García-Almanza, A.L. & Tsang, E.P.K., Detection of stock price movements using chance discovery and genetic programming, International Journal of Knowledge-based and Intelligent Engineering Systems, Vol.11, No.5, December 2007, 329-344

The Repository Method for Chance Discovery in Financial Forecasting

Alma Lilia Garcia-Almanza and Edward P.K. Tsang

Department of Computer Science
University of Essex
Wivenhoe Park, Colchester, CO4 3SQ, UK
algarc@essex.ac.uk, edward@essex.ac.uk

Abstract. *The aim of this work is to forecast future events in financial data sets, in particular, we focus our attention on situations where positive instances are rare, which falls into the domain of Chance Discovery. Machine learning classifiers extend the past experiences into the future. However the imbalance between positive and negative cases poses a serious challenge to machine learning techniques. Because it favours negative classifications, which has a high chance of being correct due to the nature of the data. Genetic Algorithms have the ability to create multiple solutions for a single problem. To exploit this feature we propose to analyse the decision trees created by Genetic Programming. The objective is to extract and collect different rules that classify the positive cases. It lets model the rare cases in different ways, increasing the possibility of identifying similar cases in the future. The over-learning produced by this method attempts to compensate the lack of positive cases. To illustrate our approach, it was applied to a data set composed by closing prices from the London Financial Market for finding investment opportunities with very high returns. From experiment results we showed that the Repository Method can consistently improve both the recall and the precision.*
Keywords: Chance Discovery, Classification, Genetic Programming.

1 Introduction

Financial forecasting is one of the important areas in computational finance [1]. Based on Genetic Programming (GP) [2] and aided by constraints [3], EDDIE is a machine learning tool for financial forecasting [4, 1]. However, in some situations, the number of profitable investment opportunities is extremely small, this occurs, for example, in finding arbitrage opportunities [5]. The interest in finding rare opportunities motivates our research in chance discovery [6, 7].

Machine learning classifiers, like other forecasting techniques, involve extending the past experiences into the future. However the imbalance between positive and negative cases poses a serious challenge to machine learning. Specifically GP, which is a technique that tries to emulate the evolution, has limitations to deal with imbalanced data sets. Because it favours negative classifications, which has a high chance of being correct due to the nature of the data. In imbalanced data

sets the performance of a classifier must not be measured only by the *accuracy* [1] [8, 9]. A common measure for a classifier performance, in imbalanced classes, is the geometric mean of the product of *precision*[2] and *recall* [3][9].

The objective of the Repository Method (RM) is to increase the recall in the classification without sacrificing the precision (i.e. without substantially increase the total number of false positives). Genetic algorithms are able to produce multiple solutions for a single problem. However, the standard procedure is to choose only the best individual of the evolution as the optimal solution of the problem and discard the rest of the population. A GP process spends a lot of computational resources evolving entire populations for many generations. For this reason we presume that the remaining individuals could contain useful information that is not necessarily considered in the best individual. We propose to analyse the decision trees in a wider part of the population and in different stages of the evolutionary process. The idea behind this approach is to collect different rules that model the rare cases in diverse ways. The over-learning produced by this method attempts to compensate the lack of positive cases in the data set. This work is illustrated with a data set composed by closing prices from the London Financial Market. The remainder of this paper is organized as follows: Section 2 contains an overview of the problem that illustrates our method; Section 3 presents our approach, while Section 4 describes the experiments to test our method. Section 5 presents the experiment results. Finally, Section 6 summaries the conclusions.

2 Problem Description

To illustrate our method, it was applied to a problem for discovering classification rules in a financial stock data set. The goal is to create a classifier to predict future movements in the stock price. This problem has been addressed previously by Tsang *et al.* [10, 4, 5]. Every case in the dataset is composed by a *signal* and a set of attributes or *independent variables*. The signal indicates the opportunities for *buying* or *not buying* and *selling* or *not selling*. The signal is calculated looking ahead in a future horizon of n units of time, trying to detect an increase or decrease of at least $r\%$. The independent variables are composed by financial indicators derived from financial technical analysis. Technical analysis is used in financial markets to analyse the price behaviour of stocks. This is mainly based on historical prices and volume trends [11].

