{29}	Historic number of judicial incidences	Since the company was created	Integer
x_{30}	Number of judicial incidences	Last year	Integer
x_{31}	Number of partners		Integer
x_{32}, x_{33}, x_{34}	Auditor's opinion	Favourable/Qualification/Unfavourable	Categorical
x_{35}	Delay	If the company has submitted its annual accounts on time	Binary
x_{36}	Linked to a group	If the company is part of a group holding	Binary

Table 2: GP parameters

Initialization method	Ramped half and half
Replacement operator	Generational with elitism (0.2%)
Selection operator	Tournament selection
Tournament group size	7
Cloning rate	0.1
Crossover operator	Bias tree crossover
Internal node selection rate	0.9
Crossover rate	0.4
Mutation rate	0.1
Tree maximum initial depth	7
Tree maximum depth	18
Population size	1000
Termination criterion	250 generations

As a method of bloat control we have included a new genetic operator that occurs with a probability of 0.4. This operator implements a bloat control approach described in [15] and inspired in the "prune and plant" strategy used in agriculture. It consist of pruning some branches of trees and planting them in order to grow new trees. The idea is that one of the branches of the selected tree will be "pruned" and "planted" as a new individual in the next generation. This way the offspring trees will be of smaller size than the ancestor, effectively reducing bloat.

We have implemented a strongly typed GP (STGP) [26]. STGP is an enhanced version of GP that enforces data-type constraints, since standard GP is not designed to handle a mixture of data types. In STGP, each function node has a return-type, and each of its arguments also have assigned types. STGP permits crossover and mutation of trees only with the constraint that the return type of each node matches the corresponding argument type in the node's parent.

A STGP has been implemented in order to ensure that in the resulting classifying models the functions operate on appropriate data types so that the final model has a physical meaning. That is, the objective is to avoid results that operate on data which are not compatible, for instance, models which add up the liabilities and the age of a company.

The terminal set used consists of the independent variables from Table 1 (initially all of them, later on a reduced set identified as more relevant by a SOM) plus Koza's ephemeral random constant. Table 3 shows the function set used and the chosen typing.

3.2 Self-Organizing Map

The self organizing map (SOM) was introduced by Teuvo Kohonen in 1982 [19]. It is a non-supervised neural network that tries to imitate the self-organization done in the sensory cortex of the human brain, where neighbouring neurons are activated by similar stimulus. It is usually used as a clustering/classification tool or used to find unknown relationships between a set of variables that describe a problem. The main property of the SOM is that it makes a nonlinear projection from a high-dimensional data space (one dimension per variable) on a regular, low-dimensional (usually 2D) grid of neurons.

SOM is further processed using Ultsch method [32], the Unified distance matrix (U-matrix), which uses SOM's codevectors (vectors of variables of the problem) as data source and generates a matrix where each component is a distance measure between two adjacent neurons. It allows us to visualize any multi-variated dataset in a two-dimensional display, so we can detect topological relations among neurons and infer about the input data structure. High values in the U-matrix represent a frontier region between clusters, and low values represent a high degree of similarities among neurons on that region, clusters.

Table 3: Function set

Functions	Number of arguments	Arguments type	Return type
+, -, *, /	2	real	real
If $arg_1 \leq arg_2$ then arg_3 else arg_4	4	real	real
If arg_1 then arg_2 else arg_3	3	arg_1 is a boolean arg_2, arg_3 are real	real
If $arg_1 \leq int$ then arg_2 else arg_3 (*int* is randomly chosen)	3	arg_1 is an integer arg_2, arg_3 are real	real

Although Kohonen's SOMs are not outstanding at the task of classification, they can be applied to many different types of data, yielding visualization of natural structures in the data and their relations, highlighting groupings and allowing the user to visually discover the number of clusters and their topological relationships. In addition, SOM makes it easy the estimation of the variables that have more influence on these groupings. Other statistical and soft computing tools can also be used for this purpose, but since Kohonen's SOMs offers a visual way of doing it, it is much more intuitive, and takes advantage of the capabilities of the human brain as a pattern recogniser.

3.3 GProp Method

The classification results obtained with GP have been compared with those yielded using an evolutionary neural network method named GProp [12].

