Handling Incomplete Quantitative Data for Supervised Learning based on Fuzzy Entropy

Been-Chian Chien, *Member*, *IEEE*, Cheng-Feng Lu, and Steen-J. Hsu

***Abstract*—In recent years, machine learning and knowledge discovery techniques have attracted a great deal of attention in the information area. Classification is one of the important research topics on these research areas. Most of the researches on classification concern that a complete data set is given as a training set and the test data know all values of attributes clearly. Unfortunately, incomplete data are commonly seen in real-world applications. In this paper, we propose a new strategy to deal with the incomplete quantitative data and introduce a supervised learning method based on genetic programming to handle the classification problem with incomplete data in the attributes. Two experiments are designed to evaluate the effectiveness of the proposed approaches.**

***Index Terms*—Classification, genetic programming, fuzzy entropy, incomplete data.**

I. Introduction

CLASSIFICATION is an important research topic in knowledge discovery and machine learning. The classification problem is a task of supervised learning that consists of two phases. The first phase is to learn a classification model from a training set with predefined classes. In the second phase, the learned classification model is used to classify an unknown case to one of the predefined classes. Typically, the training set is assumed to be collected completely without loss or errors inside. Thus, reliable classification models can be learned and represented in the form of classification rules, decision trees, or mathematical functions. However, imperfect data happen usually in real world applications. The existence of imperfect data will generally degrade the quality of classification models. Since imperfect data may be caused by erroneous recording, uncertain decision, incomplete storing or vague description [1], different strategies of handling imperfect data are necessary. In this paper, we investigate the problem of incomplete data in datasets. A dataset with at least one missing attribute value is referred as an incomplete dataset. Since the incomplete samples do not provide perfect information for training process, most of the traditional classification algorithms can not deal with incomplete data directly but generate inaccurate classifiers from an incomplete data. Hence, the incomplete data must be tackled well so that good classification models can be developed for real life applications.

Many techniques on dealing with missing values have been studied in the past decade. However, most of the related researches focus on the process of missing data in the training set only. Since the class of each object in the training set is known, it is easy to fill the missing attributes by processing the existing data of attributes in the same class. For the missing values appearing in the unknown cases, the current approaches can not work well. In this paper, we introduce a new method to handle the problem of incomplete data and propose a classification method using discriminant functions to classify the unknown cases having missing values in their attributes. For the proposed method of handling incomplete data, the technique of granular process based on fuzzy entropy is used to fill the values of missing attributes. Then, a GP-based supervised learning method is applied to generate a set of discriminant functions and the classification algorithms based on the set of discriminant functions are developed to classify unknown cases with missing values. To evaluate the performance of the proposed schemes, two experiments containing complete test datasets and incomplete test datasets are designed. The sets of test data also include real and artificial incomplete datasets generated from UCI Machine Learning repository [2]. By comparing with other classifiers and different resolutions of incomplete data, we can find that the granular scheme can provide better classification accuracy in the datasets with low missing rate than the datasets with high missing rates. First, in the following, we review some related researched on incomplete data handling and the techniques of generating classification models using genetic programming briefly.

A. Review of Incomplete Data Handling

Learning from incomplete datasets is more difficult than learning from complete datasets. Designing a sophisticated learning algorithm being able to deal with incomplete datasets presents a challenge to researchers. In the past, many methods were proposed to handle the problem of incomplete datasets [3]-[8]. Usually, incomplete data first can be transformed into complete data (such as using similarity measure) before the learning program starts. For example, objects with unknown values may be directly removed from datasets [3] or processed in a particular way [4]-[6]. We summarize the related methods of preprocessing incomplete data as follows.

B.-C. Chien is with the Department of Computer Science and Information Engineering, National University of Tainan, Tainan 700, Taiwan, R.O.C (e-mail: bcchien@mail.nutn.edu.tw).

C.-F. Lu is with the Department of Information Engineering, I-Shou University, Kaohsiung 840, Taiwan, R.O.C.

Steen-J. Hsu is is with the Department of Information Engineering, I-Shou University, Kaohsiung 840, Taiwan, R.O.C.

1) Ignore the samples or attributes [3]: The objects or attributes with incomplete information in the training set are all deleted. This method will decrease plenty information and drive ineffective classification model unless the objects or attributes contain too many missing values.

2) Fill in the missing value manually: In general, this method is time-consuming and may be not feasible when a large dataset have many missing values. Further, what values should be filled is another problem.

3) Use the mean of existing values in the attribute to fill in the missing value [4]: This is a simple method which can only work for quantitative data. At the same time, this method is effective only when the distribution of the attribute values is discriminative with the class label.

4) Concept most common attribute value [5]: The value of the attribute which occurs most common within the class label is selected to be the value of the missing attribute values. This method is commonly used if the class label is dependent on the attribute.

