Lingnan University

Digital Commons @ Lingnan University

Theses & Dissertations

Department of Computing and Decision
Sciences

4-8-2015

Financial fraud detection by using grammar-based multiobjective genetic programming with ensemble learning

Haibing LI

Follow this and additional works at: https://commons.ln.edu.hk/cds_etd

Part of the [Business Administration, Management, and Operations Commons](#), and the [Management Information Systems Commons](#)

Recommended Citation

Li, H. (2015). Financial fraud detection by using grammar-based multiobjective genetic programming with ensemble learning (Master's thesis, Lingnan University, Hong Kong). Retrieved from http://commons.ln.edu.hk/cds_etd/13/

This Thesis is brought to you for free and open access by the Department of Computing and Decision Sciences at Digital Commons @ Lingnan University. It has been accepted for inclusion in Theses & Dissertations by an authorized administrator of Digital Commons @ Lingnan University.

Terms of Use

The copyright of this thesis is owned by its author. Any reproduction, adaptation, distribution or dissemination of this thesis without express authorization is strictly prohibited.

All rights reserved.

FINANCIAL FRAUD DETECTION BY USING GRAMMAR-BASED
MULTIOBJECTIVE GENETIC PROGRAMMING WITH ENSEMBLE
LEARNING

LI HAIBING

MPHIL

LINGNAN UNIVERSITY

2015

FINANCIAL FRAUD DETECTION BY USING GRAMMAR-BASED
MULTIOBJECTIVE GENETIC PROGRAMMING WITH ENSEMBLE
LEARNING

by
LI Haibing

A thesis
submitted in partial fulfillment of the
requirements for the Degree of
Master of Philosophy in Business
(Computing and Decision Sciences)

Lingnan University

2015

ABSTRACT

Financial Fraud Detection by using Grammar-based Multiobjective Genetic Programming with ensemble learning

by

LI Haibing

Master of Philosophy

Financial fraud is a criminal act, which violates the law, rules or policy to gain unauthorized financial benefit. As an increasingly serious problem, it has attracted a lot of concerns. The major consequences are loss of billions of dollars each year, investor confidence and corporate reputation. Therefore, a study area called Financial Fraud Detection (FFD) is obligatory, in order to prevent the destructive results caused by financial fraud. In general, traditional modeling approaches are applied and based on pre-defined hypothesis testing of causes and effects for FFD problems. In addition, the evaluation criteria are often based on variable significance level or Goodness-of-fit only.

FFD has many common features like other data mining problems. It has accumulated vast amounts of data records of different forms (e.g. financial statements or annual reports) over a period of time. It is very difficult to observe the interesting information just by relying on traditional statistical methods. However, data mining techniques can be used to extract implicit, previously unknown and potentially useful patterns, rules or relations from massive data repositories. Such discovered patterns are appropriate to executive leadership, stakeholders and related regulatory agencies to reduce or avoid the losses.

As real-life problems, it is not sufficient for FFD to consider only a single criterion (e.g. Goodness-of-fit or accuracy). Instead, FFD can also seek multiple objectives (e.g. accuracy versus interestingness). It is not easy to consider multiple objectives at the same time unless applying combination methods (e.g. linear combination) by assigning different weights to present the importance for each criterion by using data mining techniques with a single evaluation criterion. For example, accuracy is more important than interestingness with weights of 0.9:0.1. But it is still difficult to decide the appropriate or exact values for weights. Therefore, multi-objective data-mining techniques are required to tackle FFD problems.

In this study, FFD is targeted, and comprehensively evaluated by a number of methods. The proposed method is based on Grammar-Based Genetic Programming (GBGP), which has been proven to be a powerful data mining technique to generate compact and straightforward results. The major contributions are three improvements of GBGP for FFD problems. First, multi-criteria are considered by integrating the concept of multi-objectives into GBGP. Second, minority prediction is applied to demonstrate the class prediction with unmatched rows

in their rules. Lastly, a new meta-heuristic approach is introduced for ensemble learning in order to help users to select patterns from a pool of models to facilitate final decision-making. The experimental results showed the effectiveness of the new approach in four FFD problems including two real-life problems. The major implications and significances of the study can concretely generalize for three points. First, it suggests a new ensemble learning technique with GBGP. Second, it demonstrates the usability of classification rules generated by the proposed method. Third, it provides an efficient multi-objective method for solving FFD problems.

DECLARATION

I declare that this is an original work based primarily on my own research, and I warrant that all citations of previous research, published or unpublished, have been duly acknowledged.

1
SIGNED

LI Haibing

12.07.2015

Date

CERTIFICATE OF APPROVAL OF THESIS

FINANCIAL FRAUD DETECTION BY USING GRAMMAR-BASED
MULTIOBJECTIVE GENETIC PROGRAMMING WITH ENSEMBLE
LEARNING

by
LI Haibing

Master of Philosophy

Panel of Examiners:

SIGNED (Chairman)
Prof. CUI Geng

SIGNED (External Member)
Prof. CHENG Chun-hung

SIGNED (Internal Member)
Prof. WONG Man-Leung

SIGNED (Internal Member)
Prof. GAO Jin

Supervisor:

Prof. WONG Man-Leung

Approved for the Senate:

SIGNED
Prof. SHARMA Shalendra
Chairman, Postgraduate Studies Committee

08 APR 2015

Date

Library Rights Statement

In presenting the thesis, *Financial Fraud Detection by using Grammar-based Multiobjective Genetic Programming with ensemble learning*, in partial fulfilment of the requirements for a master of philosophy degree at Lingnan University, I agree that the Library shall make it freely available for inspection. I further agree that permission for copying, as provided for by the Copyright Law of Hong Kong, of this thesis for scholarly purposes may be granted by the Librarian. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my written permission.

I hereby grant permission to the Library of Lingnan University to use my thesis for scholarly purposes.

1
SIGNED

LI Haibing

12.07.2015

Date

TABLE OF CONTENTS

TABLE OF CONTENTS	i
LIST OF TABLES	iii
LIST OF FIGURES	iv
ACKNOWLEDGEMENTS	v
CHAPTER	
1 Introduction	1
1.1 Financial Fraud	1
1.2 Financial Fraud Detection (FFD) and Data Mining (DM) . . .	2
1.3 Data Mining Techniques in Financial Fraud Detection	3
1.4 Financial Fraud Datasets	4
1.5 Evolutionary Algorithms (EAs)	5
1.6 Ensemble Learning	6
1.7 Research Motivation	6
1.8 Organization of the Thesis	8
2 Background and Literature Review	10
2.1 Data Mining and Data Mining Applications	10
2.2 Financial Fraud	12
2.2.1 Credit Card Fraud	13
2.2.2 Corporate Securities Fraud	13
2.3 Data Mining Techniques in Financial Fraud Detection	15
2.4 Introduction to Data Mining Techniques	18
2.4.1 Logistic Regression	18
2.4.2 Neural Networks	19
2.4.3 Support Vector Machine (SVM)	21
2.4.4 Bayesian Networks	23
2.4.5 Classification and Decision Trees (CARTs)	24
2.5 Multi-Objective Problems and Multi-Objective Evolutionary Algorithms	27
2.5.1 Non-dominated Sorting Genetic Algorithm II (NSGA-II)	31
2.6 Genetic Programming and Grammar-Based Genetic Program- ming	34
2.6.1 Genetic Programming (GP)	34
2.6.2 Grammar-Based Genetic Programming (GBGP)	37
2.7 Ensemble Learning	39
2.7.1 Bagging	39
2.7.2 Boosting	40
2.7.3 Stacking	41
3 Grammar-Based Multi-objectives Genetic Programming with Ensemble Learning	42
3.1 Grammar-based Multi-objective Genetic Programming (GB- MGP)	43
3.1.1 Classifier Representation	45

	Page
3.1.2 Grammar-based Genetic Programming (GBGP)	46
3.1.3 Objective Functions	51
3.1.4 Non-dominated Sorting Genetic Algorithm II (NSGA-II)	55
3.1.5 Token Competition	56
3.2 Statistical Selection Learning	59
3.2.1 Majority Voting	60
3.2.2 Weighted Voting	60
3.2.3 Statistical Selection	61
3.3 Minority Prediction	62
4 Experiments and Results	65
4.1 Introduction to Experiment Preparation	65
4.1.1 Waikato Environment for Knowledge Analysis (WEKA)	65
4.1.2 Data Description	66
4.1.3 Grammar Design	67
4.1.4 Synthetic Minority Over-sampling Technique (SMOTE)	67
4.1.5 K-folds Cross-Validation	68
4.1.6 Model Evaluation Criteria	69
4.2 GBMGP with SSL	70
4.3 Compared Approaches	72
4.3.1 Results and Analysis	73
4.3.2 Empirical Analysis	78
4.3.3 Statistical Analysis	81
5 Conclusion	84
5.1 Major Findings	84
5.2 Contributions	87
5.3 Implications	89
5.4 Limitations and Directions for Future Research	90
 APPENDIX	
Appendix A	92
A.1 Grammar	92
A.1.1 Australian Credit	92
A.1.2 Credit Approval	93
A.1.3 U.S. Corporate Securities Fraud	94
A.1.4 China Corporate Securities Fraud	95

LIST OF TABLES

Table		Page
1	Simple example of a context-free grammar.	37
2	General procedure of using GBMGP with an Ensemble Learning technique	43
3	Example of a grammar to control the evolutionary process . .	49
4	Compact training dataset regarding firms	51
5	Compact database about firms	53
6	Data Description.	66
7	Contingency table with four outcomes of binary classification .	69
8	Parameters and values for the proposed method	71
9	Parameters and values for the compared approaches	72
10	Accuracies of data mining techniques and the proposed method	73
11	Classification accuracies of the proposed method and its variants	75
12	Abbreviations of all the approaches	76
13	Classification accuracies of the proposed method and its variants without using token competition	77
14	RAI test result	82

LIST OF FIGURES

Figure		Page
1	Example of a multilayer perceptron neural network with three layers	19
2	Example of a SVM	22
3	Simple Bayesian Network	23
4	Illustration of the decision tree splitting process	25
5	Example of Pareto Front	28
6	Crowding distance calculation	33
7	Simple GP flowchart	35
8	Four individuals in GP	35
9	GP crossover	36
10	GP mutation and reproduction	36
11	Individual program in Grammar-based GP represents if term2 > 4 is true then yes; otherwise no	38
12	Example of “noisy” data	42
13	General process of GBMGP	44
14	Possible design of a chromosome	47
15	Example of invalid mutation	48
16	Example of two individuals	49
17	Example of a population sorted by NSGA-II	55
18	Example of a population	57
19	Example of a population with tokens	59
20	Ensemble learning example using majority voting	60
21	Ensemble learning example using weighted voting	61
22	The flowchart of the proposed method.	64
23	Ten-folds cross-validation example	68
A.1	Australian Credit Grammar.	92
A.2	Credit Approval Grammar.	93
A.3	U.S. Corporate Securities Fraud.	94
A.4	China Corporate Securities Fraud Grammar.	95

ACKNOWLEDGEMENTS

In the last two year, a lot of interesting things happened and I also have met a lot of people. I had made a new career plan. For the coming journey of work, my feelings are also some complex, because only two months left for me to have a student identity when I am writing this acknowledgements. Before I coming to Hong Kong, I was full of expectation for the life in Hong Kong. When I studying at Lingnan University, I was really enjoy the student life. At present, I am looking forward to develop the future of my family and business. Each of them will affect my whole life, but most of the time is spent on the study over the two years. During the study, I have also experienced a lot of happiness, excitement, distress and confusions. At the same time, I feel that it needs to put huge effort on each research study. Therefore, firstly I owe my deepest gratitude to my supervisor: Dr. Wong Man-Leung at Lingnan University. His patience, encouragement, immense knowledge and insightful advice were key motivations throughout my MPhil research. Without his supervision, it is impossible to finish this study. In addition to academic support, he also gave me some suggestions on my career development.

I would like to express my deepest thanks to Mrs. Xu Qian-Yue, who became my wife in this year. For her love, patience, and understanding, she allowed me to spend most of the time on this thesis.

I would like to thank to Mr. Liu Si-Bo for preparing the U.S. Corporate Securities Fraud and China Corporate Securities Fraud (CCSF) datasets. He also provided many useful comments and help for my research.

I would also like to thank to Mr. Zhuang Meng-Zhou, who was my room-

mate. We often play basketball or hiking in our spare time.

I should also give my best regards to my colleagues and friends over last two years in the Department of Computing and Decision Sciences. Thanks especially to Prof. Cui Geng, Dr. Peng Ling, Zhang Chun, Huang Wen-Xin, Li Chun-Yu, Eva Cheung, Yang Jing-Yu, Luo Hua-Jiang, Yu Qian, Wang Hai-Xiu, Linda Wang, May Wang, Li Yong-Bo and Xue Xin-Shu.

Last but not least, I also want to express my gratitude and love to my parents and my wife's parents. They always encouraged and inspired me to achieve my goals.

CHAPTER 1

Introduction

1.1 Financial Fraud

Financial fraud is a serious problem that often produces destructive results in the world and it is growing fast in many countries, such as China. It is a criminal act, which violates the law to gain unauthorized financial benefit (Ngai et al., 2011). Financial fraud refers to many activities, such as credit card fraud, money laundering, insurance fraud, corporate fraud, etc. Credit card fraud and corporate fraud have attracted a great deal of attention from the year of 1998, and are still in the trend of escalation (Ngai et al., 2011). Credit card fraud is about unauthorized usage of a credit card, unusual transaction behaviour or transactions on an inactive card (Syeda et al., 2002). In the era of rapid development of information technology, a vast volume of information can be created every second, but there can be a lack of powerful techniques that can analyze the information. It is costly to detect the potential fraudulent transactions manually. The results may be destructive if one chooses to ignore them or detect them incorrectly. At the same time, credit cards are the most popular transaction method with increasing users, but the credit card fraud rate is also increasing. Corporate securities fraud in this study is related to corporate fraud in listed firms. For example, it may be perpetrated to increase the stock prices of fraudulent firms, to obtain more loans from banks or repay lesser dividends to shareholders (Ravisankar et al., 2011). In the U.S., financial analysts have been confirmed to contribute to corporate fraud detection. Effective external monitoring can increase the confidence of shareholders or investors, which is crucial to the functioning of any capital market (Cumming et al., 2011). It is also important for China's securities market, as corporate fraud can impede China's economic development since it has serious consequences for shareholders, employees and society (Cumming et al., 2011). No matter what type of fraud is involved, it results in losses of billions of dollars every year (Kou et al., 2004). Since the amount of fraud is increasing rapidly,

the workload of auditors is also increasing. They have become overburdened with the task of detection of fraud. Various efficient financial fraud detection (FFD) techniques are required to detect which will commit a fraud.

1.2 Financial Fraud Detection (FFD) and Data Mining (DM)

Financial Fraud Detection (FFD) is vital to prevent the destructive consequences of financial fraud. It can distinguish fraudulent information from data, thereby discovering fraudulent activities or behaviour and enabling decision makers to develop appropriate policies and strategies to decrease the influences of fraud (Ngai et al., 2011). In recent years, FFD has become a hot spot domain, because it has many features like other Data Mining (DM) problems. DM plays an important role in FFD, since it uses model(s) to automatically discover useful patterns from massive data repositories (Han et al., 2011; Witten et al., 2011). DM is also called Knowledge Discovery in Database (KDD), which is a process of knowledge extraction. Bose and Mahapatra (2001) define DM as a process of identifying interesting patterns in datasets, which can be used in decision-making. Turban et al. (2011) specify DM as a process that uses some techniques such as statistical, mathematical, artificial intelligence and machine learning to extract useful information from datasets. Phua et al. (2010) state that fraud detection has become one of the best real-world applications of data mining in industry and government.

The general process of data mining involves *data selection*, *data preprocessing*, *data transformation*, *data mining* and *model evaluation or data interpretation*. In general, data mining can solve different kinds of tasks including Rule Mining, Classification, Outlier Detection and Regression (Chen et al., 2014; Han et al., 2011; Witten et al., 2011). Rule mining is used to discover the patterns that can describe the information in a dataset. Classification applies some learn-

ing algorithms (e.g. decision tree learning methods) to build classifiers based on a training dataset, and predict the class label of instances in a testing dataset. Outlier detection is used to detect the instances that have different features than the majority. Regression is a statistical method (e.g. logistic regression), which is used to test a pre-defined hypothesis or reveal relationships between independent variables and dependent variables (Chen et al., 2014; Han et al., 2011; Witten et al., 2011). In the tasks described above, classification is the most popular and frequently applied data mining task. In the classification domain, a number of learning algorithms have been developed to generate classification classifiers. Artificial Neural Networks (ANNs), Decision Trees (DTs) and Bayesian networks (BNs) are well-known and powerful learning algorithms.

1.3 Data Mining Techniques in Financial Fraud Detection

Over the past few years, a number of data mining techniques have been developed and applied in FFD, such as Logistic Regression, Artificial Neural Networks and Decision Trees. Logistic Regression and its variants are the most popular methods used by financial researchers (Chen et al., 2006; Agrawal and Chadha, 2005; Wang et al., 2010; Gillan, 2006; Dyck et al., 2010; Hermalin and Weisbach, 2012; Chen et al., 2005; Cox et al., 2003). It is a generalized linear model, which is also called binomial logistic regression. The predictor variables (i.e. dependent variable) can be either numerical or categorical (Ngai et al., 2011; Spathis, 2002; Yeh et al., 2009). Except for traditional statistical approaches, Artificial Neural Networks (ANNs) might be the most popular data mining technique used in FFD. ANNs imitate the functionality of the human brain by using a number of interconnected nodes with activation functions (Ngai et al., 2011; Haykin, 1994; Yeh et al., 2009). The main advantages of using ANNs are adaptiveness and robustness. The structure (i.e. weights between interconnect nodes) of ANNs can be adaptively changed based on the training data and the performance is relatively stable (i.e. robust). It is widely used in solving FFD problems (Syeda et al.,

2002; Lin et al., 2003; Brause et al., 1999). Decision Trees (DTs) are predictive methods that create tree-based graphs or models from observations to possible consequences to support decision-making (Ngai et al., 2011; Han et al., 2011). The DTs can be generated via different machine learning algorithms. The most popular algorithms used in training DTs include ID3 and C4.5. The constructed DTs are also widely used in credit card and corporate fraud detection (Ngai et al., 2011).

Currently, the methods being used for FFD problems have some issues. We want to study them and try to find out whether it is possible to propose a better method that can be used and evaluated for financial fraud datasets.

1.4 Financial Fraud Datasets

In this study, four datasets were considered in relation to financial fraud detection. Two of them are benchmark problems from UCI machine learning repository (Asuncion and Newman, 2007), which are related to credit card fraud. The other two are real-life problems, concerning China Corporate Securities Fraud and U.S. Corporate Securities Fraud. Three important features should be considered carefully when solving real-life FFD problems. The first is that it is more complicated to solve real-life problems than other simulated or benchmark datasets, because real-life datasets often contain a number of “noisy” instances. They are not consistent with other normal instances in the given data. Therefore, they can disturb the boundary between the different classes and affect the performance of the model. The second is about missing values. In general, real-life datasets contain a number of missing values that may also affect the model performance. The last feature of real-life datasets is about the data distribution. In general, real-life financial data have extremely biased data distribution. As mentioned in Section 1.3, a number of existing popular data mining techniques, such as Logistic Regression, Neural Networks and Decision Trees, can be applied to solve

FFD problems. However, they are not efficient and perform poorly in the given FFD problems, especially for the two real-life datasets. Therefore, we attempted to use other techniques such as Evolutionary Algorithms (EAs) and the idea of ensemble learning methodology to improve the performance in solving the given FFD problems.

1.5 Evolutionary Algorithms (EAs)

EAs have been used to solve classification problems for many years and they have become particularly popular data mining algorithms in producing sets of classification rules (Pappa and Freitas, 2009; Wong and Leung, 2000). EAs are inspired by natural evolution. In nature, weak individuals within their environment become extinct because of natural selection. The strong individuals have greater opportunity to deliver their genes to the next generation via reproduction. In the general process of reproduction, some new individuals become dominant in their population if genes are correctly and successfully combined from strong individuals. At the same time, random changes may occur in their genes, which is called mutation.

In EA, each individual is called a *chromosome*, which is made up of *genes*. Each gene controls one or more characteristics (i.e. the colour of eyes) of the chromosome. EA operates with a set of individuals, called a *population*. In addition, two genetic operators are used to generate new individuals from the current (i.e. existing) generation. In crossover, two individuals are selected, called *parents*, and the new individuals produced by the parents are called offspring. The parents with better fitness values (i.e. strong individuals) are selected from the population with higher preference. Thus new offspring are able to inherit good features or genes from their parents. In mutation, the feature of an individual is randomly changed according to a probability (i.e. mutation rate). The variation of the mutated individual is either huge or small. During the evolutionary process,

weak individuals will be eliminated and strong individuals will survive. After a number of generations, the population evolves with a number of individuals and converges to an overall good solution (Konak et al., 2006). Eventually, the evolved population can be regarded as a model and evaluated by the given datasets.

1.6 Ensemble Learning

In general, it becomes more difficult to have significant improvements in the performance of a single classifier (e.g. logistic regression). It is also difficult to select models if multiple classifiers are developed (Chen et al., 2014). Ensemble learning methods are used to build a predictive model by selecting or integrating multiple models (i.e. classifiers). The main idea behind ensemble learning methods is to assign weights to several individual classifiers and combine them into a classifier that can outperform every one of them (i.e. each individual classifier). In real world applications, when decision makers face several strategies and opinions, ensemble learning methods can be used to select individual opinions and combine them to achieve the final objective (Rokach, 2010). If the individual classifiers in an ensemble are trained from the same algorithm (e.g. logistic regression), then it is called a homogeneous ensemble. Popular homogeneous ensemble learning techniques include Bagging (Breiman, 1996) and Boosting (Schapire, 1990). If the classifiers in an ensemble are trained from different algorithms (e.g. neural networks and decision trees), then it is called a heterogeneous ensemble. Popular heterogeneous ensemble learning methods include Stacking (Wolpert, 1992).