In this method, an EA carries out the evolution of a population of multi-layer perceptrons (MLP), searching for the best architecture (network structure and initial weights) for that problem and trying to optimize the network classification ability. This method leverages the capabilities of two types of algorithms: the ability of EA to find a solution close to the global optimum, and the ability of the QP (quickpropagation, a multilayer perceptron training method) algorithm to tune it and to reach the nearest local minimum by means of local search from the solution found by the EA. This EA uses an elitist [33] algorithm and is specified in Figure 1.

In GProp, an individual is a complete MLP with two hidden layers. The representation ability of a neural network depends on the number of layers, on the number of neurons per layer and on the connectivity between layers. It was demonstrated that a network with two hidden layers can solve any pattern classification problem [6, 22].

An EA requires that each individual is encoded as a chromosome for it to be handled by the genetic operators of the EA. Some authors use binary or real encoding (representation of the networks in a binary or real number string), as proposed by [13, 14], or indirect coding, as proposed by [8], but GProp evolves the initial parameters of the network (initial weights and learning constants) using specific genetic operators.

There are six genetic operators designed to evolve individual-MLPs: mutation, crossover, training (applying QP), adding and removing hidden neurons, and substituting hidden units (macromutation).

These operators act directly upon the ANN object (instead of performing hierarchical evolution at the neuron level [28]), but only *initial weights* and the *learning constant* are subject to evolution, not the weights obtained after training.

When a genetic operator changes a MLP, it considers each hidden neuron (and its input and output weights) as a "gene", so that if two MLPs are crossed, complete hidden layer neurons are interchanged (and weights to and from it are treated as one unit), as proposed in [25, 31].

The fitness function of an individual (MLP) is given by the number of correctly classified patterns obtained on the validation process that follows training. In the case of two individuals showing an identical classification error (measured as the number of incorrectly classified patterns), the one with the hidden layer containing the least number of neurons would be considered the best (the aim being small networks with a high generalization ability).

Regarding the GProp implementation, at the lowest level, an MLP is an object instantiated from the MLP C++ class. The data structure of this class is an array of vectors of neurons, where each neuron is a vector of weights. However, the EA does not use binary strings, but MLP objects and neurons.

We used Evolutionary Objects (EO)[3] library [24], because of the facility that this library offers to evolve any object with a fitness function.

The parameters used to run the algorithm have been set considering the statistical study performed in [10], where the most important parameters (relating to their influence on the results) were identified, and the most suitable values for those parameters were established. These values are shown in Table 4.

Table 4: GProp parameter set

Parameter	Value
number of training epochs	500
number of generations	500
population size	500
selection rate	20%
initial weights range	$[-0.05, 0.05]$
mutation operator priority	2.0
crossover operator priority	0.5
addition operator priority	1.0
elimination operator priority	0.5
training operator priority	0.5
mutation probability	0.4
weight mutation range	$[-0.001, 0.001]$
learning constant mutation range	$[-0.010, 0.010]$

The number of generations and the population size needed for greater diversity should, of course, be higher or lower depending on the difficulty of the problem.

4. EXPERIMENTS AND RESULTS

For starters we have run as a benchmark a set of experiments using all available data in the dataset, and GP as a

[3] http://geneura.ugr.es/~jmerelo/EO.html

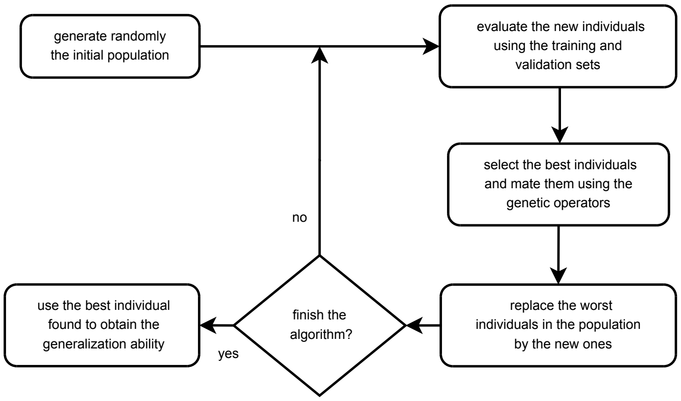
generate randomly the initial population
evaluate the new individuals using the training and validation sets
select the best individuals and mate them using the genetic operators
replace the worst individuals in the population by the new ones
finish the algorithm?
no
yes
use the best individual found to obtain the generalization ability

Figure 1: EA pseudocode.

classification method. The mean and standard deviation of 30 runs were calculated.

