Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Operations Research 32 (2005) 749–765
computers & operations research
www.elsevier.com/locate/dsw

Genetic programming for the prediction of insolvency in non-life insurance companies

Sancho Salcedo-Sanz[a,*], José-Luis Fernández-Villacañas[a], María Jesús Segovia-Vargas[b], Carlos Bousoño-Calzón[a]

[a] *Department of Signal Theory and Communications, Universidad Carlos III de Madrid 28911, Spain*
[b] *Department of Financial Economy and Accounting I, Universidad Complutense de Madrid, Spain*

Abstract

Prediction of non-life insurance companies insolvency has arised as an important problem in the field of financial research, due to the necessity of protecting the general public whilst minimizing the costs associated to this problem, such as the effects on state insurance guaranty funds or the responsibilities for management and auditors. Most methods applied in the past to predict business failure in non-life insurance companies are traditional statistical techniques, which use financial ratios as explicative variables. However, these variables do not usually satisfy statistical assumptions, what complicates the application of the mentioned methods. Emergent statistical learning methods like neural networks or SVMs provide a successful approach in terms of error rate, but their character of black-box methods make the obtained results difficult to be interpreted and discussed. In this paper, we propose an approach to predict insolvency of non-life insurance companies based on the application of genetic programming (GP). GP is a class of evolutionary algorithms, which operates by codifying the solution of the problem as a population of LISP trees. This type of algorithm provides a diagnosis output in the form of a decision tree with given functions and data. We can treat it like a computer program which returns an answer depending on the input, and, more importantly, the tree can potentially be inspected, interpreted and re-used for different data sets. We have compared the performance of GP with other classifiers approaches, a Support Vector Machine and a Rough Set algorithm. The final purpose is to create an automatic diagnostic system for analysing non-insurance firms using their financial ratios as explicative variables.

Keywords: Genetic programming; Insolvency; Non-life insurance companies; Support vector machines; Rough set

* Corresponding author. Tel.: +34-91-624-5973; fax: +34-91-624-8749.
E-mail address: sancho@tsc.uc3m.es (S. Salcedo-Sanz).

0305-0548/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2003.08.015

1. Introduction

Many financial decisions involve the classification of a set of observations (firms, stocks) into one or several groups of categories, what leads many researches to apply operational research methods to management problems. There exists an extensive literature devoted to the study of classification problems in the financial field, i.e. the work by Chen and Yeh [1] dealing with financial management or the works by Dimitras et al. [2] and Tam [3] in business failure. In this paper, we focus on the latter problem. Unlike other financial problems, there are a great number of agents (auditors, management, analysts, creditors and government) facing business failure, so research in this topic has been of growing interest in the last decades (see [4,5]). Following [6], the number of bankrupt firms and the relative trend indicate the robustness for the economy of a country in a worldwide scale, and the substantial costs associated to insolvency can become a national political issue. It has long been recognized that there needs to be some form of supervision of such entities to attempt to minimize the risk of failure. In fact, insolvency of non-life insurance companies have been a concern of parties such as insurance regulators, investors, management, financial analysts, banks, auditors, policy holders and consumers. This concern has arised from the necessity of protecting the general public against the consequences of insurers insolvencies, as well as minimizing its effects on state insurance guarantee funds or the responsibilities for management and auditors. Nowadays, *Solvency II* project is intended to lead to the reform of the existing solvency rules in European Union. Therefore, developing new methods to tackle prudential supervision in insurance companies is a highly topical question.

A large number of methods have been proposed to predict business failure (see [7–9]); however, the special characteristics of the insurance sector have made most of them unfeasible, and just a few have been applied to this sector. Most approaches to prediction of failure in non-life insurance companies are statistical methods, such as discriminant or logic analysis, which use financial ratios as explicative variables. However, this kind of variables does not usually satisfy any statistical assumptions, what introduces an extra difficulty for the use of statistical methods in insurance business failure prediction. Alternatively, statistical learning approaches such as neural networks [3] or SVM [10] has been successfully applied to this kind of problems, achieving results in terms of classification rate (less error in the classification procedure) than traditional statistical approaches. However, their black-box character make them difficult to interpret, and hence the obtained results cannot be clearly analysed and related to the economical variables for discussion.

In this paper, we propose an approach to predict insolvency of non-life insurance companies based on genetic programming (GP) techniques. The final purpose is to create an automatic diagnostic system for analysing non-insurance firms using their financial ratios as explicative variables. GP is a special kind of evolutionary algorithm [11], which operates directly on the representation of the problem, by codifying it as a population of LISP trees. The tree is itself a program with functions and terminals. After generating the initial population, GP performs a genetic-like cycle (see [12]) of fitness evaluation, selection of parents and reproductions by means of the application of genetic operators to produce successive generation of trees. GP has many configurations in tree generation, selection and variation. In particular, crossover is performed by randomly choosing nodes form each parent and swapping them; while mutation consists in reproducing a node with a randomly grown sub-tree.

There is a clear benefit in using GP for data classification: This type of algorithm provides a diagnosis output in the form of a decision tree with given functions and data. We can treat it like a computer program which returns an answer depending on the input, and, more importantly, the tree can potentially be inspected, interpreted and re-used for different data sets.

The rest of the paper is structured as follows: In Section 2 the main features of insolvency prediction in non-life insurance companies, introduced as a particular case of the *multiattribute classification problem*, are given. In Section 3 a brief introduction to genetic programming is provided, whereas Section 4 describes the main characteristics of the GP used. Section 5 includes the analysis of the test data used, the experiments performed in order to test the proposed algorithm and the results obtained. The discussion of these results, and its comparison with a Support Vector machine and a Rough Set approaches are also provided in this section. Finally, Section 6 closes the paper with some concluding remarks.

2. Problem definition

In this paper, we tackle the prediction of non-life insurance firms failure, which can be considered a particular example of the so called *multiattribute classification problem*. This problem consists in the assignment of an *object*, described by values of *attributes*, to a predefined class of category.