5) Use rough set approach to handle the missing value [6]-[8]: That idea is graceful and many researchers are interesting in this area. But the method may be time-consuming [8] when the data set has many attributes and the attributes include many different values.

All above related researches are limited on processing incomplete data in the training datasets. While the missing values are appeared in the unknown test data, the previous methods cannot work since the class label of the object is unknown. For solving the problem of incomplete data containing in the test data, a new style of classifier should be developed. A classifier using a set of discriminant functions thus is proposed in here.

B. Genetic Programming based Classification Model

The main idea of generating discriminant functions is based on genetic programming. Several supervised learning methods based on genetic programming have been proposed these years [9]-[12]. The genetic programming (GP) is one of techniques on evolutionary computation which was proposed by Koza [13], [14] in 1987. The genetic programming has been applied to several applications like symbolic regression, the robot control programs, and classification, etc. Genetic programming can discover underlying data relationships and presents these relationships by expressions. An expression is constructed by terminals and functions. There are several types of functions can be applied to the genetic programming:

1) Arithmetic operations: addition, subtraction, etc.

2) Trigonometric functions: Sine and Cosine, etc.

3) Conditional operators: IF, ELSE and OR, etc.

4) Other add-on and user-specific functions.

The algorithm of a genetic programming begins with a population that is a set of randomly created individuals. Each individual represents a potential solution that is represented as a binary tree. Each binary tree is constructed by all possible compositions of the sets of functions and terminals. A fitness value of each tree is calculated by a suitable fitness function. According to the fitness value, a set of individuals having better fitness will be selected. These individuals are used to generate new population in next generation with three genetic operators: reproduction, crossover, and mutation. The evolution will be continued if the fitness value of the individual still does not satisfy the specified conditions of the solution until the specified number of generations is reached. Then, classification rules or discrimnant functions can be learned for classification from the training set.

II. HANDLING INCOMPLETE QUANTITATIVE DATA

For handling incomplete quantitative data, we first apply fuzzy entropy to discriminate the best number of intervals for each quantitative attribute. Each quantitative interval is granulated as a fuzzy linguistic term with a membership function. Then, we employ the linguistic terms to infer the missing attribute values based on the max-min composition method according to their class labels.

A. Fuzzy Entropy

Fuzzy entropy is a measure of the amount of uncertainty of fuzzy set. We use the fuzzy entropy proposed by Lee et al. [15] to discriminate the best number of intervals for the quantitative attribute. The fuzzy entropy is defined as follows.

Definition 1: Fuzzy entropy of an interval for each feature:

1) Let $X = \{x_1, x_2, ..., x_n\}$ be a universal set with elements x_i distributed in a pattern space, where $i = 1, 2, . . ., n$.
2) Let A be a fuzzy set defined on an interval of pattern space which contains k elements ($k<n$). The mapped membership degree of the element x_i with the fuzzy set A is denoted by $\mu_A(x_i)$.
3) Let $C_1, C_2, ..., C_K$ represent K classes into which the n elements are divided.
4) Let $S_j(x_n)$ denote a set of elements of class j on the universal set X. It is a subset of the universal set X.
5) The match degree D_j with the fuzzy set A for the elements of class j in an interval, where $j = 1, 2, ..., K$, is defined as

$$D_j = \sum_{x \in S_j(x_n)} \mu_A(r) \Big/ \sum_{x \in X} \mu_A(r). \tag{1}$$

6) The fuzzy entropy $FE_j(A)$ of the elements of class j in an interval is defined as

$$FE_j(A) = -D_j \log_2 D_j. \tag{2}$$

7) The fuzzy entropy $FE(A)$ on the universal set X for the elements within an interval is defined as

$$FE(A) = \sum_{j=1}^{K} FE_{C_j}(A). \tag{3}$$

B. Membership Functions of Intervals

We use a triangular form membership function to represent each fuzzy set. When assigning a membership function to an interval, we have to consider three cases: the left-most interval, the internal interval and the right-most interval. The membership grade for an element x belonging to a fuzzy set is defined as follows:

Case 1: The left-most case:

$$\mu(x)=\begin{cases}1, & \text{for } x\le c_1\\ \max\left\{0,1-\dfrac{|x-c_1|}{|c_2-c_1|}\right\}, & \text{for } x>c_1\end{cases} \tag{4}$$

Case 2: The interval case:

$$\mu(x)=\begin{cases}\max\left\{0,1-\dfrac{|x-c_i|}{|c_i-c_{i-1}|}\right\}, & \text{for } x\le c_i\\ \max\left\{0,1-\dfrac{|x-c_i|}{|c_i-c_{i+1}|}\right\}, & \text{for } x>c_i\end{cases} \tag{5}$$

Case 3: The right-most case:

$$\mu(x)=\begin{cases}\max\left\{0,1-\dfrac{|x-c_r|}{|c_r-c_{r-1}|}\right\}, & \text{for } x\le c_r\\ 1, & \text{for } x>c_r\end{cases} \tag{6}$$

where c_1 is the center of the left-most interval, c_r is the center of the right-most interval, and c_i is the center of the internal interval.