The average error obtained using GP for the prediction of book losses using all available variables was 34.32% ± 2.11.

The results show the difficulty of predicting book losses, since the same GP strategy obtained better classification rates when using the database for predicting bankruptcy [1, 3] (a different dependent variable).

As a second step, we have exploited the ability of SOM in identifying relevant variables in a dataset so, we have built a reduced set based on the conclusions obtained in [27], where SOM and U-Matrix methods were applied to the same dataset (but using all data for training) and considering the same variables (independent and dependent). In that work three clusters were identified and, in every one, some key variables were marked:

- *Warm spot*: large companies (*size3* set to 1), which have been audited, with *type of company1* set to 1 (which means they are joint stock companies), with a favourable auditor's opinion (*auditor opinion1* set to 1), high *acid test*, high *current ratio*, *delay in reporting the annual accounts*, high *leverage* and *belonging to a group*. There are more failed companies in this cluster than successful ones, so the cluster probably corresponds to old members of company conglomerates, with a big size, which are conveniently closed without incurring in big losses. However, there are very few companies in this zone, since most companies in the database are not linked to a group.
- *Hot spot*: successful companies, with small size (*size1* set to 1) and most economic indicators in a healthy shape.
- *Small spot*: most companies with losses with *type of company3* set to 1 (neither joint stock companies nor limited liability ones) and no other distinctive value.

The planes analysis used in that paper, also confirms that just a few variables are needed to differentiate between failed (presenting book losses) and successful companies; these variables are summarized in Table 5.

In addition, the final trees GP has converged to in the first set of runs were analyzed in order to see which variables the GP algorithm uses more frequently to solve the prediction problem. The percentage of final trees that use each variable is shown in Table 6.

Table 5: Variables identified as relevant by the SOM

Variable	Type	Variable	Type
Size1	Binary	Size3	Binary
Type of company1	Binary	Type of company3	Binary
Audited	Binary	Auditor's opinion1	Binary
Delay	Binary	Linked to a group	Binary
Leverage	Real	Acid Test	Real
Current Ratio	Real		

Table 6: Percentage of final trees that use each variable in the first set of GP results

Var.	% Trees	Var.	% Trees	Var.	% Trees
x_0	13.33	x_{13}	73.33	x_{25}	46.67
x_1	23.33	x_{14}	96.67	x_{26}	13.33
x_2	16.67	x_{15}	60.00	x_{27}	23.33
x_3	16.67	x_{16}	73.33	x_{28}	30.00
x_4	26.67	x_{17}	86.69	x_{29}	26.67
x_5	30.00	x_{18}	70.00	x_{30}	13.33
x_6	20.00	x_{19}	66.67	x_{31}	23.33
x_7	10.00	x_{20}	60.00	x_{32}	0.00
x_8	60.00	x_{21}	80.00	x_{33}	13.33
x_9	83.33	x_{22}	66.67	x_{34}	20.00
x_{10}	60.00	x_{23}	53.33	x_{35}	20.00
x_{11}	56.67	x_{24}	56.67	x_{36}	6.67
x_{12}	70.00				

In Table 1 the correspondence between GP variables and financial variables can be found.

It can be seen that the highest percentages of use correspond to real variables: the financial variables (x_8 to x_{23}), the historic amount of money spent on judicial incidences (x_{24}) and the amount of money spent on judicial incidences (x_{25}). All of them but the amount of money spent on judicial incidences (x_{25}) were used in more than half of the final trees. This prevalence of the real variables is related to the chosen typing. Five out of the seven functions in the function set take real arguments. This imposes a higher need of real variables than integer or binary ones.

The variables that are used more frequently are *leverage* (x_{14}), used in 96.67% of the final trees, *return on equity* (x_{17}), used in 86.69% of the final trees and *debt cost* (x_9), used in 83.33% of the final trees. Out of this three variables, the leverage had already been identified by the SOM as relevant. The other two variables have been included in the reduced set of variables due to the good results their inclusion yielded in [1, 2].

So a set of experiments was run where the book losses prediction problem was solved using the variables identified by the SOM as relevant plus the two previously marked variables used frequently by GP: *return on equity* and *debt cost*.