Mathematically a multiattribute classification problem can be stated as follows:

Let $\{\mathbf{x}_i\}$, $\mathbf{x}_i \in \mathbb{R}^n$, $i \in \{1, \ldots, l\}$ a set of observations (objects) drawn from some unknown probability distribution $P(\mathbf{x}, y)$, and $\{y_i\} \in \{-1, 1\}$ (categories), a set of associated true labels.

A classification machine is defined by a set of possible mappings $\mathbf{x} \mapsto f(\mathbf{x}, \alpha)$, where a particular choice of parameters α generates what is called a "trained machine". As example, in a general neural network with fixed architecture, α corresponds to the weights and biases of the neural network, in the case of GP α are the number of nodes, branches and functions in nodes of the final tree.

The expectation of the test error for a trained machine can be defined as

$$R(\alpha) = \left(\int V(y, f(\mathbf{x}, \alpha)) \, \mathrm{d}P(\mathbf{x}, y) \right), \tag{1}$$

where $V(\cdot, \cdot)$ is a loss functional, $P(\mathbf{x}, y)$ is the unknown probability function the data was sampled from and the function $y = f(\mathbf{x}, \alpha)$ is the classification engine.

For the multiattribute classification problem the loss functional can be defined as

$$V(y, f(\mathbf{x}, \alpha)) = \tfrac{1}{2} |y - f(\mathbf{x}, \alpha)| \tag{2}$$

and the expected test error for a trained machine yields:

$$R(\alpha) = \left(\int \frac{1}{2} |y - f(\mathbf{x}, \alpha)| \, \mathrm{d}P(\mathbf{x}, y) \right). \tag{3}$$

An *empirical* test error can be defined as

$$R_{\text{emp}}(\alpha) = \frac{1}{2l} \sum_{i=1}^{l} |y_i - f(\mathbf{x}_i, \alpha)|. \tag{4}$$

Note that $R_{\text{emp}}(\alpha)$ is a fixed number for a particular choice of α and for a particular training set $\{\mathbf{x}_i, y_i\}$, since no probability distribution is involved in this calculation.

In the general case, the multiattribute classification problem consists in finding the machine which learn the mapping $\mathbf{x}_i \mapsto y_i$ with the highest generalization ability possible. According to the statistical learning theory, the generalization error of a learning machine can be analysed considering the machine's capacity and its empirical risk [13]. The capacity factor represents the machine's complexity, whereas the empirical risk factor measures its quality. To ensure high generalization ability, the tradeoff between these two factors should be addressed. In this paper, the complexity of the GP model is considered by controlling the *depth* of the developed trees, whereas the empirical risk is considered by minimizing empirical test error $R_{\text{emp}}(\alpha)$ defined above.

2.1. Prediction of business failure

The multiattribute classification problem is applicable in a straight forward manner to the business failure prediction problem, with $\{\mathbf{x}_i\}$, $i=1,\ldots,l$, a set of firms described by a set of n financial ratios (every component x_{ij}), and $y_i \in \{-1, 1\}$ a label which describes the state of the firm as "healthy" $y_i = 1$ or failed $y_i = -1$.

Thus, the complete problem this paper faces consists in characterizing a firm as healthy or failed, providing an automatic diagnostic system for detecting failed non-life insurance firms in the future.

3. Brief overview of genetic programming

Genetic programming can be considered an extension of Genetic Algorithms [12], in which the genetic population consists of computer programs, i.e. compositions of primitive functions and terminals (we consider the reader familiarized with the main concepts of Genetic Algorithms, see [14,12] for reference). Specifically, GP is able to evolve a computer program for solving, or approximately solving, a large variety of classification problems, from a wide variety of fields.

GP starts with a set of randomly generated computer programs composed of available programmatic ingredients and genetically evolves the population using the principle of survival of the fittest and an analogue of naturally occurring genetic crossover operation. Thus, GP provides a way to search the space of possible computer programs to find a program which solves, or approximately solves, a problem [11]. The following description of a GP procedure can be found in [15]:

1. Generate an initial population of random computer programs composed of the primitive functions and terminals of the problem.
2. Iteratively perform the following substeps until the termination criterion has been satisfied:
   2.1. Execute each program in the population and assign it a fitness value according to how well it solves the problem.

2.2. Create a new population of programs by applying the following three operations:
  2.2.1. *Reproduction*: Copy an existing program to the new population.
  2.2.2. *Crossover*: Create two new offspring programs for the new population by genetically recombining randomly chosen parts of two existing programs. This operator operates on two parental computer programs producing two offspring programs using parts of each parent.
  2.2.3. *Mutation*: Cut a branch of the tree off and substitute it by a randomly created one.
3. The single best computer program in the population produced during the run is considered the solution to the problem and evaluated on a test set.

Thus, there are five preparatory steps for applying GP to a given problem:

- determining the set of terminals;
- determining the set of functions;
- determining the fitness function to be used;
- parameters and variables for controlling the run; and
- criterion for designating a result and terminating the run.

4. Genetic programming for solving the business failure prediction problem

The main task of GP in the business failure prediction problem is to evolve a decision tree capable of discriminating between healthy and failed firms, based on firm's financial ratios. In order to perform this study, we consider different types of terminals, set of functions and fitness functions:

The representation of a solution for the problem provided by the GP algorithm is a decision tree. Each node of this tree is a function node taking one of the values from the set {+, −, OR-TH, AND-TH, NOT-TH, IFLTE}: "+" adds up two numbers, "−" subtracts them, "OR-TH" returns 1 when at least one of the two inputs are positive and −1 when none of them is positive, "AND-TH" returns 1 when both inputs are positive and −1 when one of both of them are negative, "NOT-TH" returns 1 when its single input is negative and −1 when positive. Finally, "IFLTE" is a macro that has 4 entries, two input parameters (e.g. a and b) and two conditional nodes (e.g. c and d); when "a" is larger or equal than "b" the macro evaluates "c" (it could be a function leading to a sub-tree for example) otherwise it evaluates to "d".