1.7 Research Motivation

In this study, we had four financial fraud datasets. Two of them were benchmark datasets about credit card fraud, and collected from UCI. The other two were real-life problems: U.S. Corporate Securities Fraud (U.S. CSF) and China Corporate Securities Fraud (CCSF). The U.S. CSF dataset was collected from

the Securities and Exchange Commission (SEC). It had 41 attributes and 63 fraudulent instances. The CCSF dataset was collected from the China Securities Regulatory Commission (CSRC). It had 17 attributes and 855 fraudulent instances.

In order to solve these FFD problems, the first thing we needed to consider was what kind of information should be extracted from these datasets. In the research area of Finance, researchers can find different information from the data to examine hypotheses about the factors in relation to committing financial fraud, such as corporate governance (Chen et al., 2006; Agrawal and Chadha, 2005; Hermalin and Weisbach, 2012), investor beliefs about industry business conditions in regard to initial public offerings (IPO) (Wang et al., 2010) and impact of enforcement actions (Chen et al., 2005). In general, researchers prefer to apply traditional statistical methods, such as logistic regression, to achieve their goals. In data mining, some researchers apply some data mining techniques to find out the patterns that can be used to explain the reasons for financial fraud, and then use the discovered patterns to prevent and predict financial fraud. The main difference between the data mining studies on FFD problems is that they apply or propose different data mining techniques to learn models from different FFD problems and evaluate their classification ability. For example, some existing popular data mining techniques on FFD problems include Decision Trees (Kirkos et al., 2007; Kotsiantis et al., 2006), Neural Networks (Lin et al., 2003; Kirkos et al., 2007) and Bayesian Network (Kotsiantis et al., 2006; Kirkos et al., 2007). In this study, we applied the accuracy as the model evaluation criterion to compare the performance of different methods.

In order to study the existing data mining methods, such as Logistic Regression, Neural Networks and Decision Trees, we comprehensively evaluated

them and compared them based on the classification accuracy on the given FFD datasets. In addition, we also evaluated and compared some existing ensemble learning methods, such as Bagging, LogitBoost and AdaBoost, on the given FFD datasets. The preliminary experiments provided an overview of the performance of these techniques. However, they could not handle the FFD problems well, especially the real-life datasets. Most of them performed poorly. Therefore we wanted to develop a new method that can outperform the existing approaches for the given problems. Recent evidence from the literature on data mining shows that some variants of Evolutionary Algorithms (EAs) methods can be applied (Cui et al., 2006; Coello et al., 2007; Ponsich et al., 2013; Deb et al., 2002; Ngan and Leung, 1998). Grammar-based Genetic Programming (GBGP) is one of the most appropriate methods among the variants of EAs (Ngan and Leung, 1998). It can generate a set of classification rules to represent different patterns from the dataset. Compared with other data mining techniques, such as Neural Networks, the generated classification rules are more understandable for general users. We also evaluated GBGP on the FFD datasets. It can produce competitive results among the existing methods and it still has much room to be improved. We think that it is not sufficient to use a single objective (i.e. accuracy) to evaluate the performance of each classification rule. However, the traditional GBGP cannot handle problems with multiple objectives (e.g. accuracy versus interestingness). In order to improve the performance of each rule, multi-objective optimization methods (Konak et al., 2006; Coello et al., 2007; Ponsich et al., 2013; Eskandari and Geiger, 2008; Deb et al., 2002) can be integrated with GBGP to produce a set of non-dominated solutions (i.e. classification rules) on all objectives and then an ensemble learning method be used to make selections from the non-dominated solutions to form a new set of classification rules. Then we can evaluate the final set of classification rules on the given FFD datasets. If we extend GBGP to Multi-objective GBGP and apply the concept from ensemble learning for models (i.e. classification rules) selection, the new method may be able to generate better

classification rules for solving the FFD problems and produce better results than the other existing data mining methods.

1.8 Organization of the Thesis

This thesis is organized as follows. In Chapter 1, the overview, the motivation and the organization of the study are presented. In Chapter 2, the background and literature review of this work is discussed, including data mining and data mining application, data mining techniques in FFD problems, general data mining techniques, Multi-Objective Problems and Multi-Objective Evolutionary Algorithms, Genetic Programming and Grammar-Based Genetic Programming and Ensemble Learning techniques. In Chapter 3, the proposed approach is described in detail. The motivations for some applied methods or components and the framework of the proposed method are discussed. In Chapter 4, a number of conducted experiments are introduced to compare the performance of the proposed method with other data mining techniques. The experiment results are also presented and discussed in the chapter. In Chapter 5, the conclusion is given with our contributions, limitations and the future possible extensions of the study.

CHAPTER 2

Background and Literature Review

The section is organized as follows. Section 2.1 describes data mining and some data mining problems and applications. Section 2.2 describes financial fraud and financial fraud detection problems. Section 2.3 examines a number of data mining techniques that are used for solving financial fraud detection problems. Section 2.4 introduces several popular data mining techniques in detail. Section 2.5 discusses the multi-objective problems and multi-objective evolutionary algorithms (MOEAs). Section 2.6 describes the genetic programming and grammar-based genetic programming (GBGP) in detail. Section 2.7 introduces the ensemble learning techniques.

2.1 Data Mining and Data Mining Applications

Due to the vast amounts of data records rapidly accumulating, it is important but also difficult to observe the useful information from data just by relying on traditional statistical methods. Therefore, the development of new techniques is required to tackle the tasks and this is becoming an attractive research area. The corresponding solutions are called data mining techniques, which can extract implicit, previously unknown and potentially useful patterns, rules or relations from massive data repositories. The discovered patterns maybe very useful for decision makers to apply.

Data mining problems are everywhere. Parsons and Wand (2008) demonstrated the use of cognitive principles in classification methods to determine what the useful classes should be. Before evaluating the method, the authors proposed several initial rules based on their analysis to guide the development of conceptual models, and then the conceptual models will find out which classes are important. Classification rules have several features. For example, they can generate understandable results for analysts in identifying whether a class is important or

not, and the rules are evaluated by a panel of experts.

Huo et al. (2006) proposed a tree pruning method for solving a number of benchmark problems. The proposed method focuses on minimizing a loss function, which is called Complexity Penalised Loss Function (CPLF). The idea of this study is that the system creates a number of linear functions for each node. The coefficient of each linear function indicates the size of its subtree. At the beginning of learning, the size of the whole tree is large, and some of them are redundant (i.e. are not used). By using tree pruning to change the structure (i.e. remove the redundant nodes) of the tree to minimize the loss function, the tree can obtain optimal performance. Meyer et al. (2014) introduced a machine learning technique called Prediction of Control Errors in Dynamic Contexts (PROCEDO) for assisting decision-making. PROCEDO applies a decision tree to discover the conditions under which a dynamic decision-making strategy fails and uses the discovered conditions to improve the decision strategy. The proposed method was tested on two datasets of diabetes patients and it achieved significant improvements to the treatment strategies. Churilov et al. (2005) applied self-organizing maps (SOM) and a clustering optimization algorithm as the data mining techniques to explore the existing classification rules or decision trees provided by experts, to solve a prostate cancer problem. The objective of using SOM is to explore the scores or values of the existing rules based on different attributes in the data, and the goal of using a clustering algorithm is to discover the relationships between patients without any supervision. Moreover, the proposed method can improve the accuracy of risk assessment and support experts to make final decisions. Cui et al. (2006) applied evolutionary programming techniques to evolve Bayesian networks to solve a cost-sensitive problem in direct marketing response models. Direct marketing problems refer to promoting new products or services directly by mail to customers. Companies aim to maximize the response rate and profit from customers due to the budget constraint, instead of maximizing the accuracy of the model in classifying buyers and non-buyers. Thus, mail is sent only

to targeted customers in the database. The proposed method obtains a high level of predictive accuracy compared to neural networks, classification and regression tree (CART) and latent class regression. Padmanabhan et al. (2011) developed a logistic regression method for predicting customer churn problem. Customer churn is an important business problem in competitive industries. It is necessary to understand and discover what factors affect customer churn. The proposed approach was not compared with other methods, but it identified several factors that affect customer churn, such as service quality, customer demographics and behaviour. In addition, it can also predict future customer churn. Due to the fierce market competition, it is important for a fashion company to predict future sales. Sun et al. (2008) investigated the relationships between sales amount and other factors by using a novel Artificial Neural Network (ANN) method. Moreover, a regression integration technique is involved in the ANN method to compute the average predicted sales over a period of time. The experimental results show that the proposed methods outperform some other variants of ANN methods.

2.2 Financial Fraud

Wang et al. (2006) defined financial fraud as “a deliberate act that is contrary to law, rule, or policy with intent to obtain unauthorized financial benefit”. It has always been a very important research topic, and also has attracted a lot of concern. As an increasingly serious problem, financial fraud results in the loss of billions of dollars each year (Kou et al., 2004); therefore financial fraud detection (FFD) is required in order to prevent the destructive results caused by financial fraud. FFD has many common features like other data mining problems. It has drawn a lot of research interest and a number of different techniques from many areas have been applied to tackle this problem. Especially in the field of artificial intelligence, a number of novel and advanced approaches have been developed in financial fraud detection. Ngai et al. (2011) summarized 49 journal articles on the

subject published between 1997 and 2008, and found credit card fraud (14.35%) and corporate securities fraud (34.7%) have attracted a great deal of attention during that period, and are still escalating. In this thesis, credit card fraud and corporate securities fraud are considered comprehensively.

2.2.1 Credit Card Fraud

Credit card fraud concerns the illegal usage of credit cards, such as unusual transactions (Syeda et al., 2002). It is difficult to determine the level of credit card fraud, since banks and companies are reluctant to release fraud figures to the public and these figures are growing over time (Bolton and Hand, 2002). Although these companies and banks lose billions of dollars every year due to credit card fraud, identifying which customers are included in the fraud figures is a complicated task (Bolton and Hand, 2002). With the constant rise of people's consumption standard, the users of credit cards are also increasing rapidly. At the same time, with credit cards being the most popular transaction method, the number of credit card frauds is also increasing. Detecting credit card fraud has drawn a lot of research interest and many different advanced techniques have been developed (Srivastava et al., 2008).

2.2.2 Corporate Securities Fraud

Corporate securities fraud in this study is close to corporate fraud in listed companies, rather than securities fraud only, since the definition of securities frauds includes someone manipulating the securities market, modifying securities accounts or committing wire fraud (Geis and Jesilow, 1993). On the other hand, corporate securities fraud is related to falsification of financial reports, self-dealing by corporate insiders and hiding important information from stakeholders (FBI, 2007). In other words, corporate fraud is closely associated with their own inside problems.

In the U.S., financial analysts have been confirmed to contribute to corporate fraud detection. Effective external monitoring can increase investors' confidence, which is crucial to the functioning of any capital market (Cumming et al., 2011). It is also important for China's securities market, as corporate fraud can impede China's economic development since it has serious consequences for stakeholders, employees and society (Cumming et al., 2011). In recent years, corporate securities fraud detection has become a hot spot domain in finance and there is a wave of research papers that have studied effective policies to detect and reduce fraud.

In China, the Securities Regulatory Commission (CSRC) serves as the main regulator of securities markets in China, which is devoted to investigating the potential violations of securities regulations and instigate different enforcement actions on those fraudulent corporations that have violated the related laws. Any of the enforcement actions by CSRC will affect the stock price of the firm, even resulting in bankruptcy (Chen et al., 2006). Prior studies on the causes of securities fraud have focused on different types of determinants, such as agency problems, business pressures and corporate governance (Agrawal and Chadha, 2005; Wang et al., 2010; Gillan, 2006). In Yu and Yu (2012), the authors investigated the relationship between corporate lobbying and fraud detection. They used lobbying expenses as the learning data, and found that the corporate lobbying could be an important factor in detecting corporate fraud. That is, most fraudulent firms have higher lobbying expenses than non-fraudulent firms. Dyck et al. (2010) deeply analyzed the corporate governance system of many U.S. firms and found Securities and Exchange Commission (SEC) played a very minor role in the discovery process, but analysts, employees and newspapers have strong roles to play in determining whether a firm will commit fraud or not. In Hermalin and Weisbach (2012), the authors discuss the relation between the corporate governance of a firm and information disclosure. The most important finding of the study was that larger firms adopt stricter disclosure rules than smaller firms, and firms with better disclosure rules have capable employees at management level. More-

over, firms with better disclosure rules can probably reduce the incidence rate of outright fraud by insiders. In China, it is also necessary to verify whether larger firms will have less enforcement actions by SEC or not. In addition, Chen et al. (2006) examined these enforcement actions to explain whether the ownership and governance structures of corporations have impacts on committing fraud. The authors concluded that the proportion of outside directors, the tenure of the chairman and number of board meetings are factors related to committing fraud. Farber (2005) examined the association between the financial reporting system and the quality of the corporate governance system. They considered the board members, number of financial experts and number of board meetings in the firm. As found in prior research, poor governance occurs in fraudulent firms. Chen et al. (2005) investigated enforcement actions from the viewpoint of fraudulent firms rather than what factors lead to fraud. They found that many of these firms have problems with published financial statements and irregular reports, such as inflated profit, false statements and major failure to disclose information, which are the common problems identified by CSRC. Considering the laws on federal securities, Cox et al. (2003) examined the four attributes that might associate with fraud including the number of defrauded investors, assets size, losses and financial distress of the firm. The authors concluded that only financial distress has a significant impact on the presence or absence of an enforcement action. In general, since the result of the enforcement action is either yes or no (i.e. 1 or 0), it is more reasonable to use a bivariate probit model as the learning method to analyze the data. There is a large dataset on China's listed firms collected based on the above studies and findings for this research, in order to find out corresponding relationships to detect whether a company is fraudulent or non-fraudulent in China.

2.3 Data Mining Techniques in Financial Fraud Detection

The probit model, logistic regression and their variants are the most popular methods used by financial researchers (Chen et al., 2006; Agrawal and Chadha, 2005; Wang et al., 2010; Gillan, 2006; Dyck et al., 2010; Hermalin and Weisbach, 2012; Chen et al., 2005; Cox et al., 2003). In addition to traditional statistical approaches, there is a number of machine learning techniques applied in solving financial fraud detection problems. Except for regression, neural networks might be the first and also the most popular machine-learning technique used for solving different real-world problems. Brause et al. (1999) applied neural networks for credit card fraud detection. Instead of using resampling techniques for the given unbalanced fraud data, the authors devoted themselves to increasing the inherent correct diagnosis for legal cases from 99.9% to 99.955%. However, this performance measurement may not be the best choice for fraud detection since the accuracy always biased to the majority class (i.e. the class with a higher number of instances). Shen et al. (2007) investigated the efficacy of using decision trees, neural networks and logistic regression for credit card fraud detection problems in order to reduce banks' risk. Moreover, the authors found that the conventional neural networks and logistic regression approaches obtained better results than decision trees. In Chan and Stolfo (1998), the authors used a multi-classifier meta-learning approach for real-world credit card fraud detection. The approach is based on creating data subsets with appropriate class distribution, since most fraud detection problems have unbalanced class distributions. In addition, they applied four learning algorithms (C4.5, CART, RIPPER and BAYES) as the baseline classifiers, and applied BAYES to train the baseline classifiers to generate the final model, which is also called the ensemble-learning technique. The proposed method could handle the learning tasks efficiently. In Srivastava et al. (2008), frauds are detected by using a Hidden Markov Model (HMM) modelling the sequence of actions in credit card transactions. The case is fraudulent if the incoming credit card transaction is rejected by the trained HMM with a threshold

probability. The HMM is able to outperform other conventional techniques, such as neural networks, meta-learning method and Naïve Bayesian networks. Since most fraud detection problems have imbalanced class distributions, Dal Pozzolo et al. (2014) evaluated the performance of using the Undersampling, Synthetic Minority Over-sampling Technique (SMOTE) and EasyEnsemble to solve the problem in the unbalanced dataset, and the latter two methods returned higher accuracies than Undersampling. Furthermore, the authors suggested three incremental learning approaches, called the “static”, “update” and “forget” approaches. The best results were obtained by using the “forget” approach as the incremental learning method with EasyEnsemble. Association rules are also useful classifiers in solving the credit card fraud detection problem. The performance measurement of each rule is determined by a confidence and support framework. A rule with high support value and high confidence value will be ranked as the top level. By using this framework, the results are more straightforward in facilitating fraud analysis (Sánchez et al., 2009). Phua et al. (2010) discuss and compared the performance of a number of different machine learning techniques, such as fuzzy logic, genetic programming and neural networks that are used in credit card fraud detection, and pointed out the advantages of these techniques based on different criteria.

Yuan et al. (2008) applied traditional logistic regression on financially fraudulent listed firms in China from 2002 to 2004. The major findings of the study are helping corporations and auditors to detect financial frauds. For example, it has a negative relationship between the market competition and the probability of the firm to commit fraud in their financial statements. Spathis (2002) applied the same methods, but for Greece listed firms. Kirkos et al. (2007) evaluated the effectiveness of Decision Trees, Neural Networks and Bayesian Belief Networks in detecting and identifying the factors associated with fraudulent financial statements (FFS). In terms of their performance, Bayesian Belief Networks outperform

others in regard to accuracy rate. Spathis (2002) evaluated the performance of a multi-criteria decision aid classification tool in detecting the financial problems from listed firms' financial statements. They selected variables that are often used in the falsified financial statements, such as the ratio of total debt to total assets, sales ratio and net profit. The multi-criteria decision aid classification tool is able to obtain high accuracy in estimating the probability that occurs in fraudulent firms. An improved version of neural networks with fuzzy logic is presented in Lin et al. (2003). The data were collected from the U.S. Securities and Exchange Commission (SEC) between 1980 and 1995 with enforcement actions, but the size was small (i.e. 200 cases in total) and out of date. However, the proposed method outperforms most traditional statistical models and neural networks in previous studies in classifying fraudulent cases. Bai et al. (2008) evaluated the financial problems in China by using classification and regression trees (CART), and compared it with traditional Logit regression only. The CART separated variables into a number of parts. Each part is a decision rule that presents a simple relation of the data. The size of each part is small, so it makes it easy for users to understand. In addition, decision trees are not sensitive to the learning data with outliers since the splitting happens based on different branches of sub-trees. Therefore the outliers can also be classified (Berry and Linoff, 2004). In their study, an improved data representation is introduced to describe each data item easily, and the proposed version of CART produces better results and outperforms Logit regression. Kotsiantis et al. (2006) highlighted the importance of a number of financial ratios, which can significantly determine the classification results. The main method in this study is called stacking ensemble that combines a final classifier from a number of different base-line classifiers. The results of the final classifiers are better than the results of other base-line classifiers.

2.4 Introduction to Data Mining Techniques

2.4.1 Logistic Regression

Logistic Regression (Shmueli et al., 2007) is one of the most widely used traditional techniques for solving data mining problems. It is also a popular technique that is applied as a standard technique to compare with other advanced data mining techniques. It is similar to linear regression, which has a relationship between the predictor variables and response variables. The users have to determine which predictor variables can be selected in the regression. Therefore, logistic regression has a dependent (response) variable, Y , which is categorical. The value of Y is determined by the number of categories. For example, if Y denotes fraudulent and non-fraudulent for a firm, then it has two classes. That is, some firms belong to the fraudulent class and others belong to the non-fraudulent class. It is the same as multiple linear regression, which also includes independent (predictor) variables. The differences are that, logistic regression estimates the probabilities of each class, and uses a cutoff value (e.g. 0.5) to determine whether the class is fraudulent (e.g. greater than 0.5) or non-fraudulent (e.g. smaller than 0.5). The general form with multiple predictor variables can be seen in Equations 1 and 2.

$$\text{logit}(Y) = \ln\left(\frac{\pi}{1-\pi}\right) = \alpha + \beta_0 + \beta_1x_1 + \beta_2x_2 + \dots + \beta_qx_q \quad (1)$$

where

$$\pi = \text{Probability } (Y = \text{class} | X_1 = x_1, \dots, X_q = x_q) = \frac{e^{\alpha + \beta_1x_1 + \dots + \beta_qx_q}}{1 + e^{\alpha + \beta_1x_1 + \dots + \beta_qx_q}} \quad (2)$$

2.4.2 Neural Networks

Artificial neural networks (ANNs) are very popular techniques in classification problems. The multilayer perceptron (MLP) is a feedforward artificial neural network model that maps sets of input data onto a set of appropriate outputs. Neural networks emulate the functionality of the human brain by using a set of

weighted edges and neurons. These neurons are processing units and interconnected with associated weights and distributed in a number of layers. In general, most neural networks contain three types of layers, which are input, hidden and output. One example (Haykin, 1994) is shown in Figure 1.

Figure 1. Example of a multilayer perceptron neural network with three layers

After each neuron in the hidden layer receives the signals from input layer neurons, the neuron (receiver) sums the total received values and compares them to the threshold value by using a sigmoid function. The sigmoid function is shown in Equation (3). The neuron (receiver) will be active (i.e. exceeds the threshold) and sends a value to other neurons in the output layer; otherwise it will not be active (Witten and Frank, 2005; Haykin, 1994).

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

In the output layer, each neuron receives the values from previous layers and sums them all up. The actual output is compared with the desired output to obtain a value of error (differences). Errors are iteratively propagated backwards in the network in order to re-modify the weights coming to the output layer by using a weight updating equation in order to calculate the output errors (Witten and Frank, 2005). Equation (4) is one of the error functions called Sum Squared Error (SSE) function (Haykin, 1994) used to quantify the difference between the

target and real output.

$$E_{sse} = \frac{1}{2} \sum_i \sum_j (target_{ij} - output_{ij})^2 \quad (4)$$

where $output_{ij}$ is the real output values generated by the current neural networks from node i (i.e. training patterns) to node j (i.e. output units), and $target_{ij}$ is the desired outputs (i.e. the real class values from the training data). In general, the good neural networks should often have very low such errors. Therefore, the purpose of learning is to minimize the error function by changing the weights between the nodes, i.e. W_{ij} . The weights can be adjusted by Equation (5), called the weight updated function (Haykin, 1994).

$$\Delta w_{ij}(t) = -\eta \frac{\partial E}{\partial w_{ij}} + \alpha \Delta w_{ij}(t-1) \quad (5)$$

where η is a positive value called learning rate, which specifies the portion of the adjustment to the old weight. It may not converge if the learning rate is too large, say 0.95. For example, the current neural networks can produce very low errors; if η is large, the performance of the updated neural networks can be worse. In order to avoid a large change, a small η is needed. Then if the neural networks are good enough, they cannot be changed significantly. On the other hand, it may converge extremely slowly with a very small learning rate, say 0.0001. To set an appropriate value for η is not easy. In general, the η can be large at the beginning, but needs to be reduced depending on the problems (Haykin, 1994).

