Multiple Imputation and Genetic Programming for Classification with Incomplete Data

Cao Truong Tran, Mengjie Zhang, Peter Andreae and Bing Xue
School of Engineering and Computer Science
Victoria University of Wellington, PO Box 600, Wellington 6140, New Zealand
{cao.truong.tran,mengjie.zhang,peter.andreae,bing.xue}@ecs.vuw.ac.nz

ABSTRACT

Many industrial and research datasets suffer from an unavoidable issue of missing values. One of the most common approaches to solving classification with incomplete data is to use an imputation method to fill missing values with plausible values before applying classification algorithms. Multiple imputation is a powerful approach to estimating missing values, but it is very expensive to use multiple imputation to estimate missing values for a single instance that needs to be classified. Genetic programming (GP) has been widely used to construct classifiers for complete data, but it seldom has been used for incomplete data. This paper proposes an approach to combining multiple imputation and GP to evolve classifiers for incomplete data. The proposed method uses multiple imputation to provide a high quality training data. It also searches for common patterns of missing values, and uses GP to build a classifier for each pattern of missing values. Therefore, the proposed method generates a set of classifiers that can be used to directly classify any new incomplete instance without requiring imputation. Experimental results show that the proposed method not only can be faster than other common methods for classification with incomplete data but also can achieve better classification accuracy.

CCS CONCEPTS

•**Computing methodologies** → **Genetic programming;** *Supervised learning by regression;* Classification and regression trees;

KEYWORDS

Incomplete Data, Missing Data, Genetic Programming, Classification, Multiple Imputation

ACM Reference format:
Cao Truong Tran, Mengjie Zhang, Peter Andreae and Bing Xue . 2017. Multiple Imputation and Genetic Programming for Classification with Incomplete Data. In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071181

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '17, Berlin, Germany
© 2017 ACM. 978-1-4503-4920-8/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3071178.3071181

1 INTRODUCTION

Classification is one of main tasks in machine learning and data mining. Classification has been widely applied to many scientific areas like computer science, engineering, biology, ect. Due to its importance, difficulty and complexity, classification has received a great attention over many decades, but there are still open issues in classification, one of the issues is incomplete data [8].

An incomplete dataset is a dataset which does not have values in some fields. Many real-world datasets have an unavoidable problem of missing values. One clear example is that 45% of datasets in the UCI machine learning repository [15] have missing values [8]. There are various reasons behind the severe deficiency. For example, medical datasets frequently contain missing values because not all possible tests can be run on all patients [6]; datasets collected from social surveys often lack some values because respondents usually ignore some questions [16]; industrial datasets are often missing because of mechanical failures [8].

Missing values cause some serious issues for classification. First of all, missing values result in the non-applicability of most classification algorithms because most classification algorithms require their input to be complete. Moreover, missing values often lead to large classification errors [2, 8] .

One of the most common approaches to classification with incomplete data is to use imputation methods to fill missing values with plausible values before applying classification algorithms. For example, mean imputation replaces all missing values in each feature by the average of all complete values in the feature. Imputation methods can transform incomplete data into complete data which then can be used by any classification algorithm. An ordinary obvious way to use an imputation method for classification is that the imputation needs to be performed both in the training process to generate a classifier and in the application process of applying the classifier to a new incomplete instance [8].

Multiple imputation is a powerful approach to dealing with incomplete data by finding multiple suitable values for each missing values. In statistical analysis, multiple imputation has become increasingly popular thanks to its convenience and flexibility [16]. Multiple imputation also has been a powerful method to address classification with incomplete data [6, 21, 22, 26]. Although multiple imputation is suitable for batch imputation tasks, it is often very expensive to impute missing values for a single incomplete instance, as is required in classification problems. The main reason is that to impute missing values for a single instance, it must rebuild the imputation models from all the training data combined with the new instance [24, 25]. Therefore, how to efficiently and effectively use multiple imputation for classification with incomplete data should be investigated.

Genetic programming (GP) is an evolutionary technique [14]. The capability of GP in learning the definition of a function from examples makes it a good choice for evolving good classifiers. Therefore, GP has been widely used to classification tasks [4].

Although GP has been successfully used to learn classifiers, it has been mainly applied to complete data. Since standard GP cannot work directly with incomplete data, imputation methods are often required to transform incomplete data into complete data before using GP [23]. However, the combination of GP and a simple imputation method such as mean imputation often leads to a weak classifier. Therefore, GP needs be combined with a sophisticated imputation method such as multiple imputation to evolve a good classifier. Unfortunately, applying multiple imputation to classification with incomplete data in the ordinary obvious way is very inefficient [24]. Therefore, how to effectively and efficiently combine GP and multiple imputation also should be investigated.