3 Repository Method

The objective of this approach is to compile different rules that model the positive cases in the training data set. Since the number of positive cases is very small,

[1] Accuracy is the proportion of the total number of predictions that were correctly predicted

[2] Precision is the proportion of the predicted positive cases that were correct

[3] Recall is the proportion of positive cases that were correctly identified

Table 1. Discriminator Grammar.

G	→ <Root>
<Root>	→ "If-then-else",<Conjunction>|<Condition>,"Class","No Class"
<Conditional>	→ <Operation>, <Variable>, <Threshold> | <Variable>
<Conjunction>	→ "and"|"or",<Conjunction>|<Conditional>, <Conjunction>|<Conditional>
<Operation>	→ "<", ">"
<Variable>	→ Variable_1 | Variable_2 | ... Variable_n
<Threshold>	→ Real number

it is important to gather all available information about them. RM analyses a wider part of the population to collect useful rules. This analysis is extended to different stages of the evolutionary process. The selection is based on the performance and novelty of the rule. Decision tree analysis and rule collection has been previously addressed by Quinlan [12]. RM involves the following steps: 1) Rule extraction 2) Rule simplification 3) New rule detection. The above procedures will be explained in the following sections.

3.1 Rule extraction

Rule extraction involves the analysis of the decision tree in order to delimit its rules. For this reason decision trees are generated and evolved using Discriminator Grammar (DG), see Table 1. This grammar[4] produces trees that classify or not a single class, Figure 1 illustrates a decision tree that was created using DG.

To extract the tree rules, let T be a tree with syntax DG, it means that T is composed by rules and it can be expressed as the union of its rules such as $T = (R_1 \cup R_2 \cup \cdots R_n)$ where R_i is a rule and n is the total number of rules in T. A rule R_k is a minimum set of conditions that satisfy the tree T, to satisfy R_k every condition in the rule has to be satisfied. Rule extraction is concerned with the discovery of all minimal sets of conditions that satisfy T (see Figure 1). Once a rule $R_k \in T$ has been extracted this is individually evaluated against the training data. If the precision of R_k achieves a predefined Precision Threshold (PT), then R_k is considered for the next step (rule simplification), otherwise R_k is discarded.

3.2 Rule Simplification

The procedure to identify new rules involves the comparison between R_k and the rules in repository. However noisy and redundant conditions are an impediment to make an effective rule comparison. For this reason rule simplification is an essential part of RM. Rule simplification is a hard task, specially in latest stages of the evolutionary process, due to decision trees generated by GP tend

[4] The term grammar refers to a representation concerned with the syntactic components and the rules that specify it [13]

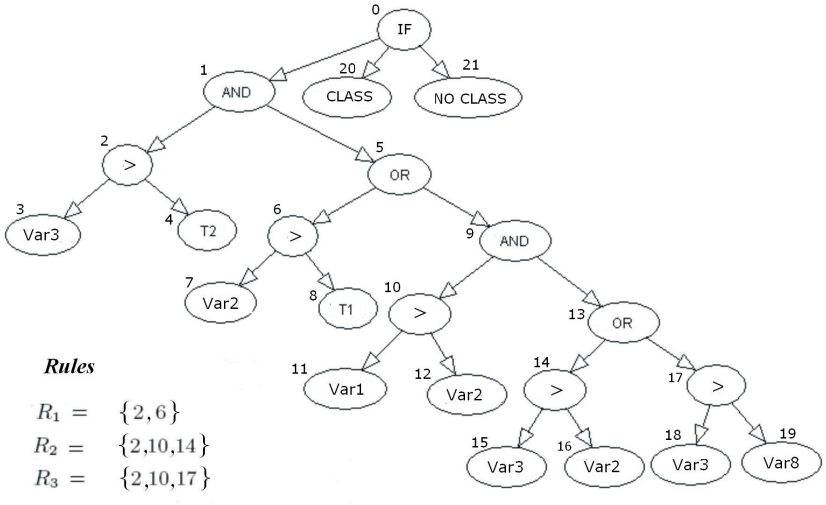

Fig. 1. The figure shows a decision tree generated by DG and its rules. Every rule is composed by a set of conditions, which are represented by the number of the *conditional node* (i.e. node with syntax <Conditional> in DG).

