Bagging and Feature Selection 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. Missing values are an unavoidable issue of many real-world datasets. Dealing with missing values is an essential requirement in classification problem, because inadequate treatment with missing values often leads to large classification errors. Some classifiers can directly work with incomplete data, but they often result in big classification errors and generate complex models. Feature selection and bagging have been successfully used to improve classification, but they are mainly applied to complete data. This paper proposes a combination of bagging and feature selection to improve classification with incomplete data. To achieve this purpose, a wrapper-based feature selection which can directly work with incomplete data is used to select suitable feature subsets for bagging. The experiments on eight incomplete datasets were designed to compare the proposed method with three other popular methods that are able to deal with incomplete data using C4.5/REPTree as classifiers and using Particle Swam Optimisation as a search technique in feature selection. Results show that the combination of bagging and feature selection can not only achieve better classification accuracy than the other methods but also generate less complex models compared to the bagging method.

Keywords: incomplete data, ensemble, feature selection, classification, particle swam optimisation, C4.5, REPTree

1 Introduction

Classification is one of the main tasks in machine learning and data mining, and has been successfully applied to many areas such as computer science, engineering and biology. Moreover, classification has been continuously received a great attention. However, there are still open issues in classification, and one of the issues is classification with incomplete data [6,9].

Incomplete data is data which contains some fields without values. Missing values are a common problem in many datasets. For example, 45% of the datasets in UCI machine learning repository, which is one of the most popular collection of benchmark datasets for machine learning, contain missing values [6]. Reasons for datastes containing missing values are various. For example, social survey sheets often contain missing values because respondents refuse to answer some questions; medical patient records also usually have missing values because all tests often cannot be done on patients [13].

Missing values cause serious problems for classification. One of the most serious problems is the non-applicability of majority classifiers with incomplete data. Majority classifiers require complete data; therefore, they cannot directly work with incomplete data. Moreover, missing values often lead to big classification errors [6,20].

One of the most popular approaches to solving classification with incomplete data is to use a classifier which can directly classify incomplete data. For example, C4.5 can directly deal with incomplete data in both training and testing process. Although this approach can tackle incomplete data to some extent, it often results in more complex learnt models and bigger classification errors [19]. Therefore, further approaches to improving classifiers able to directly classify incomplete data should be investigated.

Feature selection is the process to select a suitable feature subset from the original features. Feature selection has been proven capable of improving classification accuracy and reducing the complexity of learnt models [14]. Although feature selection is mainly applied to complete data, it is also used to improve classification with incomplete data [18,22].

Ensemble is a machine learning method that builds a set of classifiers instead of a single classifier for classification tasks. Ensemble methods have been demonstrated to enhance classification accuracy [4]. One of the most popular ensemble methods is bagging. Although bagging helps improve classification accuracy, it often generates more complex learnt models [17]. Moreover, bagging is mainly applied to complete data. Therefore, researches on improving bagging for classification with incomplete data should be investigated.

1.1 Research Goals

The goal of this paper is to propose a new method which improves bagging for classification with incomplete data. In order to achieve the goal, a combination of bagging and feature selection is proposed to classify incomplete data. The proposed method is compared with three benchmark methods for classification with incomplete data. The first benchmark method is to use a classifier which can directly classify incomplete data. The second benchmark method is to combine feature selection and a classifier which can directly classify incomplete data. The third benchmark method is to combine bagging and a classifier which can directly classify incomplete data. The experimental results are used to address the following objectives:

1. Whether the combination of bagging and feature selection can achieve better classification accuracy compared to the other methods for classification with incomplete data.
2. Whether the combination of bagging and feature selection can generate less complex learnt models compared to bagging with all features for classification with incomplete data.

1.2 Organisation

The rest of this paper is organised as follows. Section 2 outlines related work. Section 3 presents the proposed method. After that, section 4 presents comparison method and experiment design. Section 5 shows results and analysis. Finally, section 6 presents conclusions and future work.

2 Related Work

This section outlines related work including classification with incomplete data, feature selection and ensemble learning.

2.1 Classification with Incomplete Data

There are two main approaches to classification with incomplete data. One approach is to use imputation methods to transform incomplete data to complete data before using classification algorithms. The other approach is to use classification algorithms which can directly work with incomplete data without using imputation methods [6].