The process to train a neural network can be divided into six steps. First, prepare data for the neural network to learn with input and output values, e.g. $\{input_i, output_j : i = 1, 2 \dots n, j = 1, 2 \dots n\}$. Second, create a neural network with several input neurons, a layer of nonlinear hidden units and two output units (i.e. for binary problem), and fully linked to each other with connection weights w_{ij} . Third, generate random initial weights from range -1 to 1. Fourth,

select an error function as shown in Equation (4) and value for learning rate (e.g. 0.2). Fifth, apply a weight update equation as shown in Equation (5) to change weights for each connection. Last, repeat the previous step until the termination criteria are reached (e.g. until no error changes). A trained neural network can be used to explore the patterns within the training instances; then it can predict the target value given new testing data (Haykin, 1994).

2.4.3 Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a very powerful tool for solving many data mining problems. The general idea of a SVM is to find out a small number of cases to form a critical boundary line called support vectors for each class and generate a linear discriminant function to separate the data with the largest distance. When a new instance is coming, the side of its location will determine its class (Witten and Frank, 2005; Platt, 1998). An illustration is shown in Figure 2.

Figure 2. Example of a SVM

There is a line between two classes, which is a linear SVM, and the formula is shown in Equation (6).

$$u = \vec{w} \cdot \vec{x} - b \quad (6)$$

where $\vec{w}$ is the normal vector to the hyperplane (classifier) and $\vec{x}$ is the input vector. The distance of the nearest points to the hyperplane is when $u = \pm 1$. The formula of margin m is shown in Equation (7).

$$m = \frac{1}{\|\vec{w}\|^2} \quad (7)$$

It is necessary to maximize the margin to obtain the largest distance. Thus an optimization problem is formulated in Equation (8).

$$\min \frac{1}{2} \|\vec{w}\|^2 \quad \text{subject to} \quad y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1. \quad (8)$$

where $\vec{x}_i$ is the i^{th} training example, and y_i is the correct output variable. It has two values, which are +1 for positive examples and -1 for negative examples. In addition, the optimization problem can be converted to a dual form by using some Lagrangian coefficients, and it becomes an optimization problem, which is shown in Equation (9).

$$L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \mathbf{x}_j) \quad (9)$$

where N is the number of training examples, and $\alpha_i, 1 \leq i \leq N$ are the Lagrange coefficients. Once the Lagrange coefficients are determined, the value of $\vec{w}$ and threshold b can be calculated from the Lagrange coefficients, which is shown in Equation (10).

$$\vec{w} = \sum_{i=1}^N y_i \alpha_i \vec{x}_i, \quad b = \vec{w} \cdot \vec{x}_k - y_k \text{ for some } \alpha_k > 0. \quad (10)$$

After obtaining the value of $\vec{w}$ and b , the discriminant function shown in Equation (6) can be solved (Platt, 1998).

2.4.4 Bayesian Networks

As one of the most effective classifiers, Bayesian Networks (BNs) have been widely used for solving data mining problems. They are based on the Bayesian theorem and aim to estimate the probabilistic relationships among a number of variables from data. There are several advantages of using BNs (Heckerman, 1997). First, they can handle datasets with missing instances easily. Second, they can learn the causal relationships to discover the knowledge of the data and predict unknown cases. Third, it can be easily combined with other techniques, such as Bayesian statistical techniques, to provide efficient models for learning the data (Heckerman, 1997). Figure 3 is a simple Bayesian Network.

Figure 3. Simple Bayesian Network

In Figure 3, the nodes with no children are called leaf nodes, which are independent of each other. Each leaf node indicates an attribute of the data and the root node is a class attribute. If there is a directional connection from node A to B that indicates that node A is the parent node of node B, it is denoted by $Pa(B)$. The name of variables are presented by capital letters such as X,Y,Z, and their values are presented by lower case letters such as x,y,z. P is denoted as a joint probability distribution among the variables. In general, a Bayesian Network consists of two components (Heckerman, 1997). First, there is a directed acyclic graph with a number of nodes, denoted by G. The directional connections present the conditional independence between variables (i.e. nodes). Second, the conditional probability between each node and its parents is denoted by $P(X_i|Pa_i)$. Therefore, the BNs can calculate the conditional probability for

each node to show the degree of dependence on its parent node. Formally, a Bayesian Network can be expressed by Equation (11).

$$P(X_1, \dots, X_N) = \prod_{i=1}^n P(X_i | Pa_i) \quad (11)$$

where Pa_i indicates the parent nodes of X_i .

2.4.5 Classification and Decision Trees (CARTs)

CART is a tree-building technique, which is different to the traditional data mining methods. It is a binary recursive partitioning procedure for processing nominal attributes or continuous variables by using classification and regression trees respectively (Steinberg and Colla, 2009).

Breiman et al. (1984) proposed the classical CART algorithm, which includes two binary decision trees: regression and classification. Regression trees are appropriate if the dependent variables are continuous, which can be any value over a range of observations. Classification trees are appropriate if the dependent variables are nominal (i.e. named) attributes. For example, we want to know which firm will or will not commit a fraud. These could be examples of binary classification problems, since the categorical dependent variable can only be two distinct and mutually exclusive values. Therefore, the type of dependent variable determines which type of model will be chosen (i.e. regression trees or classification trees). The regression trees cannot be used for categorical data, since there are only two values (e.g. binary classification) in the dependent variable. However, the classification tree can be applied to continuous data by generalizing the data into two types (e.g. binary classification) (Xu, 2006).

Decision trees are used to represent decision processes for classifying patterns

in data. The node at the top of a decision tree is called the root node. Starting from the root node, the data flow “down” a tree and is split into children (i.e. nodes), in which case the original node is called the parent node. Decision trees grow downwards and terminate when no further splits are possible from the data. In most general terms, the objective of using decision trees is to determine a set of logical condition of the “if-then” form to classify or predict the cases. Figure 4 illustrates the splitting process.

Figure 4. Illustration of the decision tree splitting process

Initially, the tree starts with a number of training cases in which the label (e.g. fraudulent vs. non-fraudulent) is known for each case. All of the training cases are grouped together at the root node. The algorithm will separate them into two groups on the basis of a splitting line in that variable (i.e. $\text{employees} \leq 10,000$ and $\text{employees} > 10,000$). The splitting is then applied again to each of the new separations until no further splits are possible (Xu, 2006).

There are four steps in decision tree analysis. The first step consists of “tree building”. It starts at the root node that includes all the training instances. The decision tree finds the best possible attribute to split the node into two children. All possible attributes will be checked in order to find the best one. After finding the best attribute, an appropriate splitting line is also required. In general, “Gini Index” and “Least Squares” are the most common splitting functions that are

used for classification trees and regression trees respectively. The second step consists of “stopping tree building” the tree will terminate if no further splits are possible. The tree building process is terminated if one of the following conditions is satisfied: (1) There is only one observation in each of child node; (2) The splitting is impossible when all training cases in each child node have the same distribution of independent variables. (3) It reaches the predefined maximal levels (i.e. the depth) of the tree. The third step consists of “tree pruning” a number of decision trees are generated, and each of them represents a candidate for the final tree. It uses a parameter called α to measure the complexity of each candidate, and α is initialized to zero and gradually increased during the tree pruning process. The tree pruning process starts from the terminal nodes at the last level of the tree, and nodes are pruned away if the misclassification cost after pruning is less than the α multiplied by the changes in tree complexity. In other words, the nodes can be removed if there is no significant change after pruning away the nodes. The higher value of α indicates the simpler decision tree. The last step consists of “optimal tree selection”. In general, the maximal tree should have higher accuracy than any other tree. However, the optimal tree should not only have high accuracy, but also have correct complexity level α , because the tree with very low α may have an over-fitting issue. Generally, using an independent set to evaluate the candidates can select the optimal tree. Eventually, the selected tree can have the best performance (Xu, 2006).

2.5 Multi-Objective Problems and Multi-Objective Evolutionary Algorithms

Many real-world problems can be regarded as optimization problems. It is always difficult to obtain a single solution for a multi-objective problem. For example, if a stock buyer tries to select a stock with the highest expected return, it may not be very difficult. However, there is a relationship between return and risk, which means that the stock, which the buyer selected may also have very

high risk. If the buyer is also concerned about the risk of the selected stock, then it will have other choices with lower risk. In general, researchers search for a single solution by assigning different weights for each objective. The weight is used to describe the importance of the objective. For example, if users want to optimize Equation (12)

$$y = \text{Maximize}(w_1 \cdot \text{objective}_1 + w_2 \cdot \text{objective}_2) \quad (12)$$

where w_1 is the weight for objective_1 , w_2 is the weight for objective_2 , and $w_1 + w_2 = 1$, the first issue is to assign the value for each weight. If the users think objective_1 is more important than the other, then the weight of w_1 should be higher than w_2 . However, what exact value of w_1 should be assigned is difficult. It will affect the results. In addition, if there are three more objectives, the problem of assigning values for weights becomes more difficult. Therefore, methods to solve multi-objectives optimization problems are required.

Multi-objective optimization problems are also called Multi-criterion optimization. The main purpose of this method is to find a number of trade-off solutions (i.e. Pareto solutions or non-dominated solutions), which can meet all criteria, and then the users can make a final decision based on the optimal solutions (Coello et al., 2007). The optimal solutions form a curve among all objectives, which is shown in Figure 5.

Without loss of generality, we define multi-objective maximization problems by Equation (13).

$$\text{maximize } f(x) = [f_1(x), f_2(x) \dots f_k(x)] \quad (13)$$

subject to

Figure 5. Example of Pareto Front

$$g_i(x) \leq 0 \quad i = 1, 2, \dots, m \quad (14)$$

$$h_j(x) = 0 \quad j = 1, 2, \dots, p \quad (15)$$

where $x = [x_1, x_2, \dots, x_n]$ is a vector of n decision variables, $f(x) = [f_1(x), f_2(x) \dots f_k(x)]$ is called objective functions, and $g_i, h_j, i = 1, 2, \dots, m$, and $j = 1, 2, \dots, p$ are the constraint functions of the problem. The number of p (i.e. equality constraints) must be smaller than the number of n (i.e. objectives); otherwise the problem is over-constrained (i.e. no solutions) (Coello et al., 2007).

Evolutionary Algorithms (EAs) are very appropriate and have been widely used to solve multi-objective problems. The advantage of using EAs is that they can operate a number of solutions (i.e. individuals) and find out which of the them are the non-dominated solutions easily, instead of using traditional mathematical programming techniques to handle the calculations separately (Coello et al., 2007). The methods with EAs are called Multi-Objective Evolutionary Algorithms (MOEAs). MOEAs have solved many multi-objective optimization problems in different domains, such as engineering, and scientific and finance

fields(Coello et al., 2007). There are many state-of-the-art MOEAs. In general, they have two major components, which are individual selection scheme and diversity maintenance scheme (Ponsich et al., 2013). The first scheme is to select the optimal solutions among all the objectives, and the second scheme is to avoid Pareto solutions converging to a single solution. As shown in Figure 5, the Pareto solutions are located along the curve (i.e. Pareto front) rather than only a single solution on the curve. A diversity maintenance scheme is required, such as crowding distance (Deb et al., 2002). The crowding distance is the metric to calculate the distance between a solution and others (Deb et al., 2002). Therefore, a diverse population should have large crowding distances between the non-dominated solutions. Eskandari and Geiger (2008) developed a fast Pareto genetic algorithm approach (FastPGA) to find out the non-dominated solutions. The fitness value of each solution is based on the crowding distance with other solutions, and then FastPGA ranks the solutions based on their fitness values. The proposed method obtains similar performance to the non-dominated sorting genetic algorithm (NSGA-II) (Deb et al., 2002). Nebro et al. (2009) proposed a cellular genetic algorithm called MOCeII to solve the multi-objective problems. The main idea of the study is to randomly replace individuals in an external set, which is used to store the non-dominated solutions obtained in that time. In addition, the genetic operators only operate a pair of individuals, which have the smallest crowding distance (i.e. neighbours). Zhang and Li (2007) designed a MOEA method and used an aggregation method to separate the multi-objectives into several small groups with a number of individual objectives, and then solved each small group simultaneously. Therefore, it has lower computational complexity due to the advantage of small group operation compared to other MOEAs such as NSGA-II (Deb et al., 2002). Fonseca et al. (1993) proposed a MOEA method, which is widely used in solving multi-objective problems. The method does not use the elitist scheme, which keeps the best solution for the next generation. The advantage of using the elitist scheme is to reduce the converge time; at

the same time it is easy for an over-fitting problem to occur. Instead, the authors designed a novel ranking scheme to use an extra parameter, say α , for each solution. The α is 1 if the solution belongs to the Pareto front (i.e. non-dominated solution). For other dominated solutions, the values of α are determined on the basis of their density. Deb et al. (2002) proposed one of the most widely used MOEA, called Non-dominated Sorting Genetic Algorithm (NSGA-II). NSGA-II has several advantages compared to the original NSGA (i.e. NSGA-I) (Srinivas and Deb, 1994). For a single solution, NSGA-II calculates the number of solutions dominating it, and the number of solutions dominated by it to determine whether it is a non-dominated solution or not. In addition, it also calculates the crowding distance for each solution to measure the density of solutions surrounded by each solution. Simulation results show that the efficiency of NSGA-II has much better performance than other constrained multi-objective optimizers on a number of test problems. Zitzler et al. (2001) developed an improved Strength Pareto Evolutionary Algorithm (SPEA2) to solve multi-objective problems. It updates the external population with observed Pareto solutions at each generation in order to maintain the diversity of the Pareto solutions. The diversity maintaining scheme is also based on estimating the neighbourhood density for each solution. In addition, a value to measure the strength of each Pareto solution in the external population is calculated, which is based on the number of solutions it is dominated by others and the number of solutions in which it dominates others. The SPEA2 is better than NSGA-II on some testing datasets, but not all of them. Ghosh and Nath (2004) developed a multi-objective rule learning system by using genetic algorithms. The objectives are quite different from the other studies discussed above. There are three objectives: *support count*, *comprehensibility* and *interestingness*. *Support count* measures the number of instances that a rule can cover (i.e. both antecedents part and consequence part). *Comprehensibility* indicates the number of attributes included in a rule. *Interestingness* describes the interestingness of a rule. Their proposed method has been evaluated

on a market-basket dataset, and it can obtain around 2% Pareto solutions of the whole population (Ghosh and Nath, 2004).

2.5.1 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

There are two main features in NSGA-II (Deb et al., 2002). The first feature is to *sort* the individuals into different level of fronts (i.e. rank). The individuals in the first front are the non-dominated solutions (i.e. Pareto solutions). The individuals in subsequent ranks are poorer than the individuals in previous ranks. The algorithm can be found in algorithm 1. In algorithm 1, it compares each individual with all others to determine whether it can dominate them or not. After a number of comparisons for each individual p , two important values will be assigned. The first is about the set of individuals that is dominated by individual p (i.e. S_p). For example, if there are 10 individuals, and only the first one is the non-dominated solution (i.e. highest value on support and confidence) the set of individuals that is dominated by the first solution is the other nine individuals. The second is about the number of individuals that dominate the individual p (i.e. n_p). In the previous example, there are no other individuals better than the first non-dominated solution; therefore its n_p is zero. The process is repeated until all individuals are evaluated. Finally, each individual is sorted in descending order into different front levels according to its n_p . For example, if the n_p of an individual is zero, which is the lowest, then it is sorted at the first front. Other individuals with the second lowest n_p will be sorted at the second front.

The second feature is to measure the *crowding distance* between each pair of two individuals that are the neighbour of each other, which is shown in Figure 6.

In Figure 6, the crowding distance $i_{distance}$ of individual i is the average side length of the cuboid (Deb et al., 2002). It is an important measurement when

Algorithm 1 Fast non-dominated sorting algorithm

Require: population (P)

```
1: for each  $p \in P$  do
2:    $S_p = \phi$  {Set of solutions dominated by p}
3:    $n_p = 0$  {Number of solutions which dominate the p}
4:   for each  $q \in P$  do
5:     if p dominates q then
6:        $S_p = S_p \cup q$ 
7:     else if q dominates p then
8:        $n_p = n_p + 1$ 
9:     end if
10:  end for
11:  if  $n_p = 0$  then
12:     $p_{rank} = 1$ 
13:     $F_1 = F_1 \cup p$ 
14:  end if
15:   $i = 1$  {Increase the front counter}
16: end for
17: while  $F_i \neq \phi$  do
18:    $Q = \phi$ 
19:   for each  $p \in F_i$  do
20:     for each  $q \in S_p$  do
21:        $n_q = n_q - 1$ 
22:       if  $n_q = 0$  then
23:          $q_{rank} = i + 1$ 
24:          $Q = Q \cup q$ 
25:       end if
26:     end for
27:   end for
28:    $i = i + 1$ 
29:    $F_i = Q$ 
30: end while
```

Figure 6. Crowding distance calculation

two individuals are selected from the same front. Then the individual with a larger crowding distance will be selected, since an individual with a large crowding distance can have better diversity. Diversity is important in all evolutionary algorithms. Without diversity, all individuals in the population are similar to each other. A crossover operator will not work because it always selects similar individuals and produces similar individuals. Mutation can increase the diversity but not fundamentally. Therefore, the algorithm can maintain the diversity adaptively based on crowding distances. Algorithm 2 shows the crowding distance calculation.

Algorithm 2 Crowding distance calculation (I)

Require: solutions (I)

- 1: $l = |I|$
 - 2: for each i , set $I[i]_{distance} = 0$ {initialize all distances to zero}
 - 3: **for** each objective m **do**
 - 4: $I = \text{sort}(I, m)$ {sort the solutions based on the objective}
 - 5: $I[1]_{distance} = I[l]_{distance} = \infty$ {set boundary to infinite}
 - 6: for $i = 2$ to $(l - 1)$ {for all other points}
 - 7: $I[i]_{distance} = I[i]_{distance} + (I[i + 1].m - I[i - 1].m) / (f_m^{max} - f_m^{min})$
 - 8: **end for**
-

2.6 Genetic Programming and Grammar-Based Genetic Programming

2.6.1 Genetic Programming (GP)

Evolutionary Algorithms (EAs) are generic population-based optimization techniques inspired by natural evolution (Wong and Leung, 2000). EAs have several variants, such as Genetic Algorithm (GA) and Genetic Programming (GP) (Keane et al., 2006). They all have a pool of individuals, which can be numbers, functions or models. The main difference between GA and GP is the representation of individuals. In GA, each individual is a chromosome of a fixed length, i.e. the length of individuals cannot be changed during the evolution. Moreover, the process in GA uses the *encode* scheme which converts the information of the individual when producing new individuals (i.e. offspring), and then uses the *decode* scheme to transform the generated individuals back to their original form. For example, if the given problem aims to maximize a function $y = x^2$, firstly, a pool of different values (i.e. individuals) for variable x is randomly generated, e.g. 13, 24, 8, 19. These numbers have decimal format, which are the original form. A simple *encode* scheme converts decimal to binary. For example, if 13 and 24 are selected as parents, their encoded binary strings with size 5 are 01101 and 11000 respectively, which are also called *genes* of the chromosome. If the crossover operator only swaps the first position from both strings, then the new strings become 11101 and 01000. The new individuals have to be decoded to decimal format again, which are 29 and 8 respectively. The fitness evaluation scheme will calculate the score (i.e. the value of y) of these two new individuals, and obtain $y=841$ when $x = 29$, and $y = 64$ when $x = 8$. Therefore the fitness value of individual $x=29$ is the highest in the generation.

Genetic Programming (GP) has a tree structure in the representation of an individual. Other EAs do not have similar structure like GP, but the general evolutionary process of GP is similar to other EAs, as shown in Figure 7.

Figure 7. Simple GP flowchart

It creates a number of computer programs (i.e. individuals), which are composed of functions and terminals to form an initial population (generation 0). Some possible computer programs with a function set and terminal set are shown in Figure 8. In Figure 8, the terminal set is $\{0,1,2,-1,-2,x\}$, and the value of each terminal node is selected from the terminal set, which is located in the leaves of individual program trees. For example, in Figure 8 (a), $x, 1$ and 0 are the terminal nodes. The function set is $\{+,-,*\}$, and the value of a function node is selected from the function set, which is the connector of two nodes, such as $-$ and $+$ in Figure 8 (a). In Figure 8, four computer programs are randomly generated from the terminal set and the function set (Keane et al., 2006).

Figure 8. Four individuals in GP

Afterwards, it iteratively selects some individuals based on their fitness values and breeds them into a new generation of individuals by using different genetic operators. Fitness value is the score to measure the quality of the individual,

which means that a good computer program has a high fitness value. The selection is based on fitness values; therefore poor computer programs have lower probability of being selected. When one or two individuals are selected, genetic operators will be applied to produce new individuals. Genetic operators include crossover, mutation and reproduction. As shown in Figure 9, crossover is executed with a probability to swap a subtree from one individual with a subtree from another individual once two individuals are selected (Keane et al., 2006).

Figure 9. GP crossover

It is also straightforward for mutation and reproduction which is shown in Figure 10. The former is executed with a probability to change a subtree to a randomly generated subtree, and the latter is to reproduce itself.

Figure 10. GP mutation and reproduction

The new individuals (i.e. offspring) will replace some individuals (i.e. parent) according to the altering scheme at each generation. In general, poor individuals have high probability of being replaced by new individuals. The evolution is repeated until the termination criterion is satisfied, e.g. number of iterations. Finally, the evolved population contains a number of good individuals to solve the given problem (Keane et al., 2006). In order to apply GP for a problem, the user needs to pre-define a number of parameters, which include a set of primitive

functions $\mathbf{F}$, a set of terminals $\mathbf{T}$, a fitness function, a set of related parameters for evolution (e.g. crossover rate, mutation rate and selection rate) and the termination criteria (Wong and Leung, 2000).