to grow and accumulate introns [14–17]. To simplify rules we have defined two types of conditions: *hard conditions* and *flexible conditions*. A hard condition is a comparison of two variables (e.g. $var_1 < var_2$). A flexible condition is the equation between a variable and a threshold (e.g. $var_1 < .8$). When two flexible conditions have the same variable and operator they are defined as *similar conditions* (e.g. $var_1 < 3$ and $var_1 < 2$ are similar conditions). Conditions were divided in two types (hard and flexible) because the conditions that compare thresholds are difficult to differentiate (e.g. $var_1 < .08998$ and $var_1 < .08991$). Besides these can be easily simplified.

Next, we describe briefly the procedure to simplify rules. Details are omitted in this paper due to lack of space, however a detailed explanation is available in [18]. Redundant and noisy conditions have to be removed and flexible conditions have to be simplified. Let $R_k = \{c_1, c_2 \dots c_n\}$ where c_i is a condition and n is the number of conditions in R_k

- If $c_1, c_2 \in R_k$ are hard conditions and $c_1 = c_2$ then $R_k = R_k - c_2$
- If $c_1, c_2 \in R_k$ are flexible conditions and c_1 and c_2 are similar conditions then c_1 and c_2 are simplified using the simplification table in [18]. (e.g. conditions $var_1 < 12$ and $var_1 < 10$ are similar and they can be replaced by $var_1 < 10$)
- If $c_i \in R_k$ and $Performance(R_k) = Performance(R_k - c_i)$ then $R_k = R_k - c_i$

3.3 New rule detection

Once the rule R_k has been simplified, we are able to determine if this is a new rule (i.e. $R_k \neq R_j \forall R_j \in Rep$, where $Rep = \{R_i\}$ is the set of rules in the repository

). Let R_i be a *hard rule* if R_i is comprised exclusively of hard conditions. Let R_i be a *flexible rule* if it has at least one flexible condition. R_k and R_i are *similar rules* if they have the same hard conditions and similar flexible conditions. The following procedure determines if R_k is added or not to the rule repository.

- If R_k is a hard rule and $R_k \notin Rep$ then $Rep = Rep \cup R_k$
- If R_k is a flexible rule and $\exists\ R_i \in Rep$ such as R_k and R_i are similar rules and $Fitness(R_k) > Fitness(R_i)$ then $Rep = (Rep - R_i) \cup R_k$
- If R_k is a flexible rule and there is not a $R_i \in Rep$ such as R_k and R_i are similar rules then $Rep = Rep \cup R_n$

4 Experiments Description

To test our approach a series of experiments was conducted. The performance was measured in terms of the recall, precision and accuracy. A population of 1,000 individuals was evolved and every ten generations the entire population was saved, let's call them $P_{10}, P_{20}, \ldots, P_{100}$. Subsequently RM analysed these populations and compiled the useful rules. RM accumulated all the useful rules during the entire process. The experiment was tested using different values for the precision threshold (PT = 60%, 70%, 80%). This process was repeated twenty times, the results of the experiment were grouped and averaged by generation and PT. Table 2 presents the GP parameters used to evolve the populations.

4.1 Training data description

The data sets to train and test the GP in the experiment came from the London stock market. Every data set contains 890 records each from Barclays stock (from March, 1998 to January, 2005). The attributes of each record are composed by indicators derived from financial technical analysis; these were calculated on the basis of the daily *closing price*[5], volume and some financial indices as the FTSE[6]. We looked for *selling* opportunities of 15% in ten days. The number of positives cases is 39 in the training data set and 23 in the testing. The opportunities are naturally grouped in clusters, close to the peak to predict.

5 Main Results

This section documents the results obtained by applying RM to the set of populations described in the previous section. All figures and tables given in this section denote average results from series of twenty test runs.

