Classification Rule Mining for Automatic Credit Approval using Genetic Programming

Sum Sakprasat and Mark C. Sinclair

***Abstract*—Automatic credit approval is important for the efficient processing of credit applications. Eight different genetic programming (GP) approaches for the classification rule mining of a credit card application dataset are investigated, using both a Booleanizing technique and strongly-typed GP. In addition, the use of GP for missing value handling is evaluated. Overall, on the Australian Credit Approval dataset, those GP approaches that had poorer classification correctness on the training data often proved better at generalizing for the test set.**

I. Introduction

THE granting of credit, e.g. loans and credit cards, is a necessary part of a developed economy. In today's interconnected world, even in developing countries such as Cambodia, the use of, say, credit cards is starting to become a reality. However, for lenders, there remains the problem of credit approval – which customers represent an acceptable credit risk, and should be granted credit. This is particularly true in situations, such as developing countries, where established guidelines and models from developed countries may not be reliably applied. There is thus a need to research effective methods for automatic credit approval that can assist credit professionals in assessing consumer credit applications.

This paper examines the use of strongly-typed genetic programming for automatic credit approval. The problem is essentially one of classification rule mining in an existing database of credit applications, aiming to classify applications as reliably as possible into those that should receive credit, and those that should not. Eight different genetic programming (GP) approaches were applied and compared, as well as the use of GP for missing value handling.

Independently of the work described here, Huang *et al.* [1] have also recently applied GP for automatic credit approval, but their approach used a two-stage GP to obtain classification rules of very limited size, combined with a discrimination function evolved to deal with confusors. They used a much smaller test set, and did not consider missing data, so their results are not directly comparable with the research reported here.

Section II provides a review of knowledge discovery in databases, classification rule mining, and the use of GP and strongly-typed GP in particular. Section III discusses credit risk, credit scoring and automatic credit approval. Section IV gives details of the credit application dataset used for the experimental work; Section V presents the different GP approaches; and Section VI details two techniques to address missing values in the dataset. The experimental results are summarized in Section VII, and Section VIII concludes with some discussion and an outline of further work.

Manuscript received March 15, 2007. This research was carried out by Sum Sakprasat, supervised by Mark C. Sinclair, as part of a Master of Science in Information Technology degree at the Royal University of Phnom Penh, Cambodia.

Sum Sakprasat is with Build Bright University, Phnom Penh, Cambodia (e-mail: sum.sakprasat@gmail.com).

Mark C. Sinclair was with the Royal University of Phnom Penh, Cambodia. He is now with the National Polytechnic Institute of Cambodia, Phnom Penh (e-mail: mcs@ieee.org).

II. Data Mining

A. Knowledge Discovery in Databases

Knowledge discovery in databases (KDD) is the process of finding accurate, comprehensive, useful, and innovative patterns in data from large databases [2]. It is an interactive and iterative process that consists of the following steps [3]. Data integration – the data from different sources are combined; inconsistencies in attribute names and attribute value names between the datasets from different sources are corrected; and redundant data are removed. Data cleaning – missing values are filled in; noisy data such as outliers that do not conform to the general data model expected are identified, and then replaced or deleted; and inconsistent data are corrected or removed. Data transformation – the data are normalized; and aggregation and discretization are applied to make the data appropriate for the mining algorithm and to improve the accuracy and efficiency of the mining process. Data selection – the portion of task-relevant data that the users want to investigate is specified for the mining algorithm. Data mining – an algorithm is applied to the data to extract the hidden patterns. Pattern evaluation – the interestingness of the discovered patterns is evaluated. Knowledge representation – the interesting patterns are presented as knowledge to the user.

B. Classification Rule Mining

There are several types of patterns discussed in the data mining literature, including classification rules, association rules, clusters, and outliers [3]. In this paper, however, the focus is on classification rules. The basic techniques for mining these are: [3] decision tree induction, Bayesian, and artificial neural networks. Another approach, the one

examined here, is GP [4].

C. Genetic Programming

For classification rule mining, each GP individual represents a classification rule. An initial population is created consisting of randomly generated rules. Each rule is evaluated to assign its fitness based on its classification accuracy on the training samples. Rules that have high fitness (high accuracy) have more opportunities to survive by reproducing, which are then modified by genetic operators to produce new rules, which are subsequently placed in the new population using some replacement model. Each rule in the new population is evaluated to assess its fitness on the same training samples. The generation of new rules is repeated until a given number of rules in the population satisfy a pre-specified fitness threshold, or the maximum number of generations is reached.

The main motivation for employing GP in classification rule mining is that it is a robust and adaptive search method than can more effectively discover interesting patterns than the greedy search performed by many other mining algorithms [5]. By applying several different functions available in the function set to the original attributes, GP can create derived attributes which have greater predictive power, even if the original attributes do not have much predictive power by themselves [2].