C. The Algorithm for Handling Incomplete Quantitative Data

Input: An incomplete quantitative data set with the attributes $\boldsymbol{A}$.
Output: A complete quantitative data set.

```
k = 0.
do {
  k = k + 1.
  Set the initial number of intervals I = 2.
  do {
    Set the initial center of each interval.
    do {
      Assign the membership function for each interval.
      Assign interval label to each element.
      Re-computing the interval centers.
    } while ( Each interval center be changed )
    Compute fuzzy entropy FE(I) for I interval.
    if (FE(I) < FE(I-1))
      { I ← I + 1. }
  } while ( FE(I) < FE(I - 1) )
  Use fuzzy inference to replace missing attribute values.
  Defuzzify the value of each missing attribute value.
} while ( k < |A| )
```

III. The Classification Algorithms

For learning discriminant functions from training sets, the fitness function is important for genetic programming to generate effective solutions. We define two types of fitness functions, the first one is *fix interval* and the other is *dynamic interval*.

The fitness function with fix interval is that we consider a discriminant function f_i of a class C_i, and define two a specified constant a and p. To achieve the objectivity of f_i, we define the error measure of a positive instance to be

$$D_p=\begin{cases}0 & \text{if } c_j=C_i \text{ and } |f_i(x_j)-a|\ge 0\\ [p-f_i(x_j)]^2 & \text{if } c_j=C_i \text{ and } |f_i(x_j)-a|<0\end{cases}, \tag{7}$$

and the error measure of a negative instance to be

$$D_n=\begin{cases}0 & \text{if } c_j\ne C_i \text{ and } |f_i(x_j)-a|<0\\ [f_i(x_j)-q]^2 & \text{if } c_j\ne C_i \text{ and } |f_i(x_j)-a|\ge 0\end{cases}. \tag{8}$$

The dynamic interval fitness function is that we urge the values of $f_i(x_j)$ for positive instances to fall in the interval $[\overline{X}_i^{(gen)}-r_i,\ \overline{X}_i^{(gen)}+r_i]$. At the same time, we also wish that the values of $f_i(x_j)$ for negative instances are mapped outside the interval $[\overline{X}_i^{(gen)}-r_i, \overline{X}_i^{(gen)}+r_i]$, where $\overline{X}_i^{(gen)}$ is the mean value of an individual $f_i(x_j)$ in the gen-th generation of evolution,

$$\overline{X}_i^{(gen)}=\frac{\sum_{\substack{<x_j,c_j>\in TS',\\ c_j=C_i}} f_i(x_j)}{m_i},\ 1\le j\le m_i,\ 1\le i\le K. \tag{9}$$

Let r_i be the maximum distance between $\overline{X}_i^{(gen)}$ and positive instances $<x_j, c_j>$ for $1\le j\le m_i$. That is,

$$r_i=\max_{1\le j\le m_i}\{|\overline{X}_i^{(gen)}-f_i(x_j)|\},1\le i\le K. \tag{10}$$

We measure the error of a positive instance for (gen+1)-th generation by

$$D_p=\begin{cases}0 & \text{if } c_j=C_i \text{ and } |f_i(x_j)-\overline{X}_i^{(gen)}|\le r_i\\ 1 & \text{if } c_j=C_i \text{ and } |f_i(x_j)-\overline{X}_i^{(gen)}|> r_i\end{cases}, \tag{11}$$

and the error of a negative instance by

$$D_n=\begin{cases}1 & \text{if } c_j\ne C_i \text{ and } |f_i(x_j)-\overline{X}_i^{(gen)}|\le r_i\\ 0 & \text{if } c_j\ne C_i \text{ and } |f_i(x_j)-\overline{X}_i^{(gen)}|> r_i\end{cases}. \tag{12}$$

Then, the fitness value of an individual f_i is evaluated by the following fitness function:

$$fitness(f_i, T)=\sum_{j=1}^{m}(D_p+D_n), \tag{13}$$

where T stands for the training set. The fitness value of an individual represents the degree of error between the target function and the individual. The fitness value should be as small as possible.

After generating the classification functions by genetic programming from the training set, we obtain a set of discriminant functions F. However, these functions may conflict each other in practical cases and can't classify the cases with missing attribute values. To avoid the situations of conflict, we proposed the scheme [11], the Z-value measure, is used. If the Z-value of an unknown object x_j for the discriminant function f_i is minimum, then x_j belongs to the class C_i. For the problem of unknown data with missing attribute values, we introduce a simple idea to solve this problem. We present the classification approach in the following.