The average error rate obtained in this set of experiments was 32.79% ± 1.33, smaller than when using all the variables. Student's T-Test have been used to compare the results; when comparing the latter results with those obtained using all variables, differences were significant to level 99%.

Finally, we have run a new set of experiments using GProp for solving the classification problem, considering all the variables and the reduced set. We want to compare its results with the previous in order to assess how good they are.

The average error rate obtained using this last method was 38.75% ± 2.16 in the first case (complete set of variables), and 35.55% ± 1.24. Both of them are worse than GP results, even when it uses all the variables. Student's T-Test states that GP results are better with a confidence level of 99%.

Table 7 summarizes the error rates obtained in each set of runs.

Table 7: Average results for the prediction of book losses obtained in each set of experiments.

	Testing error
GP + All variables	34.32 ± 2.11
GP + SOM subset	**32.79 ± 1.33**
ANN + All variables	38.75 ± 2.16
ANN + SOM subset	35.55 ± 1.24

Next, Figure 2 presents an example of a GP tree obtained for the classification in order to show an example of what kind of solution we get with GP. This particular tree was chosen because it is an 'average' tree with a classification error very close to the mean. In the figure, x_5 represents the binary variable *type of company1*, x_9 is the debt cost, x_{14} is the leverage, x_{17} is the return on equity, x_{22} is the current ratio, x_{23} is the acid test, x_{32} represents the binary variable *auditor opinion1* and x_{36} represents the binary variable *linked to a group*. If $y > 0$ the company is classified as in financial distress, otherwise the company is considered healthy. This tree achieves a classification error of 32.75%.

We are going to analyze what happens in the resulting tree when a company is not attached to a group. This assumption simplifies considerably the analysis and more than 97% of the companies in the database satisfy this condition; however, it is interesting to note that if the value of that variable is true, the company would be in the *warm spot* we mentioned before, which would mean it is very likely a company that will be in financial distress.

Under this assumption we can express y_0 (see Figure 2 at the right hand side) as two nested conditional clauses:

$$\begin{aligned} y_0 \quad = \quad & \text{if } x_5 = 1 \\ & \text{then } x_{14}x_{22} \\ & \text{else if } x_{17} \leq x_{23} \\ & \qquad \text{then } {x_{23}}^2 \\ & \qquad \text{else } -20.6x_{23} \end{aligned} \tag{4}$$

And the overall output would be:

$$y = x_{14} - \text{ if } y_0 \leq x_{17} \text{ then } x_{14} \text{ else } x_{17} \tag{5}$$

Therefore,

$$\begin{aligned} & \text{if } y_0 \leq x_{17} \\ & \text{then } y = x_{14} - x_{14} = 0 \;\Rightarrow\; \text{healthy company} \\ & \text{else } y = x_{14} - x_{17} \;=\; \begin{cases} > 0 \Rightarrow \text{company with losses} \\ \leq 0 \Rightarrow \text{healthy company} \end{cases} \end{aligned} \tag{6}$$

Basically it predicts that if $y_0 \leq x_{17}$ the company is healthy. Otherwise, that is, assuming $y_0 > x_{17}$, the company will have losses if the leverage is greater than the return on equity, while if the return on equity (ROE) can compensate the leverage the company will be healthy. Given that leverage $= \frac{\text{liabilities}}{\text{equity}}$ and ROE $= \frac{\text{net income}}{\text{equity}}$, the previous statement could be rewritten as: assuming $y_0 > x_{17}$, the company will have losses if the liabilities are greater than the net income, while if the net income can compensate the liabilities the company will be healthy.

Let us analyze the inequality $y_0 \leq x_{17}$. y_0 can take 3 different values:

$$y_0 = \begin{cases} x_{14}x_{22} & : \text{if } x_5 = 1 \\ {x_{23}}^2 & : \text{if } x_5 = 0 \text{ and } x_{17} \leq x_{23} \\ -20.6x_{23} & : \text{if } x_5 = 0 \text{ and } x_{17} > x_{23} \end{cases} \tag{7}$$

If $x_5 = 0$ and $x_{17} \leq x_{23}$ then y_0 takes the value ${x_{23}}^2$ which yields to $y_0 \leq x_{17}$ being false. That is, for a company type 2 or 3, the prediction is that if the acid test value is greater than the return on equity, the company will have losses if the liabilities are greater than the net income, otherwise the company will be healthy.