D. Strongly-Typed Genetic Programming

One of the difficulties in designing a function and terminal set for canonical GP is satisfying the closure property [4]. Possible approaches in the context of classification rule mining are Booleanizing [2], strongly-typed GP [6] and grammar-based GP [7]. However, only the first two were applied in this research, leaving grammar-based GP for further work.

For the Booleanizing approach, all the terminals in the function set return a Boolean value, and only logical operators (and, or, not, etc.) are used in the function set. Therefore, the returned value of any node in the tree can be used as the argument of any logical operator in the corresponding parent node.

In strongly-typed GP (STGP), the value of every terminal and function is associated with a data type. Each function in the function set must specify the data type of its returned value, and the data types of each of its arguments. The returned value of the root node of the tree must be of the type expected by the problem, and the returned value of each internal node or leaf node must be of the type expected by its parent node. Consequently, there is type restriction on which function or terminal can be chosen at a particular node. The function chosen as root node must have the data type expected by the problem. The functions or terminals chosen as internal nodes or leaf nodes must in turn have the data types expected by their parent nodes. Initialization procedure and genetic operators are modified so that they generate only syntactically correct parse trees. For the crossover operator, the nodes at the crossover points in both parents must have the same data type. For the mutation operator, the randomly generated subtree must have the same data type as the replaced subtree. It has been observed that the use of such type restrictions also reduces the search space, which is likely to improve the search [8].

III. CREDIT APPROVAL

A. Credit Risk

Credit risk refers to the risk of loss to lenders due to customers defaulting on their credit obligations because of unwillingness, inability, bankruptcy or any number of other reasons why the customers cannot repay their credit [9]. It is the probability that a customer will default, and this probability represents the level of credit risk of the customer. The assessment of credit risk is very important for lenders to help determine the likely risk and so make correct decisions about the approval of credit, the interest rate, the repayment term, etc.

B. Credit Scoring

Credit scoring is an evaluation system designed to enhance lenders' abilities in determining the creditworthiness or credit risk of a customer in the process of credit risk analysis.

Credit scoring methods can be divided into deductive credit scoring and empirical credit scoring. In deductive credit scoring, relevant customer attributes are assigned points. These points are then used to figure out a credit score. The selection of the relevant attributes, the determination of the points, and how the credit scores are calculated, are all based on the experience of credit professionals. In empirical credit scoring, scoring models are constructed by analyzing past customer credit data using appropriate algorithms to identify characteristics relating to the credit risk or creditworthiness of customers. These scoring models can then be used to evaluate the credit risk or creditworthiness of new customers [10].

Lenders use credit scores as an indication of the level of credit risk to help decide whether or not customers should be approved for credit, and, if approved, at what interest rate, on what repayment terms, and what other conditions imposed. The lower the credit risk, the higher the chance of getting the credit at a lower interest rate and on a longer repayment term. If the credit risk is high, but below the cut-off credit risk, the customer is not automatically disqualified from getting the credit, but such a score prompts the lender to review the customer's application more carefully before deciding whether to approve or deny the credit. If approved, a high credit risk usually results in a higher interest rate and shorter repayment term than those offered to customers with lower credit risk.

C. Automatic Credit Scoring and Approval

One of the most important advantages of credit scoring is the possibility of automating credit approval. Automatic credit scoring and approval means computerized credit scoring and approval. With the use of computers and the Internet, such a system can quickly gather the necessary information, evaluate, and determine whether to approve or deny credit applications, thereby playing a very important role in effective credit decisions, particularly in the case of consumer, rather than business, credit.

Automatic credit scoring and approval may never take the place of the credit professional, but it can help make rapid decisions to approve or deny the majority of cases. Those credit applications that are identified as good credit risk, and those as bad credit risk, may be automatically approved, or denied, while those of intermediate risk may still be passed to credit analysts for more detailed review before deciding whether to approve or deny credit. This can reduce the number of credit applications that need more detailed review and reduce wasted time, thus allowing credit analysts to concentrate only on those credit applications that are difficult or important [11].

IV. AUSTRALIAN DATASET

For the experimental work in this paper, the authors were not yet able to obtain a suitable Cambodian dataset. Consequently, a dataset of credit card applications and approval decisions, Australian Credit Approval, from UCI Repository of Machine Learning Databases and Domain Theories, was used. The dataset was originally provided by Quinlan in his studies of ID3 and C4.5 system in 1987 and 1992, to induce decision trees for assessing credit card applications [12]. The dataset has 15 attributes plus the class label attribute. All attribute names and values were changed, before being released, to meaningless symbols to protect the confidentiality of the data. The dataset is summarized in Table I.