For a discriminant function $f_i \in F$ corresponding to the class C_i and T_{c_i} is the set of positive instances for class C_i belonging to the training set T,

$T_{C_i} = \{x_j \mid <x_j, c_j> \in T \text{ with } c_j = C_i\}$ and $|T_{C_i}| = m_i$.

Let $\overline{X}_i$ be the mean of values of $f_i(x_j)$. The standard deviation of values of $f_i(x_j)$, $1 \le j \le m_i$, is defined as

$$\sigma_i = \sqrt{\frac{\sum_{<x_j, c_j> \in T_{C_i}} (f_i(x_j) - \overline{X}_i)^2}{m_i}}, 1 \le j \le m_i, 1 \le i \le K. \quad (14)$$

Now, for an data set S, let a data $x \in S$ and a discriminant function $f_i \in F$, the Z-value of data x for f_i is defined as

$$Z_i(x) = \frac{|f_i(x) - \overline{X}_i|}{\sigma_i}, \quad (15)$$

where $1 \le i \le K$. We used the Z-value to determine the class to which the data should be assigned. The detailed classification algorithm is listed as follows.

For an unknown class data include missing attribute values, we use the value r_{ik} inferred in Section II-*C* to replace the k-th missing attribute value for the classification function f_i, where $1 \le i \le K$ and $1 \le k \le |A|$. Furthermore, we propose another form to replace the missing attribute value, and the idea is similar to previous described method. We use the value v_{ik} to replace the missing attribute value for the classification function f_i, the value v_{ik} is defined as follows and where k is the attribute of missing data.

$$v_{ik} = \frac{\sum_{<x_j, c_j> \in T_{C_i}} x_{jk}}{m_i}, 1 \le j \le m_i, 1 \le k \le |\boldsymbol{A}|, 1 \le i \le K. \quad (16)$$

Finally, based on the Z-measure, two algorithms: Algorithm Z and Algorithm Z-min for resolving conflict. Algorithm Z-min evaluates the Z-values of unknown case on all functions and assigns the case to the class with minimum Z-value. Instead of evaluating all the Z-values, Algorithm Z considers only the Z-values that conflict occurs.

IV. Experimental Results

We design two experiments to demonstrate and compare the performance of the proposed strategy for handling missing attribute values. The proposed learning algorithm based on genetic programming is implemented by modifying the GPQuick 2.1 [16]. The parameters of GPQuick used in the experiments are listed in TABLE I. The two experiments were conducted on a PC with 3.4 GHz CPU and 256 MB RAM. In order to understand the effect of our proposed strategy for the problem of incomplete data, we generated incomplete datasets artificially selected from UCI Machine Learning repository [2]. The selected datasets contain Iris, wine, bupa, and pima. The selected datasets are modified to obtain the incomplete data sets by randomly selecting a specified percentage of cases to set to be null. The generation of missing values follows the constraints:

C1) Each original case retains at least one attribute value.

C2) Each attribute has at least one value presents null.

The performance of the classification scheme is evaluated by the average classification error rate of 10-fold cross validation for 10 runs. Each run works using a new generated incomplete dataset. In the experiments, we compare the effectiveness with different classification models. These models include statistical models like Naïve Bayes, NBTree; decision tree based classifiers like C4.5 [17], and the support vector machines classifier [18]. TABLE II describes the classification tools we used. The training time of each set of discriminant functions for all selected data sets are shown in TABLE III.

For evaluate the effectiveness of the proposed strategy, we use two experiments to archive the goal and compare with other classifiers. In the two experiments, we compare the proposed strategy with the simple method, i.e. using the attribute mean of same concept to fill in the missing value. Because this method is often used and easy be implemented.

A. Experiment 1

The first experiment separates the complete cases from incomplete cases. The training set contains the incomplete cases with the complete cases but the testing samples have complete cases only. The objective of the experiment is that we hope to evaluate the effectiveness of the proposed strategy to fill the missing attribute values into training samples. TABLE IV represents the results of classification error rate using the selected incomplete datasets. In TABLE IV, the percentage of missing data is only up to 30%. This situation explains that the 40% missing rate is not applicable for Experiment 1.

From the experimental results, we observed that the proposed strategy is better than the attribute mean of same concept method when the missing rate is under 10%, and GPIZ

TABLE I
THE PARAMETERS OF GPQUICK USED IN THE EXPERIMENTS

Parameters	Values	Parameters	Values
Node mutate weight	43.5%	Crossover weight annealing	20%
Mutate constant weight	43.5%	Mutation weight	8%
Mutate shrink weight	13%	Mutation weight annealing	40%
Selection method	Tournament	Population size	1000
Tournament size	7	Set of functions	{+, -, ×, ÷}
Crossover weight	28%	Generations	10000