On the other hand if $x_5 = 0$ and $x_{17} > x_{23}$ then y_0 takes the value $-20.6x_{23}$ which yields to $y_0 \leq x_{17}$ being true if $x_{23} > 0$ (which happens in around 98% of the companies in the database). That means that for a company of type 2 or 3 if the acid test value is positive and smaller than the return on equity the company is healthy.

The last possible case is that the company is of type 1. Then y_0 takes the value $x_{14}x_{22}$. Again, leverage $= \frac{\text{liabilities}}{\text{equity}}$ and ROE $= \frac{\text{net income}}{\text{equity}}$, the condition could be expressed as: if the multiplication of the liabilities and the current ratio is smaller than the net income the company will be healthy otherwise the prediction depends on whether the liabilities are greater than the net income or not. Regarding the condition that the multiplication of the liabilities and the current ratio must be smaller than the net income for a type 1 company to be healthy, it penalizes high current ratios that may indicate that the firm has too many assets tied up in current assets and is not making efficient use to them.

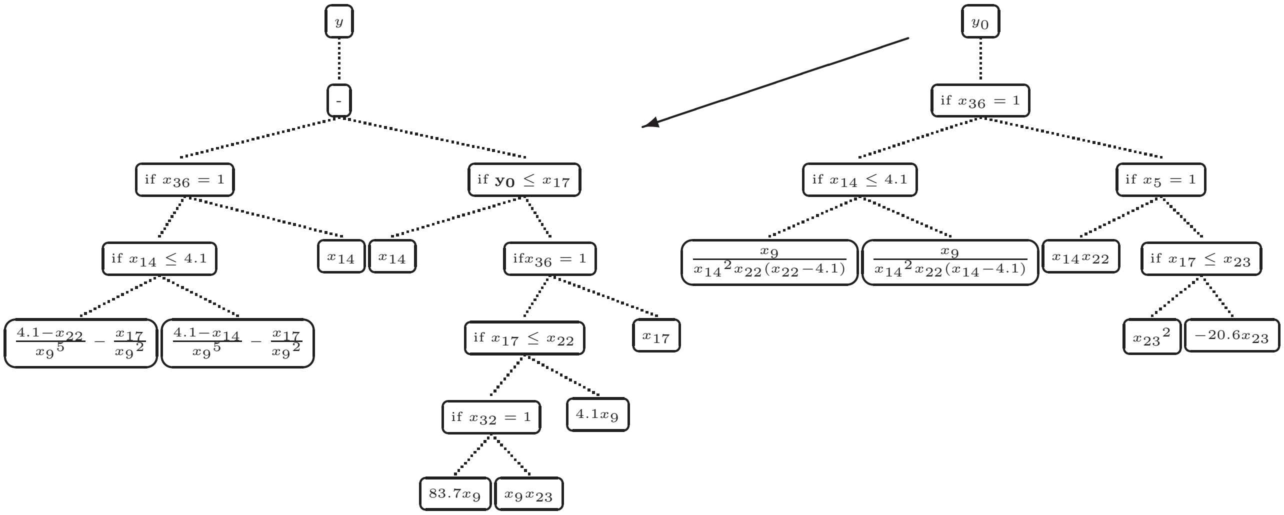

Figure 2: Example of resulting GP tree. The y_0 symbol in the left hand side tree (mark in boldface) has been expanded on the right hand side.

Thus, in general the prediction of financial distress for companies not linked to a group in this GP tree is based on the comparison of two values: liabilities and net income. If the liabilities are greater than the net income the company will suffer losses, otherwise the company will not. If the companies in the testing set are classified according to this rule the classification error is 34.75%. In addition the GP has detected two groups that do not follow the rule, improving the classification error to 32.75%. Therefore, if a company is of type 1 and the multiplication of the liabilities and the current ratio is smaller than the net income or if a company is of type 2 or 3 and its acid test value is positive and smaller than the return on equity, the company is predicted as healthy regardless of the value that takes the substraction liabilities - net income.

5. CONCLUSIONS AND FUTURE WORK

In this study we compare two soft-computing methods for prediction of financial distress (book losses) that have proved their adequacy to this prediction problem in previous studies: Genetic Programming (GP) and Evolutionary Artificial Neural Networks (ANNs). Both classifiers work on a reduced set of variables, which has been found using two methods: the analysis of final GP trees and the Kohonen's Self-Organizing Maps. As a result, we have got a set of variables that are significant for the book losses prediction problem, and which make sense from an economic point of view. This reduced set of variables makes it easier to explain the decision reached by any method by analyzing which values yield a positive or negative outcome.