The purpose of imputation methods is to replace missing values with plausible values. For example, mean imputation fills all missing values in each feature with the average of all complete values in the same feature. The main benefit of using imputation methods is that they can provide complete data which can be used by any classification algorithm. However, simple imputation methods such as mean imputation often lead to the big classification error. Moreover, more sophisticated imputation methods are often computationally intensive to estimate missing values before using classification algorithms [20].

The majority of classification algorithms cannot directly work with incomplete data. However, there are some classification algorithms which are able to directly classify incomplete data. For example, C4.5 [19] use a probabilistic approach to tackle missing values in both the training set and test set. The main benefit of using classification algorithms able to directly classify incomplete data is that the classification algorithms do not require any time for estimating missing values. However, when the classification algorithms work with incomplete data, they often generate more complexed models and lead to large classification errors [20]. Therefore, further approaches to improving the classification algorithms should be investigated.

2.2 Feature Selection

Feature selection is the process of selecting a relevant subset of features from the original features. The underlying reason for using feature selection is that the data often contains redundant/irrelevant features, which should be removed without much loss of information. By removing redundant/irrelevant features, feature selection can help improve classification accuracy. Moreover, thanks to providing a smaller number of features, feature selection can help to speed up the training process and make simpler learnt classifiers which are easier to interpret [1,14,24].

Feature selection includes two main procedures: a search procedure and an evaluation procedure. The search procedure is used to search feature subsets while the evaluation procedure is used to measure the quality of feature subsets. The performance of feature selection strongly depends on the quality of both of the procedures [1,14,24].

Search methods for feature selection can be categorised into traditional search methods and evolutionary search methods. For example, sequential forward selection and sequential backward selection are two common traditional search methods for feature selection. In recent times, evolutionary algorithms have been successfully used as search methods in feature selection. Genetic algorithms and particle swarm optimisation (PSO) are two popular evolutionary search methods for feature selection [24].

Evaluation methods for feature selection can be categorised into wrapper methods and filter methods. A wrapper method uses a classifier to evaluate feature subsets while a filter method uses a measure such as information gain to evaluate. In order to evaluate each feature subset, wrapper methods need to train a classifier and then test its performance; therefore, they are often computationally expensive. In contrast, evaluation measures in filter methods are often computationally cheap; therefore, filter methods are often more efficient and general than wrapper methods. However, wrapper methods are often more accurate than filter methods [1,14,24].

Particle swarm optimisation (PSO) is a swarm intelligent algorithm proposed by Kennedy and Eberhart in 1995 [10]. Recently, PSO has been widely used as a search method for feature selection. Continuous PSO is usually used for feature selection, where the dimensionality of each particle is equal to the total number of features and a threshold θ is often used to determine whether or not a feature is selected. If the value is smaller than θ, the corresponding feature is not selected, otherwise, it is selected. PSO has been used for both wrapper and filter. In a PSO-based wrapper, a classifier is required to evaluate particles, and the fitness of each particle is the accuracy of the classifier by using selected features. In a PSO-based filter, an evaluation measure is required to evaluate particles, and the fitness of each particle is estimated by the measure with selected features [24].

Feature selection methods have been mainly applied to complete data. However, in recent times, there are some feature selection approaches to incomplete data. In [5], the mutual information criterion which is based on k-nearest neighbours is expanded to tackle with missing values. The experimental results show that the method can select important feature subsets without using any imputation method and help enhance the performance of the prediction models. In [18], a combination of the mutual information measure and rough sets is proposed to evaluate feature subsets with incomplete data. The empirical results show that the proposed method is effective for selecting feature subsets with incomplete data. A wrapper-based feature selection for incomplete data is proposed in [22], where PSO is used as a search technique and C4.5 which is able to directly classify incomplete data is used to evaluate feature subsets. The experimental results show that the proposed methods not only can improve the accuracy of the classifier, but it also can reduce the complexity of the classifier.

2.3 Ensemble Learning

Ensemble learning is a machine learning method which constructs a set of classifiers for a classification task. It classifies a new instance by voting the decision of individual classifiers. The set of classifiers has been proved capable of achieving higher accuracy than any of the individual classifiers [4].