TABLE I
THE AUSTRALIAN CREDIT APPROVAL DATASET

Attribute	Type	Values
A1	nominal	a, b
A2	continuous	13.75 – 80.25
A3	continuous	0 – 28
A4	nominal	u, y, l, t
A5	nominal	g, p, gg
A6	nominal	c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff
A7	nominal	v, h, bb, j, n, z, dd, ff, o
A8	continuous	0 – 28.5
A9	nominal	t, f
A10	nominal	t, f
A11	continuous	0 – 67
A12	nominal	t, f
A13	nominal	g, p, s
A14	continuous	0 – 2000
A15	continuous	0 – 100000
class	nominal	+, –

TABLE II
CLASS DISTRIBUTION FOR THE AUSTRALIAN CREDIT APPROVAL DATASET

Class	Frequency
+	307 (44.5%)
–	383 (55.5%)

TABLE III
STATISTICS OF MISSING VALUES IN THE AUSTRALIAN CREDIT APPROVAL DATASET

Attribute	Missing Values
A1	12
A2	12
A4	6
A5	6
A6	9
A7	9
A14	13

The dataset is interesting because there is a good mix of attributes: continuous, nominal with small numbers of values, and nominal with larger numbers of values. There are 690 instances in this dataset, with 307 (44.5%) being positive (credit approved) and 383 (55.5%) being negative (credit denied). There are 37 (5%) instances which have some missing values. The class distribution and the statistics of missing values in the Australian Credit Approval dataset are summarized in Table II and Table III.

In the Statlog project, the dataset was modified by replacing missing values: for nominal attributes, with their mode, and for continuous attributes, with their mean. The labels were also changed for the convenience of the statistical algorithms. For example, attribute A5 originally had three labels g, p, gg and these were changed to labels 1, 2, 3. Attributes A4 and A5 of the original data were apparently identical, so attribute A4 of the original data was removed for the convenience of the statistical algorithms, and A5 to A15 relabeled A4 to A14. The Statlog project suggests that only attributes A4, A7, A8, A12 and A13 (or A5, A8, A9, A13 and A14 in the original dataset) are relevant, and improved results are often obtained if only these attributes are used [12].

In this experiment, the problem of missing values is one of those to be solved by GP. Thus, the original Australian Credit Approval dataset is used, with the exception that the attribute A4, as in the Statlog project, is removed. The nominal attribute values, when processing are converted into numerical. The modified version of the Australian Credit Approval dataset, with the corresponding numerical values for nominal attributes, is summarized in Table IV.

Here, the dataset is divided into the training set and the test set, each consisting of 345 (50%) samples. The training set is created by randomly selecting 345 samples from the whole dataset, and the remaining samples are the test set. There are two versions of the training and the test sets: the 15-attribute set and the 6-attribute set. The six attributes of the 6-attribute set are the six attributes suggested by the Statlog project. For each of the four datasets, there are two further versions: one with missing values filled in with their

TABLE IV
THE MODIFIED AUSTRALIAN CREDIT APPROVAL DATASET USED IN THE EXPERIMENT

Attribute	Type	Values
A1	nominal	a(0), b(1)
A2	continuous	13.75 – 80.25
A3	continuous	0 – 28
A4	nominal	g(0), p(1), gg(2)
A5	nominal	c(0), d(1), cc(2), i(3), j(4), k(5), m(6), r(7), q(8), w(9), x(10), e(11), aa(12), ff(13)
A6	nominal	v(0), h(1), bb(2), j(3), n(4), z(5), dd(6), ff(7), o(8)
A7	continuous	0 – 28.5
A8	nominal	t(1), f(0)
A9	nominal	t(1), f(0)
A10	continuous	0 – 67
A11	nominal	t(1), f(0)
A12	nominal	g(0), p(1), s(2)
A13	continuous	0 – 2000
A14	continuous	0 – 100000
class	nominal	+, -

modes for nominal attributes, and with their medians for continuous attributes; the other with missing values left unfilled, leaving it to GP to handle by evolving substitute values.

For comparison purposes, of the 27 data mining approaches reported by Statlog [13], the classification accuracy on the training set ranged from 56 to 100%, and on the test set, from 56 to 87%, of those approaches that did not fail outright.

V. GP APPROACHES

A. The Eight Approaches

There are eight GP approaches for the Australian Credit Approval problem in this paper, referred to as Approach1 to Approach8. All eight use STGP, with the main difference in the terminal sets and the function sets used. Approach1 to Approach4 are similar to Approach5 to Approach8, respectively, with the first four approaches using compound terminals for enumerated attributes (a Booleanizing approach), while the last four use `IfLeA*` functions (see below). Their common parameters are recorded in Table V.

The population size of 1024 was taken as a reasonable default, and initial experimental runs indicated that little improvement occurred after 15 generations, so a value of 30 generations beyond the initial random population was judged to be sufficient. The terminal set and the function set for each of the eight approaches are described in the following subsections.