TABLE II
THE INFORMATION OF CLASSIFICATION TOOLS

Models	Tools	References
Naïve Bayes	Weka 3.4.3	[19]
NBTree	Weka 3.4.3	[19]
C4.5	c4.5 release 8	[17]
SVM	LIBSVM 2.71	[20]

TABLE III
THE TRAINING TIME OF CLASSIFICATION FUNCTIONS (IN SECOND)

Datasets	Classification functions	GP-I				GP-D			
		Ave.	min	max	S.D	*Ave.*	min	max	S.D.
Iris	f_{Setosa}	0.02	0.01	0.03	0.004	0.02	0.02	0.03	0.004
	$f_{Versicolor}$	1.13	1.04	1.23	0.06	1.19	1.12	1.28	0.05
	$f_{Virginica}$	1.03	0.96	1.09	0.04	1.12	1.07	1.17	0.03
wine	f_{wine1}	1.29	1.19	1.33	0.06	1.56	1.47	1.70	0.06
	f_{wine2}	0.99	0.83	1.07	0.09	1.36	1.16	1.49	0.10
	f_{wine3}	0.83	0.74	0.93	0.07	1.10	0.92	1.32	0.11
bupa	f_1	1.91	1.69	2.21	0.16	2.62	2.36	2.73	0.10
	f_2	2.15	1.83	2.44	0.18	2.45	2.30	2.59	0.07
pima	$f_{position}$	3.58	2.76	4.18	0.44	5.57	5.38	6.04	0.22
	$f_{negative}$	3.54	3.02	4.37	0.38	4.21	3.43	4.74	0.43

classifier performs better than other proposed GP-based classifiers.

B. *Experiment 2*

In this experiment, the training samples combine the incomplete cases with the complete cases and the testing samples also include the incomplete cases. The objective of the second experiment is that we hope to evaluate the effectiveness of the proposed method for handling the missing attribute values for testing samples. Except for C4.5 [17], most of the famous classifier can not work in this experiment because the class label is not known in the test data. Hence, we just compare the proposed GP-based classifier with C4.5 classifier here. From TABLE V to TABLE VIII, the tables represent the results of classification error rate using the selected incomplete datasets.

From the experimental results, we observed that the proposed method for GP-based classifiers to classify unknown class data with missing attribute values is better than C4.5 method.

V. Conclusion

Classification is one of the important research topics in these research areas. However, learning a classifier from incomplete data is more difficult than learning one from complete data. This paper introduces a new strategy to solve this problem of incomplete data and a supervised learning scheme based on genetic programming to generate classifiers. The set of dicriminant functions can handle and classify the unknown cases with missing attribute values. The results of experiment 1 demonstrate that the proposed scheme is only better than the attribute mean of same concept method when the missing rate is under 10%. Further, the results of experiment 2 show that the proposed method can classify unknown class data with missing attribute values and have better accuracy rate than C4.5. In the future, we are interested in handling both of incomplete qualitative and nominal data.