Table 3 shows the recall, precision and accuracy (columns 2,6,10) of the best individual (according to the fitness function defined in Table 2) of a GP

[5] The settled price at which a traded instrument is last traded at on a particular trading day

[6] An index of 100 large capitalization companies stock on the London Stock Exchange, also known as "Footsie.

Table 2. Summary of Parameters.

Parameter	Value
Population size	1,000
Initialization method	Growth
Generations	100
Crossover Rate	0.8
Mutation Rate	0.05
Selection	Tournament (size 2)
Elitism	Size 1
Control bloat growing	Tarpeian method, 50% of trees whose largest branch exceed 6 nodes are penalized with 20% of the fitness for each node that surpassed the largest branch allowed.
Fitness Function	$\sqrt{Recall \cdot Precision}$

evolution. In the same way the recall, precision and accuracy obtained by RM, using PT =60%,70%,80%, are described in Table 3, all the results were obtained using the testing data set. The standard GP recall fluctuates from generation to generation, because this was evaluated with the testing data set. In contrast the recall obtained by RM increases consistently with the generations. Table 3 shows that, except for earliest generations, RM can obtain better recall than the best GP tree. The low performance in earliest generations is probably due to the fact that most of the trees were not too far from being random. Combining semi-random trees do not necessarily produce better results. However when the evolutionary process advances, it tends to generate more and better rules, in consequence RM performance improves. As can be seen in Table 3, precision improvement shows a high level of consistency. However, in some cases the improvement in recall and precision is paid by decrease in accuracy, as it can be seen in generation 90. This is explained because the evolution pressure discourages positive classifications in GP, given that they have a small chance of being correct (a standard feature in chance discovery). Experiments show that RM out more rules with positive classifications (which is part of the design). In addition, most of the extra positive classifications were correctly made. This is reflected in both increase in recall and precision. But, since more errors were made (not as much, in proportion, as the correct classifications), the overall accuracy has been decreased. Given that our goal is to improve recall and precision, this is an acceptable price to pay.

6 Conclusions

The objective of RM to mine the knowledge acquired by the evolutionary process in order to compile more features from the training data. The procedure involves compiling rules from different individuals and stages of the evolutionary process. Our experimental results showed that by combining rules from different classification trees, we can classify more positive cases. RM outperformed the best tree generated by a standard GP, improving the precision and recall in

Table 3. Recall, Precision and Accuracy of a standard GP and repository method using PT = 60%, 70%, 80%

Gen	RECALL				PRECISION				ACCURACY			
	GP	Repository method			GP	Repository method			GP	Repository method		
		PT=60%	70%	80%		PT=60%	70%	80%		PT=60%	70%	80%
10	1%	1%	0%	0%	1%	1%	0%	0%	97%	97%	97%	97%
20	0%	4%	3%	2%	0%	2%	2%	1%	97%	95%	95%	96%
30	13%	11%	6%	3%	2%	5%	3%	2%	90%	93%	93%	95%
40	10%	21%	16%	10%	4%	6%	5%	5%	92%	91%	91%	93%
50	13%	30%	21%	12%	6%	7%	6%	5%	93%	89%	90%	92%
60	8%	39%	28%	19%	7%	9%	8%	7%	94%	88%	89%	91%
70	16%	44%	35%	23%	5%	9%	8%	7%	88%	87%	88%	90%
80	10%	44%	39%	29%	4%	9%	8%	8%	93%	87%	87%	89%
90	6%	46%	40%	32%	6%	9%	8%	8%	94%	86%	87%	88%
100	21%	47%	43%	34%	3%	9%	9%	8%	82%	85%	86%	87%