1.1 Research goals

The goal of this paper is to propose a method combining multiple imputation with GP to evolve classifiers for incomplete data that allow efficient and effective classification. To achieve this goal, multiple imputation is used to transform the training incomplete data to the training complete data. Furthermore, the proposed method identifies all common patterns of missing values. After that, GP is used to learn a classifier for each pattern of missing values. As a result, the proposed method builds a set of classifier which can be then used to directly classify incomplete instances without using any imputation. The proposed method is compared with other common combinations of GP and imputation methods which use imputation methods to estimate missing values for both the training set and the test set. Experimental results are used to show that:

(1) The proposed method can achieve better classification accuracy than the other combinations of GP and imputation methods, and
(2) The proposed method can be faster to classify incomplete instances than the other combinations of GP and imputation methods.

1.2 Organisation

The rest of the paper is organised as follows. Section 2 discusses related work. Section 3 shows the proposed method. Section 4 presents the experiment design. Section 5 states experimental results and analysis. Finally, section 6 makes conclusions and mentions future work.

2 RELATED WORK

This section presents related work including imputation methods and GP for classification.

2.1 Imputation Methods

The purpose of imputation methods is to transform incomplete data into complete data by replacing missing values with plausible values. Because a majority of classification algorithms require complete data, using imputation methods is a main approach to addressing classification with incomplete data. A traditional way to use an imputation method for classification with incomplete data is that the imputation method is used to estimate missing values for both the training data and a new incomplete instance that needs be classified in the application process [8].

Imputation methods can be divided into single imputation methods and multiple imputation methods. While single imputation methods estimate one value for each missing value, multiple imputation methods estimate multiple values for each missing value [16].

2.1.1 Single imputation. Single imputation methods try to find one suitable value for each missing value. Two single common imputation methods are hot deck imputation and Knn-based imputation [5].

Hot deck imputation replaces missing values for an incomplete instance by searching the most similar with the incomplete instance, and then filling missing values with the complete values in the most similar instance. This method can replace missing values by real values. However, it only uses the information of one instance; therefore, it ignores global properties of the data [1].

Knn-based imputation is based on K-nearest neighbors algorithm. To replace missing values for each incomplete instance, firstly, it searches the K most similar instances with the incomplete instance, and then replaces missing values of the incomplete instance with the average of complete values in the K most similar instances. Knn-based imputation often performs better than hot deck imputation. However, this method is often computationally intensive because it has to search through all instances to find the K most similar instances for each incomplete instance [2].

2.1.2 Multiple imputation. Multiple imputation methods try to find a set of suitable values for each missing value. Multiple imputation is often computationally more expensive than single imputation. However, multiple imputation has becomes more and more popular. The main reason is that multiple imputation often reflects better the uncertainty of missingness [6, 21, 22, 26]. Moreover, many recent software developments have based on the multiple imputation framework [10].

MICE [27] is one of the most advanced multiple imputation methods. MICE uses some regression models to predict missing values. Firstly, each missing field is randomly replaced with one complete value in the same feature. Next, each incomplete feature is estimated on other features to build a better estimation for the feature. The process is done several times for all incomplete features to produce one imputed dataset. The whole procedure is done N times ($N > 1$) to produce N imputed datasets. Finally, the N imputed datasets are averaged to make the final imputed dataset.

Multiple imputation is originally designed for statistical analysis tasks [16]. Multiple imputation also has been demonstrated as a powerful imputation approach to estimating missing values for classification with incomplete data [6, 26]. Multiple imputation is suitable for batch imputation, so it is efficient to estimate missing values for the training data. However, multiple imputation is computationally expensive to estimate missing values for single incomplete instance that needs be classified in the application process [24]. Therefore, an effective and efficient way to apply multiple imputation for classification with incomplete data should be more investigated.

2.2 Genetic Programming for Classification

GP has been widely used to construct discriminant functions for classification tasks. A discriminant function is a mathematical expression that represents a combination of the features of an instance which needs be classified. The value returned by the discriminant function determines the predicted class by using a single threshold or a set of thresholds. The obvious way to construct discriminant functions by using GP is that each individual in GP population represents one discriminant function. The function set of GP is able to contain any type of functions and operations which is able to work on the data. GP has been used to construct both binary discriminant functions and multiple discriminant functions [4].

With binary classification problems, one discriminant function is adequate to distinguish two classes. When the function output of an instance is less than a given threshold, the instance is classified to a particular class, otherwise it is classified to the other one. The threshold is usually set zero, so a positive output of the function associates with a certain class, and a non-positive output of the function associates with the other one. In [20], a zero-threshold discriminant function is constructed, where a multi-objective approach is used to simultaneously optimise the class distribution posteriori entropy and classification accuracy. In [28], a single threshold discriminant function is evolved, where a fitness function is the combination of size penalty and classification accuracy. In [11], a single threshold function is also built, where a fitness function is the combination of classification accuracy and a measure of certainty. In [18], a discriminant function with single threshold is constructed, where class imbalance is addressed by two special fitness functions.