Unexpectedly, the classification rates achieved by GP improve the results obtained with the ANN approach named GProp (which is a method for evolving multilayer perceptrons). In addition, the ANN model acts as a black box and its predictions are difficult to explain (typical disadvantage in this soft-computing method). On the other hand, the GP approach presents the advantage of producing the rules that an analyst could use to predict and to explain the book losses. In any case, even as error rate might seem high for a pattern recognition method, it is quite usual to obtain error rates close to 50% in economic prediction methods. These predictions are used as a tool that aids the decision making process of a human operator, not as an oracle that yields a definitive decision.

In this case, part of the reason why the GProp method is at a disadvantage might be due to the different way that fitness is evaluated in them. Possible, due to the fact that failed/successful companies are dealt with differently in the GP case (by giving a higher value to predicting correctly the successful companies) accounts for that two point difference, which means that its success in this case is more due to the lack of balance of the existing database than to any intrinsic advantage.

That is why, as a future line of work, it would be interesting to use a multiobjective approach to solving the problem, by dealing separately with type I (false positive) and type II (false negative) errors, as has been attempted before [9]. We will also try to improve the classification accuracy by fine-tuning parameters in both methods, or even combining them in ensembles.

The analysis of GP trees for feature extraction is also an interesting field that might be worth exploring, so in the future we will try to establish the conditions that make a variable significant in the final result obtained by the GP-synthesized function.

Acknowledgements

This work has been supported by project TIN2007-68083-C02 (Ministry of Education and Culture, NOHNES Project).

6. REFERENCES

[1] E. Alfaro-Cid, A. Cuesta-Cañada, K. Sharman, and A. I. Esparcia-Alcázar. *Natural Computing in Computational Economics and Finance*, chapter Strong Typing, Variable Reduction and Bloat Control for Solving the Bankruptcy Prediction Problem Using Genetic Programming. Springer, 2008.

[2] E. Alfaro-Cid, A. Mora, J. Merelo, K. Sharman, and A. Esparcia-Alcázar. A SOM and GP tool for reducing the dimensionality of a financial distress prediction problem. *LNCS*, 2008. Accepted for publication.

[3] E. Alfaro-Cid, K. Sharman, and A. Esparcia-Alcázar. A genetic programming approach for bankruptcy prediction using a highly unbalanced database. *LNCS*, 4448:169–178, 2007.

[4] E. Altman. The success of business failure prediction models. An international survey. *J. of Banking, Acc. and Finance*, 8:171–198, 1984.

[5] W. Beaver. Financial ratios as predictors of failures. Empirical research in accounting: Selected studies. *J. of Acc. Research*, 5:71–111, 1966.

[6] C. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press. Oxford University Press Inc., 1996.

[7] A. Brabazon and M. O'Neill. *Biologically inspired algorithms for finantial modelling*. Springer, 2006.

[8] A. Cangelosi, D. Parisi, and S. Nolfi. Cell Division and Migration in a Genotype for Neural Networks. *Network: Computation in Neural Systems*, 5:497–515, 1994.

[9] P. Castillo, M. Arenas, J. Merelo, V. Rivas, and G. Romero. Multiobjective optimization of ensembles of multilayer perceptrons for pattern classification. In *Proceedings PPSN IX*, pages 453–462, 2006.

[10] P. Castillo, J. Merelo, G. Romero, A. Prieto, and I. Rojas. Statistical Analysis of the Parameters of a Neuro-Genetic Algorithm. *IEEE Trans. on Neural Networks*, 13(6):1374–1394, 2002.

[11] P. A. Castillo, J. M. D. la Torre, J. J. Merelo, and I. Román. Forecasting business failure. A comparison of neural networks and logistic regression for spanish companies. In *Proc. of the 24th Eur. Acc. Assoc.*, Athens, Greece, 2001.

[12] P. A. Castillo, J. J. Merelo, V. Rivas, G. Romero, and A. Prieto. G-Prop: Global Optimization of Multilayer Perceptrons using GAs. *Neurocomputing*, 35(1–4):149–163, 2000.