An ensemble of classifiers is accurate if the individual classifiers in the ensemble is accurate and diverse. Two popular methods to construct accurate ensembles are Bagging and Boosting. Both of the methods use "resampling"techniques to construct different training sets for each of the classifier. Bagging manipulates the original training data by randomly drawing with replacement instances. Consequently, some of the original instances might appear multiple times in the resulting training data while others might disappear. Bagging is usually helpful with "unstable"classification algorithms like neural networks and decision trees where small changes in the training data often result in major changes in predictions. Experimental results reveal that Bagging ensemble almost always achieves better accuracy than a single classifier. Boosting also manipulates the original training data by drawing with replacement, but it uses the performance of the previous classifier(s) to calculate the probability of selecting each instance. Boosting tries to construct new classifiers that are better to classify instances for which the current ensemble's performance is poor. Therefore, in Boosting, instances which are incorrectly classified by previous classifiers are more often selected than instances which are correctly classified. Experimental results reveal that with little or no classification noise, Boosting ensemble also almost always achieve better accuracy than a single classifier, and it is sometimes more accurate than Bagging ensemble. However, with substantial classification noise, Boosting ensemble is often less accurate than a single classifier since Boosting ensemble often overfits noisy datasets [16].

Feature selection also has been used to improve ensemble learning. In [17], a genetic algorithm (GA) is used to search a suitable set of feature subsets for ensemble. Initially, a set of classifiers is generated where each classifier is built by randomly picking a set of features. After that, new classifiers are created by using the genetic operators. Finally, the best fit individuals are chosen to build an appropriate set of feature subsets which is then used to create an ensemble. Neural network is used as a classifier and the fitness of each individual is the combination of accuracy and diversity. Experiment results show that the feature selection ensemble can achieve better accuracy than the popular and powerful ensembles of Bagging and Boosting. In [7], GA is also applied to search a set of feature subsets for ensemble, where the GA runs multiple times with different training data to provide different feature subsets. C4.5 and Euclidean Decision

Tables are used as classifiers and the fitness of each individual is the classification accuracy. Experiments show that the feature selection ensemble is more accurate than other existing ensemble methods. In [15], a multi-objective GA is used to select a set of feature subsets which is used to construct a set of classifiers. After that the multi-objective GA is used again to select an optimal set of classifiers. The experimental results show that the proposed method is more effective than Boosting and Bagging for the handwriting recognition problem.

Ensemble learning also has been used to solve classification with incomplete data. In [11], a set of classifiers is constructed to classify incomplete data, where each base classifier is trained with a random subset of features. In [2], incomplete data is firstly grouped into complete subsets, and then each subset is used to train one classifier. Although the two methods can tackle incomplete data in some extent, they cannot ensure to classify all incomplete instances.

Popular ensemble methods such as bagging/boosting have been mainly applied to complete data. Therefore, the application of popular ensemble methods for incomplete data should be investigated. Moreover, combining popular ensemble methods with feature selection has not been investigated. Feature selection can improve the performance of classification with incomplete data. Therefore, a combination of popular ensemble methods with feature selection for incomplete data also need be investigated.

3 The Proposed Method

The key idea of the proposed method is that bagging is combined with feature selection to improve the accuracy and diversity of a set of learnt classifiers. The underlying reason is that to construct a set of classifiers, bagging repeatedly resamples the training dataset to build a set of training resampled datasets. The resampled datasets often contain redundant/irrelavant features. Moreover, feature selection has been proven capable of remove redundant/irrelavant features. Therefore, feature selection could be applied to each resampled dataset to eliminate redundant/irrelavant features. By eliminating redundant/irrelavant features, feature selection can help improve resampled datasets which in turn can help build more accurate and less complex learnt classifiers.

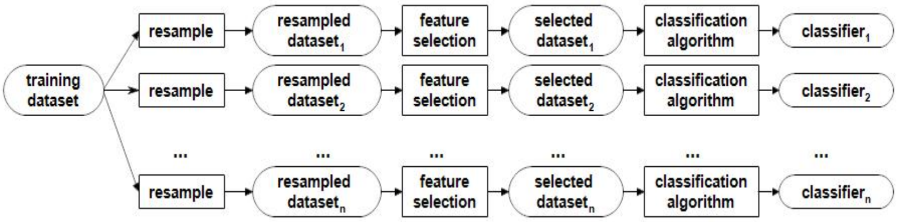

Fig. 1. The training process of classification with incomplete data by combining bagging and feature selection.

Fig. 1 shows main steps of the training process of the proposed method. In the training process, firstly, the training dataset is put into a resampling procedure several times to generate a set of training resampled datasets. After that, each training resampled dataset is put into a feature selection procedure to select a suitable feature subset which is then used to transform the training resampled dataset into the training selected dataset. Subsequently, each the training selected dataset is used by a classification algorithm to learn a classifier. As a result, the training process generates a set of classifiers. In the application process, the set of classifiers is combined to classify a new instance.