With multi-class classification, two main methods can be followed. One method is to solve a *n*-class classification problem by solving *n-1* binary classification problems; therefore, *n-1* binary discriminant functions are able to use to distinguish the n classes [13, 19]. The other method is only to construct one discriminant function for discriminating all the classes. In this approach, *n-1* threshold values are needed to make *n* intervals, and then each interval associates with one class [29, 30]. In [13], an n-class classification problem is considered as *n-1* binary classification problems, and classification accuracy is considered as the fitness measure. In [19], an n-class classification problem is also considered as *n-1* binary classification problems, but a fitness function is designed to estimate the overlapping between classes given by classifiers. In [29, 30], multiclass classification is tacked by constructing a multiple threshold discriminant function. The function discriminates the differences between n classes by n-1 threshold values. These thresholds defines n intervals, and each interval is used to represent a particular class. In [29], two methods are proposed to dynamically determine thresholds for classes. In this approach, instead of using static boundaries to discriminate different classes, the two methods gradually construct the boundaries during evolutionary process. The experimental results showed that the two dynamic methods can perform much better than static methods, especially with difficult classification problems.

GP cannot directly work with incomplete data. Therefore, GP needs to combine with an imputation method to build classifiers for incomplete data. However, a combination of GP and a simple imputation method usually leads to a weak classifier. In contrast, a combination of GP and a multiple imputation method often results in a smaller classification error, but it is computationally expensive [24]. Therefore, an effective and efficient combination of GP and multiple imputation should be investigated.

3 PROPOSED METHOD

Multiple imputation is a powerful approach to estimating missing values for incomplete data. It was originally designed for batch imputation, and is therefore suitable for estimating missing values for training data. However, multiple imputation is computationally intensive when estimating missing values for a single incomplete instance that needs be classified. Therefore, this paper proposes an efficient and effective way to use multiple imputation for classification with incomplete data.

The proposed method has two key ideas. The first key idea is to identify common patterns of missing values in the training data. After that, a classifier is built for each of these patterns that does not use the missing features in that pattern. This means that any new incomplete instance with the same pattern of missing values can be classified by the classifier without using imputation. The second key idea is to build classifiers using GP which is able to do further feature selection by only using a subset of original features in the classifiers. The result is that frequently more than one of the classifiers will apply for a new incomplete instance and the multiple predictions can be combined to give a better result.

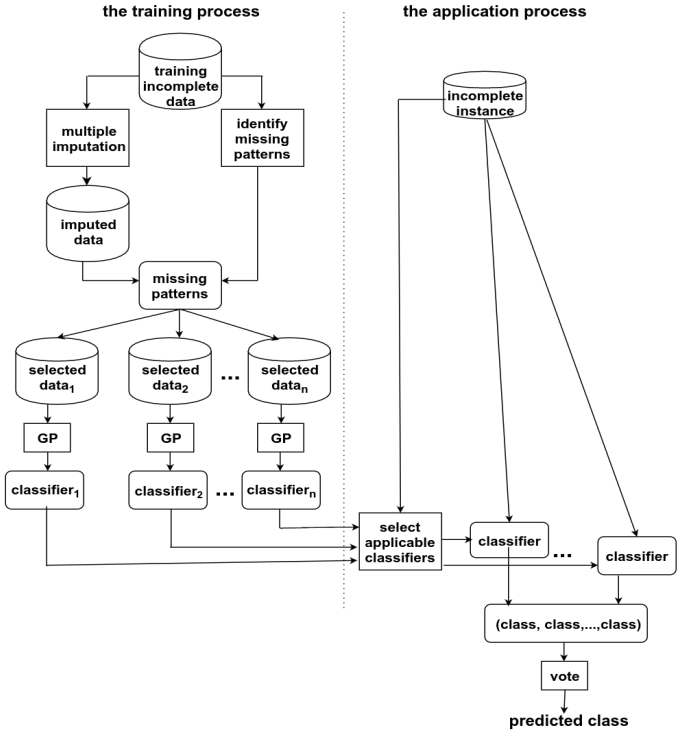

Figure 1: Classification with incomplete data by using multiple imputation *only* in the training process.

Fig. 1 shows main steps of the proposed method. The proposed method has two phases: the training process and the application process. The training process combines multiple imputation and GP to construct a set of classifiers. The application process classifies an incomplete instance by choosing a subset of applicable classifiers from the set of classifiers to directly classify the incomplete instance without using any imputation.