B. Approach1

In this approach (see Table VI), only one ERC (ephemeral random constant) terminal is used. This ERC terminal is used to generate constants that are to be compared to values of the continuous attributes `A2`, `A3`, `A7`, `A10`, `A13`, and `A14`, in the functions `IfLeA2`, `IfLeA3`, `IfLeA7`, `IfLeA10`, `IfLeA13`, and `IfLeA14`, respectively. Each of these functions returns true if its single argument, which is a constant generated by the ERC terminal, is less than or equal to the corresponding attribute value, or otherwise returns false. The ERC values range from 0.0, which is the minimum number for the six continuous attributes, to 100000.0, which is the maximum number for attribute `A14`.

For the nominal attributes `A8`, `A9`, and `A11`, that have attribute values 1(`t`) and 0(`f`) representing the Boolean values true and false, there is a terminal for each, which returns true if the corresponding attribute value is 1, or false, if the corresponding attribute value is 0. This is similar for the attribute `A1` that has two attribute values, 0(`a`) and 1(`b`), for which the terminal set returns true if the corresponding attribute value is 0, or false if the corresponding attribute values is 1.

For nominal attributes `A4`, `A5`, `A6`, and `A12`, that have more than two attribute values, there is a compound terminal for each. For instance, the attribute `A5` has four terminals `A5A`, `A5B`, `A5C`, and `A5D`. This is because the attribute `A5` has 14 attribute values that require four bits to represent all the attribute values. The attribute value of `A5` is determined by the returned values of the four terminals. This is

TABLE V
COMMON TABLEAU OF THE EIGHT GP APPROACHES FOR THE AUSTRALIAN CREDIT APPROVAL PROBLEM

Objective	Find a classification rule that is able to determine whether to approve a credit or not, or to justify whether a credit is either safe or risky.
Terminal set	(differs between approaches)
Function set	(differs between approaches)
Fitness cases	345
Fitness	missclassification[2]
Parameter	Population size = 1024, Generations = 31

TABLE VI
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH1

Terminal Set	`ERC[0.0, 100000.0], True, False, A1, A4A, A4B, A5A, A5B, A5C, A5D, A6A, A6B, A6C, A6D, A8, A9, A11, A12A, A12B`
Function Set	`IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor`

TABLE VII
THE VALUES OF ATTRIBUTE A5

Attribute Values	Numeric Equivalent	A5A	A5B	A5C	A5D
c	0	0	0	0	0
d	1	0	0	0	1
cc	2	0	0	1	0
i	3	0	0	1	1
j	4	0	1	0	0
k	5	0	1	0	1
m	6	0	1	1	0
r	7	0	1	1	1
q	8	1	0	0	0
w	9	1	0	0	1
x	10	1	0	1	0
e	11	1	0	1	1
aa	12	1	1	0	0
ff	13	1	1	0	1

TABLE VIII
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH2

Terminal Set	ERC[13.75, 80.25], ERC[0.0, 28.0], ERC[0.0, 28.5], ERC[0.0, 67.0], ERC[0.0, 2000.0], ERC[0.0, 100000.0], True, False, A1, A4A, A4B, A5A, A5B, A5C, A5D, A6A, A6B, A6C, A6D, A8, A9, A11, A12A, A12B
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor

TABLE IX
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH3

Terminal Set	ERC[0.0, 100.0], True, False, A1, A4A, A4B, A5A, A5B, A5C, A5D, A6A, A6B, A6C, A6D, A8, A9, A11, A12A, A12B
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, Add, Sub, Mul, Div

TABLE X
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH4

Terminal Set	ERC[0.0, 13.0], True, False, A1, A4A, A4B, A5A, A5B, A5C, A5D, A6A, A6B, A6C, A6D, A8, A9, A11, A12A, A12B
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, Add, Sub, Mul, Div

TABLE XI
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH5

Terminal Set	ERC[0.0, 100000.0], ERC[0.0, 13.0], True, False, A8, A9, A11
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, IfLeA1, IfLeA4, IfLeA5, IfLeA6, IfLeA12

TABLE XII
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH6

Terminal Set	ERC[13.75, 80.25], ERC[0.0, 28.0], ERC[0.0, 28.5], ERC[0.0, 67.0], ERC[0.0, 2000.0], ERC[0.0, 100000.0], ERC[0.0, 13.0], True, False, A8, A9, A11
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, IfLeA1, IfLeA4, IfLeA5, IfLeA6, IfLeA12

TABLE XIII
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH7

Terminal Set	ERC[0.0, 100.0], ERC[0.0, 13.0], True, False, A8, A9, A11
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, Add, Sub, Mul, Div, IfLeA1, IfLeA4, IfLeA5, IfLeA6, IfLeA12

TABLE XIV
THE TERMINAL SET AND THE FUNCTION SET FOR APPROACH8

Terminal Set	ERC[0.0, 13.0], True, False, A8, A9, A11
Function Set	IfLeA2, IfLeA3, IfLeA7, IfLeA10, IfLeA13, IfLeA14, And, Or, Not, Nand, Nor, Add, Sub, Mul, Div, IfLeA1, IfLeA4, IfLeA5, IfLeA6, IfLeA12

illustrated in Table VII.