A tree which returns a positive (or zero) real value for a training or test case previously marked as "healthy firm", is said to have correctly classified that case, and is labelled as a correct positive. Alternatively, when the tree returns a negative real value for the same case, it is taken as a false negative. Trees that return positive (or zero) real values for cases marked as "failed firms" are known as false positives and correct negatives when the return value is a negative real value.

The fitness function is evaluated over a set of training or test cases. It is parameterized by the number of correct positives CP, false positives FP, correct negatives CN, false negatives FN, the total number of positives N_{pos} and the total number of negatives N_{neg}. Several fitness functions were investigated.

The first one seeks to maximize two values known as *precision* and *recall* of the failed firms. This fitness is given by

$$f_1 = 1 - (\alpha_1 \cdot f^{\mathrm{N}}_{\mathrm{rec}} + \beta_1 f^{\mathrm{N}}_{\mathrm{prec}}), \tag{5}$$

where $f^{\mathrm{N}}_{\mathrm{rec}} = CN/N_{\mathrm{neg}}$ and $f^{\mathrm{N}}_{\mathrm{prec}} = CN/(CN + FN)$, and α_1 and β_1 are parameters in the range $[0, 1]$ which allow us to shift the emphasis of the selection on precision or recall.

This fitness function has the drawback that only manages precision and recall values of failed firms. If a balance between failed and healthy firms parameters is required, the following function should achieve better results in terms of balance between precision and recall:

$$f'_1 = \frac{(f^{\mathrm{N}}_1 + f^{\mathrm{P}}_1)}{2}, \tag{6}$$

where

$$f^{\mathrm{N}}_1 = 1 - (\alpha_1 \cdot f^{\mathrm{N}}_{\mathrm{rec}} + \beta_1 f^{\mathrm{N}}_{\mathrm{prec}}), \tag{7}$$

$$f^{\mathrm{P}}_1 = 1 - (\alpha_1 \cdot f^{\mathrm{P}}_{\mathrm{rec}} + \beta_1 f^{\mathrm{P}}_{\mathrm{prec}}) \tag{8}$$

with $f^{\mathrm{P}}_{\mathrm{rec}} = CP/N_{\mathrm{pos}}$ and $f^{\mathrm{P}}_{\mathrm{prec}} = CP/(CP + FP)$.

Note that f^{N}_1 is exactly the function f_1 (we have renamed it as f^{N}_1 here in order to maintain the notation), and $f^{\mathrm{N}}_{\mathrm{rec}}$, $f^{\mathrm{N}}_{\mathrm{prec}}$, $f^{\mathrm{P}}_{\mathrm{rec}}$ and $f^{\mathrm{P}}_{\mathrm{prec}}$ stand for recall and precision values of failed and healthy firms, respectively. Note also that parameters α_1 and β_1 have been consider to be equal for both functions. Function f'_1 can be written in the following form:

$$f'_1 = 1 - \frac{\alpha_1}{2}\left(\frac{CN}{N_{\mathrm{neg}}} + \frac{CP}{N_{\mathrm{pos}}}\right) - \frac{\beta_1}{2}\left(\frac{CN}{CN + FN} + \frac{CP}{CP + FP}\right). \tag{9}$$

The second fitness function we analyse is designed to maximize the number of correct positives (CP) and correct negatives (CN):

$$f_2 = 1 - \frac{CP + CN}{N_{\mathrm{pos}} + N_{\mathrm{neg}}}. \tag{10}$$

It is expected that this function also provides a balance between precision and recall.

Note that all fitness functions considered lie in the range $[0, 1]$ with 0 being the best possible fitness, 1 the worst. The aim is, therefore, to minimize the fitness functions above.

Another important characteristic of the GP is the tree depth considered. In this investigation we test the performance of trees with two different depths, trees of depth 2 and trees of depth 4.

The GP algorithm used in this work was Simple Genetic Programming in C (SGPC) [16].

4.1. A note on precision and recall

Precision and recall are two important parameters for measuring the behaviour of a classification system. Recall, defined as CN/N_{neg} for failed firms and as CP/N_{pos} if one want to extract healthy firms, measures the ability of our classifier for extracting all cases we are interested in, whereas

precision, ($CN/(CN+FN)$ for analysing failed firms and $CP/(CP+FP)$ for healthy firms) measures the quality of the extracted cases.

Let us consider the problem of classifying insurance firms in healthy or failed, where we are interested in detecting firms which will fail. A high decision value of recall means that our tree is able to extract the majority of the firms which potentially will fail, although some healthy ones can be classified by the tree as failed. On the other hand, a high value of precision means that the majority of the firms marked as failed by the tree are truly failed, although it is possible that the decision tree has not detected all failed firms.

Depending on the application, a higher value of recall or precision may be required. However, in the general case, a balance between precision and recall parameters should be provided, in order to avoid capital losses due to investments in firms that will go bankrupt and, at the same time, minimize the rejection of profitable investments in healthy firms [17].

5. Experiments and results

5.1. Test data and input variables

In this section, we show the main characteristics of the data and variables that will be used to test our algorithm. We have used the sample of firms employed by Sanchis [18]. This data sample consists of Spanish non-life insurance firms data 5 years prior to failure. The firms were in operation or went bankrupt between 1983 and 1994. In each period, 72 firms (36 failed and 36 non-failed) are selected. As a control measure, a failed firm is matched with a non-failed one in terms of size (premiums volume). In addition, each firm is described by 21 financial ratios that have come from a detailed analysis of the variables and previous bankruptcy studies for non-life insurance. Table 1 shows the 21 ratios which describe the firms. Ratios 15 and 16 have been removed in our study, due to most of the firms not having "other income"; this reduces the total number of ratios to 19. Note that the special financial characteristics of insurance companies require general financial ratios as well as those that are specially proposed for evaluating insolvency of insurance sector. The ratios have been calculated from the last financial statements (balance sheets and income statements) issued 1 year before the firms declared bankruptcy. Thus, it has to be noted that the prediction of the insolvency achieved by our method will be 1 year in advance.