In the training process, the training incomplete data is put into a multiple imputation method to generate the training imputed data which is complete. The training incomplete data is also searched to identify all missing patterns from the training incomplete data. A missing pattern is a feature subset such that there is at least one instance in the training data that is missing data for exactly the features in the feature subset. Subsequently, for each missing pattern, a training selected data is created from the training imputed data by removing features which appear in the missing pattern. After that, GP uses the training selected data to build a classifier that is applicable to all incomplete instances with the same missing pattern. As a result, the training process generates a set of classifiers.

In the application process, to classify a new instance, the proposed method first identifies all applicable classifiers which do not require fields that are missing in the instance. After that, the instance is classified by the applicable classifiers, and then a majority vote is used to decide the final class label.

A similarity between the proposed method and the tradition method to use imputation for classification with incomplete data is that both of them use the imputation to estimate missing values for the training data. In the application process, the traditional method also requires the imputation to estimate missing values for a new incomplete instance that needs be classified. However, the proposed method builds a set of classifiers that can directly classify the new incomplete instance without requiring the imputation.

4 EXPERIMENT DESIGN

This section shows detailed experiment design including the comparison method, datasets, the imputation methods used in the experiments and GP settings.

4.1 The Comparison Method

The experiments are designed to evaluate the ability of the proposed method to classify incomplete data. To achieve this objective, the experiments are designed to compare the proposed method as shown in Fig.1 with a common approach using GP for classification with incomplete data as shown in Fig. 2. Fig. 2 shows the common approach to classification with incomplete data by using GP. In the benchmark approach, an imputation method is used to estimate missing values in both the training process and the application process. In training process, the incomplete data is put into to an imputation method to generate training imputed data which is then used by GP to build a classifier. In the application process, to classify an incomplete instance, the incomplete instance is firstly put into the imputation method to generate imputed instance which is then classified by the classifier. Both multiple imputation and single imputation are used in the benchmark approach.

The proposed method also uses a multiple imputation method to estimate missing values of the training incomplete data. However,

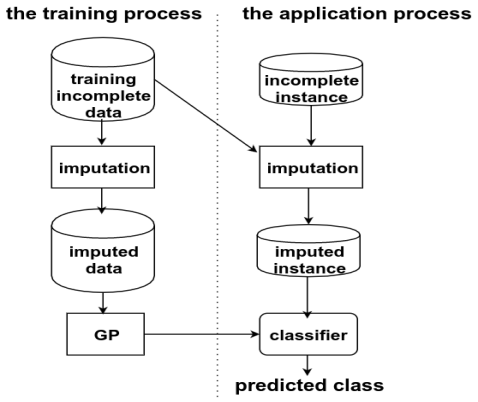

Figure 2: Classification with incomplete data by using an imputation method in *both* the training and testing process.

in the application process, instead of using an imputation method to estimate missing values, the proposed method constructs a set of classifiers which can directly classify incomplete instances without using any imputation.

4.2 Datasets

The experiments compared the proposed methods with the other methods on eight datasets. The datasets were chosen from the University of California at Irvine Machine Learning repository (UCI) [15]. The main characters of the selected datasets are shown in Table 1 including name, the number of features, the number of instances, the number of classes and the percentage of instances containing at least one missing field.

Table 1: Datasets used in the experiments

Dataset	#Features	#Inst	#Classes	Incomplete Inst (%)
Automobile	25	205	6	26.83
Bands	19	539	2	32.28
Hepatitis	19	155	2	48.39
Horsecolic	23	368	2	98.1
Ecoli	7	336	2	0
Parkinsons	22	197	2	0
Seedst	7	210	3	0
Wine	13	178	3	0

The first four datasets are "natural" incomplete datasets. These datasets have high percentages of incomplete instances such as Horsecolic dataset containing 98.1% incomplete instances. To evaluate the proposed methods on incomplete datasets with different levels of missing values, we used the four complete datasets, and removed some values in some features to create "artificial" incomplete datasets. To more precisely test the proposed methods, we only introduced missing values into important features. The correlation-based feature selection method (CFS) [9] was used to select the important features. Six levels of missing values: 5%, 10%, 15%, 20%,

25% and 30% were randomly introduced into the important features to generate incomplete datasets with different levels of missing values. For each level of missing values, we generated 30 incomplete datasets by randomly introduce the level of missing values in the important features. Therefore, from one complete dataset, 180 (30×6) artificial incomplete datasets were conducted. As a results, there are 720 (180×4) artificial incomplete datasets used in the experiments.

None of the datasets were separated into a training set and a test set. Furthermore, some datasets have a small number of instances. Therefore, a ten-fold cross-validation method was used to divide the datasets into training sets and test sets. The ten-fold cross-validation method was done 30 times on each dataset containing natural missing values. With artificial datasets, with each level of missing values, the ten-fold cross-validation method was done the 30 incomplete datasets. Therefore, 300 pairs of training and test sets were conducted.