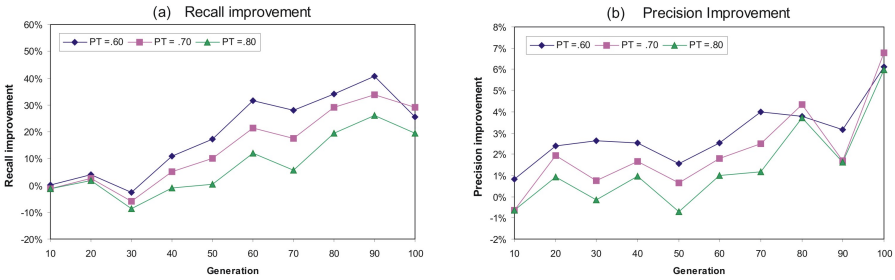

Fig. 2. Experimental Results. Improvement achieved by RM. (a) Recall improvement (b) Precision improvement

classification. Our approach is a general method, and therefore, results should not be limited to financial forecasting. It should be useful for all classification problems where chances are rare, i.e. the data set is imbalanced. Therefore, the Repository Method is a promising general tool for chance discovery.

Acknowledgement

Authors would like to thank anonymous reviewers for their helpful comments. The first author thanks to Consejo Nacional de Ciencia y Tecnología (CONACyT) to support her studies at the University of Essex.

References

1. E. P. Tsang and S. Martinez-Jaramillo, "Computational finance," in *IEEE Computational Intelligence Society Newsletter*, 2004, pp. 3–8.

2. J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, Massachusetts: The MIT Press, 1992.
3. J. Li, *A genetic programming based tool for financial forecasting*. Colchester CO4 3SQ, UK: PhD Thesis, University of Essex, 2001.
4. E. P. Tsang, P. Yung, and J. Li, "Eddie-automation, a decision support tool for financial forecasting" in *Journal of Decision Support Systems, Special Issue on Data Mining for Financial Decision Making*, ser. 4, vol. 37, 2004.
5. E. P. Tsang, S. Markose, and H. Er, "Chance discovery in stock index option and future arbitrage," in *New Mathematics and Natural Computation, World Scientific*, ser. 3, vol. 1, 2005, pp. 435–447.
6. A. Abe and Y. Ohsawa, "Special issue on chance disocvery," in *New Generation Computing*, ser. 1, vol. 21. Berlin: Springer and Tokyo: Ohmsha, November 2002, pp. 1–2.
7. Y. Ohsawa and P. McBurney, *Chance discovery*. Springer, 2003.
8. F. J. Provost, T. Fawcett, and R. Kohavi, "The case against accuracy estimation for comparing induction algorithms," in *In Proc. Fifteenth Intl. Conf. Machine Learning*, W. Madison, Ed., 1998, pp. 445–553. [Online]. Available: citeseer.ist.psu.edu/provost97case.html
9. M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," in *Machine Learning*, vol. 30. 195-215, 1998.
10. E. P. Tsang, J. Li, and J. Butler, "Eddie beats the bookies," in *International Journal of Software, Practice and Experience*, ser. 10, vol. 28. Wiley, August 1998, pp. 1033–1043.
11. W. F. Sharpe, G. J. Alexander, and J. V. Bailey, *Investments*. Upper Saddle River, New Jersey 07458: Prentice-Hall International, Inc, 1995.
12. J. R. Quinlan., "Rule induction with statistical data" in *Journal of the operational research Society*, 38, 1987, pp. 347-352
13. N. Chomsky, *Aspects of the theory of syntax*. Cambridge M.I.T. Press, 1965.
14. P. Angeline, "Genetic Programming and Emergent Intelligence," in *Advances in Genetic Programming*, K. E. Kinnear, Jr., Ed. MIT Press, 1994, ch. 4, pp. 75–98.
15. P. Nordin, F. Francone, and W. Banzhaf, "Explicitly Defined Introns and Destructive Crossover in Genetic Programming," in *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, J. P. Rosca, Ed., Tahoe City, California, USA, 9 July 1995, pp. 6–22.
16. T. Soule and J. A. Foster, "Code size and depth flows in genetic programming," in *Proceeding of the Second Annual Conference*, J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. R. Riolo, Eds. Morgan Kaufmann, 1997, pp. 313–320.
17. W. B. Langdon, "Quadratic bloat in genetic programming," in *Proceedings of the Genetic and evolutionary Computation Conference*, 2000, pp. 451–458.
18. A. L. Garcia-Almanza, "Technical report, rule simplification," http://privatewww.essex.ac.uk/~algarc/documents/Rule-simplification.doc.