2.6.2 Grammar-Based Genetic Programming (GBGP)

Comparing Grammar-based Genetic Programming (GBGP) with traditional GP, the concept of grammar is employed. Using grammar can provide a powerful mechanism to describe many complex structures, such as languages, graphs and mathematical expressions. As a component in GBGP, grammar can be very effective to evolve a population. By using grammar, GBGP can generate individuals with desired structure and avoids occurrence of invalid genetic operations during the evolution. In other words, a grammar guides the evolution on what can be done and what should be done. GBGP supports logic grammars, context-free grammars (CFGs) and context-sensitive grammars (CSGs) to generate tree-based programs (Wong and Leung, 2000). An appropriate grammar can be used to solve a particular problem.

<i>Expression</i>	$\rightarrow$	<i>Boolean</i>	<i>Yes</i>	<i>No</i>
<i>Boolean</i>	$\rightarrow$	<i>Operator</i>	<i>Term</i>	<i>Term</i>
<i>Boolean</i>	$\rightarrow$	true false		
<i>Term</i>	$\rightarrow$	term1 term2 term3		
<i>Term</i>	$\rightarrow$	0 1 2 3 4 5 6 7 8 9		
<i>Operator</i>	$\rightarrow$	= >= <= > <		
<i>Yes</i>	$\rightarrow$	yes		
<i>No</i>	$\rightarrow$	no		

Table 1. Simple example of a context-free grammar.

In order to have a better understanding about the designed grammar, the simple example in Table 1 can be used to illustrate the idea of using grammars. By using grammars, only valid programs can be generated during the evolutionary process. Table 1 is an example of a context-free grammar. Expression is the start symbol. The items with capital letters are the non-terminal symbols, and others are the terminal symbols. Each statement indicates a rule with the form $\alpha \rightarrow \beta$

to show how a non-terminal symbol is expanded to a number of non-terminals or terminal symbols. The representation of an individual in GBGP is a tree-based structure. The root node of an individual is the start symbol of the grammar. Figure 11 is an example of an individual in GBGP (Ngan and Leung, 1998), which is generated by using the grammar in Table 1. The evolutionary steps of GBGP are similar to traditional GP, but the genetic operations are different, especially for a crossover. It is different to the example shown in Figure 9; the selected nodes can be swapped if the new individual fulfils the given grammar. There is a mechanism to check the validation of the new individual before making a genetic operation. In order to apply GBGP for a problem, the user also needs to pre-define a number of parameters. In addition to the parameters used in GP, a problem-specific grammar is required (Wong and Leung, 2000).

Figure 11. Individual program in Grammar-based GP represents if $\text{term2} > 4$ is true then yes; otherwise no

In recent years, many modified versions of GBGP have been developed to solve different kinds of problems. McKay et al. (2010) investigated a number of GBGP variants. They applied different grammar representations, used different search strategies and solved several different problems. Knowledge discovering problems are widely discussed in the survey (McKay et al., 2010). Whigham (1995); Geyer-Schulz and Geyer-Schulz (1997) and citetngan1998using applied context-free grammars (CFGs) to evolve a population with derivation trees for learning knowledge rules. Wong and Leung (1997) applied logic grammar (DCGs) to generate programs to learn logical functions such as even-n-parity problems. Wong and Leung (2000) developed a GBGP called LOGENPRO for handling

several data mining problems, such as medical knowledge discovery. O’Neil and Ryan (2003) applied Grammatical Evolution (GE) for predicting real-life trading market indices. The data were collected from the UK FTSE 100, German DAX and Japanese NIKKEI stock exchange. The GE produced superior performance in the FTSE and NIKKEI, but poor in the DAX market. The authors found that over-fitting of the evolved individuals was the main reason.

2.7 Ensemble Learning

It becomes more and more difficult to improve the performance of a single classifier significantly. Moreover, it is also not sufficient to apply a single classifier to solve a problem. Ensemble approaches are introduced as learning algorithms that can select a set of classifiers and then classify the unknown data (Dietterich, 2000). The ensemble techniques are proven empirically and theoretically to give better results than any single classifier in most cases (Chen et al., 2014). There are two levels in learning an ensemble. The first level is called base-level, where the base learning algorithms are applied to learn base-level classifiers from the training data. The second level is called meta-level, where an algorithm on a model is used to combine the outputs from the base-level classifiers. If the classifiers in an ensemble are trained from the same algorithm (e.g. neural networks), then it is called a homogeneous ensemble. Popular homogeneous ensemble learning techniques includes Bagging (Breiman, 1996) and Boosting (Schapire, 1990). If the classifiers in an ensemble are trained from different algorithms (e.g. neural networks and decision trees), then it is called a heterogeneous ensemble. The popular heterogeneous ensemble methods include Stacking (Wolpert, 1992).

2.7.1 Bagging

Bagging is one of the most well-known independent ensemble methods, which is also called bootstrap aggregating. It aims to increase accuracy by selecting a

number of classifiers and forming them into a single predictor. The general process of bagging is to generate a number of subsets with training instances, which are randomly sampled with replacement from the original training set. Therefore, some of the instances from the original training set will appear more than once in each subset, and some of the instances may even not be used. The classifiers in the base-level are trained by the same learning algorithms, such as decision tree method, but they are trained by using different training subsets. Each base-level classifier returns a predicted value, and the bagged classifier combines all values by using a voting method (i.e. most often predicted values) to determine the final prediction (Rokach, 2010).

2.7.2 Boosting

Boosting uses a voting scheme to combine the results from all base-level classifiers. New base-level classifiers are trained iteratively by changing the distribution of the training dataset based on previous misclassified instances. That is, the misclassified instances have higher probability (i.e. weights) of being selected for training. On the other hand, the weights of correctly classified instances are decreased. The boosted classifier combines the predictions from all base-level classifiers to generate the final prediction. Except for using a majority voting scheme to determine the final decision, several other schemes can be applied. For a binary classification problem, the boosting method can be regarded as an approximation method like logistic regression. A logitboost was developed by (Friedman et al., 2000) to learn the base-level classifiers in order to achieve better performance. Each variable in the logistic regression is replaced by a base-level classifier in the logitboost method. Therefore, the result of the ensemble provides a probability of the prediction for each class, and it is widely applied to solving ensemble learning problems. In addition, Adaboost is another popular method extended from the boosting scheme (Freund et al., 1996). The main improvement is to consider the classification error of the base-level classifiers. Each base-level

classifier has a weight to measure the importance. All weights are equally assigned at the beginning of the learning process. And then each base-level classifier will have a classification error based on the training data. The weights are updated based on their classification errors, and also the probabilities of the misclassified instances are increased. When all classifiers are learnt, the ensemble will make a final prediction based on their weights (Freund et al., 1996).

2.7.3 Stacking

Compared to Bagging and Boosting methods, Stacking is concerned with using several classifiers obtained from *different* learning algorithms. The instances s_i can be represented by (x_i, y_i) , where x_i is the vector of the training data, and y_i is the classified value. In base-level, a number of classifiers $C_1, \dots, C_N$ are generated. In meta-level, a classifier is learnt with meta-data with the form of $((y_i^1, \dots, y_i^n), y_i)$, where y_i^n is the prediction of the n^{th} base-level classifier on the i^{th} instance and y_i is the actual label of the instance. Therefore, a meta-level classifier applies the outputs from all base-level classifiers to generate the final prediction (Rokach, 2010). The challenge of using stacking is to determine the number of base-level classifiers as well as which classifiers should be included.

CHAPTER 3

Grammar-Based Multi-objectives Genetic Programming with Ensemble Learning

As described in Section 2.2, financial fraud is a criminal act and an increasingly serious problem. Each year, billions of dollars are lost through failure to detect financial fraud. Financial Fraud Detection (FFD) is required to prevent destructive activities. FFD is challenging, as several issues should be considered. Firstly, it is more complicated to solve real-life problems than other kinds of problems (e.g. benchmark or simulated data sets), because real-life datasets often contain a large number of instances, and several of them are “noisy” data, which may affect the model performance.

Figure 12. Example of “noisy” data

For example, in Figure 12 (a), black points indicate the fraudulent instances, and white points indicate the non-fraudulent instances. Point A is regarded as the outlier or “noisy” instance in the data since it should belong to the area of black points. In addition, it may affect the generalization performance of a classifier, if the classifier over-learns the outlier (Ripley, 1996), which is shown in Figure 12 (b). When such classifier is applied and evaluated on a testing dataset, some white points (i.e. non-fraudulent instances) cannot be correctly classified because of the poor generalization performance. Secondly, when using some modelling approaches (i.e. Neural Networks), the results are not straightforward for general

users to understand. Thirdly, except for noisy data, real-life financial data have other features, such as biased data distribution. We have collected four FFD datasets: two benchmark datasets and two real-life datasets. In order to study the existing data mining methods, we evaluate them using these datasets. As we have observed from the results of preliminary experiment, they perform poorly and some of them generate biased results. Therefore, we want to propose a new method that can be used to handle the given FFD problems.

The general procedure of the proposed method for solving financial fraud problems is shown in Table 2.

The input to the system:	
•	Datasets: Training instances.
•	Objectives: A number of objectives, and maximization or minimization of each objective.
•	Pre-defined grammar for the specific problem.
•	Parameters for evolution: Number of generations and number of individuals.
•	Ensemble learning technique.
1.	Grammar-based Multi-objective Genetic Programming (GBMGP):
•	Applying genetic programming to the classification rules.
•	Training by pre-defined grammar.
•	Output the best population with evolved individuals.
2.	Generating ensemble:
•	Applying ensemble approach for the population.
3.	Testing:
•	Applying the final ensemble on the testing set.

Table 2. General procedure of using GBMGP with an Ensemble Learning technique

Three major components are included in this framework. The first consists of Grammar-based Multi-objective Genetic Programming (GBMGP), which is described in Section 3.1. The second consists of ensemble learning, which is described in Section 3.2. The third consists of minority prediction in model testing, which is discussed in Section 3.3.

3.1 Grammar-based Multi-objective Genetic Programming (GBMGP)

This subsection describes the key components of the GBMGP and corresponding motivations, designs and implementations in detail.

Figure 13. General process of GBMGP

Figure 13 shows the general process of GBMGP. Compared with the general process of Genetic Programming (GP), GBMGP has three more components, which are Step 4, Step 5 and Grammar framework, to handle the multi-objective problems, maintain the diversity of base classifiers and control the evolutionary process respectively. A famous multi-objective learning algorithm called Non-dominated Sorting Genetic Algorithm II (NSGA-II) is applied in our multi-objective optimization procedure (i.e. Step 4). Another diversity maintenance scheme called Token Competition is applied in Step 5. Other procedures are similar to traditional GP. Before the evolution starts, the representation of the base classifiers should be determined first (i.e. Step 2). Section 3.1.1 describes the representation of the base classifiers at the beginning of the evolutionary process. Section 3.1.2 shows how Grammar-based Genetic Programming (GBGP) (Wong and Leung, 2000) is used as the evolutionary approach (i.e. entire process). Section 3.1.3 elaborates the objective functions applied in the framework. Section 3.1.4 describes how NSGA-II is applied as the multi-objective optimization technique and incorporated with GBGP (i.e. Step 4). Section 3.1.5 elaborates Token Competitions among base classifiers (i.e. Step 5).

3.1.1 Classifier Representation

Many existing traditional methods and other advanced data mining techniques can be applied to handle FFD problems. However, they may have different issues and even provide poor results. It is not easy to learn a good classifier from a real-life dataset with a large number of records. For example, the performance of a Logistic Regression model cannot be significantly improved. In addition, as described before, a real-life dataset often contains a number of “noisy” cases that may affect the performance of the model obtained by the learning process. However, such “noisy” instances in FFD datasets cannot be removed immediately since they may be important for model learning. Therefore, we propose to use several classifiers to represent the relationships and knowledge learnt from the real-life dataset. Each classifier represents a relation or information that is different from other classifiers. Compared to using a logistic regression model to represent the hidden relationship in the whole dataset, the proposed method is more flexible in handling “noisy” examples. Moreover, the understandability of the produced classifiers or models is also important. Only experts or users who have related background knowledge can understand the results generated by some data mining methods (e.g. support vector machine).

Classification rules are one of the the most straightforward representations for the given problems. Each classification rule consists of two parts: *antecedent* and *consequence*. The antecedent part describes the relations and the consequence part shows the corresponding classification result. In general, the structure of a rule is shown in Equation (16).

$$if \langle X_1 = a_1 \rangle \text{ and } \langle X_2 = a_2 \rangle \text{ and } \dots \text{ and } \langle X_n = a_n \rangle, \text{ then } \langle Class = class_t \rangle \quad (16)$$

where X_i , $1 \leq i \leq n$, are the names of attributes. $\langle X_i = a_i \rangle$, $1 \leq i \leq n$,

indicate the conditions (i.e. antecedent part). $< Class = class_t >$ indicates the classification result (i.e. consequence part). Consider the following examples:

1. if Location is Shanghai and NOE between 500 to 600, then class = yes
2. if Industry is manufacturing and ChairCEO is no, then class = no

We define class = yes for fraudulent instances, and class = no for non-fraudulent instances. The first rule means that the firms are fraudulent if they are located in Shanghai and their number of employees (i.e. NOE) is between 500 to 600. The second rule means that the firms are non-fraudulent if their principal work is related to manufacturing and the chairman and CEO are not the same person (i.e. ChairCEO). In these two rules, each rule indicates a different relation in the data and users can easily understand and decide whether the firm is fraudulent or not. Therefore, using a number of classification rules can capture many relations from the given problem.

After determining classifier representation, it is necessary to apply an appropriate learning method to train the classification rules. Several techniques can be applied, but not all of them are suitable and efficient. An evolutionary algorithm (EA) can be a very powerful searching algorithm in training a number of classification rules. The following section will describe the applied evolutionary algorithm in detail.

3.1.2 Grammar-based Genetic Programming (GBGP)

Genetic Algorithm (GA) and Genetic Programming (GP) are members of the Evolutionary Algorithms (EAs). They all have a pool of individuals, which can be numbers, functions or models. In GA, the individual normally has a fixed size, i.e. the length of individual cannot be changed during the evolution. It is

necessary to design the *encode* and *decode* scheme for a classification rule if GA is applied to solve the FFD problems. In GP, the individual is more flexible and it has variable length (i.e. the size of an individual can be changed). For example, consider the following rules:

1. if $(X_1 = X_2)$, then class = yes.

In GA, such rule cannot be represented in an individual, because comparison between variables cannot be denoted readily in GA. On the other hand, many rules have to be used to describe the meanings of the above rules. If there are two values (a and b) for the variables X_1 and X_2 , two rules have to be used:

1. if $(X_1 = a)$ and $(X_2 = a)$, then class = yes.
2. if $(X_1 = b)$ and $(X_2 = b)$, then class = yes.

In GP, such rule can be represented directly if an appropriate structure of individuals in GP is designed. In addition, it is limited and inflexible to generate a complex individual if the size of a chromosome is fixed. For example, consider a problem with two independent variables and one dependent variable, and the two following rules:

1. if $(X_1 = a)$, then class = yes.
2. if $(X_1 = a)$ and $(X_2 = b)$, then class = no.

In GA, a possible design of the two individuals is shown in Figure 14.

(1)	1	0	1	0	0	#	#	#	#	#	1
(2)	1	0	1	0	0	0	1	0	1	1	0
	← X ₁ →					← X ₂ →					← class →

Figure 14. Possible design of a chromosome

In Figure 14, two chromosomes (i.e. individuals) are used to represent the two rules in the example respectively. It uses five bits to represent the value of X_1 and X_2 . The last bit represents the class of a rule, which is “1” if class is *yes* and “0” if class is *no*. Since the size of the individual is fixed, the first individual has redundant bits if it does not have values for X_2 , which can also be a drawback of the representation in GA. On the other hand, GP uses a tree-based structure for each individual with a variable length (i.e. the length of an individual can be changed). It is more flexible compare to the representation of an individual with a fixed length in GA. GP does not consider the redundant information in the individual. Therefore, in summary, it is more appropriate to use GP than GA in the framework.

However, traditional GP has no mechanism to control the evolutionary process. It may generate a number of invalid individuals, for example, if a classification rule with tree-based structure is selected for mutation, as shown in Figure 15.

Figure 15. Example of invalid mutation

In Figure 15, if the node *Operator* is selected, and the value of the node “>” may be changed (i.e. mutate) to a real number, say “2”, in this case, it can be regarded as an invalid individual, because it is meaningless to assign a value to *Operator*. On the contrary, the value of the node *Operator* must be other operators, such as “<” or “=”. Therefore, a mechanism is required to monitor the evolutionary process to ensure that only valid individuals are generated. The mechanism in the framework is called “grammar”.

Table 3 shows an example of a simple grammar. The genetic operations (e.g. crossover and mutation) will be made based on the grammar. Then the

$Expression \rightarrow Boolean\ Yes\ No$
$Boolean \rightarrow Operator\ Term\ Value$
$Boolean \rightarrow true\ \ false$
$Term \rightarrow meeting\ \ board$
$Term \rightarrow [0,100]$
$Operator \rightarrow =\ \ >=\ \ <=\ \ >\ \ <$
$Yes \rightarrow yes$
$No \rightarrow no$

Table 3. Example of a grammar to control the evolutionary process

situation shown in Figure 15 can be avoided. In addition, “grammar” has another important feature, which can describe the given FFD problems readily.

Figure 16. Example of two individuals

For example, consider two individuals shown in Figure 16, which are generated based on the grammar in Table 3. Individual (a) indicates that if the number of board meetings is greater than 40, then the expression (e.g. firm) is “yes” (e.g. fraudulent); otherwise it is “no”. Individual (b) indicates that if the number of board members is smaller than 15, then the expression is “yes”; otherwise it is “no”. Therefore, GP with grammar can be very useful in solving the given problems. Genetic programming which incorporates grammar is called Genetic-based Genetic Programming (GBGP) (Wong and Leung, 2000).

The original GBGP uses a single fitness function (e.g. accuracy) to measure the performance of each individual. It may be easy to solve a simple data mining task by using the original GBGP with a single fitness function. But it is not sufficient to solve a real-life problem by only using a single objective function (i.e. fitness function) to measure the performance of an individual. The original

GBGP applies confidence and support framework, which has two objectives. It is not enough to using *confidence* (i.e. accuracy) solely to measure performance of a classification rule. The classification rule with the highest value of confidence may not able to recognize (i.e. cover) many other instances. In this case, such rules are not useful to future instances if considering confidence as the single metric. In another situation, the results are also week if using *support* only. The classification rule with the highest value of support may not able to correctly classify many instances, even though it can cover many instances. In this case, such rule is not useful because of its poor classification accuracy Wong and Leung (2000). We have applied the original GBGP with single objective function (i.e. support and confidence) to solve the given problems. The preliminary result shows that the original GBGP works well in benchmark problems, but cannot handle real-life problems very well. The major potential issue of the original GBGP is the fitness function, where the fitness value cannot measure the performance of an individual appropriately. In addition, we also applied the GBGP with different single objectives separately (i.e. support or confidence). The preliminary result shows that the GBGP with single objective (i.e. support or confidence) cannot handle the given problems very well. To solve the given FFD problems, by using combined version of confidence and support, it is able to consider both objectives, but it still hard to determine the trade-off between each objective. Because there is no single solution that can optimize each objective simultaneously. Confidence and support are conflicting, and the optimal solutions can be obtained by using some multi-objective methods. The goal of using multi-objective methods is to find out these non-dominated solutions. Each non-dominated solution has equally good objectives values like other non-dominated solutions, and researchers can apply them for task of classification. Many prior studies apply evolutionary algorithms with multi-objective functions to solve financial or economic problems. Therefore, we want to extend the original GBGP system to handle multi-objective optimization problems. Before developing the GBGP with multi-objective (GB-

MGP), we first study the objectives that can be used in the framework in detail in the following section.

3.1.3 Objective Functions

Many researchers have developed a number of different evolutionary algorithms with multiple objectives (i.e. MOEAs) for solving finance and economics applications (Ponsich et al., 2013). For example, in portfolio optimization problems, most applied objectives are *risk* and *return*. The Multi-objective Evolutionary Algorithms (MOEAs) are maximize the return and minimize the risk from different combinations of stocks in portfolios in order to obtain the optimal combination (Ponsich et al., 2013). There are not as many developed MOEAs in FFD applications portfolio optimization applications. Several objectives have been applied in FFD applications, such as *accuracy*, *comprehensiveness* and *interestingness*.

Industry	AB	Location	Dividend	Nshhold	Nemploy	Chairceo	Meeting	Fraud
supply	B	Shanghai	yes	174,536	3,947	no	6	yes
social_service	B	Tianjin	no	34,626	826	no	6	yes
manufacturing	B	Tianjin	yes	40,660	1,479	no	11	yes
manufacturing	A	Shanghai	yes	23,146	29,452	no	0	no
IT	A	Shanghai	yes	14,855	7,008	no	11	no

Table 4. Compact training dataset regarding firms

- *Accuracy* measures the classification ability of classifiers or models.

For example, consider the following two rules:

1. if Location is Shanghai, then class = yes
2. if Location is Shanghai, then class = no

The first rule indicates that the fraudulent firms are located in Shanghai. The second rule indicates that the non-fraudulent firms are located in Shanghai. There is a compact dataset in Table 4. Three instances (i.e.

firms) are located in Shanghai, but only the first instance is fraudulent (i.e. class = yes) and the other two are non-fraudulent (i.e. class = no). For the first rule, the number of instances that are correctly classified is one, and the number of instances that are misclassified is two. Therefore, the accuracy of the first rule is $1/3$. For the second rule, the number of instances that are correctly classified is two, and the number of instances that are misclassified is one. Therefore, the accuracy of the second rule is $2/3$. In general, good classifiers often have a high accuracy rate, which means that the performance of the second rule is better than the first rule in this example.