4.3 Imputation algorithms

The proposed method was compared with imputation methods combined with GP. Three imputation methods were used to combine with GP including two single imputations: hot deck imputation, Knn-based imputation, and a multiple imputation: MICE. Hot deck imputation and Knn-based imputation were in-house implementation. With KNN-based imputation, the number of neighbors K was set 10. MICE's implementation in [3] was used for multiple imputation. In MICE, random forest was used as a regression method to estimate missing values. Each incomplete feature was repeatedly regressed 20 times on other features. With each incomplete dataset, the multiple imputation was repeatedly done 10 times to generate 10 imputed datasets before averaging them to generate a final imputed dataset.

4.4 GP settings

We used the ECJ package [17] to implement GP. Table 2 shows the parameters of GP used in all the experiments.

Table 2: GP parameters.

Parameter	Value
Function set	+, -, x, / (protected division)
Variable terminals	$\{f_1, f_2, .., f_n\}$
Population size	1024
Initialization	Ramped half-and-half
Generations	50
Crossover probability	60%
Mutation probability	30%
Reproduction rate	10%
Selection type	Tournament(size=7)

5 RESULTS AND ANALYSIS

This section shows the comparison between the proposed methods and the other methods on accuracy and computation time. Moreover, a comprehensive analysis is done to demonstrate the advantage of the proposed method.

5.1 Classification Accuracy

Table 3 presents the average of classification accuracy along with standard deviation of the proposed method and the other methods on the first four datasets. The average of classification accuracy in Table 3 was calculated based on accuracies of each method on 30 times performing ten-fold cross-validation on each dataset. Table 4 shows the average of classification accuracy along with standard deviation of the proposed method and the other methods on the last four datasets with six levels of missing values. With each dataset and each level of missing values, the averages of classification accuracy in Table 4 was calculated based on accuracies of each method on 30 generated incomplete datasets by using ten-fold cross-validation on each incomplete dataset.

In Table 3 and Table 4, the MiceFSGP column presents results from the proposed method as shown in Fig.1. The MiceGP, HotGP and KnnGP columns present results from the benchmark experimental setup as shown in Fig.2 by combining GP with MICE imputation, hot deck imputation and Knn-based imputation, respectively.

For each incomplete dataset, Friedman test [7], which is a non-parametric test for multiple comparisons, is used to statistical test the null hypothesis in classification accuracies over 300 times at a 5% level of significance. The test shows that for all tasks, there is a significant difference in classification accuracies for the four methods, so the null hypothesis is rejected. Therefore, a post hoc multiple comparisons test using the Holm method [12] is used to determine the statistically significant differences between group means. "T" in the two tables show significant tests of the columns before them against MiceFSGP, where "+" means MiceFSGP is significantly more accurate, "=" means not significantly different and "-" means significantly less accurate.

It can be seen clearly from Table 3 that with the natural incomplete datasets, MiceFSGP can achieve significantly better classification accuracy than the other methods in almost all cases. MiceFSGP is significantly more accurate than MiceGP on three datasets, and similar to MiceGP on one dataset. Moreover, MiceFSGP is significantly more accurate than both HotGP and KnnGP in all cases.

It also can be seen clearly from Table 4 that with the artificial incomplete datasets, MiceFSGP can achieve significantly better accuracy than all the other methods on all cases. MiceFSGP achieves significantly better classification accuracy than all the other methods on all the 24 cases.

In order to confirm if the proposed methods are really significantly better than the other methods, we perform the Friedman test on the average of accuracies of all the algorithms in all incomplete datasets as shown in Table 3, and Table 4. The test indicates that there is a significant difference in classification accuracies in the four methods, so the null hypothesis is rejected. Therefore, the Holm method [12] is used to determine the statistically significant differences between pairs of algorithms on all the incomplete datasets. Table 5 shows the significance comparison between all pairs of algorithms. As demonstrated from Table 5, on all the incomplete datasets, MiceFSGP is significantly more accurate than the other methods. As also can be seen from Table 5 that MiceGP is significantly better than single imputation methods combined with GP showing that multiple imputation generates a more reliable imputed dataset. Table 6 shows the ranking of the algorithms using

Table 3: Classification accuracy with datasets containing natural missing values.

Dataset	MiceFSGP	MiceGP	T	HotGP	T	KnnGP	T
Automobile	**50.31**±2.79	46.64±3.06	+	45.01±2.66	+	45.76±3.15	+
Bands	**68.31**±0.78	67.73±1.27	=	61.12±2.18	+	61.25±1.58	+
Hepatitis	**82.47**±1.28	81.01±1.83	+	80.20±2.69	+	80.33±2.58	+
Horsecolic	**85.45**±0.37	84.36±0.74	+	81.56±2.38	+	81.75±2.41	+

Table 4: Classification accuracy with datasets containing artificial missing values.