TABLE IV

The Results of Classification Using Incomplete Datasets

Missing rate	*Inference by FE*															
	GPDZ		GPDZ-min		GPIZ		GPIZ-min		C4.5		SVM		NB		NBTree	
Iris	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
0%	3.9	1.7	3.5	1.3	3.7	1.2	4.0	1.3	4.7	0.0	2.7	0.0	4.0	0.0	7.3	0.0
5%	4.9	1.8	5.1	1.9	5.2	1.8	5.8	1.9	5.7	0.0	4.2	0.7	5.9	0.5	5.0	1.0
10%	5.5	2.2	5.6	2.1	4.8	1.9	5.0	2.0	6.2	0.6	5.7	0.8	7.2	1.1	6.0	1.7
20%	5.5	4.3	5.5	4.4	4.9	4.3	5.5	4.1	7.7	5.6	6.8	3.2	7.9	2.5	7.5	5.5
30%	9.1	7.4	10.1	7.5	9.0	8.2	8.9	7.8	12.1	8.4	11.5	8.1	9.0	8.1	14.6	5.1
	Attribute mean of same concept															
5%	5.6	1.8	5.3	1.9	5.4	2.0	5.8	2.3	5.0	1.1	5.2	0.4	5.5	0.7	5.2	1.1
10%	6.2	2.2	5.4	1.9	4.9	1.7	5.4	2.2	5.5	1.0	4.5	1.0	5.2	0.6	6.1	0.3
20%	5.6	3.0	4.7	2.5	5.3	2.5	4.9	1.9	4.7	2.4	3.9	0.9	4.0	1.2	7.0	2.9
30%	5.9	4.4	5.9	4.0	7.2	5.6	8.3	6.5	6.3	5.5	4.5	2.8	5.7	4.0	9.3	8.3
wine	*Inference by FE*															
0%	7.1	2.6	7.3	2.5	6.7	3.2	8.1	3.1	9.0	0.0	2.3	0.0	3.4	0.0	3.9	0.0
5%	6.3	2.8	6.2	2.9	6.7	2.7	7.5	3.0	6.5	1.7	1.8	0.9	2.5	0.9	3.3	1.0
10%	9.3	6.6	11.5	6.8	9.0	4.1	9.9	5.4	4.9	1.8	2.7	1.7	4.9	1.8	6.6	0.4
	Attribute mean of same concept															
5%	8.7	3.9	8.6	3.3	6.0	2.8	6.8	2.7	7.5	1.1	1.8	0.9	2.5	0.8	4.0	1.4
10%	10.0	4.6	9.7	4.6	9.1	4.6	9.5	4.1	9.9	3.0	1.8	0.9	4.0	1.0	4.4	2.0
bupa	*Inference by FE*															
0%	38.6	2.4	45.3	2.9	32.0	1.2	40.8	2.8	33.9	0.0	41.4	0.0	44.6	0.0	33.6	0.0
5%	39.1	2.7	46.3	3.2	35.7	2.5	41.6	4.8	38.2	0.4	43.4	1.2	45.4	3.2	37.0	1.6
10%	40.7	4.4	44.7	4.3	36.9	3.7	42.9	4.3	43.3	4.2	46.4	3.8	41.8	0.7	41.9	1.4
20%	43.6	4.2	49.9	4.3	40.6	6.3	45.1	4.3	39.4	1.7	44.6	3.9	47.4	3.4	35.5	2.4
30%	41.1	4.0	45.6	7.1	44.1	8.7	48.9	5.1	44.8	1.2	41.2	2.6	34.4	2.0	33.7	3.0
	Attribute mean of same concept															
5%	38.7	2.7	46.0	2.2	33.8	1.6	41.2	2.8	36.7	0.7	43.0	1.3	43.3	2.3	39.1	4.0
10%	39.2	2.8	44.9	3.2	36.5	3.4	40.3	2.4	40.7	1.6	46.2	3.0	40.5	1.3	41.3	1.9
20%	39.2	5.2	45.3	4.6	37.0	3.9	41.0	4.3	43.3	2.3	42.0	1.0	39.1	1.9	41.0	4.0
30%	40.4	6.0	40.6	6.1	41.0	5.5	44.8	7.7	43.7	6.2	38.1	4.9	37.3	4.5	37.0	6.5
pima	*Inference by FE*															
0%	32.3	2.3	36.3	2.9	26.4	0.9	34.6	3.9	25.5	0.0	23.4	0.0	23.7	0.0	25.5	0.0
5%	34.9	6.2	37.4	4.8	28.8	4.3	33.7	3.5	25.7	2.0	23.7	1.2	25.4	1.3	25.0	1.2
10%	33.4	5.3	36.9	3.7	28.9	3.2	33.5	4.0	26.3	1.9	23.7	2.0	25.7	2.7	26.6	3.4
20%	35.5	5.0	39.3	6.3	29.1	6.1	35.6	6.6	29.1	2.9	22.0	4.2	24.8	4.9	25.5	4.3
30%	31.4	6.3	33.2	7.2	28.8	6.6	34.0	6.8	28.6	7.1	18.1	6.7	22.1	5.9	22.6	5.4
	Attribute mean of same concept															
5%	35.8	5.1	38.3	4.4	28.4	4.8	34.8	4.1	25.9	1.3	23.6	1.2	25.4	1.4	26.8	2.4
10%	34.7	4.1	37.7	3.5	28.2	3.9	32.2	3.6	27.2	3.3	23.7	2.2	25.8	2.6	26.4	2.4
20%	35.6	5.3	38.5	6.1	27.8	5.7	32.1	6.9	26.5	5.2	22.7	4.7	24.9	5.0	25.1	4.5
30%	30.3	7.5	32.3	8.1	25.0	8.8	30.8	8.5	24.1	6.2	17.6	7.6	21.5	4.9	24.4	4.3

TABLE V

THE RESULTS OF CLASSIFICATION USING INCOMPLETE *IRIS* DATASETS

Iris	*Inference by FE*									
	GPDZ		GPDZ-min		GPIZ		GPIZ-min		C4.5	
	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
0%	3.9	1.2	3.4	0.9	3.7	0.4	4.0	0.5	4.7	0.0
5%	5.7	1.2	5.2	1.4	5.2	1.1	5.6	1.0	10.4	0.8
10%	6.5	1.2	6.2	0.9	6.3	1.8	6.8	1.6	13.2	0.9
20%	11.0	2.2	11.1	2.0	9.9	3.5	10.0	3.2	18.8	0.6
30%	17.9	2.6	18.5	3.2	16.8	2.4	17.1	2.6	27.8	0.4
40%	23.4	4.1	23.4	4.0	21.0	2.6	21.5	1.8	40.0	0.0
50%	27.1	2.3	27.0	2.4	27.5	3.5	28.5	3.5	46.6	1.6
Attribute mean of same concept										
5%	5.9	1.0	6.3	1.3	6.8	1.3	7.0	1.5	10.4	1.3
10%	8.8	1.7	9.1	1.4	7.1	1.1	7.5	1.5	11.6	1.2
20%	14.4	2.6	14.2	2.7	12.8	4.1	13.3	3.8	19.2	0.7
30%	18.2	2.8	18.6	3.1	17.8	1.6	18.2	1.2	22.8	1.4
40%	25.4	4.9	26.1	4.6	25.9	2.3	26.1	2.1	34.8	0.6
50%	32.6	3.3	32.4	3.4	28.9	3.4	29.0	3.5	36.1	0.6