- *Comprehensiveness* indicates the comprehensive level (i.e. number of included attributes) of the classifiers or models. In (Ghosh and Nath, 2004), comprehensibility is minimized for a model to have clearer presentation. For example, consider the following two rules:

1. if Location is Shanghai, then class = yes
2. if Location is Shanghai and Industry is manufacturing, then class = yes

The first rule has a higher comprehensive level than the second one, since it has only one variable.

- *Interestingness* describes the interestingness of a rule.

As described in the previous section, GBGP with classification rules is applied, and it may not be sufficient to use only a single objective function to evaluate an individual for FFD problems. It has been used in solving some data mining problems with good performance (Ngan and Leung, 1998). It still has space for improvement if it is extended to Grammar-based Multi-objective Genetic Programming (GBMGP). Researchers in related domains have devoted themselves

to exploring more useful objectives for the FFD problems. In our study, we proposed to apply a famous framework called support-confidence (Agrawal et al., 1993). It has been successfully applied with GBGP to solve data mining problems, such as medical data mining (Ngan and Leung, 1998).

- *Confidence* measures accuracy, which is an important objective of a rule, since the principle purpose of FFD is to identify potentially fraudulent firms to prevent destructive consequences. The model will be risky without considering accuracy. However, it cannot ensure that a classification rule is good even it has 100% accuracy, because it may only cover one training instance, but ignore all the other instances. *Confidence* is a ratio of the number of cases covered by a rule (i.e. the attribute values and class value both fulfil the antecedent part and consequence part of a rule respectively) to the number of cases covered by the antecedent part only (i.e. the attribute values fulfil the antecedent part of a rule) (Wong and Leung, 2000).
- *Support*. In addition to accuracy, another objective called *support* is also necessary to establish the coverage of a classification rule. Therefore a good classification rule not only has high accuracy but also covers many cases. *Support* is a ratio of the number of cases covered by a rule to the total number of cases (Wong and Leung, 2000).

Table 5 is a compact database with 10 instances (i.e. firms). The first row consists of variable names. The classes of firms are shown in the last column.

Consider the following rule:

1. if Location is Shanghai, then class = yes.

According to the rule, only location and class type are considered; thus we only focus on these two attributes to calculate the support and confidence of

Industry	AB	Location	Dividend	Nshhold	Nemploy	Chairceo	Meeting	Fraud
retail	A	Shanghai	no	69,163	1,226	no	4	yes
supply	B	Shanghai	yes	174,536	3,947	no	6	yes
social_service	B	Tianjin	no	34,626	826	no	6	yes
manufacturing	B	Tianjin	yes	40,660	1,479	no	11	yes
manufacturing	A	Shanghai	yes	23,146	29,452	no	0	no
IT	A	Shanghai	yes	14,855	7,008	no	11	no
manufacturing	A	Shanghai	no	39,337	2,723	no	9	no
manufacturing	A	Sichuan	yes	29,861	3,878	no	7	no
mining	A	Sichuan	no	8,882	459	yes	6	no
supply	B	Tianjin	yes	47,264	826	no	6	no

Table 5. Compact database about firms

the rule. The total number of instances is 10. For support, only the first two instances are covered by the rule, whose location and class type are the same as the rule. For confidence, there are five instances covered by the antecedent part of the rule, which is Shanghai. Therefore, the support is 0.2 and confidence is 0.4 in this example. However, in traditional GBGP, support and confidence framework are combined into a single objective function, which is shown in Equation 17.

$$f = w_1 * support + w_2 * normalized_confidence \quad (17)$$

where f is the fitness value of an individual, *support* is defined before (Wong and Leung, 2000). A normalized confidence is used in their framework. *normalized_confidence* equals $confidence * \log(\frac{confidence}{prob})$, where *confidence* is defined before, *prob* is a ratio of the number of instances that have the same class (e.g. fraudulent) as the rule to the total number of instances. w_1 and w_2 are user-defined weights to control the balance between support and confidence respectively (Wong and Leung, 2000). However, it is not easy to determine an appropriate weight for each term. For example, if accuracy (i.e. confidence) is more important than the number of instances that the classifier covers (i.e. support), the value of w_2 can be 0.9 and the value of w_1 is 0.1. These values may not be the optimal combination for learning good classification rules. Therefore, we want to extend the traditional GBGP to GBMGP, which is described in the following section.

3.1.4 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

GBGP can genetically generate new individuals (i.e. classification rules), but the selection scheme is not appropriate in handling multi-objective problems. In the general process of Evolutionary Algorithm (EA) for single objective optimization, the selection scheme is used to select individuals as parents to produce new individuals. The good individuals have higher probabilities of being selected as parents for simple problems. The traditional selection scheme is only based on a single value, which is not appropriate for multi-objective problems. The main idea of multi-objective optimization is to obtain a number of non-dominated solutions, which are also called Pareto solutions. The general background of multi-objective optimization problems has been discussed in Section 2.5. As shown in Figure 17, the non-dominated solutions (i.e. black points) will form a curve called a Pareto front. It is necessary to optimize the curve, and then the performance of the entire population can be optimized. A number of MOEAs can be applied to achieve the task a non-dominated sorting genetic algorithm (NSGA-II) is an appropriate MOEA (Deb et al., 2002). It has been proven to be a powerful, fast and efficient algorithm in solving multi-objective problems. Therefore, the concept of NSGA-II is incorporated into the GBGP method.

Figure 17. Example of a population sorted by NSGA-II

There are two main features in NSGA-II (Deb et al., 2002), as shown in Figure 17. The first feature is to *sort* the individuals into different level of fronts (i.e. rank). The individuals in the first front are the non-dominated solutions (i.e. Pareto solutions). The individuals in subsequent ranks are poorer than the individuals in previous ranks. The second feature is to measure the *crowding distance* between each pair of two individuals that are the neighbour of the other. Therefore, each individual has a value to present its average distance. If the value of average distance is small, it means that the individual is close to its neighbours. If the value of average distance is large, it indicates that the individual is located in a less dense region (i.e. far away from its neighbours). Incorporating NSGA-II with GBGP, each classification rule is sorted into different fronts to obtain an average distance value. Then the classification rules that are located in less dense regions (i.e. large crowding distances) and a high front will have higher probability of being selected as parents, and GBGP will produce new classification rules based on the selected individuals.

3.1.5 Token Competition

Although all individuals are expected to be optimal on both objective functions, some potential good individuals may be ignored during the learning process. In NSGA-II, it prefers the individual of high rank if two individuals are located in differing ranks, and prefers the individual located in the less crowded region if two individuals belong to the same rank. The less crowded region means that the individuals are relatively far away from each other in the region. In other words, the individuals located in the relatively denser region may have low probability of being selected and reproduced. Figure 18 shows an example of a population.

In Figure 18, the eight black points are the Pareto solutions with index number. Points 1, 2 and 3 can be regarded as the less crowded region, and points 4, 5, 6, 7 and 8 are located in the crowded region. Points 1 and 8 can be selected

Figure 18. Example of a population

because they are boundary points (i.e. only have one neighbour) and therefore they have the largest crowding distances. The situation of point 4 is better than points 5, 6 and 7, since it is relatively far away from point 3, so its crowding distance is also higher than the other points in crowded regions (i.e. points with small crowding distances). Points 5, 6 and 7 may not be selected because they have very small crowding distances. Although the points 5, 6 and 7 are on the Pareto front (i.e. non-dominated solutions), they may not be selected during the learning process. Each point is a classification rule. Even though rules 5, 6 and 7 have similar support and confidence values, they may have different meanings. For example, there are three rules that indicate points 5, 6 and 7 in the following:

1. if Location is Shanghai, then class = yes
2. if Industry is manufacturing, then class = no
3. if NOE between 500 to 1000, then class = no

These three classification rules may have similar values for confidence and support, but they have totally different meanings. Therefore, we want to apply a mechanism to further evaluate this kind of individual by adjusting their locations. Token competition (Wong and Leung, 2000) can be used to achieve this purpose.

The basic idea of token competition is that each training case has a token. If a rule can cover a training case, then its token is seized by the rule. Other rules cannot take the token even if they can classify the case. The priority of obtaining tokens is determined by the performance of the individuals, which has been done in our implementation of NSGA-II. In other words, all individuals are sorted from high to low ranks. In each rank, the individuals are also sorted from large to small crowding distances. The sorted individuals will be evaluated for each training case at each generation. Only the first individual that can cover the training case will obtain a token and others cannot. After evaluating all individuals in all training cases, each individual will know the number of tokens it obtained. For each individual, its support and confidence values are then modified according to the formula shown in Equation (18).

$$adjusted_objective_value_m = raw_objective_value_m \times \frac{count}{ideal_total} \quad (18)$$

where, $raw_objective_value_m$ is the original value of objective m (e.g. confidence or support), $count$ is the number of tokens that the rule obtained and $ideal_total$ is the total number of tokens that it can obtain ideally.

Effectively, the location of each individual in the objective value space is changed. Figure 19 shows an example of three individuals with adjusted objective values.

In Figure 19, the locations of the three individuals 2, 5 and 7 are changed based on the number of obtained tokens respectively. Then the density of the region with individuals 4, 5, 6, 7 and 8 is decreased; consequently the remaining individuals 4, 6 and 8 can have higher probabilities of being selected during the learning process. For example, the original individual 5 is a non-dominated solution and its crowding distance is relatively small because it is close to individuals 4 and 6. Therefore, individuals 5 and 6 have low probability of being selected, be-

Figure 19. Example of a population with tokens

cause they are in the same front and similar (i.e. close) to each other in objective value space. After adjusting by token competition, the location of individual 5 is changed to 5' if it obtains few tokens. Therefore, individual 5 is dominated by other individuals. The crowding distance of individual 6 is increased because its neighbours are changed to individuals 4 and 8. Compared to the original Pareto front, the probability of selecting individual 6 is improved.

3.2 Statistical Selection Learning

As mentioned in Section 2.7, ensemble learning is able to improve the performance of the classifiers. In this study, a number of classification rules are evolved by GBMGP. It is difficult to determine which of them should be selected eventually. If all of them are selected as an ensemble, the result will not be optimal, because some non-Pareto individuals or even poorer individuals may produce incorrect results. In general, the result could be improved if an appropriate ensemble method is used. Therefore, we adopt the ideas from the ensemble learning techniques, such as bagging and boosting, which have been described in Section 2.7. In this study, we developed and examined three ensemble techniques for solving the FFD problems.

3.2.1 Majority Voting

The main idea of using majority voting is that, for a testing case, a number of individuals that cover the case will predict a result. The final prediction for the testing case is determined by the votes. If more than half of them predict that the case is class A and the others predict that it is class B, then the final result is A. Figure 20 is an example of using majority voting for making a prediction.

Figure 20. Ensemble learning example using majority voting

In Figure 20, eight individuals cover the testing case, and five of them predict that the firm is fraudulent. Therefore, the final prediction is fraudulent.

3.2.2 Weighted Voting

The main idea of using weighted voting is that, for a testing case, a number of individuals that cover the case will predict a result. The final prediction for the testing case is determined by the weighted votes. Each individual has a weight, which is calculated by its average value of support and confidence (i.e. $0.5 * \text{support} + 0.5 * \text{confidence}$). Moreover, all the class values of class are changed to 1 (i.e. class A) and -1 (i.e. class B). Then the prediction for each individual is its weight multiplied by its class value (i.e. 1 or -1). The final prediction of a testing case is the average of the covered individuals' predictions. The final prediction is 1 if the average value is greater than 0; otherwise it is -1. Figure 20 is an example of using majority voting for a prediction.

In Figure 21, the class of fraudulent firm is 1 and the class of non-fraudulent

Figure 21. Ensemble learning example using weighted voting

firm is -1. The average weight of this example is 0.0125, which is greater than 0. Therefore, the final prediction is fraudulent.

3.2.3 Statistical Selection

In addition to the previous ensemble learning techniques, we designed a new method to select and combine the evolved rules. Two targets should be achieved. The first is about performance, which is the accuracy of the final ensemble. The second is also diversity maintenance, which aims to have a wide variety of solutions to cover more cases. From the point view of statistics, a diverse population is composed of a number of different small groups of individuals, which are significantly different to each other. Therefore, we propose an ensemble method called statistical selection learning (SSL). The idea of the SSL is straightforward. Suppose there is a population with different evolved individuals, and each individual contains two terms $ind_i(fit_i, s_i)$, where fit_i is the fitness value of the individual. Fitness value is calculated by its average value of support and confidence (i.e. $0.5 * \text{support} + 0.5 * \text{confidence}$). s_i is the status of i^{th} individual, which indicates if the individual is selected or not. At the beginning, a set of individuals with small size (e.g. 3) were randomly selected from the Pareto front as the primary set, and then the same number of individuals were randomly selected from the whole population as the secondary set. We calculated the difference between these two sets by using paired t-test. The two sets were merged to form a new primary set if they were significantly different to each other at the 5 percent significance level, and then above steps were repeated to compare the primary set with another

secondary set. On the other hand, the secondary set were re-selected if they were not significantly different to the primary set. Once the termination conditions were satisfied, the final ensemble was constructed.

The whole process of the GBMGP is shown in algorithm 3.

3.3 Minority Prediction

The last step of the proposed framework is to evaluate the constructed ensemble on the testing dataset. In general, for a classification rule, a testing case will be evaluated only if the values of attributes from the testing are covered by the antecedent part of the rule. Otherwise, the testing case will skip the rule, and keep looking for other rules in turn until all rules are considered. However, there is no prediction for the testing case if no rules can be applied for it. Therefore, we suggest setting a default rule and predict the result of such testing case as the minority class (i.e. fraudulent). If the real class of the testing case is the same as the minority class (i.e. the testing case is a fraudulent firm), then we say it is correctly classified. Otherwise, it is misclassified. Therefore, minority prediction should improve the performance in classifying the fraudulent class. In this study, we think the detection of minority class (i.e. fraudulent) is much more important than the detection of majority class (i.e. non-fraudulent) in the given FFD problems. For example, if a firm is fraudulent (i.e. positive class), and it is incorrectly classified as non-fraudulent, then the loss to relative people (e.g. shareholders) may be destructive. However, if a firm is non-fraudulent (i.e. negative class), and it is incorrectly classified as fraudulent, it may need to be investigated by the Securities Regulatory Commission (SRC) or Securities and Exchange Commission (SEC) at relatively much lower cost (i.e. investigation fees) compared to the destructive consequence caused by fraud without any investigation. Therefore, it is more important to classify the fraudulent firms correctly than non-fraudulent firms. In summary, the flowchart of the complete framework is shown in Figure

Algorithm 3 GBMGP with statistical selection

```
1: Define the objectives  $Objs = \{(obj_1, s_1), \dots, (obj_N, s_N)\}$ 
2: Define and initialize variables:
3:    $R_t, P_t, Q_t, E_t$  {Pop. of whole, parent, offspring and ensemble}
4:    $N, S_{ensemble}$ , {The population size and ensemble size}
5:   max_gen,
6:   selection_method,
7:   crossover_rate, mutation_rate
8: for  $i = 1$  to  $N$  do
9:   initialize the individual  $ind_i$  and store in  $P_t$ .
10: end for
11: set  $t = 0$  {The initial generation.}
12: while terminate criterion does not match do
13:    $R_t = P_t \cup Q_t$  {The initial generation.}
14:    $F = fast\_non\_dominated\_sort(R_t)$ 
15:   set  $P_{t+1} = \phi$  and  $i = 1$ 
16:   while  $|P_{t+1}| + |F_i| \leq N$  do
17:     crowding_distance_assignment( $F_i$ )
18:      $P_{t+1} = P_{t+1} \cup F_i$ 
19:      $i = i + 1$ 
20:   end while
21:   Sort( $F_i$ )
22:    $P_{t+1} = F_i[1 : (N - |P_{t+1}|)]$ 
23:    $Q_{t+1} = produce\_new\_population(P_{t+1})$ 
24:    $t = t + 1$ 
25: end while
26: set  $t = 0$  {reset t to zero}
27: define  $k$  {The size for initial ensemble}
28: define  $C_t$  {The provisional ensemble for comparison.}
29:  $E_t = random\_select\_individuals(k, R_t)$ 
30:  $R_t := R_t - E_t$  {Remove the selected individuals from  $R_t$ }
31: while terminate criterion does not match do
32:   set  $C_t = \phi$ 
33:    $C_t = random\_select\_individuals(k, R_t)$ 
34:   if ( $E_t \neq C_t$ ) then
35:      $E_t := E_t \cup C_t$ 
36:      $k := k + k$ 
37:   end if
38:    $t = t + 1$ 
39:   return  $E_t$ 
40: end while
```

Figure 22. The flowchart of the proposed method.

CHAPTER 4

Experiments and Results

This chapter describes the experiment preparation and experiment results. In this study, a number of data mining techniques were applied to solve four financial fraud detection problems. We applied Waikato Environment for Knowledge Analysis (WEKA) for the experiments, which are described in Section 4.1.1 providing the well-developed data mining methods for comparison. Section 4.1.2 contains the descriptions of the datasets. Section 4.1.3 gives the grammar designs for each problem. Section 4.1.4 describes the over-sampling technique used in the experiment. Section 4.1.5 describes the K-folds cross-validation mechanism. Section 4.1.6 describes the evaluation criteria used for each method in the experiment. Section 4.2 shows the parameter setting for the proposed method and briefly introduces several variants of proposed methods. Section 4.3 presents the experiment results of all methods and discusses the results in detail.

4.1 Introduction to Experiment Preparation

4.1.1 Waikato Environment for Knowledge Analysis (WEKA)

In order to compare the performance of Grammar-based Multi-objective Genetic Programming with Statistical Selection Learning (GBMGP-SSL) and the other well-known data mining techniques, we applied Waikato Environment for Knowledge Analysis (WEKA) (Hall et al., 2009) for the experiments. WEKA has already included a number of well-developed techniques that can be used in data mining, ensemble learning and machine learning. A traditional statistical method, four machine learning techniques, three ensemble learning approaches and 14 variants of GBGP (including the proposed method) were evaluated in the study.

4.1.2 Data Description

Four financial fraud problems were considered. Two of them were collected from the UCI machine learning repository (Asuncion and Newman, 2007) and the other two were real-life financial fraud problems. The description of datasets is shown in Table 6.

Dataset	Attributes	Instances	Classes	Class Ratio
Australian credit	14	690	2	307:383
Credit approval	15	690	2	307:383
U.S. corporate fraud	41	68332	2	63: 68269
China corporate fraud	17	18373	2	855:17518

Table 6. Data Description.

“Australian credit” and “credit approval” are similar, but the latter has one more attribute, which may affect the results. However, they are often used together as benchmark problems in many data mining studies. The class distributions are not imbalanced.

For “U.S corporate fraud” data, the original data had nearly 200 variables with duplicated and useless attributes, such as firm id or name. The data were extremely imbalanced, which may affect the results if models are used immediately. In general, the number of fraudulent firms is much smaller than the number of non-fraudulent firms. Therefore, it is better to maintain the fraudulent instances. Otherwise, it is difficult to learn the fraudulent information based on the few instances. If the fraudulent firms had too many missing values (e.g. more than 40% missing values) in some attributes, we removed those attributes directly. On the other hand, if the fraudulent firms had few missing values in some attributes, we replaced them based on the data distributions of those attributes (e.g. took a mean of the variable as the value for the missing data). For non-fraudulent firms, we removed the instances with many missing values. For the attributes with few missing values, we also replaced them based on the data distributions of those attributes.

“China corporate securities fraud” (CCSF) data contained records of corporations with their firm, financial, governance and trade characteristics. The variables were selected on the basis of the relative literature discussed in Section 2.2.2. Moreover, including more attributes may provide more interesting information of the fraudulent firms for the system to learn. The original database has 21,396 instances with 25 attributes for all listed firms from 1998 to 2011. Each instance with more than 20 missing values in these 25 attributes was directly removed. Moreover, there are seven attributes about trade characteristics were removed since more than two-thirds of firms had no such trade data. The final dataset had 18,373 records with 18 attributes. It was also highly imbalanced with 5.8% fraudulent and 94.2% non-fraudulent examples.

4.1.3 Grammar Design

The grammars for all problems are attached in the appendix. A.1.1 and A.1.2 are the grammar design for Australian credit and credit approval dataset respectively, A.1.3 describes the grammar design for U.S. corporate securities fraud detection problem and A.1.4 concerns grammar design for the China corporate securities fraud detection problem. In *Australian credit* and *credit approval*, the attribute names and values were changed to meaningless symbols in order to protect the confidentiality of the data (Asuncion and Newman, 2007).

4.1.4 Synthetic Minority Over-sampling Technique (SMOTE)

Imbalanced data cannot be directly used in any of mentioned methods. Without prior consideration of the imbalance, the classifier(s) will always generate biased results for the majority class. Such classifiers are not useful, as their performance could be very poor (Liu et al., 2007). A number of approaches have

been introduced to address imbalanced datasets, such as resampling techniques or pre-processing methods. The Synthetic Minority Over-sampling Technique (SMOTE) is a data pre-processing method, which can process data and generate synthetic examples by taking each minority class example along the line joining all of its k nearest neighbours. For example, if the number of minority class examples needed is triple (i.e. 300%), and the number of its nearest neighbours are limited to 5 (i.e. $k=5$). 3 of 5 nearest neighbours are selected as three directions and one synthetic example is generated along each direction, and then three new examples are introduced. This study applied SMOTE for a variety of reasons. First, the standard SMOTE is very simple to implement in practice. Second, empirically, SMOTE has been shown to perform well against random oversampling techniques in a lot of experiments (Liu et al., 2007; Chawla et al., 2011). Third, the synthetic examples are generated in a less application-oriented manner. In other words, the new examples are operated in feature space (i.e. based on each variable) rather than data space (i.e. whole dataset) (Chawla et al., 2011). Therefore, it can be widely applied in imbalanced dataset applications.

4.1.5 K-folds Cross-Validation

For a robust experiment, a ten-fold cross-validation mechanism was applied for each dataset. The ten-fold cross-validation split the data into ten mutually exclusive and exhaustive folds. For each experiment, one fold was regarded as the testing dataset and the other nine folds were combined together as the training dataset. Figure 23 is a graphic illustration of ten-folds cross-validation.