[13] I. de Falco, A. Iazzetta, P. Natale, and E. Tarantino. Evolutionary neural networks for nonlinear dynamics modeling. *LNCS*, 1498:593–602, 1998.

[14] P. Durr, C. Mattiussi, and D. Floreano. Neuroevolution with Analog Genetic Encoding. *LNCS*, 4193:671–680, 2006.

[15] F. Fernández de Vega, M. Rubio del Solar, and A. Fernández Martínez. Implementación de algoritmos evolutivos para un entorno de distribución epidémica. In *Actas del MAEB'05*, pages 57–62, Granada, Spain, 2005.

[16] N. Japkowicz and S. Stephen. The class imbalance problem: a systematic study. *Int. Data Analysis*, 6(5):429–449, 2002.

[17] S. Kaski, J. Sinkkonen, and J. Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Trans. Neural Networks*, 12(4):936ss, 2001.

[18] K. Kiviluoto. Predicting bankruptcies with the self-organizing map. *Neurocomputing*, 21(1-3):191–201, 1998.

[19] T. Kohonen. *The Self-Organizing Maps*. Springer, 2001.

[20] T. Lensberg, A. Eilifsen, and T. E. McKee. Bankruptcy theory development and classification via genetic programming. *Eur. J. of Op. Research*, 169:677–697, 2006.

[21] F. Leung, H. Lam, S. Ling, and P. Tam. Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Trans. on Neural Networks*, 14(1):79–88, 2003.

[22] R. Lippmann. An introduction to computing with neural nets. *IEEE ASSP Magazine*, 3(4):4–22, 1987.

[23] H. Mayer, R. Schwaiget, and R. Huber. Evolving topologies of artificial neural networks adapted to image processing tasks. In *Proc. of 26th Int. Symp. on Remote Sensing of Environment*, pages 71–74, Vancouver, BC, Canada, 1996.

[24] J. J. Merelo, M. G. Arenas, J. Carpio, P. A. Castillo, V. M. Rivas, G. Romero, and M. Schoenauer. Evolving objects. In *Proc. of FEA'2000 & JCIS'2000*, pages 1083–1086, Atlantic City, NJ, 2000.

[25] J. J. Merelo, M. Patón, A. Cañas, A. Prieto, and F. Morán. Optimization of a competitive learning neural network by genetic algorithms. *LNCS*, 686:185–192, 1993.

[26] D. J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3(2):199–230, 1995.

[27] A. M. Mora, J. L. J. Laredo, P. A. Castillo, and J. J. Merelo. Predicting financial distress: A case study using self-organizing maps. In F.Sandoval and et al., editors, *Proc. of the 9th International Work Conference on Artificial Neural Networks (IWANN 2007)*, volume 4507 of *LNCS*, pages 765–772, San Sebastian, Spain, June 2007.

[28] D. Moriarty and R. Miikkulainen. Hierarchical evolution of neural networks. In *Proc. of the ICEC'98*, pages 428–433, Anchorage, AK, 1998.

[29] I. Román, M. E. Gómez, J. M. D. la Torre, J. J. Merelo, and A. M. Mora. Predicting financial distress: Relationship between continued losses and legal bankrupcy. In *Proc. of the 27th Annual Congress Eur. Acc. Assoc.*, Dublin, Ireland, 2006.

[30] S. Salcedo-Sanz, J. L. Fernández-Villacañas, M. J. Segovia-Vargas, and C. Bousoño-Calzón. Genetic programming for the prediction of insolvency in non-life insurance companies. *Computers and Op. Research*, 32:749–765, 2005.

[31] D. Thierens, J. Suykens, J. Vandewalle, and B. D. Moor. Genetic weight optimization of a feedforward neural network controller. In *Proc. of the Conf. on Artificial Neural Nets and Genetic Algorithms*, pages 658–663, 1993.

[32] S. Ultsch. Kohonen's self-organizing maps for exploratory data analysis. In *Proc. of the INNC'90*, pages 305–308, 2000.

[33] D. Whitley. The GENITOR algorithm and selection presure: Why rank-based allocation of reproductive trials is best. In *Proc. of the 3th Int. Conf. on Genetic Algorithms*, pages 116–121, 1989.

[34] X. Yao. Evolving artificial neural networks. *Proc. of the IEEE*, 87(9):1423–1447, 1999.