TABLE VI

THE RESULTS OF CLASSIFICATION USING INCOMPLETE *WINE* DATASETS

wine	*Inference by FE*									
	GPDZ		GPDZ-min		GPIZ		GPIZ-min		C4.5	
	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
0%	7.1	1.4	7.3	2.5	6.7	2.5	8.1	2.4	9.0	0.0
5%	7.8	1.7	8.2	1.7	9.1	2.3	10.6	3.0	13.1	2.4
10%	11.1	3.2	12.5	3.8	11.3	1.7	13.1	2.4	21.8	2.8
20%	19.8	3.7	21.2	4.5	17.1	3.3	19.3	3.2	33.3	1.5
30%	27.2	4.0	28.8	3.7	24.6	3.6	28.9	5.0	42.0	4.3
40%	38.6	3.8	40.0	4.2	34.9	5.3	40.5	6.0	45.3	3.3
50%	45.1	4.2	47.0	4.2	45.7	4.7	48.4	4.9	50.9	2.1
Attribute mean of same concept										
5%	10.5	1.7	10.9	1.7	9.5	1.7	11.0	1.8	15.2	1.8
10%	10.9	1.4	11.1	1.6	13.0	2.9	14.0	3.2	21.0	1.5
20%	17.4	3.3	17.9	3.2	16.8	1.8	17.3	2.5	31.5	2.9
30%	25.1	3.3	25.3	3.0	24.4	5.1	24.1	4.8	40.5	3.7
40%	32.1	2.2	31.8	2.6	30.6	4.5	30.9	4.7	44.3	5.0
50%	36.7	5.1	36.8	5.2	37.7	5.2	37.9	5.1	46.8	4.3

TABLE VII

THE RESULTS OF CLASSIFICATION USING INCOMPLETE *BUPA* DATASETS

bupa	*Inference by FE*									
	GPDZ		GPDZ-min		GPIZ		GPIZ-min		C4.5	
	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
0%	38.6	2.4	45.2	2.9	32.0	1.2	40.7	2.8	33.9	0.0
5%	41.0	2.5	46.8	1.9	34.9	1.7	41.3	3.2	37.3	2.1
10%	40.4	2.0	46.0	2.6	35.7	2.3	42.4	2.5	40.0	1.7
20%	45.3	2.0	48.8	2.3	42.8	2.6	48.2	4.0	39.6	4.4
30%	47.2	1.6	49.7	1.6	45.0	1.2	47.7	1.6	43.0	0.3
40%	48.6	3.5	49.9	2.8	45.0	3.0	46.8	3.4	46.1	1.6
50%	48.2	3.1	49.3	3.1	48.2	2.0	48.6	2.1	45.6	3.0
Attribute mean of same concept										
5%	40.9	2.5	46.0	1.4	35.4	2.7	43.1	3.4	37.2	3.5
10%	39.0	2.1	45.9	3.4	35.1	1.3	41.5	1.5	41.7	0.4
20%	46.9	2.1	49.7	1.8	42.7	1.6	47.1	2.5	42.5	1.3
30%	48.3	2.5	50.3	3.2	46.4	2.0	47.5	1.4	45.2	3.1
40%	48.1	3.0	48.7	3.2	46.7	2.5	48.0	3.0	46.6	1.4
50%	49.5	3.4	50.7	3.1	47.9	2.8	48.5	2.1	40.8	1.3