Figure 23. Ten-folds cross-validation example

All the approaches used the same training data, tested on the same testing

data and generated corresponding results. The experiments were repeated ten times until each fold was evaluated as a testing dataset. The final result was computed by taking the average (i.e. mean) of the generated rules. In addition, two datasets (i.e. U.S.CSF and CCSF) have unbalanced data distribution. When evaluating data mining methods on these two datasets, only the training data were pre-processed by SMOTE. In other words, the testing datasets maintained the original data distribution in the experiment.

4.1.6 Model Evaluation Criteria

As discussed in Section 3.1, accuracy is the most important criteria of a model when solving FFD problems. Each problem has two classes. The first class is regarded as *positive* (i.e. fraudulent) and the other is *negative* (i.e. non-fraudulent). If a testing case matches the antecedent part of the rule, and the consequence part of the rule also matches the class of the testing case, then the testing case is correctly classified. Table 7 shows the possible outcomes for binary classification.

	Classified as True	Classified as False
Actual is True	True Positive (TP)	False Negative (FN)
Actual is False	False Positive (FP)	True Negative (TN)

Table 7. Contingency table with four outcomes of binary classification

The accurate rate of positive class is called *true positive rate (TPR)*, which is calculated by Equation (19).

$$TPR = TP / (TP + FN) \quad (19)$$

where TP is the number of positive examples that are correctly classified. $TP + FN$ is the total number of positive examples including the number of correctly classified positive examples (i.e. TP) and the number of incorrectly classified as negative class (i.e. FN). The accuracy rate for the negative class is called *true*

negative rate (TNR), which is calculated by Equation (20).

$$TNR = TN / (TN + FP) \quad (20)$$

where TN is the number of negative examples that are correctly classified. $TN + FP$ is the number of total negative examples including the number of correctly classified negative examples (i.e. TN) and the number of negative examples that are incorrectly classified as positive (i.e. FP). It is easy to observe the performance of each model for each class by using TPR and TNR as evaluation criteria.

4.2 GBMGP with SSL

Table 8 shows the parameter setting for the Grammar-based Multi-objectives Genetic Programming with Statistical Selection Learning (GBMGP-SSL). Except for the GBMGP with SSL, several GBGP variants (i.e. single objective GBGP) and GBMGP (i.e. multi-objective GBGP) variants are developed for model comparisons. GBGP variants contain GBGP(s,c), GBGP(s,c) with majority voting and GBGP(s,c) with weighted voting. GBMGP variants include GBMGP(s,c), GBMGP(s,c) with majority voting and GBMGP(s,c) with weighted voting, where s, c indicates support and confidence respectively. In GBGP variants, support and confidence are combined into a linear equation. On the other hand, support and confidence are the two independent objectives in GBMGP variants. Majority voting and weighted voting are the ensemble techniques. The former counts the number of votes in two classes for each testing case, and the final prediction is the class with higher votes. The latter is similar to majority voting, but each rule has different weight for voting. The weight is the average value of two objectives from each individual. The strong rules with high objective values have higher weights to make the final decision.

Single objective GBGP variants (e.g. GBGP only) used *elitism* to select the best individual(s) of the current population for the next generation directly. The *elitism* operator always selects the individual with the highest objective value for

GBMGP-SSL	
Parameter	Value
Population size	200
Max. no. of generation	500
Use elitism	no
Selection scheme	tournament
Keep parent	yes
Use token competition	yes
Crossover rate	0.8
Mutation rate	0.2
Ensemble method	statistical selection
Max. ensemble size	0.6

Table 8. Parameters and values for the proposed method

the next generation directly without using any genetic operators. Multi-objectives GBGP variants (e.g. GBMGP) did not use *elitism*, since the Pareto solutions (i.e. non-dominated solutions) in the current population have already been considered in the evolutionary process automatically. Other experiments settings were the same as shown in Table 8. In addition, *tournament* selection was used. It randomly selects a number of solutions with tournament size k , and chooses the best (i.e. winner) for genetic operation (e.g. crossover or mutation). The default tournament size is 2. *Keep parent* has to be used in the GBMGP variants, since the NSGA-II has to sort the parent (P_t) and offspring (Q_t) population at the same time, which has been shown in Section 2.5.1. During the evolutionary process, the individuals were selected from a population (i.e. parents) to produce new individuals (i.e. offspring). *Keep parent* operator selects all parents and merges them with new individuals. Therefore, the size of the population is doubled. The last two parameters were only used in the proposed method for ensemble learning. It applied the proposed statistical selection with maximum 60 percents of the whole population. For example, if the population size is 100 and ensemble size setting is 0.6 (i.e. 60%), then at most 60 individuals will be selected to form an ensemble. On the other hand, majority voting and weighted voting used all evolved solutions (i.e. ensemble size is 100%).

4.3 Compared Approaches

Method	Parameter	Value
Logistic	ridge	1.0E-8
	max.iterations	-1
Neural Networks	Learning rate	0.3
	Momentum value	0.2
	no.hidden layers	1
	weight update	Back-propagation
	Training epochs	500
	random seed	0
SMO	kernel function	Polykernel
	complexity	1
	tolerance rate	0.001
	exponent value	1
Bayesian Networks	estimator	Simple estimator
	search algorithm	Hill climbing
Decision Tree	min number of nodes	2
	No pruning	False
	number of folds	3
	min variance probability	0.001
AdaBoost	classifier	Decision stump
	number of iterations	20
	seed	1
	use resampling	False
	weight threshold	100
Bagging	classifier	REPTree
	no.iterations	20
	bag size percent	100
LogitBoost	classifier	Decision stump
	no.iterations	20
	use resampling	False
	seed	1
	weight threshold	100
	likelihood threshold	-1.798

Table 9. Parameters and values for the compared approaches

The parameter settings for a traditional statistical method, four machine learning techniques and three ensemble learning approaches are shown in Table 9 and separated by a double line. The compared approaches are often applied in other data mining studies. These methods were briefly discussed in Section 2.4.

	Australia credit		Credit approval		U.S.CSF		CCSF		
Methods	TPR	TNR	TPR	TNR	TPR	TNR	TPR	TNR	w/t/1
Logistic	0.81 ⁺⁺	0.81	0.85 ⁺⁺	0.84 ⁻	0.41 ⁺⁺	0.90 ⁻⁻	0.41 ⁺⁺	0.47 ⁺⁺	6/0/2
	0.058	0.063	0.046	0.042	0.222	0.013	0.045	0.017	
MultilayerNN	0.80 ⁺⁺	0.83	0.80 ⁺⁺	0.83	0.47 ⁺⁺	0.94 ⁻⁻	0.31 ⁺⁺	0.83 ⁻⁻	5/0/3
	0.089	0.064	0.047	0.048	0.191	0.014	0.067	0.053	
SMO	0.87	0.79 ⁺⁺	0.80 ⁺⁺	0.79	0.51 ⁺	0.94 ⁻⁻	0.41 ⁺⁺	0.73 ⁻⁻	6/0/2
	0.027	0.069	0.043	0.051	0.183	0.006	0.037	0.019	
BayesNetwork	0.80 ⁺⁺	0.88	0.80 ⁺⁺	0.81	0.52 ⁺	0.91 ⁻⁻	0.28 ⁺⁺	0.94 ⁻⁻	4/0/4
	0.059	0.017	0.046	0.029	0.185	0.008	0.022	0.006	
Decision Tree	0.82 ⁺⁺	0.84	0.82 ⁺⁺	0.84	0.10 ⁺⁺	1.00 ⁻⁻	0.24 ⁺⁺	0.93 ⁻⁻	5/0/3
	0.079	0.105	0.078	0.103	0.071	0.001	0.029	0.009	
AdaBoost	0.81 ⁺⁺	0.84	0.77 ⁺⁺	0.81	0.51 ⁺	0.88 ⁻⁻	0.47 ⁺⁺	0.71 ⁻	5/0/3
	0.053	0.062	0.046	0.069	0.174	0.016	0.046	0.031	
Bagging	0.79 ⁺⁺	0.82	0.78 ⁺⁺	0.80	0.11 ⁺⁺	1.00 ⁻⁻	0.23 ⁺⁺	0.89 ⁻⁻	5/1/2
	0.068	0.067	0.057	0.058	0.076	0.001	0.025	0.008	
LogitBoost	0.81 ⁺⁺	0.82	0.83 ⁺⁺	0.82	0.50 ⁺	0.90 ⁻⁻	0.44 ⁺⁺	0.80 ⁻⁻	5/0/3
	0.038	0.069	0.032	0.047	0.211	0.013	0.046	0.028	
GBMGP(s,c,S) _i	0.89	0.85	0.89	0.80	0.64	0.76	0.66	0.67	n/a
	0.063	0.069	0.058	0.055	0.118	0.133	0.074	0.069	

1. ⁺⁺ Using paired t-test, the average accuracy is significantly worse than that of GBMGP(s,c,S)_i at the 0.05 level.
2. ⁺ Using paired t-test, the average accuracy is significantly worse than that of GBMGP(s,c,S)_i at the 0.1 level.
3. ⁻ Using paired t test, the average accuracy is significantly better than that of GBMGP(s,c,S)_i at the 0.1 level.
4. ⁻⁻ Using paired t-test, the average accuracy is significantly better than that of GBMGP(s,c,S)_i at the 0.05 level.

Table 10. Accuracies of data mining techniques and the proposed method

4.3.1 Results and Analysis

Table 10 summarizes the results of average accuracy for each class on the four financial datasets. The name of each method is shown in the first column. Four datasets were evaluated by nine methods in this experiment. Each dataset had two classes: positive and negative, and the corresponding accuracies are indicated by TPR and TNR respectively, which are shown in the second row of Table 10. The Standard Deviation (S.D.) of each method is also given below the corresponding accuracy result. For example, logistic regression obtains 81% in classifying positive class on the Australian dataset, and its S.D. is 5.8%.

In Australian credit and credit approval, all the approaches are promising with regard to TPR and TNR. In the two real-life datasets, U.S.CSF and CCSF, the performances were not stable using different methods. Some methods such as Decision Tree and Bagging generated extremely biased results with very low TPRs and very high TNRs. The base classifier of Bagging is also a kind of Decision Tree. Moreover, Logistic regression obtained about 41% in regard to classifying fraudulent firms in both real-life datasets. SMO, BayesNetwork, AdaBoost and

LogitBoost could obtain about 50% in regard to classifying fraudulent firms for the U.S.CSF dataset only, and worse TPR for the CCSF dataset. The proposed method with minority prediction, which is located in the last row of Table 10 can achieved better TPR results than most of the other techniques, but its TNR values for each dataset were relatively lower at the same time. According to the characteristics of financial datasets, especially for real-life FFD problems, the detection of positive class (i.e. fraudulent) is much more important than the detection of negative class (i.e. non-fraudulent). For example, if a firm is fraudulent (i.e. positive class), and it is incorrectly classified as non-fraudulent, then the loss to interested people (e.g. shareholders) may be destructive. However, if a firm is non-fraudulent (i.e. negative class), and it is incorrectly classified as fraudulent, it may need to be investigated by the Securities Regulatory Commission (SRC) at relatively much lower cost (i.e. investigation fees) compared to the destructive consequence caused by fraud without any investigations. Therefore, it is more important to classify fraudulent firms correctly than non-fraudulent firms. In order to have a more comprehensive comparison for the proposed method, a number of GBGP variants and GBMGP variants were developed and the corresponding results are shown in Table 11.

In Table 11, a number of developed GBGP variants and GBMGP variants which were evaluated by the same datasets are shown. The name of each method is located in the first column, and the meanings of notations are indicated in Table 12. For example, the first method is $\text{GBGP}(\text{s}, \text{c})_a$, which is the original GBGP. It is a single objective method, which uses an average linear combined version of **support** (i.e. **s**) and **confidence** (i.e. **c**) framework. The value of the objective function is $1 * \text{support} + 8 * \text{normalized_confidence}$. The symbol “a” means that the first method used **m**ajority prediction.

As another example, the last method is $\text{GBMGP}(\text{s}, \text{c}, \text{S})_i$, which is the pro-

With tokens	Australia credit		Credit approval		U.S.CSF		CCSF		
Methods	TPR	TNR	TPR	TNR	TPR	TNR	TPR	TNR	w/t/1
GBGP(s,c) _a	0.84 ⁺	0.86	0.84 ⁺⁺	0.85 ⁻⁻	0.33 ⁺⁺	0.84 ⁻⁻	0.48 ⁺⁺	0.78 ⁻⁻	4/0/4
	0.063	0.095	0.027	0.045	0.201	0.049	0.059	0.025	
GBGP(c) _a	0.68 ⁺⁺	0.79 ⁺⁺	0.66 ⁺⁺	0.75 ⁺	0.29 ⁺⁺	0.83 ⁻	0.33 ⁺⁺	0.65	7/0/1
	0.049	0.024	0.045	0.060	0.046	0.054	0.062	0.045	
GBGP(s) _a	0.09 ⁺⁺	0.97 ⁻⁻	0.09 ⁺⁺	0.97 ⁻⁻	0.02 ⁺⁺	0.93 ⁻⁻	0.07 ⁺⁺	0.94 ⁻⁻	4/0/4
	0.048	0.027	0.015	0.018	0.009	0.031	0.033	0.034	
GBGP(s,c, M) _a	0.82 ⁺⁺	0.79 ⁺⁺	0.82 ⁺⁺	0.79	0.39 ⁺⁺	0.85 ⁻⁻	0.25 ⁺⁺	0.92 ⁻⁻	6/0/2
	0.034	0.075	0.032	0.031	0.254	0.021	0.034	0.021	
GBGP(s,c, W) _a	0.82 ⁺⁺	0.80 ⁺	0.84 ⁺⁺	0.79	0.30 ⁺⁺	0.89 ⁻⁻	0.54 ⁺⁺	0.67	6/1/1
	0.046	0.053	0.031	0.041	0.120	0.046	0.055	0.051	
GBMGP(s,c) _a	0.80 ⁺⁺	0.91 ⁻⁻	0.80 ⁺⁺	0.84	0.44 ⁺⁺	0.86 ⁻⁻	0.48 ⁺⁺	0.73 ⁻⁻	4/0/4
	0.054	0.027	0.074	0.066	0.229	0.034	0.058	0.054	
GBMGP (s,c,W) _a	0.85	0.90 ⁻⁻	0.85 ⁺	0.89 ⁻⁻	0.32 ⁺⁺	0.83 ⁻	0.38 ⁺⁺	0.88 ⁻⁻	4/0/4
	0.061	0.027	0.053	0.036	0.165	0.046	0.108	0.052	
GBMGP (s,c, M) _a	0.84 ⁺⁺	0.91 ⁻⁻	0.84 ⁺⁺	0.89 ⁻⁻	0.45 ⁺⁺	0.83	0.48 ⁺⁺	0.87 ⁻⁻	4/0/4
	0.040	0.047	0.034	0.049	0.235	0.084	0.127	0.061	
GBMGP(s,c,S) _a	0.82 ⁺⁺	0.90	0.86	0.87 ⁻⁻	0.54 ⁺	0.89 ⁻⁻	0.53 ⁺⁺	0.81 ⁻⁻	4/0/4
	0.081	0.079	0.056	0.049	0.122	0.089	0.081	0.084	
GBGP(s,c) _i	0.82 ⁺⁺	0.78 ⁺⁺	0.83 ⁺⁺	0.80	0.43 ⁺⁺	0.81	0.59 ⁺	0.59 ⁺⁺	6/1/1
	0.043	0.072	0.038	0.059	0.179	0.047	0.092	0.096	
GBGP(c) _i	0.75 ⁺⁺	0.69 ⁺⁺	0.73 ⁺⁺	0.68 ⁺⁺	0.34 ⁺⁺	0.58 ⁺⁺	0.43 ⁺⁺	0.49 ⁺⁺	8/0/0
	0.061	0.050	0.075	0.058	0.155	0.137	0.050	0.047	
GBGP(s) _i	0.13 ⁺⁺	0.99 ⁻⁻	0.14 ⁺⁺	0.97 ⁻⁻	0.08 ⁺⁺	0.97 ⁻⁻	0.10 ⁺⁺	0.96 ⁻⁻	4/0/4
	0.031	0.018	0.035	0.027	0.036	0.027	0.047	0.038	
GBGP(s,c, M) _i	0.84 ⁺⁺	0.78 ⁺⁺	0.85 ⁺⁺	0.79	0.47 ⁺⁺	0.86 ⁻⁻	0.61 ⁺	0.60 ⁺⁺	7/0/1
	0.040	0.067	0.025	0.042	0.189	0.033	0.058	0.057	
GBGP(s,c, W) _i	0.85 ⁺⁺	0.79 ⁺⁺	0.86	0.79	0.44 ⁺⁺	0.76	0.60 ⁺⁺	0.58 ⁺⁺	7/1/0
	0.038	0.036	0.037	0.062	0.116	0.065	0.062	0.040	
GBMGP(s,c) _i	0.82 ⁺⁺	0.88	0.84 ⁺⁺	0.86 ⁻⁻	0.45 ⁺⁺	0.73	0.48 ⁺⁺	0.58 ⁺⁺	6/0/3
	0.037	0.070	0.045	0.048	0.258	0.143	0.052	0.055	
GBMGP (s,c,W) _i	0.86	0.89	0.85 ⁺	0.87 ⁻⁻	0.35 ⁺⁺	0.81	0.55 ⁺⁺	0.68	4/0/4
	0.067	0.050	0.055	0.036	0.113	0.063	0.053	0.031	
GBMGP (s,c, M) _i	0.85	0.86	0.82 ⁺⁺	0.84 ⁻	0.44 ⁺⁺	0.75	0.59 ⁺⁺	0.64	6/0/2
	0.064	0.087	0.047	0.022	0.186	0.154	0.042	0.075	
GBMGP(s,c,S) _i	0.89	0.85	0.89	0.80	0.64	0.76	0.66	0.67	n/a
	0.063	0.069	0.058	0.055	0.118	0.133	0.074	0.069	

1. ++ Using paired t-test, the average accuracy is significantly worse than that of GBMGP(s,c,S)_i at the 0.05 level.
2. + Using paired t-test, the average accuracy is significantly worse than that of GBMGP(s,c,S)_i at the 0.1 level.
3. - Using paired t-test, the average accuracy is significantly better than that of GBMGP(s,c,S)_i at the 0.1 level.
4. -- Using paired t-test, the average accuracy is significantly better than that of GBMGP(s,c,S)_i at the 0.05 level.

Table 11. Classification accuracies of the proposed method and its variants

posed method. It is a multi-objective GBGP (i.e. GBMGP), and the abbreviation “M” indicates that the system has the component: **multi-objective**. Therefore, it uses support (i.e. s) and confidence (i.e. c) as the two objectives. The symbol “S” means that the ensemble learning technique is a **statistical selection method**, as shown in Table 12. The symbol “i” means that the last method uses **minority** prediction.

In the Australia credit and Credit approval datasets, the original GBGP can obtain about 85% accuracy for both TPRs and TNRs. Ensemble learning techniques (i.e. majority voting and weighted voting) cannot improve the original GBGP, no matter whether majority prediction or minority prediction is used. In addition, variants with multi-objective (i.e. GBMGP) have slightly poorer perfor-

Abbreviation	Description
s	Objective: s upport
c	Objective: c onfidence
W	Ensemble: w eighted voting
M	Ensemble: m ajority voting
S	Ensemble: s tatistical selection
a	M ajority prediction
i	M inority prediction

Table 12. Abbreviations of all the approaches

mance in regard to TPRs and slightly better performance in regard to TNRs than the original GBGP. However, variants with multi-objective (i.e. GBMGP) and ensemble techniques perform similarly and even obtains better TPRs and TNRs than the original GBGP except for the proposed method. The proposed method obtain the highest TPRs and had slightly poorer performance in regard to TNRs.

In U.S.CSF, all methods using majority prediction had good performance in regard to TNRs. However the corresponding TPRs were very low, with only the GBMGP(s,c,S)_a obtaining a result that was more than 50% for TPR. The TPRs were relatively improved by using minority prediction, but still less than 50%. The proposed method achieved 64%, which is the highest TPR value among all variants.

In the CCSF dataset, GBMGP without using any ensemble learning techniques could not improve the results over the original GBGP. The original GBGP with majority voting even produced poorer TPR results. However, compared to the original GBGP, the TPR using original GBGP with minority prediction had about 22.9% improvements. Except for the proposed method, the GBGP with majority voting and minority prediction obtained the second highest TPR, but the corresponding TNR was greatly reduced. The minority prediction performed well in this dataset, especially for GBGP(s,c,M). Finally, the pro-

posed method produced the highest TPR and relatively higher TNR compared to $\text{GBGP}(\text{s,c,M})_i$. In addition, we also evaluated the performance of the proposed method and its variants without using token competition, which is shown in Table 13.