In order to test the predictive accuracy of the GP, it is necessary to split the set of original data to form a training set, and a holdout sample to validate the obtained model, i.e. the test set. It is a well-known fact that overfitting of the training data set can occur, leading to a poor performance of the generated tree on the test set. In order to avoid overfitting, a possible solution is to generate multiple subsets from the original firms. The set of 72 firms are split in 10 different, randomly generated training and testing sets (hereafter *k-folds*), every set consisting of 50 firms for training (25 failed and 25 non-failed) and 22 firms for testing (11 failed and 11 non-failed). The results will be averaged for the 10 different k-folds.

5.2. Results

We have tested our algorithm for the data introduced above. First, we study the dependence of our results on the algorithm's free parameters: model for tree generation, model for selection, population

Table 1
Definition of the ratios

Ratio	Definition
R1	$\frac{\text{Working Capital}}{\text{Total Assets}}$
R2	$\frac{\text{Earnings Before Taxes (EBT)}}{(\text{Capital} + \text{Reserves})}$
R3	$\frac{\text{Investment Income}}{\text{Investments}}$
R4	$\frac{\text{EBT} + \text{Reserves for Depreciation} + (\text{Extraordinary Income} - \text{Extraordinary charges})}{\text{Total Liabilities}}$
R5	$\frac{\text{Earned Premiums}}{(\text{Capital} + \text{Reserves})}$
R6	$\frac{\text{Earned Premiums net of Reinsurance}}{(\text{Capital} + \text{Reserves})}$
R7	$\frac{\text{Earned Premiums}}{(\text{Capital} + \text{Reserves} + \text{Technical Provisions})}$
R8	$\frac{\text{Earned premiums Net of Reinsurance}}{(\text{Capital} + \text{Reserves} + \text{Technical Provisions})}$
R9	$\frac{(\text{Capital} + \text{Reserves})}{\text{Total Liabilities}}$
R10	$\frac{\text{Technical Provisions}}{(\text{Capital} + \text{Reserves})}$
R11	$\frac{\text{Claims Incurred}}{(\text{Capital} + \text{Reserves})}$
R12	$\frac{\text{Claims Incurred Net of Reinsurance}}{(\text{Capital} + \text{Reserves})}$
R13	$\frac{\text{Claims Incurred}}{(\text{Capital} + \text{Reserves} + \text{Technical Provisions})}$
R14	$\frac{\text{Claims Incurred Net of Reinsurance}}{(\text{Capital} + \text{Reserves} + \text{Technical Provisions})}$
R15	$\frac{\text{Claims Incurred}}{\text{Earned Premiums}} + \frac{\text{Other Charges and Commissions}}{\text{Other Income}}$
R16	$\frac{\text{Claims Incurred Net of Reinsurance}}{\text{Earned Premiums Net of Reinsurance}} + \frac{\text{Other Charges and Commissions}}{\text{Other income}}$
R17	$\frac{\text{Claims Incurred} + \text{Other Charges and Commissions}}{\text{Earned Premiums}}$
R18	$\frac{\text{Claims Incurred Net of Reinsurance} + \text{Other Charges and Commissions}}{\text{Earned Premiums Net of Reinsurance}}$
R19	$\frac{\text{Technical provisions of Assigned reinsurance}}{\text{Technical Provisions}}$
R20	$\frac{\text{Claims Incurred}}{\text{Earned Premiums}}$
R21	$\frac{\text{Claims Incurred Net of Reinsurance}}{\text{Earned Premiums net of Reinsurance}}$

Table 2
Classification accuracy in per cent of correctly classified firms using Rough Set

Experiment	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)	6 (%)	7 (%)	8 (%)	9 (%)	10 (%)
Failed	63.6	81.8	72.7	90.9	90.9	72.7	72.7	81.8	54.5	90.9
Healthy	81.8	54.5	63.6	54.5	63.6	63.6	63.6	45.4	72.7	63.6
Total	72.7	68.2	68.2	72.7	77.3	68.2	68.2	63.6	63.6	77.3

size, number of generations, maximum tree depth after variation and probabilities of crossover and mutation. The behaviour we found while varying the different SGPS parameters showed little variance as to the model of initial tree generation (ramped growth was chosen as balance between random creation at each depth and full growth). Population size was set to 100 trees (higher populations did not show better results, comparing training and test sets). The number of generations was chosen as 1000 in order to compare improvements in fitness in train and data sets. Mutation and crossover probabilities were set to 0.2 and 0.6, respectively (higher rates showed no improvement in mean fitness while lower rates made the population converge in fitness well before the 1000 generation limit). Selection was performed through a tournament of size 2 and the depth of the trees was fixed to 2, but in an example of function f_2 which was 4 (Table 2).

We have run the three algorithms for the 10 k-folds in the three defined functions. In function f_1 we have used as values for α_1 and β_1 0.8 and 0.2, respectively. In function f_1' we have maintained $\alpha_1 = 0.8$ and $\beta_1 = 0.2$, using Eq. (9) in order to calculate its value. Function f_2 maximizes the number of CP and CN, its definition can be found in Section 4. Two variants of this function can be defined, noting that outputs of the decision tree next to 0 can be misclassified. Thus, f_2 *sharp* does not take this fact into account, whereas f_2 *soft* consider a certain margin around the "0" border to classify a sample as positive or negative. For fitness function f_2 soft, the GP was executed for trees with initial maximum depth 2 and 4. This will allow us to compare performances based on depth.

In addition to these fitness functions, we also analyse the behaviour of the GP after applying a feature selection procedure. In [10] the same data set has been analysed, applying a feature selection process performed by means of a Genetic Algorithm. It has been shown that good results can be obtained with only 3 ratios (R1, R9, R13), discarding the other 16. In this paper, we will use these three ratios in order to compare the results obtained by the GP with it against the results obtained with 19 ratios.