Dataset	Missing rate (%)	MiceFSGP	MiceGP	T	HotGP	T	KnnGP	T
Ecoli	5	**68.69**±1.26	65.40±2.35	+	63.60±2.25	+	63.96±2.18	+
	10	**69.70**±1.16	65.83±1.85	+	62.19±1.88	+	62.44±1.76	+
	15	**70.11**±1.40	65.52±1.99	+	58.59±2.50	+	60.98±2.28	+
	20	**69.77**±1.45	65.14±2.15	+	56.86±2.43	+	59.74±2.18	+
	25	**69.03**±1.36	63.82±2.33	+	54.10±3.16	+	57.74±2.50	+
	30	**67.98**±1.91	63.25±2.91	+	52.74±2.94	+	56.04±2.76	+
Parkinsons	5	**89.09**±0.80	86.46±2.01	+	84.93±1.77	+	84.76±1.75	+
	10	**88.75**±0.87	86.39±1.50	+	84.02±1.78	+	84.68±2.16	+
	15	**88.42**±1.17	86.46±1.47	+	81.45±2.07	+	83.40±1.96	+
	20	**88.50**±1.40	86.19±1.53	+	80.71±2.49	+	81.50±2.75	+
	25	**88.22**±1.08	85.31±2.17	+	79.22±2.74	+	80.14±2.79	+
	30	**87.60**±1.45	85.45±2.10	+	77.16±2.71	+	79.79±3.27	+
Seedst	5	**91.73**±1.18	86.06±2.65	+	79.22±2.98	+	79.79±2.60	+
	10	**91.46**±1.45	84.44±2.63	+	75.31±2.54	+	75.93±3.38	+
	15	**91.36**±1.28	84.98±2.37	+	71.22±2.91	+	72.11±2.96	+
	20	**90.79**±1.08	85.31±2.07	+	68.63±3.27	+	69.50±2.75	+
	25	**89.63**±1.33	83.42±2.70	+	64.84±4.49	+	66.63±3.17	+
	30	**88.98**±1.62	83.46±2.30	+	61.14±3.90	+	64.80±3.27	+
Wine	5	**90.46**±1.35	85.87±2.54	+	82.14±3.46	+	82.33±3.32	+
	10	**89.97**±1.63	86.20±2.78	+	78.43±4.26	+	78.90±4.14	+
	15	**89.20**±1.30	83.91±2.20	+	72.90±3.97	+	74.48±3.55	+
	20	**88.36**±2.17	84.66±3.17	+	67.87±3.86	+	70.94±4.36	+
	25	**87.22**±1.87	85.07±2.79	+	66.16±4.34	+	68.89±3.80	+
	30	**87.09**±1.76	84.29±2.57	+	62.30±3.36	+	66.15±4.27	+

the Friedman test *(smaller means better).* As is evident from Table 6 that the proposed methods are the best algorithms.

In summary, the proposed method can achieve better accuracy than the other methods not only with natural incomplete datasets, but also with different levels of artificial incomplete datasets.

Table 5: The significant comparison between some methods on all incomplete datasets *(Holm's procedure rejects those hypotheses that have a p-value <= 0.05).*

Algorithms	p-value
MiceFSGP vs MiceGP	0.0038
MiceFSGP vs HotGP	0.0000
MiceFSGP vs KnnGP	0.0000
MiceGP vs HotGP	0.0027
MiceGP vs KnnGP	0.0000
HotGP vs KnnGP	0.0071

Table 6: The ranking of the methods on all incomplete datasets using Friedman test (smaller means better.).

MiceFSGP	MiceGP	HotGP	KnnGP
1.0	2.0	3.03	3.96

5.2 Computation time

In order to compare the computation time of the proposed method and the other methods in the application process, the average of computation time to classify one instance on the first four "natural" incomplete datasets is estimated. Table 7 shows the average of computation time to classify one instance of the first four datasets.

Table 7: The average of computation time for classifying one (complete/incomplete) instance in the testing process (*millisecond*)

Dataset	MiceFSGP	MiceGP	HotGP	KnnGP
Automobile	0.006	3.59×10^3	0.05	0.13
Bands	0.03	1.21×10^4	0.05	0.18
Hepatitis	0.02	7.98×10^3	0.10	0.23
Horsecolic	0.11	5.87×10^4	0.06	0.08

It is clear from Table 7 that the proposed method is faster than the other methods in almost all cases. MiceFSGP is faster than both HotGP and KnnGP on the first three datasets. The underlying reason is that in the application process, to classify an incomplete instance, although MiceFSGP has to spend time to classify the instance by more than one classifier, it does not have to spend time for estimating missing fields. In contrast, the other methods have to spend time for estimating missing fields before classifying the incomplete instance. With *Horsecolic* dataset, MiceFSGP is slightly more expensive than the two single imputation methods combined with GP. It is likely that *Horsecolic* dataset contains 98.1% incomplete instances so MiceFSGP has to build a large number of classifiers; therefore, in application process, to classify an incomplete instance, MiceFSGP has to spend a long time to search suitable classifiers for classifying the incomplete instance.