Note that there is no combination of 1, 1, 1, and 0, or combination of 1, 1, 1, and 1, in this table since these will never happen, although there can be an individual with a branch of `(And (And A5A A5B) (And A5C (Not A5D)))` or `(And (And A5A A5B) (And A5C A5D))` but these two branches will always simply return false.

With the use of these compound terminals with some Boolean operators, various conditions on, say `A5`, can be represented. For example, the condition of `A5` equals 3(`i`), in a Lisp S-expression, is `(And (Not A5A) (And (Not A5B) (And A5C A5D)))`; the condition of `A5` equals 8(`q`) or 9(`w`) or 12(`aa`) or 13(`ff`) is `(And A5A (Not A5C))`.

The terminals `True` and `False` represent Boolean values true and false. The functions `And`, `Or`, `Not`, `Nand`, and `Nor` are the respective Boolean operators.

For the continuous attributes, there is a function `IfLeA*` for each. For instance, for the attribute `A14`, there is a function `IfLeA14`. This function may be use in combination with the Boolean operators to represent various conditions on `A14`. For example, the condition of `A14` less than 4000 is `(Not (IfLeA14 4000))`, and the condition of `A14` equals 4000 is `(And (IfLeA14 4000) (Not (IfLeA14 4001)))`.

C. Approach2

In this approach (see Table VIII), six ERC terminals are used. Each continuous attribute now has its own ERC ranging from the minimum value of the attribute to the maximum one, so that the constants generated by the ERC will usually match the attribute values of the corresponding attributes. With the use of these ERC terminals, it is hoped that the accuracy will be improved. The other terminals and functions are the same as in Approach1.

D. Approach3

In this approach (see Table IX), only one ERC terminal is used, but this time ranging from 0.0 to 100.0. However, there are four new functions in this approach: `Add`, `Sub`, `Mul`, and `Div`, which are the four arithmetic operators. This approach lets GP evolve the values for the continuous attributes by applying the four arithmetic operators to the generated ERC values to compose new values. The other terminals and functions remain the same as in Approach1 and Approach2.

E. Approach4

This approach (see Table X) is similar to Approach3, with the exception that the ERC terminal here ranges from 0.0 to 13.0, which is used to generate constants specifically for the enumerated attributes, and also used with the combination of the arithmetic operators for the continuous attributes as well.

F. Approach5

This approach (see Table XI) is similar to Approach1,

except that IfLeA* functions, rather than compound terminals, are used for the enumerated attributes A1, A4, A5, A6, and A12, and another ERC terminal, ranging from 0.0 to 13.0, is added for the arguments of these IfLeA* functions. The other terminals and functions are the same.

For the first four approaches, the condition of A5 equals 3(i) is (And (Not A5A) (And (Not A5B) (And A5C A5D))). With this approach, it is (And (IfLeA5 3) (Not (IfLeA5 4))). The condition of A5 equals 8(q) or 9(w) or 12(aa) or 13(ff), in the first four approaches, is (And A5A (Not A5C)). With this approach, it is (Or (And (IfLeA5 8) (Not (IfLeA5 10)))(And (IfLeA5 12) (Not (IfLeA5 14)))).

G. Approach6 to Approach8

The three remaining approaches, Approach6, Approach7, and Approach8 (see Tables XII, XIII and XIV), are similar to Approach2, Approach3, and Approach4, respectively, but they use IfLeA* functions, rather than compound terminals, for the enumerated attributes A1, A4, A5, A6, and A12, and an ERC terminal, ranging from 0.0 to 13.0, for these IfLeA* functions. The other terminals and functions are the same as in their counterparts.

VI. MISSING VALUE HANDLING

The GP approach for missing value handling in this paper is to substitute the missing values by values generated and evolved by GP. This general approach was first proposed by Backer in his study of learning with missing data using GP [14].

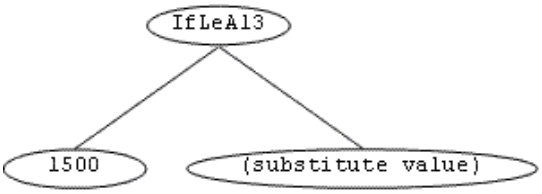

Fig. 1. An example of the GP approach for missing value handling with a continuous attribute.

For continuous attributes with missing values, A2 and A13, the IfLeA* functions are modified to take two arguments, in which the second argument is the substitute value used in the event the attribute value is missing. Fig. 1 shows an example of function IfLeA13 taking two arguments. The first argument is 1500 and the second argument is the substitute value. The latter can be just a single constant, or it can be a value composed with arithmetic operators. The function IfLeA13 first checks to see if the attribute value of A13 that is to be compared is missing. If it is not, the value will be compared with the 1500; otherwise, the substitute value will be used as the attribute value, and compared with the 1500.