We compare the results obtained by our algorithm in all functions considered, 19 and 3 ratios, with the results achieved by using a *Support Vector Machine* (SVM) [19]. SVM is a classification scheme which has gained prominence in the past couple of years. For linearly separable data, SVM is known to provide the classification frontier with maximum margin between classes. In non-linearly separable data, SVM formulation involves *kernel functions* in order to perform the classification. Thus, the solution provided by the SVM is again of maximum margin in a higher dimensional (possibly infinite) space. For a detailed description of the SVM schema see [19,20]. The best results obtained by the SVM were achieved using a polynomic kernel of order 4. In order to compare results we have used the same sets of training and test firms for the three algorithms, and the results were averaged for the 10 different k-folds as mentioned above.

Table 3
Results for data with 19 ratios

	GP	SVM
f_1	0.14(0.04)	0.40(0.16)
f_1 recall	0.95(0.07)	0.60(0.18)
f_1 prec.	0.53(0.06)	0.54(0.09)
f_2 sharp	0.30(0.08)	0.45(0.1)
f_2 sharp recall	0.66(0.27)	0.60(0.18)
f_2 sharp prec.	0.53(0.11)	0.54(0.09)
f_2 soft depth 2	0.41(0.05)	—
f_2 soft depth 2 recall	0.97(0.27)	—
f_2 soft depth 2 prec.	0.54(0.07)	—
f_2 soft depth 4	0.40(0.04)	—
f_2 soft depth 4 recall	0.90(0.25)	—
f_2 soft depth 4 prec.	0.52(0.08)	—

Table 4
Results for data with 3 ratios

	GP	SVM
f_1	0.15(0.04)	0.44(0.15)
f_1 recall	0.93(0.07)	0.55(0.15)
f_1 prec.	0.55(0.06)	0.58(0.14)
f_2 sharp	0.30(0.08)	0.42(0.14)
f_2 sharp recall	0.64(0.28)	0.55(0.15)
f_2 sharp prec.	0.63(0.19)	0.58(0.14)
f_2 soft depth 2	0.34(0.05)	—
f_2 soft depth 2 recall	0.61(0.27)	—
f_2 soft depth 2 prec.	0.64(0.19)	—
f_2 soft depth 4	0.36(0.08)	—
f_2 soft depth 4 recall	0.84(0.30)	—
f_2 soft depth 4 prec.	0.59(0.18)	—

Tables 3 and 4 show the results obtained by the GP and SVM algorithms for 19 and 3 ratios, respectively. Both tables show the values of the different fitness functions considered, as well as the precision and recall values obtained with every fitness function. Tables 5 and 6 show the results obtained using fitness function f_1'. In the case of the GP, the fitnesses have been calculated for the best performing tree of the training set, evaluated on the test set. These values are the mean of 10 independent runs for each of the 10 different k-folds. In the case of the SVM schema, we have calculated the fitnesses values evaluating the SVM performance in the test set, once the machine has been trained in the training set. The values displayed in the tables are the average of the results obtained by the SVM in the 10 k-folds. Standard deviation for the k-folds (between parenthesis in tables) was chosen as the measure of our error. It is easy to see that the fitness values obtained by

Table 5
Results for data with 19 ratios f_1'

	GP	SVM
f_1'	0.31(0.05)	0.47(0.15)
f_1^N recall	0.93(0.06)	0.60(0.18)
f_1^N prec.	0.64(0.03)	0.54(0.09)
f_1^P recall	0.19(0.1)	0.50(0.08)
f_1^P prec.	0.70(0.3)	0.58(0.11)

Table 6
Results for data with 3 ratios f_1'

	GP	SVM
f_1'	0.38(0.05)	0.43(0.15)
f_1^N recall	0.66(0.1)	0.55(0.15)
f_1^N prec.	0.61(0.06)	0.58(0.14)
f_1^P recall	0.54(0.13)	0.59(0.18)
f_1^P prec.	0.62(0.1)	0.56(0.15)

the SVM are poorer than the GP results. Since the GP can be tuning for minimizing a given fitness function, and the SVM cannot, these results are somehow expected. However, these results indicate that the GP approach is a valuable tool for classifying a set of data in such a way that a given fitness measure to be minimized (*recall* measure in this case).

Note the different precision and recall values obtained by the GP algorithm using functions f_1 and f_1'. Using fitness function f_1, a high value of recall is obtained, with a low value of precision. Using function f_1' a balance between precision and recall values is obtained. The fitness function for the GP algorithm must be carefully chosen depending on the problem to be solved, and on the analysts' objectives.

We can compare the GP performance over 3 ratios and 19 ratios data sets. Fitness function f_1 gives approximately the same value of fitness for the 19 ratios data set and for the 3 ratios data set. This results can be interpret as we can generate decision trees with similar performance only with 3 ratios instead of the initial 19 ratios data set.

We can also analyse the behaviour of the GP using the fitness function f_2 soft with depth 2 and depth 4 (note that the SVM does not admit this analysis). In this case the results obtained with the data set of 19 ratios outperforms the results of the data set of 3 ratios in precision and recall. However, the fitness value obtained with the 3 ratios data set is better than the value obtained with the 19 ratios data set.