Without tokens	Australia credit		Credit approval		U.S.CSF		CCSF	
Methods	TPR	TNR	TPR	TNR	TPR	TNR	TPR	TNR
$\text{GBGP}(\text{s,c})_a$	0.362	0.925	0.360	0.908	0.175	0.796	0.181	0.927
	0.080	0.041	0.052	0.032	0.082	0.062	0.035	0.052
$\text{GBGP}(\text{c})_a$	0.135	0.975	0.113	0.983	0.061	0.903	0.066	0.938
	0.052	0.031	0.059	0.023	0.035	0.053	0.041	0.047
$\text{GBGP}(\text{s})_a$	0.023	0.960	0.020	0.985	0.009	0.987	0.012	0.980
	0.017	0.034	0.013	0.025	0.008	0.011	0.007	0.020
$\text{GBGP}(\text{s,c,M})_a$	0.305	0.936	0.316	0.928	0.209	0.803	0.150	0.897
	0.059	0.039	0.054	0.026	0.055	0.053	0.059	0.053
$\text{GBGP}(\text{s,c,W})_a$	0.388	0.855	0.380	0.876	0.207	0.798	0.208	0.906
	0.040	0.042	0.035	0.060	0.081	0.057	0.053	0.038
$\text{GBMGP}(\text{s,c})_a$	0.366	0.858	0.349	0.877	0.246	0.818	0.240	0.930
	0.031	0.055	0.051	0.042	0.078	0.069	0.078	0.050
$\text{GBMGP}(\text{s,c,W})_a$	0.373	0.903	0.363	0.898	0.285	0.835	0.204	0.893
	0.057	0.083	0.039	0.047	0.109	0.075	0.047	0.063
$\text{GBMGP}(\text{s,c,M})_a$	0.359	0.930	0.371	0.924	0.242	0.835	0.160	0.954
	0.044	0.056	0.053	0.043	0.092	0.075	0.049	0.049
$\text{GBMGP}(\text{s,c,S})_a$	0.411	0.906	0.408	0.904	0.239	0.908	0.301	0.930
	0.065	0.058	0.057	0.045	0.091	0.063	0.061	0.050
$\text{GBGP}(\text{s,c})_i$	0.924	0.173	0.915	0.165	0.711	0.347	0.889	0.316
	0.038	0.048	0.037	0.048	0.075	0.132	0.051	0.090
$\text{GBGP}(\text{c})_i$	0.963	0.008	0.968	0.010	0.948	0.052	0.969	0.013
	0.030	0.010	0.027	0.009	0.029	0.041	0.030	0.008
$\text{GBGP}(\text{s})_i$	0.031	0.957	0.038	0.968	0.020	0.971	0.021	0.969
	0.019	0.047	0.036	0.023	0.008	0.034	0.011	0.039
$\text{GBGP}(\text{s,c,M})_i$	0.922	0.181	0.924	0.178	0.737	0.377	0.834	0.332
	0.051	0.055	0.042	0.052	0.098	0.060	0.060	0.075
$\text{GBGP}(\text{s,c,W})_i$	0.924	0.187	0.915	0.182	0.700	0.330	0.815	0.367
	0.038	0.052	0.051	0.062	0.077	0.081	0.044	0.053
$\text{GBMGP}(\text{s,c})_i$	0.926	0.272	0.915	0.268	0.831	0.350	0.877	0.359
	0.050	0.051	0.045	0.067	0.076	0.053	0.062	0.049
$\text{GBMGP}(\text{s,c,W})_i$	0.940	0.306	0.927	0.318	0.849	0.296	0.845	0.322
	0.044	0.051	0.047	0.054	0.078	0.081	0.048	0.059
$\text{GBMGP}(\text{s,c,M})_i$	0.902	0.298	0.907	0.301	0.815	0.375	0.815	0.387
	0.036	0.042	0.047	0.035	0.114	0.059	0.060	0.048
$\text{GBMGP}(\text{s,c,S})_i$	0.938	0.467	0.924	0.452	0.885	0.405	0.910	0.402
	0.050	0.094	0.053	0.052	0.116	0.048	0.089	0.044

Table 13. Classification accuracies of the proposed method and its variants without using token competition

From the results in Table 13, the overall performance was worse than the results obtained by using token competition in Table 11. In general, the result

of each method was biased towards a class due to the majority prediction and minority prediction. For example, $\text{GBGP}(s,c)_a$ without using token competition had poor results for TPR because of its poor diversity. It could achieve good results for TNR, but it is because of the majority prediction, i.e. the instances that cannot be identified will be predicted as majority class if no rules can be applied. Consider the proposed method $\text{GBMGP}(s,c,S)_i$: it obtained good results for TPR because of minority prediction. However, the results for TNR were very poor. Therefore, using token competition can improve the diversity of the population and separate the individuals with different meanings while keeping good performance. In particular with the two real-life problems, although the minority class (i.e. positive class) is more important than the majority class (i.e. negative class), sometimes the users may not expect to have very low accuracy in classifying negative examples. In this case, the proposed method with token competition should be applied. But in other situations, it also provides another choice for users if they only focused on the positive class (i.e. fraudulent firms) and are not concerned about relatively low accuracy for the negative class. In this situation, the proposed method without using token competition can be used. The following empirical and statistical tests focus on the comparison between GBMGP-SSL with minority prediction and the other approaches.

4.3.2 Empirical Analysis

Each problem had two results for TPR and TNR respectively. We set each one as a competition, and therefore eight competitions were included for four datasets. The empirical $w/t/l$ (i.e. win, tie and lose) test results are given in the last column of Table 10 and Table 11, where w means that GBMGP-SSL with minority prediction outperforms the compared approach, t means that GBMGP-SSL with minority prediction has the same results, and l means that GBMGP-SSL with minority prediction is not good as the compared approach.

In Table 11, compared with BayesNetwork, the proposed method wins all competitions in regard to TPR, but also loses all of them with regard to TNR. BayesNetwork can be regarded as a generic method for solving FFD problems. Although it obtained the highest TNR for Australia and the second highest TPR for U.S.CSF, it also obtained very poor TPR results in the CCSF dataset. In this study, the real-life datasets were more important than the benchmark datasets. Compared with the traditional statistical method, i.e. logistic regression, the proposed method won six competitions and lost two, which were TNRs from “credit approval” and “U.S. CSF”. Especially for “U.S. CSF”, the logistic regression had a biased result for TNR. However, logistic regression is still a competitive method compared with other approaches. Compared with other machine learning techniques, i.e. MultilayerNN, SMO and Decision Tree, the proposed method respectively won 5, 6 and 5 competitions. On the other hand, it respectively lost in 3, 2 and 3 competitions, which were also related to TNRs. SMO had a similar performance to that of BayesNetwork, but Decision Tree had extremely biased results with regard TNRs. Therefore, Decision Tree may not be an appropriate method for imbalanced financial datasets, since it is very sensitive to noisy instances. Compared with ensemble learning techniques, i.e. AdaBoost, Bagging and LogitBoost, the proposed method respectively won 5, 5 and 5 competitions, and respectively tied in 0,1 and 0 competitions. Therefore, the proposed method is able to perform better results than other ensemble learning techniques even in TNR competitions. Bagging generated extremely biased results in relation to TNR on two real-life financial fraud datasets, and also it may not be appropriate model for imbalanced financial datasets. The base-level classifiers of Bagging are also Decision Trees. Therefore it has the similar problem as do Decision Trees. In addition, comparing the ensemble learning techniques with the logistic regression in U.S. CSF and CCSF datasets, the overall performance by using ensemble-learning techniques (except for Bagging) improved for TPRs.

According to Table 10, compared with $\text{GBGP}(\text{s,c})_a, \text{GBMGP}(\text{s,c,W})_i$ and all the GBMGP variants with majority prediction,¹ the proposed method won all of the competitions for TPR, but also lost all of them for TNR. GBMGP variants with majority prediction produced good TNR results because of the majority prediction to enhance the majority (i.e. negative) class. Compared with other GBGP variants,² $\text{GBGP}(\text{s,c,M})_i$ and $\text{GBGP}(\text{s,c,W})_i$, the proposed method respectively won 6, 6, 6, 6, 7 and 7 competitions, and respectively tied in 0, 0, 1, 1, 0 and 1. This indicates that the proposed method outperformed other GBGP variants no matter whether majority or minority prediction was applied. Moreover, it also indicates that the multi-objective and the new statistical selection ensemble learning technique together can improve the results for most TPRs and TNRs.

Finally, compared with other remaining GBMGP variants with minority prediction,³ the proposed method respectively won 6 and lost in 2 competitions. The two lost competitions were for TNRs from the “Australian credit” and “credit approval” datasets. This indicates that the $\text{GBMGP}(\text{s,c})$ and $\text{GBMGP}(\text{s,c,M})$ with minority prediction maybe more suitable for balanced datasets. They achieved similar results for TPRs and TNRs on four problems. Using majority voting in GBMGP did not improve a lot of benchmark datasets, but it improved CCSF problem for TPR and TNR.

In addition, relative improvement (RAI) (Chen et al., 2014) was applied to evaluate the approaches, calculated by Equation (21).

$$p = \sum \frac{\alpha_i - \alpha'_i}{\alpha'_i} \quad (21)$$

where α_i denotes the accuracy of the GBMGP-SSL with minority prediction in the i^{th} dataset and α'_i refers to the accuracy of the approach being compared with.

¹i.e. $\text{GBMGP}(\text{s,c})_a, \text{GBMGP}(\text{s,c,W})_a, \text{GBMGP}(\text{s,c,M})_a$ and $\text{GBMGP}(\text{s,c,S})_a$

²i.e. $\text{GBGP}(\text{s,c,M})_a, \text{GBGP}(\text{s,c})_i, \text{GBGP}(\text{s,c,W})_a, \text{GBGP}(\text{s,c})_i$

³i.e. $\text{GBMGP}(\text{s,c,M})_i$ and $\text{GBMGP}(\text{s,c})_i$

In Table 14, RAI (TPR) indicates the relative improvements for TPR and RAI (TNR) indicates the relative improvements for TNR. According to the RAI test in Table 14, the proposed method improved the results for both TPRs and TNRs better than Logistic regression, GBGP variants with minority prediction,⁴ and GBMGP(s,c)_i. For most of the other techniques, the proposed method reduced the results for TNRs relatively little compared with the significant improvements for TPRs, especially for Bagging, Decision Trees, GBGP(s)_a and GBGP(s)_i. As discussed before, Bagging and Decision Tree may not be suitable methods for solving financial datasets. In addition, the classification accuracy was not considered in GBGP variants with single objective *support*, therefore GBGP(s)_a and GBGP(s)_i also performs very poor results.

4.3.3 Statistical Analysis

Pairwise t-test was applied to demonstrate the statistical significance of the experiments. The performance of the proposed method and other approaches was compared to calculate statistical significance. The results of the t-test are shown in Table 10 and Table 11, which highlight the statistical significance at the 5% level by the symbol “++” if the proposed method was better than the compared method, and the symbol “+” if the proposed method was significantly better than the compared method at the 10% level. On the other hand, symbol “- -” and “-” are used if the proposed method was significantly worse than the compared method at the 5% level and 10% level respectively. For example in Table 10, compared with traditional logistic regression, the proposed method significantly outperformed it in the 5 of 8 metrics at the 5% level. However, the proposed method was significantly worse than logistic regression for TNRs in “credit approval” and “U.S.CSF” at the 10% level and 5% level respectively. In addition, the proposed method was significantly better than MultilayerNN in the 4 of 8 metrics at the 5% level. Compared with SMO, it was significantly

⁴GBGP(s,c)_i, GBGP(s,c,M)_i, GBGP(s,c,W)_i

	RAI (TPR)	RAI (TNR)
Logistic	130%	25%
MultilayerNN	170%	-39%
SMO	98%	-19%
BayesNetwork	186%	-51%
Decision Tree	720%	-56%
AdaBoostM1	91%	-21%
Bagging	712%	-46%
LogitBoost	94%	-31%
GBGP(s,c) _a	141%	-31%
GBGP(c) _a	292%	6%
GBGP(s) _a	5,358%	-78%
GBGP(s,c, M) _a	243%	-30%
GBGP(s,c, W) _a	152%	-7%
GBMGP(s,c) _a	105%	-32%
GBMGP (s,c,W) _a	182%	-50%
GBMGP (s,c, M) _a	93%	-49%
GBMGP(s,c,S) _a	48%	-47%
GBGP(s,c) _i	73%	16%
GBGP(c) _i	184%	108%
GBGP(s) _i	2,397%	-84%
GBGP(s,c, M) _i	53%	9%
GBGP(s,c, W) _i	64%	23%
GBMGP(s,c) _i	96%	6%
GBMGP (s,c,W) _i	111%	-22%
GBMGP (s,c, M) _i	69%	-1%
GBMGP(s,c,S) _i	-	-

Table 14. RAI test result

superior in the 3 of 8 metrics at the 5% level, and 1 metric at the 10% level. Compared with BayesNetwork, it was significantly superior in the 3 of 8 metrics at the 5% level, and 1 metric at the 10% level. It was also significantly superior to Decision Trees in 4 metrics at the 5% level. Moreover, it also outperformed ensemble techniques such as AdaBoost and LogitBoost in 3 of the 8 metrics at the 5% level, and 1 metric at the 10% level, etc.

For the benchmark problems in Table 10, the proposed method significantly outperformed all the data mining techniques for TPR in regard to Australia credit and credit approval, except for SMO for Australia credit. Moreover, it also

significantly outperformed SMO for TNR in Australia credit at the 5% level, but was significantly worse than the logistic regression for TNR in credit approval at the 10% level. For the U.S. CSF problem, the proposed method significantly outperformed Logistic regression, MultilayerNN, Decision Trees and Bagging for TPR at the 5% level, and it also significantly outperformed SMO, BayesNetwork, AdaBoost and LogitBoost for TPR at the 10% level. However, the TNR of using the proposed method was significantly worse than all the data mining methods at the 10% level. For the CCSF problem, the proposed method significantly outperformed all the data mining methods for TPR at the 5% level, but was also significantly worse than all the data mining methods at the 10% level, except for the Logistic regression. It is easy to obtain very good results on the majority (i.e. negative) class by applying the traditional statistical method, advanced data mining techniques or ensemble learning techniques. As discussed in Section 2.2, the detection of fraudulent firms is much more important than the classification of non-fraudulent firms. The proposed method seemed to reduce by a few percent the accuracy of non-fraudulent classification, but it significantly increased the performance in identifying fraudulent firms.

CHAPTER 5

Conclusion

5.1 Major Findings

Financial fraud has become an increasingly serious problem in economics, finance and management. Financial fraud detection (FFD) is vital for the prevention of the destructive consequences of financial fraud. Data mining plays a significant role in solving FFD problems. In this study, we conducted a comprehensive comparison of data mining techniques and suggested a new approach to identify fraudulent information from four financial datasets. The applied data mining techniques included Logistic Regression (LR), Neural Networks (NNs), Support Vector Machine (SVM), Bayesian Networks (BNs) and Decision Trees (DTs). The applied ensemble learning methods are AdaBoost, Bagging and LogitBoost. We can conclude several findings from the experimental results.

- *The first is the performance of the existing data mining methods on the given FFD datasets.*

All the applied data mining techniques and ensemble-learning methods performed well on the benchmark datasets. In the Australia Credit dataset, Decision Trees could produce better classification results for positive class (i.e. fraudulent) than other methods. It also produced very high results in classifying the negative class (i.e. non-fraudulent), but the highest result was obtained by Bayesian Networks. In the Credit Approval dataset, Logistic Regression could generate the highest results for both classes. Moreover, Decision Trees also produced the highest accuracy in classifying the negative class, but the value of its variance was larger than for Logistic Regression. Ensemble learning techniques could not produce outstanding results for the benchmark datasets.

Real-life problems are not easy to handle. All the methods performed poorly in classifying fraudulent firms, but well for non-fraudulent firms classification. For the U.S. CSF dataset, all methods had very high variances, and poor results for fraudulent firms classification. Decision Trees and Bagging methods obtained the lowest accuracies for the fraudulent class, but their accuracies for the non-fraudulent class were the highest. This shows that they could not be used for this dataset, since the results were extremely biased towards the majority class (i.e. non-fraudulent). For the CCSF dataset, the situation was similar to that for the U.S. CSF dataset. All the methods could not perform well for the fraudulent class. However, their variances were much lower compared to the situation in the U.S. CSF dataset. Ensemble methods produced relatively better results than the data mining methods in the CCSF dataset, except for Bagging.

- *The second is the performance of GBGP variants and GBMGP variants for the given FFD datasets.*

In order to evaluate the features of the proposed method, such as Multi-objective optimization and ensemble techniques, we developed several GBGP variants and GBMGP variants and compared their performance with the original GBGP. All GBGP variants and GBMGP variants produced competitive results compared with data mining and ensemble methods. For the benchmark datasets, all GBGP variants and GBMGP variants that used minority prediction did not improve the classification results for both classes significantly compared to the original GBGP. But GBMGP variants that used majority prediction with ensemble methods (i.e. weighted voting and majority voting) improved the average accuracy for the negative (i.e. non-fraudulent) class compared with the traditional GBGP.

For the U.S. CSF and CCSF datasets, many of the GBMGP variants produced

better results than the original GBGP. Minority prediction improved all GBGP variants and GBMGP variants in regard to classifying the positive (i.e. fraudulent) class. Ensemble methods in GBGP variants did not improve the results significantly. But ensemble methods with GBMGP variants to varying degrees produced better results for the positive class compared with the original GBGP.

- *The third is the proposed method and comparison with all other methods.*

The proposed method produced significantly better results than most of the other methods. Especially for fraudulent classification (i.e. positive class), the proposed method obtained the highest accuracies of Tree Positive Rate (i.e. TPR) for the given FFD datasets. However, the non-fraudulent classification (i.e. negative class) accuracies dropped slightly. The results by using the proposed method are acceptable if it improves the performance of the fraudulent detection significantly, because we think that it is more important to spot fraudulent instances than non-fraudulent instances. If a non-fraudulent instance is misclassified as a fraudulent instance, related organizations (e.g. Securities Regulatory Commission or auditors) and firms may need to undertake further investigations manually at low expense for such case. But if it misclassifies a fraudulent instance as a non-fraudulent instance, the cost due to the misclassification could be much worse than in the previous scenario.

- *The fourth is the proposed method with and without using token competition.*

We also evaluated the performance of the proposed method and its variants without using token competition. The results were very different. The accuracies in classifying the positive class were significantly increased, but meanwhile the accuracies for the negative class were decreased significantly, to even less than 50%. It provides another choice for users, if they only focus on the positive class (e.g. fraudulent firms) and do not concern themselves with the relatively low accuracy for the negative class then they should apply the proposed method

without using token competition.

5.2 Contributions

There are five major contributions of this study. The first contribution is that we made a number of comprehensive comparisons between different approaches to solving the FFD problem. They included one traditional statistical method: Logistic Regression, four data mining techniques: Neural Networks, Support Vector Machines, Bayesian Networks and Decision Trees and three ensemble learning methods: AdaBoost, Bagging and LogitBoost. In addition, we also developed a number of GBGP variants and GBMGP variants with different ensemble methods for comparison.

The second contribution is that we proposed a new method called Grammar-based Multi-objective Genetic Programming (GBMGP) with Statistical Selection Learning (SSL) that can take advantage of the Evolutionary Algorithm and the concept of ensemble learning. By selecting a number of good classification rules, the proposed method can be used to predict fraudulent instances. The performance of each individual classification rule was optimized by the use of a support and confidence framework. Previously, the framework was applied as a combined version, but we separated them into two individual objectives and applied a multi-objective method (i.e. NSGA-II) to optimize them. The comprehensive experimental results show that the proposed method performs well on the given FFD problems. It is a combination of ensemble learning and Grammar-based Multi-objective Genetic Programming approach in Financial Fraud Detection, and no one has done this before.

The third contribution of the study is the diversity maintenance between individuals. In general, by using NSGA-II as the Multiobjective optimization

algorithm, the diversity can be maintained well based on the location of each solution. However, in this study, each solution was a classification rule. It is possible that some individuals were located in a crowded area (i.e. high density region), since they had close values on two objectives. However, they were not truly similar to each other if they had different meanings. In this case, we applied token competition to adjust the location (i.e. objective values) of each solution. The diversity could be maintained, since the individuals with similar objective values but different meanings were separated.

The fourth contribution is unbalanced dataset processing. In general, all methods cannot be applied directly to learn unbalanced datasets (i.e. real-life problems). In this study, we applied the SMOTE approach to the two real-life datasets in order to assist all methods, and the proposed method GBMGP-SSL was able to obtain better performance in classifying fraudulent firms than the other methods.

The last contribution is ensemble learning. We also faced the problem in selecting individuals, because many individuals were evolved by the GBMGP. Based on the previous research and significant practical applications of ensemble learning, we applied several ensemble methods, such as majority voting, combined with GBGP variants and GBMGP variants. By applying ensemble techniques, such as majority voting, the classification accuracy of fraudulent firms could be improved slightly. In addition, the proposed ensemble technique (i.e. Statistical Selection Learning) could achieve better performances in classifying fraudulent firms than others in the two real-life FFD problems.

5.3 Implications

The major findings from this thesis may have important implications for the Securities and Exchange Commission (SEC) and China Securities Regulatory Commission (CSRC) to facilitate their work. Moreover, it may also have useful implications for other data mining researchers. In this study, a GBMGP-SSL method was proposed to optimize the classification accuracy. SMOTE was applied as a data processing method to assist all used methods. We made comprehensive comparisons between traditional statistical method, data mining methods, ensemble learning techniques and the proposed method for solving the FFD problems. Therefore, the study can be utilized as a guideline for data mining research in financial fraud detection. We also highlighted some potential issues when using classification rules. For example, if some testing instances could not be classified by any rules, they were ignored. We did not want to ignore any testing instances; therefore we suggested using minority prediction and majority prediction to handle this issue. Moreover, NSGA-II also has a potential issue in diversity maintenance for crowded solutions (i.e. similar individuals). The original diversity scheme could not separate similar solutions if they had different meanings. Therefore, a token competition was applied with the proposed method.

The application of classification rules in FFD provides understandable results for users, even if they are not experts in the relevant areas. The combination of GBGP, NSGA-II, Token Competition and Statistical Selection Learning also provides a novel and powerful approach to data mining research. The proposed method outperforms many other data mining techniques in classifying the fraudulent class. The users in the Securities and Exchange Commission (SEC) and China Securities Regulatory Commission (CSRC) can refer to the results from the proposed method.

5.4 Limitations and Directions for Future Research

Although the proposed method has promising results in solving the FFD problems, there are some limitations of this research. Due to the large volume of real-life datasets, the evaluation process of using the proposed method is time-consuming. But for some other methods, such as Decision Trees, they can generate results quickly. Therefore, if possible, the executing time of the program can be shortened in future by using powerful machines.

In the research area of finance or data mining, it will be more interesting if researchers can apply different objectives (e.g. risk and return) to the proposed method, and evaluate the method for the two real-life datasets to see whether it can generate interesting results.

We have evaluated the proposed method on the given FFD problems, but it has not been used for solving other real-life problems in different areas. The proposed method can be considered as one of the methods to solve different problems. For example, the proposed method can be used for solving direct marketing problems. In direct marketing, the evaluation of a method is usually based on response rate and profit. Response rate is the ratio of the number of responders to the total number of customers in the dataset. Profit is the profit generated by the responders. A high value of response rate may not produce high profits. Therefore, it is necessary to find responders who can generate high profit. In this problem, the minority class is high-profit customers and majority class is low-profit customers. The generated rules can identify the high-profit customers, low-profit customers and non-response customers if the objectives of the proposed method are modified to response rate and profit. In addition, except for the classification rules, a regression method should be applied for predicting the profit, and then the performance of the proposed method can be easily compared with

other direct marketing methods.