TABLE VIII

THE RESULTS OF CLASSIFICATION USING INCOMPLETE *PIMA* DATASETS

pima	*Inference by FE*									
	GPDZ		GPDZ-min		GPIZ		GPIZ-min		C4.5	
	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
0%	32.3	2.3	36.2	2.9	26.4	0.9	34.6	3.9	25.5	0.0
5%	33.2	2.4	37.0	1.7	28.0	1.1	35.5	2.5	27.1	1.1
10%	34.1	2.4	39.9	3.1	29.8	1.8	35.7	2.7	27.2	0.9
20%	36.5	1.8	39.9	2.5	34.7	1.6	39.2	1.7	32.1	2.5
30%	42.7	2.5	46.7	3.1	38.7	2.2	40.8	2.4	34.6	2.0
40%	42.1	1.9	45.2	2.0	41.2	1.7	43.9	1.6	37.5	3.3
50%	44.5	0.8	46.7	0.7	43.9	1.9	45.6	1.8	40.6	2.5
Attribute mean of same concept										
5%	31.6	2.3	36.6	2.4	28.3	2.3	36.0	3.1	26.7	0.6
10%	32.4	2.4	36.4	2.7	30.4	1.5	36.6	2.3	28.9	1.2
20%	35.9	2.5	40.5	2.7	32.6	1.7	36.5	2.2	30.2	0.9
30%	37.6	1.9	42.2	3.3	37.9	1.5	39.7	2.3	34.0	0.8
40%	39.7	2.4	41.9	2.6	38.5	2.0	40.9	1.9	39.1	4.8
50%	43.9	1.7	45.4	2.3	43.5	2.9	44.9	2.9	40.1	1.3

REFERENCES

[1] J. W. Grzymala-Busse and M. Hu, "A comparison of several approaches to missing attribute values in data mining," in *Proc. of the Second International Conference on Rough Sets and Current Trends in Computing*, RSCTC'2000, pp. 378-385.

[2] C. Blake, E. Keogh and C. J. Merz, UCI repository of machine learning database, Irvine, University of California, Department of Information and Computer Science (1998). Available: http://www.ics.uci.edu/~mlearn/MLRepository.html,

[3] J. H. Friedman, "A recursive partitioning decision rule for non-parametric classification," *IEEE Trans. on Computer Science*, 1977, pp. 404-408.

[4] J. Han and M. Kamber, *Data Mining: Concept and Techniques*, Morgan Kaufmann publishers, 2001.

[5] I. Koninenko, I. Bratko and E. Roskar, "Experiments in Automatic Learning of Medical Diagnostic Rules," *Technical Report*, Jozef Stenfan Institute, Ljubljana, 1984.

[6] R. Slowinski and J. Stefanowski, "Handling various types of uncertainty in the rough set approach," in *Proc. of the International Workshop on Rough Sets and Knowledge Discovery*, 1993, pp. 366-376.

[7] J. W. Grzymala-Busse, "Rough set strategies to data with missing attribute values," in *Proc. of the Workshop on Foundations and New Directions in Data Mining, the third IEEE International Conference on Data Mining*, November 19-22, 2003, pp. 56-63.

[8] T.-P. Hong, L.-H. Tseng and B.-C. Chien, "Learning fuzzy rules from incomplete quantitative data by rough sets," in *Proc. of the 2002 IEEE International Conference on Fuzzy Systems*, pp. 1438-1443.

[9] J. K. Kishore, L. M. Patnaik, V. K. Agrawal, "Application of Genetic Programming for Multicategory Pattern Classification," *IEEE Trans. on Evolutionary Computation*, Vol. 4, No. 3, 2000, pp. 242-258.

[10] M. Bramrier and W. Banzhaf, "A comparison of linear genetic programming and neural networks in medical data mining," *IEEE Trans. on Evolutionary Computation*, Vol. 5, No. 1, Feb. , 2001, pp. 17-26.

[11] B.-C. Chien, J.-Y. Lin, and W.-P. Yang, "Learning effective classifiers with z-value measure based on genetic programming," *Pattern Recognition*, Vol. 37, No. 10, 2004, pp. 1957-1972.

[12] B.-C. Chien, J.-H. Yang, W.-Y. Lin, "Generating Effective Classifiers with Supervised Learning of Genetic Programming," in *Proc. of the 5th International Conference on Data Warehousing and Knowledge Discovery*, 2003, pp. 192-201.

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

[14] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs. Cambridge*, MA: MIT Press, 1994.

[15] H.-M. Lee, C.-M. Chen, J.-M. Chen, and Y.-L. Jou, "An efficient fuzzy classifier with feature selection based on fuzzy entropy," *IEEE Trans. on Systems, Man, and Cybernetics-part B: Cybernetics*, Vol. 31, No. 3, 2001, pp. 426-432.

[16] A. Singleton, "Genetic Programming with C++," *Byte*, Feb., 1994, pp. 171-176. Available: http://www.byte.com /art/9402/sec10/ar-t1.htm.

[17] J. R. Quinlan, *C4.5: Programs for Machine Learning*, San Mateo, California, Morgan Kaufmann Publishers (1993).

[18] S. R. Gunn, "Support vector machines for classification and regression," Technical Report, School of Electronics and Computer Science University of Southampton, (Southampton, U.K.), 1998.

[19] Ian H. Witten and Eibe Frank., *Data Mining: Practical machine learning tools with Java implementations*, Morgan Kaufmann, San Francisco, 2000.

[20] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, 2001. Available: http://www.csie.ntu. edu.tw/ ~cjlin/libsvm.