5.3. Analysis of the best decision trees obtained

Decision trees generated by the GP approach are a useful tool for the analysis of business failure. Depending on the fitness function, maximum depth allowed and the set of functions and

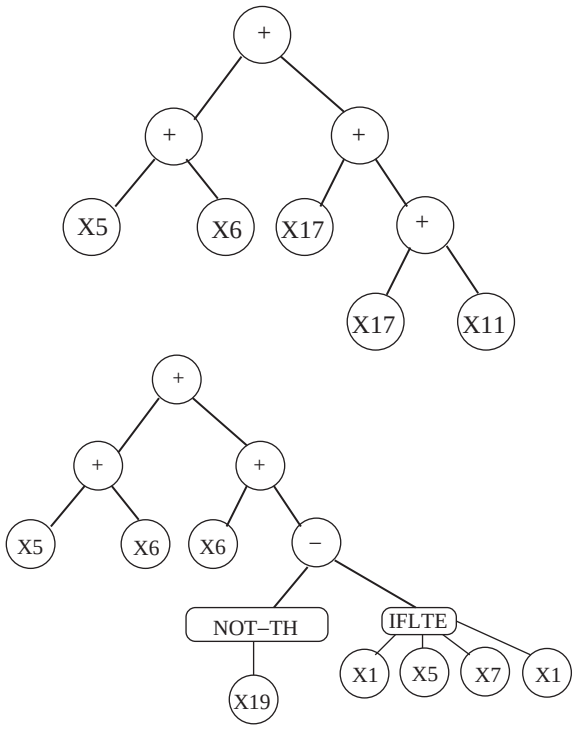

Fig. 1. An example of two different trees with the same value of fitness, precision and recall.

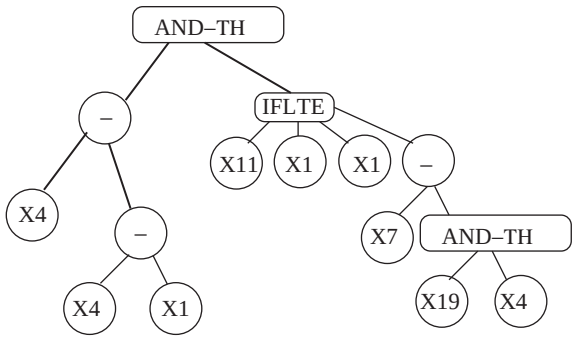

Fig. 2. An example of a tree generated by the GP using the fitness function f_1' with values of fitness, precision and recall of 0.31, 83% and 62%, respectively.

terminals, the GP will generate very different class of trees. Fig. 1 shows an example of two different trees which provide the same values of fitness, recall and precision (0.36, 100% and 58%), using fitness function f_1. However, if we consider fitness functions f_1' the trees are different. Fig. 2 shows a decision tree with value of fitness, recall and precision 0.31, 83% and 62%,

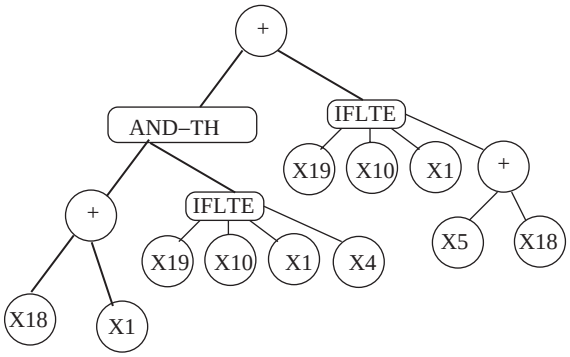

Fig. 3. An example of a tree generated by the GP using the fitness function f_1' with values of fitness, precision and recall of 0.27, 73% and 73%, respectively.

whereas the tree displayed in Fig. 3 provide a values of fitness recall and precision of 0.27, 73% and 73%.

The ratios that have the highest frequency of occurrence in the generated decision trees (considering all experiments performed with fitness function f_1') are R1, R4, R5, R7, R10, R11, R18, R20, R21. This indicates that these variables are highly discriminatory between solvent and insolvent firms in our sample. R1 and R4 are general financial ratios and the others are specific for insurance sector. Their meaning is:

(a) R1—One of the most important questions in order to assure the proper functioning of any firm is the need of having sufficient liquidity. But in the case of an insurance firm, the lack of liquidity should not arise due to "productive activity inversion" which implies that premiums are paid in before claims occur. If an insurance firm cannot pay the incurred claims, the clients and public in general could lose faith in that company. On the other hand, this ratio is a measure of financial equilibrium if it is positive as it implies that the working capital is also positive.
(b) R4—This ratio is a general measuring of profitability. Sometimes it would be better use cash-flow (the numerator of this ratio is cash-flow variable plus extraordinary results) than profits because the first one is less manipulated than the second one.
(c) R5, R7 and R11—These rates are considered as "solvency ratios in strict sense". The numerator shows the risk exposure through earned premiums (R5 and R7) or incurred claims (R11). The denominator shows the real financial support because technical provisions are considered together with capital and reserves. This demonstrates the need of having sufficient shareholder' funds and the need of complying correctly with the technical provisions to guarantee the financial viability of the insurance company. They belong to Insurance Regulatory Information System (IRIS) ratios. IRIS ratios are tests developed by the National Association of Insurance Commissioners (EEUU) as an early warning system.
(d) R10—This ratio shows what proportion of the shareholders' funds (capital and reserves) represents the technical provisions. This confirms the importance, from a solvency viewpoint, of the adequacy of this relation in case the technical provisions were not sufficient to meet the future claims obligations.

(e) R18—Combined ratio. In this case it is expressed net of reinsurance. This ratio is the traditional measuring of underwriting profitability.
(f) R20 and R21—Both of them are IRIS ratios and they are regarded as loss ratios. They measure the overall managerial efficiency.

5.4. Comparison with ratios generated by means of a Rough Set approach

Rough Set Theory was firstly developed by Pawlak [21] as a mathematical tool to deal with the uncertainty or vagueness inherent in a decision making process.

Briefly, the Rough Set approach works by discovering dependencies between attributes in an information table, and reducing the set of attributes by removing those that are not essential to characterize knowledge. A *reduct* is defined as the minimal subset of attributes which provides the same quality of classification as the set of all attributes. A reduced information table may provide decision rules of the form "if conditions then decisions". These rules specify what decisions (actions) should be undertaken when some conditions are satisfied, and can be used to assign new objects to a decision class by matching the condition part of one of the decision rule to the description of the object.

We have performed the rough set analysis using the rough set system (ROSE). For a more detailed description of the Rough Set theory and the *ROSE* software, see [22,23].