The last limitation of the study is that, we applied the NSGA-II as the multi-objective optimization method, and it may not be the best option for solving the given FFD problems. Therefore, the study could include more comparisons with other multi-objective methods, such as Strength Pareto Evolutionary Algorithm 2 (SPEA-II).

APPENDIX

Appendix A

A.1 Grammar

A.1.1 Australian Credit

Australian credit	
Rule -> "if" Ante Consq	
Ante -> "and" aPart bPart	
Consq -> "is" "type" [value type]	
#	
aPart -> "and" V1 V2 V3 V4 V5 V6 V7 V8	
V1 ->	"any" "is" "var1" [value var1]
V2 ->	"any" "between" "var2" [value var2] [randomvalue var2]
V3 ->	"any" "between" "var3" [value var3] [randomvalue var3]
V4 ->	"any" "is" "var4" [value var4]
V5 ->	"any" "between" "var5" [value var5] [randomvalue var5]
V6 ->	"any" "between" "var6" [value var6] [randomvalue var6]
V7 ->	"any" "between" "var7" [value var7] [randomvalue var7]
V8 ->	"any" "is" "var8" [value var8]
#	
bPart -> "and" V9 V10 V11 V12 V13 V14	
V9 ->	"any" "is" "var9" [value var9]
V10 ->	"any" "between" "var10" [value var10] [randomvalue var10]
V11 ->	"any" "is" "var11" [value var11]
V12 ->	"any" "is" "var12" [value var12]
V13 ->	"any" "between" "var13" [value var13] [randomvalue var13]
V14 ->	"any" "between" "var14" [value var14] [randomvalue var14]
#	

Figure A.1. Australian Credit Grammar.

A.1.2 Credit Approval

Credit approval	
Rule -> "if" Ante Consq	
Ante -> "and" aPart bPart	
Consq -> "is" "type" [value type]	
#	
aPart -> "and" V1 V2 V3 V4 V5 V6 V7 V8	
V1 ->	"any" "is" "var1" [value var1]
V2 ->	"any" "between" "var2" [value var2] [randomvalue var2]
V3 ->	"any" "between" "var3" [value var3] [randomvalue var3]
V4 ->	"any" "is" "var4" [value var4]
V5 ->	"any" "is" "var5" [value var5]
V6 ->	"any" "is" "var6" [value var6]
V7 ->	"any" "is" "var7" [value var7]
V8 ->	"any" "between" "var8" [value var8] [randomvalue var8]
#	
bPart -> "and" V9 V10 V11 V12 V13 V14	
V9 ->	"any" "is" "var9" [value var9]
V10 ->	"any" "is" "var10" [value var10]
V11 ->	"any" "between" "var11" [value var11] [randomvalue var11]
V12 ->	"any" "is" "var12" [value var12]
V13 ->	"any" "is" "var13" [value var13]
V14 ->	"any" "between" "var14" [value var14] [randomvalue var14]
V15 ->	"any" "between" "var15" [value var15] [randomvalue var15]
#	

Figure A.2. Credit Approval Grammar.

A.1.3 U.S. Corporate Securities Fraud

U.S. Corporate securities fraud (US-CSF)	
Rule -> "if" Ante Consq	
Ante -> "and" aPart bPart cPart dPart ePart fPart gPart hPart iPart jPart	
Consq -> "is" "fraud" [value fraud]	
aPart -> "and" ACO ACOMINC AO AP	
bPart -> "and" AT CEQ CH CIOther	
cPart -> "and" CITOTAL COGS CSTKE CURRTR	
dPart -> "and" DLC DLTT DO DVP	
ePart -> "and" EBIT EXCHG GP IB	
fPart -> "and" ICAPT LCO LO LSE	
gPart -> "and" LT NI NOPI NOPIO	
hPart -> "and" NP OIADP PI PSTK	
iPart -> "and" PSTKR REVT SEQ SRC	
jPart -> "and" TXT XI XIDO XIDOC XOPR	
ACO ->	"any" "between" "aco" [value aco] [randomvalue aco]
ACOMINC ->	"any" "between" "acominc" [value acominc] [randomvalue acominc]
AO ->	"any" "between" "ao" [value ao] [randomvalue ao]
AP ->	"any" "between" "ap" [value ap] [randomvalue ap]
AT ->	"any" "between" "at" [value at] [randomvalue at]
CEQ ->	"any" "between" "ceq" [value ceq] [randomvalue ceq]
CH ->	"any" "between" "ch" [value ch] [randomvalue ch]
CIOther ->	"any" "between" "ciother" [value ciother] [randomvalue ciother]
CITOTAL ->	"any" "between" "citotal" [value citotal] [randomvalue citotal]
COGS ->	"any" "between" "cogs" [value cogs] [randomvalue cogs]
CSTKE ->	"any" "between" "cstke" [value cstke] [randomvalue cstke]
CURRTR ->	"any" "between" "currtr" [value currtr] [randomvalue currtr]
DLC ->	"any" "between" "dlc" [value dlc] [randomvalue dlc]
DLTT ->	"any" "between" "dltt" [value dltt] [randomvalue dltt]
DO ->	"any" "between" "do" [value do] [randomvalue do]
DVP ->	"any" "between" "dvp" [value dvp] [randomvalue dvp]
EBIT ->	"any" "between" "ebit" [value ebit] [randomvalue ebit]
EXCHG ->	"any" "between" "exchg" [value exchg] [randomvalue exchg]
GP ->	"any" "between" "gp" [value gp] [randomvalue gp]
IB ->	"any" "between" "ib" [value ib] [randomvalue ib]
ICAPT ->	"any" "between" "icapt" [value icapt] [randomvalue icapt]
LCO ->	"any" "between" "lco" [value lco] [randomvalue lco]
LO ->	"any" "between" "lo" [value lo] [randomvalue lo]
LSE ->	"any" "between" "lse" [value lse] [randomvalue lse]
LT ->	"any" "between" "lt" [value lt] [randomvalue lt]
NI ->	"any" "between" "ni" [value ni] [randomvalue ni]
NOPI ->	"any" "between" "nopi" [value nopi] [randomvalue nopi]
NOPIO ->	"any" "between" "nopio" [value nopio] [randomvalue nopio]
NP ->	"any" "between" "np" [value np] [randomvalue np]
OIADP ->	"any" "between" "oiadp" [value oiadp] [randomvalue oiadp]
PI ->	"any" "between" "pi" [value pi] [randomvalue pi]
PSTK ->	"any" "between" "pstk" [value pstk] [randomvalue pstk]
PSTKR ->	"any" "between" "pstk" [value pstkr] [randomvalue pstkr]
REVT ->	"any" "between" "revt" [value revt] [randomvalue revt]
SEQ->	"any" "between" "seq" [value seq] [randomvalue seq]
SRC->	"any" "between" "src" [value src] [randomvalue src]
TXT ->	"any" "between" "txt" [value txt] [randomvalue txt]
XI ->	"any" "between" "xi" [value xi] [randomvalue xi]
XIDO ->	"any" "between" "xido" [value xido] [randomvalue xido]
XIDOC ->	"any" "between" "xidoc" [value xidoc] [randomvalue xidoc]
XOPR ->	"any" "between" "xopr" [value xopr] [randomvalue xopr]
#	

Figure A.3. U.S. Corporate Securities Fraud.

A.1.4 China Corporate Securities Fraud

China corporate securities fraud (CCSF)
<p>Rule -> "if" Ante Consq</p> <p>Ante -> "and" aPart bPart cPart dPart ePart fPart</p> <p>Consq -> "is" "enforcement" [value enforcement]</p> <p>aPart -> "and" Location Industry Market</p> <p>bPart -> "and" ABshare Asset ShortTerm</p> <p>cPart -> "and" Operating LongTerm Earning</p> <p>dPart -> "and" RiskLevel ROE</p> <p>ePart -> "and" HDividend Dividend NOS</p> <p>fPart -> "and" NOE ChairCEO NOM</p> <p>Location -> "any" "is" "location" [value location]</p> <p>Industry -> "any" "is" "industry" [value industry]</p> <p>Market -> "any" "is" "market" [value market]</p> <p>ABshare -> "any" "is" "abshare" [value abshare]</p> <p>Asset -> "any" "between" "asset" [value asset] [randomvalue asset]</p> <p>ShortTerm -> "any" "between" "shortterm" [value shortterm] [randomvalue shortterm]</p> <p>Operating -> "any" "between" "operating" [value operating] [randomvalue operating]</p> <p>LongTerm -> "any" "between" "longterm" [value longterm] [randomvalue longterm]</p> <p>Earning -> "any" "between" "earning" [value earning] [randomvalue earning]</p> <p>RiskLevel -> "any" "between" "risklevel" [value risklevel] [randomvalue risklevel]</p> <p>ROE -> "any" "between" "roe" [value roe] [randomvalue roe]</p> <p>HDividend -> "any" "is" "hdividend" [value hdividend]</p> <p>Dividend -> "any" "between" "dividend" [value dividend] [randomvalue dividend]</p> <p>NOS -> "any" "between" "shareholders" [value shareholders] [randomvalue shareholders]</p> <p>NOE -> "any" "between" "employees" [value employees] [randomvalue employees]</p> <p>ChairCEO -> "any" "is" "chairCEO" [value chairCEO]</p> <p>NOM -> "any" "between" "meeting" [value meeting] [randomvalue meeting]</p> <p>#</p>

Figure A.4. China Corporate Securities Fraud Grammar.

Bibliography

- Anup Agrawal and Sahiba Chadha. Corporate governance and accounting scandals. *Journal of law and economics*, 48(2):371–406, 2005.
- Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, 1993.
- Arthur Asuncion and David Newman. UCI machine learning repository, 2007.
- Belinna Bai, Jerome Yen, and Xiaoguang Yang. False financial statements: characteristics of China’s listed companies and CART detecting approach. *International journal of information technology & decision making*, 7(02):339–359, 2008.
- Michael J. Berry and Gordon S. Linoff. *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons, 2004.
- Richard J. Bolton and David J. Hand. Statistical fraud detection: A review. *Statistical Science*, pages 235–249, 2002.
- Indranil Bose and Radha K. Mahapatra. Business data mining: a machine learning perspective. *Information & management*, 39(3):211–225, 2001.
- R. Brause, T. Langsdorf, and Michael Hepp. Neural data mining for credit card fraud detection. In *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*, pages 103–106. IEEE, 1999.
- Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.
- Leo Breiman, Jerome Friedman, Charles J. Stone, and Richard A. Olshen. *Classification and regression trees*. CRC press, 1984.
- Philip K. Chan and Salvatore J. Stolfo. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 164–168. AAAI Press, 1998.
- Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *arXiv preprint arXiv:1106.1813*, 2011.
- Gongmeng Chen, Michael Firth, Daniel N. Gao, and Oliver M. Rui. Is China’s securities regulatory agency a toothless tiger? evidence from enforcement actions. *Journal of Accounting and Public Policy*, 24(6):451–488, 2005.
- Gongmeng Chen, Michael Firth, Daniel N. Gao, and Oliver M. Rui. Ownership structure, corporate governance, and fraud: Evidence from China. *Journal of Corporate Finance*, 12(3):424–448, 2006.

- Yijun Chen, Man-Leung Wong, and Haibing Li. Applying ant colony optimization to configuring stacking ensembles for data mining. *Expert Systems with Applications*, 41(6):2688–2702, 2014.
- Leonid Churilov, Adyl Bagirov, Daniel Schwartz, Kate Smith, and Michael Dally. Data mining with combined use of optimization techniques and self-organizing maps for improving risk grouping rules: application to prostate cancer patients. *Journal of Management Information Systems*, 21(4):85–100, 2005.
- Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary algorithms for solving multiobjective problems*. Springer, 2007.
- James D. Cox, Randall S. Thomas, and Dana Kiku. SEC enforcement heuristics: An empirical inquiry. *Duke Law Journal*, pages 737–779, 2003.
- Geng Cui, Man Leung Wong, and Hon-Kwong Lui. Machine learning for direct marketing response models: Bayesian networks with evolutionary programming. *Management Science*, 52(4):597–612, 2006.
- D.J. Cumming, Wenxuan Hou, and Edward Lee. The role of financial analysts in deterring corporate fraud in China. Technical report, Working paper, SSRN, 2011.
- Andrea Dal Pozzolo, Olivier Caelen, Yann-Aël Le Borgne, Serge Waterschoot, and Gianluca Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Systems with Applications*, 41(10):4915–4928, 2014.
- Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.
- Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.
- Alexander Dyck, Adair Morse, and Luigi Zingales. Who blows the whistle on corporate fraud? *The Journal of Finance*, 65(6):2213–2253, 2010.
- Hamidreza Eskandari and Christopher D. Geiger. A fast pareto genetic algorithm approach for solving expensive multiobjective optimization problems. *Journal of Heuristics*, 14(3):203–241, 2008.
- David B. Farber. Restoring trust after fraud: Does corporate governance matter? *The Accounting Review*, 80(2):539–561, 2005.
- Federal Bureau of Investigation FBI. Financial crimes report to the public fiscal year. Technical report, Department of Justice, United States, 2007.
- Carlos M. Fonseca, Peter J. Fleming, et al. Genetic algorithms for multiobjective optimization: Formulation discussion and generalization. In *ICGA*, volume 93, pages 416–423, 1993.
- Yoav Freund, Robert E. Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

- Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, 28(2):337–407, 04 2000.
- Gilbert Geis and Paul Jesilow. *White-collar crime*. Sage Periodicals Press, 1993.
- Andreas Geyer-Schulz and A. Geyer-Schulz. *Fuzzy rule-based expert systems and genetic machine learning*. Physica-Verlag Heidelberg, 1997.
- Ashish Ghosh and Bhabesh Nath. Multiobjective rule mining using genetic algorithms. *Information Sciences*, 163(1):123–133, 2004.
- Stuart L. Gillan. Recent developments in corporate governance: An overview. *Journal of corporate finance*, 12(3):381–402, 2006.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.
- Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011. ISBN 0123814790, 9780123814791.
- Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.
- David Heckerman. Bayesian networks for data mining. *Data mining and knowledge discovery*, 1(1):79–119, 1997.
- Benjamin E. Hermalin and Michael S. Weisbach. Information disclosure and corporate governance. *The Journal of Finance*, 67(1):195–233, 2012.
- Xiaoming Huo, Seoung Bum Kim, Kwok-Leung Tsui, and Shuchun Wang. FBP: a frontier-based tree-pruning algorithm. *INFORMS Journal on Computing*, 18(4):494–505, 2006.
- Martin A. Keane, Matthew J. Streeter, William Mydlowec, Guido Lanza, and Jessen Yu. *Genetic programming IV: Routine human-competitive machine intelligence*, volume 5. Springer, 2006.
- Efstathios Kirkos, Charalambos Spathis, and Yannis Manolopoulos. Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, 32(4):995–1003, 2007.
- Abdullah Konak, David W Coit, and Alice E. Smith. Multiobjective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91(9):992–1007, 2006.
- S. Kotsiantis, E. Koumanakos, D. Tzelepis, and V. Tampakas. Forecasting fraudulent financial statements using data mining. *International Journal of Computational Intelligence*, 3(2):104–110, 2006.

- Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. Survey of fraud detection techniques. In *Networking, sensing and control, 2004 IEEE international conference on*, volume 2, pages 749–754. IEEE, 2004.
- Jerry W. Lin, Mark I. Hwang, and Jack D. Becker. A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, 18(8):657–665, 2003.
- Alexander Liu, Joydeep Ghosh, and Cheryl E. Martin. Generative oversampling for mining imbalanced datasets. In *DMIN*, pages 66–72, 2007.
- Robert I. McKay, Nguyen Xuan Hoai, Peter Alexander Whigham, Yin Shan, and Michael O’Neill. Grammar-based genetic programming: a survey. *Genetic Programming and Evolvable Machines*, 11(3-4):365–396, 2010.
- Georg Meyer, Gediminas Adomavicius, Paul E. Johnson, Mohamed Elidrisi, William A. Rush, JoAnn M Sperl-Hillen, and Patrick J. O’Connor. A machine learning approach to improving dynamic decision making. *Information Systems Research*, 25(2):239–263, 2014.
- Antonio J. Nebro, Juan J. Durillo, Francisco Luna, Bernabé Dorronsoro, and Enrique Alba. Mocell: A cellular genetic algorithm for multiobjective optimization. *International Journal of Intelligent Systems*, 24(7):726–746, 2009.
- E.W.T. Ngai, Yong Hu, Y.H. Wong, Yijun Chen, and Xin Sun. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569, 2011.
- Po Shun Ngan and Kwong Sak Leung. Using grammar based genetic programming for data mining of medical knowledge. In *Genetic Programming: Proceedings Of the Third Annual Conference*. Citeseer, 1998.
- Michael O’Neil and Conor Ryan. Grammatical evolution. In *Grammatical Evolution*, pages 33–47. Springer, 2003.
- Balaji Padmanabhan, Alan Hevner, Michael Cuenco, and Crystal Shi. From information to operations: Service quality and customer retention. *ACM Transactions on Management Information Systems (TMIS)*, 2(4):21, 2011.
- Gisele L. Pappa and Alex A. Freitas. Evolving rule induction algorithms with multiobjective grammar-based genetic programming. *Knowledge and information systems*, 19(3):283–309, 2009.
- Jeffrey Parsons and Yair Wand. Using cognitive principles to guide classification in information systems modeling. *Mis Quarterly*, pages 839–868, 2008.
- Clifton Phua, Vincent Lee, Kate Smith, and Ross Gayler. A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*, 2010.
- John Platt. Sequential minimal optimization: A fast algorithm for training Support Vector Machines. Technical report, Microsoft Research, April 1998.

- Antonin Ponsich, Antonio López Jaimes, and Carlos A Coello Coello. A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *Evolutionary Computation, IEEE Transactions on*, 17(3):321–344, 2013.
- Pediredla Ravisankar, V. Ravi, G. Raghava Rao, and Indranil Bose. Detection of financial statement fraud and feature selection using data mining techniques. *Decision Support Systems*, 50(2):491–500, 2011.
- Brian D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 1996.
- Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2): 1–39, 2010.
- Daniel Sánchez, M.A. Vila, L. Cerda, and José-María Serrano. Association rules applied to credit card fraud detection. *Expert Systems with Applications*, 36(2):3630–3640, 2009.
- Robert E. Schapire. The strength of weak learnability. *Machine learning*, 5(2): 197–227, 1990.
- Aihua Shen, Rencheng Tong, and Yaochen Deng. Application of classification models on credit card fraud detection. In *Service Systems and Service Management, 2007 International Conference on*, pages 1–4. IEEE, 2007.
- Galit Shmueli, Nitin R. Patel, and Peter C. Bruce. *Data mining for business intelligence: Concepts, techniques, and applications in Microsoft Office Excel with XLMiner*. John Wiley & Sons, 2007.
- Charalambos T. Spathis. Detecting false financial statements using published data: some evidence from Greece. *Managerial Auditing Journal*, 17(4):179–191, 2002.
- Nidamarthi Srinivas and Kalyanmoy Deb. Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary computation*, 2(3): 221–248, 1994.
- Abhinav Srivastava, Amlan Kundu, Shamik Sural, and Arun K. Majumdar. Credit card fraud detection using Hidden Markov Model. *Dependable and Secure Computing, IEEE Transactions on*, 5(1):37–48, 2008.
- Dan Steinberg and Phillip Colla. Cart: classification and regression trees. *The Top Ten Algorithms in Data Mining*, 9:179, 2009.
- Zhan-Li Sun, Tsan-Ming Choi, Kin-Fan Au, and Yong Yu. Sales forecasting using extreme learning machine with applications in fashion retailing. *Decision Support Systems*, 46(1):411–419, 2008.
- Mubeena Syeda, Yan-Qing Zhang, and Yi Pan. Parallel granular neural networks for fast credit card fraud detection. In *Fuzzy Systems, 2002. FUZZ-IEEE’02. Proceedings of the 2002 IEEE International Conference on*, volume 1, pages 572–577. IEEE, 2002.

- E. Turban, R. Sharda, D. Delen, J.E. Aronson, T.P. Liang, and D. King. Decision support and business intelligence systems, 9e édition, 2011. 2011.
- Jau-Hwang Wang, You-Lu Liao, Tyan-muh Tsai, and Garfield Hung. Technology-based financial frauds in Taiwan: issues and approaches. In *SMC*, pages 1120–1124, 2006.
- Tracy Yue Wang, Andrew Winton, and Xiaoyun Yu. Corporate fraud and business conditions: Evidence from IPOs. *The Journal of Finance*, 65(6):2255–2292, 2010.
- P.A. Whigham. Grammatically-based genetic programming. In *Proceedings of the workshop on genetic programming: from theory to real-world applications*, volume 16, pages 33–41. Citeseer, 1995.
- Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.
- Ian H. Witten, Eibe Frank, and Mark A. Hall. Data mining: Practical machine learning tools and techniques. 2011.
- David H. Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.
- Man Leung Wong and Kwong Sak Leung. Evolutionary program induction directed by logic grammars. *Evolutionary Computation*, 5(2):143–180, 1997.
- Man Leung Wong and Kwong Sak Leung. Data mining using grammar based genetic programming and applications. 2000.
- Chen Xu. Customer lifetime value: an integrated data mining approach. 2006.
- I. Yeh, Che-hui Lien, et al. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480, 2009.
- Frank Yu and Xiaoyun Yu. Corporate lobbying and fraud detection. *Journal of Financial and Quantitative Analysis*, 46(06):1865–1891, 2012.
- Jianguo Yuan, Chunsheng Yuan, and Xiaolan Deng. The effects of manager compensation and market competition on financial fraud in public companies: An empirical study in China. *International Journal of Management*, 25(2), 2008.
- Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *Evolutionary Computation, IEEE Transactions on*, 11(6):712–731, 2007.
- Eckart Zitzler, Marco Laumanns, Lothar Thiele, Eckart Zitzler, Eckart Zitzler, Lothar Thiele, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm, 2001.