For the nominal attributes with missing values, A1, A4, A5, and A6, when compound terminals are used, as in Approach1 to Approach4, each terminal is modified to be a function taking a single argument, which is the substitute value used in the event the attribute value is missing. The substitute value can be just a terminal, True or False, or it can be a Boolean value composed from other Boolean terminals using operators. The A5A function first checks to see if the attribute value of A5 is missing. If it is not, the value will be used to determine whether to return true or false, otherwise, the substituted value will be returned.

For the additional IfLeA* functions, as in Approach5 to Approach8, the functions are similarly modified to take two arguments, with the second argument the substitute value used in case the attribute value is missing.

VII. EXPERIMENTAL RESULTS

A. Experimental Work

In the experiments in this paper, comparisons were made on the basis of ten runs. The average classification correctness for each approach is shown compared to that of the other approaches using the significance values from the pair-wise sign test [15]. In the tables, a value shown in bold means that it is better than the other it has been compared to. A significance value of 1.1%, say, means that the approach with the value shown in bold is highly significantly better than the other approach it is compared with. All of the experimental work was carried out using ECJ 12 [16]. Given space limitations, much of the detailed experimental results have been omitted, but full results can be obtained from [17].

B. Training

For training, where the mean classification accuracy ranged between 87 and 89%, there were four important outcomes.

With the use of compound terminals for enumerated attributes, the classification correctness is often higher than that with the use of the IfLeA* functions (compound terminals, 14 cases vs. IfLeA*, 2 cases; see Table XV).

On the 6-attribute dataset, the classification correctness is always higher than that on the 15-attribute dataset (6-attribute, 16 cases vs. 15-attribute, 0 cases; see Table XVI).

With the use of the GP approach for missing value handling, the classification correctness is often higher than that with the use of the pre-processing approach (GP approach, 15 cases vs. pre-processing, 0 cases; see Table XVII).

There is no approach that is always the best in all cases. However, there is an approach that is always the worst; it is Approach8. Approach2 performs the best when applied on the 6-attribute dataset, while Approach1 performs the best when applied on the 15-attribute dataset. There is not much difference between the eight approaches when they are applied on the 6-attribute dataset, with the use of the GP

TABLE XV
COMPARISON BETWEEN COMPOUND TERMINALS AND IfLeA* METHOD FOR ENUMERATED ATTRIBUTES

Approach			Method for enumerated attributes: Compound T. Train	Compound T. Test	IfLeA* Train	IfLeA* Test	Compound T. vs. IfLeA* Train	Compound T. vs. IfLeA* Test
GP approach	6attributes	1, 5	**89.304**	82.841	89.275	**82.899**		
		2, 6	**89.391**	**82.957**	89.217	82.899		
		3, 7	89.275	82.899	**89.333**	**82.957**		
		4, 8	**89.304**	82.870	89.159	**82.899**		
	15 attributes	1, 5	**89.130**	82.899	89.014	82.899		
		2, 6	88.696	**82.986**	**88.928**	82.812		
		3, 7	**88.638**	82.899	88.493	82.899		
		4, 8	**88.696**	82.928	88.377	**83.043**		
Pre-processing	6 attributes	1, 5	**88.986**	82.899	88.464	**83.188**	1.1%	0.1%
		2, 6	**89.391**	82.928	88.725	**83.478**	1.1%	1.1%
		3, 7	**89.101**	82.928	88.493	**83.188**	0.1%	1.1%
		4, 8	**89.072**	82.609	88.464	**83.159**	1.1%	1.1%
	15 attributes	1, 5	**88.899**	82.522	88.203	**83.043**	5.5%	
		2, 6	**88.522**	83.043	88.174	**83.101**		
		3, 7	**88.319**	83.101	87.913	**83.188**		
		4, 8	**88.435**	82.841	87.826	**83.188**	1.1%	

TABLE XVI
COMPARISON BETWEEN THE 6-ATTRIBUTE DATASET AND THE 15-ATTRIBUTE DATASET

Approach		Dataset: 6 attributes Train	6 attributes Test	15 attributes Train	15 attributes Test	6 attributes vs. 15 attributes Train	6 attributes vs. 15 attributes Test
GP approach	1	**89.304**	82.841	89.130	**82.899**		
	2	**89.391**	82.957	88.696	**82.986**		
	3	**89.275**	82.899	88.638	82.899	1.1%	
	4	**89.304**	82.870	88.696	**82.928**	1.1%	
	5	**89.275**	82.899	89.014	82.899		
	6	**89.217**	**82.899**	88.928	82.812		
	7	**89.333**	**82.957**	88.493	82.899	5.5%	
	8	**89.159**	82.899	88.377	**83.043**		
Pre-processing	1	**88.986**	**82.899**	88.899	82.522		
	2	**89.391**	82.928	88.522	**83.043**	5.5	
	3	**89.101**	82.928	88.319	**83.101**	0.1	
	4	**89.072**	82.609	88.435	**82.841**	0.1	
	5	**88.464**	**83.188**	88.203	83.043	5.5	
	6	**88.725**	**83.478**	88.174	83.101	1.1	5.5%
	7	**88.493**	83.188	87.913	83.188	1.1	
	8	**88.464**	83.159	87.826	**83.188**	0.1	

approach for missing value handling.