The reducts selected for the ten tests have been the following: {R3, R4, R9, R14, R17} for tests 1, 2, 3; {R1, R3, R9, R14, R18, R19} for tests 4, 5, 6; {R1, R4, R9, R17, R20} for test 7; {R1, R9, R13, R18, R20} for test 8; {R1, R9, R14, R17, R19} for test 9 and {R1, R4, R9, R10, R13, R17} for test 10. They have been selected taking into account three questions [24]:

(a) The reduct should have a small number of attributes as possible.
(b) It should have the most significant attributes in our opinion for the evaluation of the companies.
(c) After having selected a few reducts containing the most significant attributes, the reduct chosen should not contain ratios with a very high value for the autocorrelation coefficient.

The classifications accuracies in per cent of correctly classified firms for the ten tests are given in Table 2. Note that the Rough Set achieved good results in terms of classification error. However, ROSE approach used does not allow extracting results in terms of precision and recall, but only in terms of classification accuracy. This fact makes difficult a direct comparison between our proposed method and Rough Set. Nevertheless, we can compare the ratios selected by the Rough Set with the ratios achieved by the GP, in order to extract some conclusions about which are the best ratios to be used in the problem.

Using the Rough Set approach, the ratios that have the highest frequency of occurrence (more than 40%) in reducts are R1, R3, R4, R9, R17, R18 and R19, whereas using the decision trees generated by GP, the ratios that have the highest frequency of occurrence are R1, R4, R5, R7, R10, R11, R18, R20, R21. Therefore, we can consider that R1, R4 and R18 ratios are highly discriminatory variables between solvent and insolvent firms (they appear both in the solutions achieved by the GP and the Rough Set). Consequently, those parties interested in evaluating the solvency of non-life insurance companies should take into account the following questions: the importance of having

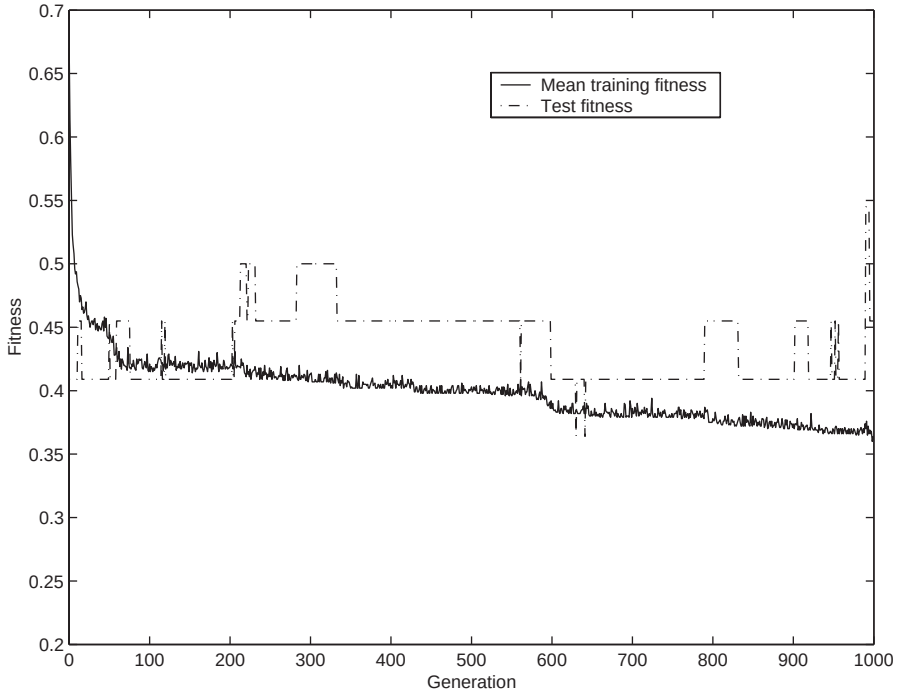

Fig. 4. An evolution of the GP for the training and test fitnesses with trees of depth 2.

sufficient liquidity (R1), the need to generate sufficient profitability to follow a right self-financing (R4) and, finally, the need to set a proper rating in order to calculate right premiums that take into account the whole costs (R18).

5.5. Overfitting analysis

One important problem associated to GP is that the generated decision trees can be adapted to the train set, giving poor results in the test set, this problem is known as overfitting of the obtained solution. Overfitting is common in real problems as the one we tackle. Figs. 4 and 5 show the evolution for the test and train fitness for 19 ratios data set, using trees of depths 2 and 4, respectively. These show the best run out of the 10 different k-folds. Overfitting can be detected by comparing the evolution of the training and test fitnesses. Note that in both figures the fitness in the training set always decreases whereas the fitness in the test set starts growing about generation 150 for trees of depth 4 and about generation 200 with trees of depth 2. In both cases overfitting occurs, however, as the number of generation increases, trees of depth 2 keep training and test fitness curves closer than trees of depth 4, attaining comparable fitness values. Therefore, trees of depth 2 are preferred to trees of depth 4 as they are smaller and easier to analyse. Logically, trees of depth 4 attain lower training fitnesses by overclassifying the training set.