It is also clear from Table 7 that MiceFSGP is thousand times faster than MiceGP. The key reason is that in application process, to classify an incomplete instance, MiceGP require rebuilding the regression functions using all the training data and the incomplete instance. In contrast, MiceFSGP does not require estimating missing fields, and only spends time for classifying the incomplete instance.

In summary, the proposed method not only is more accurate but also faster than the other methods.

5.3 Further Analysis

In order to know how MiceFSGP can achieve better classification accuracy than the other methods, we further analysed classifiers evolved by MiceFSGP. Table 8 shows the average of total number of classifiers evolved by MiceFSGP on the first four "natural " incomplete datasets. Table 9 shows the average of number classifiers evolved by MiceFSGP which are chosen to classify one incomplete instance in the application process.

Table 8: The average of the total number of classifiers evolved by MiceFSGP.

Automobile	Bands	Hepatitis	Horsecolic
6.8	36.7	19.8	70.2

Table 9: The average of number of suitable classifiers evolved by MiceFSGP for classifying one incomplete instance.

Automobile	Bands	Hepatitis	Horsecolic
5.4	17.9	14.9	40.1

In training process, for each missing pattern, MiceFSGP only constructs one classifier. However, it is clear from Table 8 and Table 9 that more than half of classifiers evolved by MiceFSGP are used to classify each incomplete instance in the application process. The key reason is that when GP constructs a classifier, it can select some relevant features and remove some redundant features. Thanks to removing some features, a classifier evolved GP can be used to classify any incomplete instance which can contain missing values in the removed features. Therefore, a classifier evolved by MiceFSGP for one missing pattern can be used to classify for other missing patterns. By evolving a set of suitable classifiers for classifying incomplete instances, MiceFSGP not only can save time for estimating missing values in the application process, but also can achieve better classification accuracy than the other methods.

In order to demonstrate the ability of GP in feature selection, we analysed a tree generated by MiceFSGP on the *Hepatitis* dataset. The Hepatitis dataset which has 19 features $\{f_1, ..f_{20}\}$ was chosen because trees generated on Hepatitis are not too big to analyse. Fig. 3 shows a tree evolved by MiceFSGP on Hepatitis using the full training set. Although the tree is constructed by MiceFSGP using *full* training set the tree only contains five features: $\{f_2, f_{11}, f_{12}, f_{13}, f_{15}\}$. As a results, the tree can be used to classify

any incomplete instance which contains missing values in the rest 14 features.

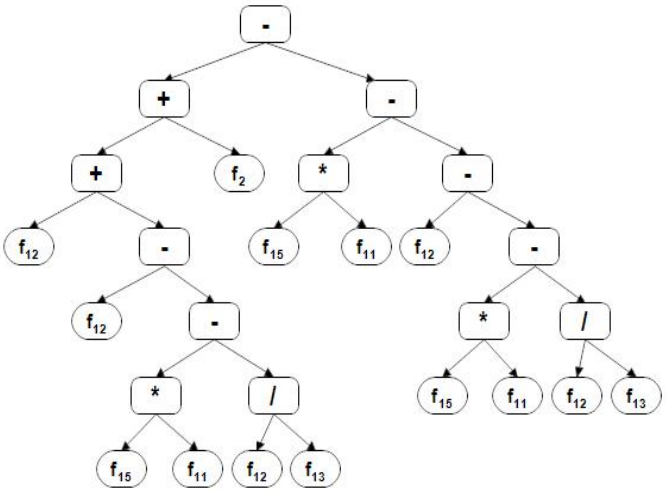

Figure 3: A tree evolved by MiceFSGP using full training *Hepatitis* data.

In summary, by suitably combining a powerful multiple imputation method and the ability of GP for evolving classifiers, the proposed method can achieve better performance than the other methods.

6 CONCLUSIONS AND FUTURE WORK

This paper proposed an effective system to combine multiple imputation and GP to evolve classifiers for classification with incomplete data. Multiple imputation is used to estimate missing values for the training incomplete data. Furthermore, the proposed method searches for all common missing patterns, and then GP is used to build a set of classifiers, one for each missing pattern. To classify an incomplete instance, a subset of applicable classifiers is used to classify the incomplete instance without requiring any imputation. The proposed method was compared with other common methods for classification with incomplete data which combine GP with imputation methods in both the training process and the application process. Experimental results showed that the proposed method can achieve better classification accuracy than the benchmark methods, and can be faster to classify incomplete data than the benchmark methods. Further analysis shows that the ability of GP to do feature selection during evolving classifiers helps the proposed method to have multiple classifiers for classifying each incomplete instance to give better accuracy.