Putting all this together, the approach that performs the best in training is Approach2 on the 6-attribute dataset either with the use of GP for missing value handling or with the pre-processing approach. The approach that performs the worst is Approach8 on the 15-attribute dataset with the use of the pre-processing approach for missing value handling.

C. Testing

For testing, where the mean classification accuracy ranged between 82 and 83%, there were also four important outcomes.

With the use of the IfLeA* functions for enumerated attributes, the classification correctness is often higher than that with the use of compound terminals (IfLeA*, 12 cases vs. compound terminals, 2 cases; see Table XV).

On the 15-attribute dataset, the classification correctness is often higher than that on the 6-attribute dataset (15-attribute, 8 cases vs. 6-attribute, 5 cases; see Table XVI).

TABLE XVII
COMPARISON BETWEEN GP APPROACH FOR MISSING VALUE HANDLING AND PRE-PROCESSING APPROACH

Approach		Missing value handling: GP approach Train	GP approach Test	Pre-processing Train	Pre-processing Test	GP approach vs. Pre-processing Train	GP approach vs. Pre-processing Test
6 attributes	1	**89.304**	82.841	88.986	**82.899**	0.1%	
	2	89.391	**82.957**	89.391	82.928		
	3	**89.275**	82.899	89.101	**82.928**		
	4	**89.304**	**82.870**	89.072	82.609	1.1%	
	5	**89.275**	82.899	88.464	**83.188**	0.1%	0.1%
	6	**89.217**	82.899	88.725	**83.478**		1.1%
	7	**89.333**	82.957	88.493	**83.188**	0.1%	
	8	**89.159**	82.899	88.464	**83.159**	0.1%	1.1%
15 attributes	1	**89.130**	**82.899**	88.899	82.522		
	2	**88.696**	82.986	88.522	**83.043**		
	3	**88.638**	82.899	88.319	**83.101**		
	4	**88.696**	**82.928**	88.435	82.841	5.5%	
	5	**89.014**	82.899	88.203	**83.043**	1.1%	1.1%
	6	**88.928**	82.812	88.174	**83.101**	1.1%	1.1%
	7	**88.493**	82.899	87.913	**83.188**		
	8	**88.377**	83.043	87.826	**83.188**		

With the use of the pre-processing approach for missing value handling, the classification correctness is often higher than that with the use of the GP approach (pre-processing, 12 cases vs. GP approach, 4 cases; see Table XVII).

Approach6 performs the best when using the pre-processing approach for missing value handling on the 6-attribute dataset. Approach8 performs the best when applied on the 15-attribute dataset. Approach2 still performs the best when applied on the 6-attribute dataset with the use of the GP approach for missing value handling. There is not much difference between the eight GP approaches with the use of the GP approach for missing value handling.

Putting all this together, the approach that performs the best on the test set is Approach6 with the use of the pre-processing approach for missing value handling on the 6-attribute dataset. The approach that performs the worst is Approach1 with the use of the pre-processing approach for missing value handling on the 15-attribute dataset.

What is worth noting is that the results from the testing are often in direct contrast with those from the training, indicating that some overfitting has occurred.

VIII. DISCUSSION AND FURTHER WORK

A. Discussion

Automatic credit scoring and approval plays a very important role in today's competitive credit industries, helping quick and intelligent credit decision making. GP, as has been demonstrated by the experiments here, can be used in data mining for the automatic credit approval problem. The experiments also show that GP can be successfully used with data containing missing values. Although the results from the approaches which use GP for missing value handling have lower classification correctness than those

which use the pre-processing approach on the test sample, nevertheless, the GP approach for missing values has the advantage that it can be trained with data containing missing values without pre-processing, and the results can then be applied to future credit approval data also containing missing values.

B. Multiple Classes of Credit Risk

The automatic credit scoring and approval here works only with two levels of credit risk: 'good' and 'bad'. Good credit risk applications may be automatically approved, but, bad credit risk applications will need to be passed to credit analysts for more detailed review. It may be more desirable if the system can assess various levels of credit risk. For example, if the system can provide three levels of credit risk: 'good', 'bad', and 'intermediate'. The automatic credit scoring and approval system may be designed to automatically approve good credit risk applications, and automatically deny bad credit risk applications, with those of intermediate risk passed to credit analysts. Moreover, if the system can produce many levels of credit risk, a credit manager could associate interest rates and repayment terms with the levels of the credit risk that are to be automatically approved.