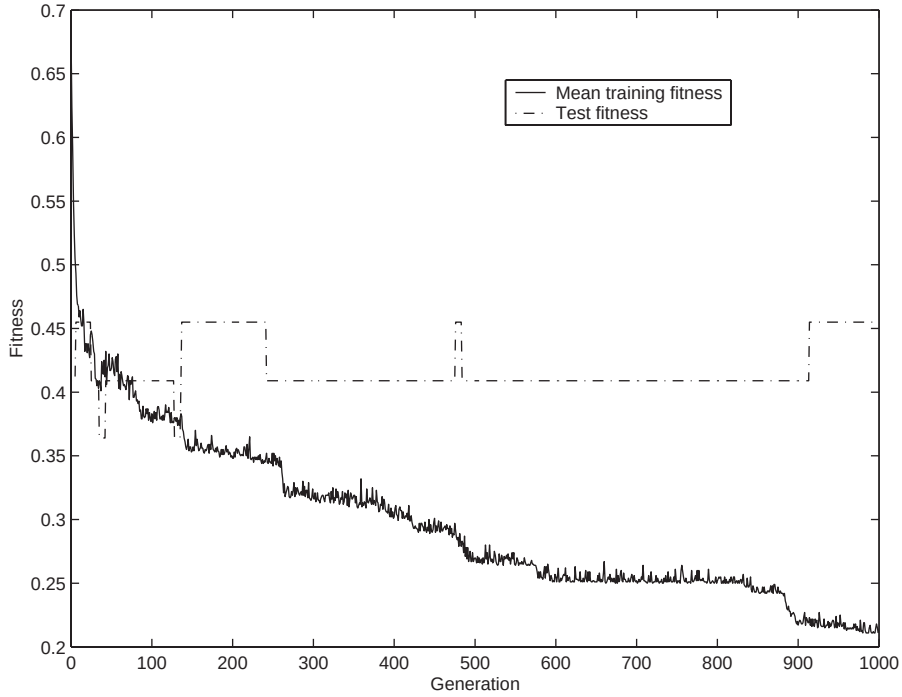

Fig. 5. An evolution of the GP for the training and test fitnesses with trees of depth 4.

6. Conclusions

In this paper, we have presented a genetic programming approach to predict insolvency of non-life insurance companies. We have applied it to a real problem of classification of non-life insurance companies into healthy or failed, formed by 72 companies described by a set of 19 financial ratios, comparing the obtained results with other classification algorithms, a Support Vector Machine and a Rough Set approach. We have shown the suitability of the genetic programming methodology as a support decision method, in such a way that the algorithm proposed in this paper could be useful for insurance regulators, investors, management, financial analysts, banks, auditors, policy holders and consumers.

Acknowledgements

The authors thank professor G. Laporte and the anonymous reviewers for their useful comments and indications.

References

[1] Chen SH, Yeh CH. Using genetic programming to model volatility in financial time series. In: Koza J, et al., editors. Proceedings of the Second Annual Conference on Genetic Programming, 1997. p. 58–63.

[2] Dimitras AI, Zanakis SH, Zopounidis C. A survey of business failures with an emphasis on prediction methods and industrial applications. European Journal of Operational Research 1996;90(3):487–513.
[3] Tam KY. Neural network models and the prediction of bankruptcy. Omega 1991;19(5):429–45.
[4] Tam KY, Kiang MY. Managerial applications of neural networks: the case of bank failure predictions. Management Science 1992;38(7):926–47.
[5] Dimitras AI, Slowinski R, Susmaga R, Zopounidis C. Business failure using rough set. European Journal of Operational Research 1998;114(2):263–80.
[6] Wilson RL, Sharda R. Bankruptcy prediction using neural networks. Decision Support Systems 1994;11:545–57.
[7] Ambrose JM, Carol AM. Using best ratings in life insurer insolvency prediction. Journal of Risk and Insurance 1994;61:317–27.
[8] Barniv R. Accounting procedures, market data, cash-flow figures and insolvency classification: the case of the insurance industry. The Accounting Review 1990;65(3):578–604.
[9] Zopounidis C, Dimitras A. Multicriteria decision aid methods for the prediction of business failure. Dordrecht: Kluwer; 1998.
[10] Segovia-Vargas MJ, Salcedo-Sanz S, Bousoño-Calzón C. Prediction of non-life insurance companies using Support Vector Machines and genetic algorithms. In: Proceedings of X SIGEF Congress in Emergent Solutions for the Information and Knowledge Economy, León, Spain, October 2003, submitted for publication.
[11] Koza J. Genetic programming. Cambridge, MA: MIT Press; 1992.
[12] Goldberg DE. Genetic algorithms in search, optimization and machine learning. Reading, MA: Addison-Wesley; 1989.
[13] Vapnik V. Statistical learning theory. New York: Wiley; 1998.
[14] Michalewicz Z. Genetic algorithms + data structures = evolution programs. Berlin: Springer; 1996.
[15] Koza J. Genetic programming for economic modeling. Statistics and Computing 1994;4(2):187–97.
[16] Tackett WA. Genetic programming for feature discovery and image discrimination. In: Forrest S, editor. Proceedings of the Fifth International Conference on Genetic Algorithms. Los Altos, CA: Morgan-Kaufmann; 1993. p. 300–9.
[17] Theodossiou P, Kahya E, Saidi R, Philippatos G. Financial distress and corporate acquisitions: further empirical evidence. Journal of Business Finance 1996;23(5–6):699–719.
[18] Sanchis A, Gil JA, Heras A. El análisis discriminante en la previsión de la insolvencia en las empresas de seguros no vida. Revista Española de Financiación y Contabilidad 2003;32(116):183–233.
[19] Burges JC. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 1998;2(2):121–67.
[20] Scholkopf B, Smola A. Learning with kernels. Cambridge, MA: MIT Press; 2002.
[21] Pawlak Z. Rough sets. Theoretical aspects of reasoning about data. London: Kluwer Academic Publishers; 1991.
[22] Predki B, Slowinski R, Stefanowski J, Susmaga R, Wilk S. ROSE: software implementation of the rough set theory. Rough sets and current trends in computing, Lecture Notes in Artificial Intelligence, vol. 1424. Berlin: Springer, 1998. p. 605–08.
[23] Predki B, Wilk S. Rough set based data exploration using ROSE system. Foundations of intelligent systems, Lecture Notes in Artificial Intelligence, vol. 1609. Berlin: Springer, 1999. p. 172–80.
[24] Segovia MJ, Gil JA, Heras A, Vilar JL, Sanchis A. Using Rough Sets to predict insolvency of Spanish non-life insurance companies. Proceedings of Sixth International Congress on Insurance: Mathematics and Economics, Lisboa, 2002.