Some other kinds of classification algorithms such as decision trees can automatically perform feature selection during building classifiers like GP. Therefore, a future work could investigate a combination of multiple imputation and the kinds of classification algorithms for constructing classifiers for incomplete data.

REFERENCES

[1] R. R. Andridge and R. J. Little. A review of hot deck imputation for survey non-response. *International statistical review*, 78(1):40–64, 2010.

[2] G. E. Batista and M. C. Monard. A study of k-nearest neighbour as an imputation method. In *Hybrid Intelligent Systems - HIS*, pages 251–260, 2002.

[3] S. Buuren and K. Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45(3), 2011.

[4] P. G. Espejo, S. Ventura, and F. Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(2):121–144, 2010.

[5] A. Farhangfar, L. Kurgan, and J. Dy. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41:3692–3705, 2008.

[6] A. Farhangfar, L. A. Kurgan, and W. Pedrycz. A novel framework for imputation of missing values in databases. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37:692–709, 2007.

[7] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937.

[8] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Computing and Applications*, 19:263–282, 2010.

[9] M. A. Hall. Correlation-based feature selection of discrete and numeric class machine learning. 2000.

[10] O. Harel and X.-H. Zhou. Multiple imputation: review of theory, implementation and software. *Statistics in medicine*, 26:3057–3077, 2007.

[11] K. Hennessy, M. G. Madden, J. Conroy, and A. G. Ryder. An improved genetic programming technique for the classification of raman spectra. *Knowledge-Based Systems*, 18:217–224, 2005.

[12] S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.

[13] J. Kishore, L. M. Patnaik, V. Mani, and V. Agrawal. Application of genetic programming for multicategory pattern classification. *Evolutionary Computation, IEEE Transactions on*, 4:242–258, 2000.

[14] J. R. Koza. *Genetic programming III: Darwinian invention and problem solving*, volume 3. 1999.

[15] M. Lichman. UCI machine learning repository, 2013.

[16] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[17] S. Luke, L. Panait, G. Balan, S. Paus, Z. Skolicki, E. Popovici, K. Sullivan, J. Harrison, J. Bassett, R. Hubley, et al. A java-based evolutionary computation research system. *Online (March 2004) http://cs. gmu. edu˜ eclab/projects/ecj*, 2004.

[18] G. Patterson and M. Zhang. Fitness functions in genetic programming for classification with unbalanced data. In *AI 2007: Advances in Artificial Intelligence*, pages 769–775. 2007.

[19] S. Silva and Y.-T. Tseng. Classification of seafloor habitats using genetic programming. In *Applications of Evolutionary Computing*, pages 315–324. 2008.

[20] W. A. Tackett. Genetic programming for feature discovery and image discrimination. In *ICGA*, pages 303–311, 1993.

[21] C. T. Tran, P. Andreae, and M. Zhang. Impact of imputation of missing values on genetic programming based multiple feature construction for classification. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pages 2398–2405, 2015.

[22] C. T. Tran, M. Zhang, and P. Andreae. Multiple imputation for missing data using genetic programming. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation*, pages 583–590, 2015.

[23] C. T. Tran, M. Zhang, and P. Andreae. Directly evolving classifiers for missing data using genetic programming. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 5278–5285, 2016.

[24] C. T. Tran, M. Zhang, and P. Andreae. A genetic programming-based imputation method for classification with missing data. In *European Conference on Genetic Programming*, pages 149–163, 2016.

[25] C. T. Tran, M. Zhang, P. Andreae, and B. Xue. Directly constructing multiple features for classification with missing data using genetic programming with interval functions. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pages 69–70, 2016.

[26] C. T. Tran, M. Zhang, P. Andreae, B. Xue, and L. T. Bui. Multiple imputation and ensemble learning for classification with incomplete data. In *Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings*, pages 401–415, 2017.

[27] I. R. White, P. Royston, and A. M. Wood. Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in medicine*, 30:377–399, 2011.

[28] L. Zhang, L. B. Jack, and A. K. Nandi. Fault detection using genetic programming. *Mechanical Systems and Signal Processing*, 19:271–289, 2005.

[29] M. Zhang and W. Smart. Multiclass object classification using genetic programming. In *Applications of Evolutionary Computing*, pages 369–378. 2004.

[30] M. Zhang and W. Smart. Using gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. *Pattern Recognition Letters*, 27:1266–1274, 2006.