C. Confusor Identification

As was seen from in Section VII, GP approaches that have higher classification correctness in training often have lower classification correctness on the test set. Whilst approaches that have higher classification correctness in training should be better at correct classification, nevertheless, it seems that they are not able to deal with noisy data so that outliers can confuse them. As GP evolution here is based solely on fitness from classification errors, this is the reason that it can be easily confused. It is, however, a difficult task for GP to identify confusors, whilst learning to classify data. A separate step for learning to identify confusors before learning to classify data may be important to allow GP to both recognize and distinguish confusors as well as learning to classify the data [1].

D. Data Selection

GP works on populations of solutions, which can be a problem if the training data is large. Moreover, a large dataset can slow down the mining process, especially for GP, or result in too many patterns. Furthermore, if the data contains redundant data or irrelevant attributes, it can confuse the mining algorithm leading to the discovery of inaccurate or useless patterns. Thus, selection of training data is important in data mining. It can not only improve the mining process, but also improve the accuracy of the discovered patterns. A possible extension to this research is thus to study the use of GP for selecting a small subset of data, which is nevertheless sufficient as a training set, in the data selection step of the KDD process.

E. Comparison with Non-GP Approaches

Whilst the overall aim of this research was to examine different GP approaches for classification rule mining in automatic credit approval, nevertheless the overall classification correctness on the training set (83%) approached the quality of the best algorithms from Statlog [13] (87%). Further work should provide more detailed comparison with non-GP approaches, as well as attempting to further improve the accuracy of the best GP approaches. In addition, it would be beneficial to work with other credit approval datasets, particularly a Cambodian specific dataset, should one become available.

REFERENCES

[1] J.J. Huang, G.H. Tzeng, and C.S. Ong, "Two-stage genetic programming (2SGP) for the credit scoring model," *Applied Mathematics and Computation*, vol. 174, pp. 1039-1053, March 2006.

[2] A.A. Freitas, "A survey of evolutionary algorithms for data mining and knowledge discovery," in *Advances in Evolutionary Computation*, A. Ghosh and S. Tsutsui, Eds. Berlin: Springer-Verlag, August 2002, pp. 819-845.

[3] J. Han, and M. Kamber, *Data Mining: Concepts and Techniques*. San Diego, CA: Morgan Kaufmann, 2001.

[4] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[5] A. Ghosh, and A.A. Freitas, "Guest editorial: Data mining and knowledge discovery with evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, pp. 517-518, December 2003.

[6] D.J. Montana, "Strongly typed genetic programming," Technical Report 7866, Bolt Beranek and Newman, Inc., March, 1994.

[7] P.A. Whigham, "Grammatically-based genetic programming," *Proc. Workshop on Genetic Programming: From Theory to Real-World Applications*, Tahoe City, CA, 1995, pp.33-41.

[8] W. Banzhaf, P. Nordin, R.E. Keller, and F.D. Francone. *Genetic Programming ~ An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*. San Francisco, CA: Morgan Kaufmann, 1998, p. 298.

[9] K.H. Ng, *Commercial Banking in Singapore*. Singapore: Addison-Wesley, 1996, pp. 252-253.

[10] Y. Liu, "New Issues in Credit Scoring Application," Arbeitsbericht Nr. 16/2001, Institut für Wirtschaftsinformatik, University of Göttingen, Germany, 2001.

[11] T. Diana, "Credit risk analysis and credit scoring – now and in the future" *Business Credit*, pp. 1-3, March 2005.

[12] "Australian credit approval," Artificial Intelligence and Data Analysis Group, Artificial Intelligence and Computer Science Laboratory, University of Porto, Portugal. Available: http://www.niaad.liacc.up.pt/old/statlog/datasets/australian/australian.doc.html

[13] "Statlog datasets: comparison of results," Department of Informatics, Nicolaus Copernicus University, Poland. Available: http://www.is.umk.pl/projects/datasets-stat.html

[14] G. Backer, "Learning with missing data using genetic programming," Masters thesis, Institut für Betriebssysteme und Rechnerverbund Technische Universität Braunschweig, Germany, 1995.

[15] P. Sprent, *Applied Nonparametric Statistical Methods*, 2nd ed., London: Chapman & Hall, pp. 102-103.

[16] S. Luke, *et al.*, "ECJ12: A Java-based evolutionary computation and genetic programming research system," Available: http://cs.gmu.edu/~eclab/projects/ecj/

[17] S. Sum, "Data mining for automatic credit approval using genetic programming," Masters thesis, Royal University of Phnom Penh, Cambodia, June 2006.