The main steps of the proposed method are presented in the following subsections.

3.1 Resampling Data

The purpose of each time resampling training dataset is to create a random redistribution of the training dataset. Each training resampled dataset is generated by randomly choosing with replacement the same number of instances in the original training dataset. As a result, many of the original instances might be repeated in the training resampled dataset while others might be left out.

3.2 Feature Selection

The key difference between the proposed method and bagging is this step. Bagging immediately uses a set of training resampled datasets to build a set of classifiers. In contrast, the proposed method applies feature selection to eliminate redundant/irrelevant features in each training resampled dataset before building a set of classifiers.

In order to remove redundant/irrelevant features in incomplete data, any search technique can be used to find feature subsets. However, to evaluate a feature subset which may contain incomplete features, the feature selection procedure requires a feature subset evaluation method which can deal with incomplete data. In [22,23], a combination of PSO and a classifier able to classify incomplete data has been successfully used to remove redundant/irrelevant features in incomplete data. Therefore, in the proposed method, PSO will be used to search feature subsets and a classifier which is able to classify incomplete data such as C4.5 will be used to evaluate feature subsets.

3.3 Combining classifiers

A set of classifiers which is built in the training process is combined to classify new instances in the application process. The majority vote chooses a class label with the most votes from the ensemble members as the ensemble output. The majority is a simple and powerful voting method [16]. Therefore, in the proposed method, the majority vote will be used to combine classifiers.

4 Method and Experiment Design

This section shows the comparison method and experiment design including datasets used in the experiment, parameter settings for feature selection and classification algorithm.

4.1 The Comparison Method

Experiments are conducted to evaluate the effectiveness of the combination of bagging and feature selection for classification with incomplete data. In order to achieve the goal, a combination of bagging and feature selection for classification with incomplete data as shown in Fig.1 is compared with three other common methods for classification with incomplete data as shown in Fig.2, Fig.3 and Fig.4. Fig.2 shows the training process of classification with incomplete data by using a classifier able to directly classify with incomplete data. Fig.3 shows the training process of classification with incomplete data by combining feature selection and a classifier able to directly classify with incomplete data. Fig.4 shows the training process of classification with incomplete data by combining bagging and a classifier able to directly classify with incomplete data.

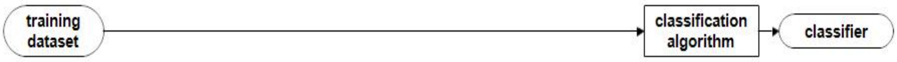

Fig. 2. The training process of classification with incomplete data by directly using a classifier able to classify incomplete data.

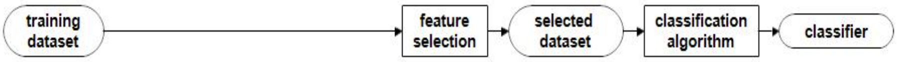

Fig. 3. The training process of classification with incomplete data by using feature selection.

In the four setups, firstly, incomplete dataset is divided into training dataset and testing dataset. In the proposed setup shown in Fig.1, the training dataset is used by bagging and feature selection to build a set of classifiers which is used to classify the testing dataset. In the setup shown in Fig.2, the training dataset is directly put into a classification algorithm to learn a classifier which is then used to classify the testing dataset. In the setup shown in Fig.3, the training dataset is put into a feature selection procedure to select a suitable feature subset which is used to transform the training dataset into the training selected dataset. After that, the training selected dataset is used by a classification algorithm to learn a

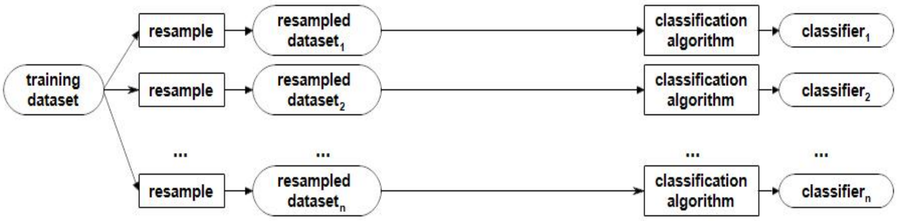

Fig. 4. The training process of classification with incomplete data by using bagging.

classifier which is used to classify the testing dataset. In the setup shown Fig.4, the training dataset is put into a resampling procedure to generate a set of training resampled dataset. After that the set of training resampled dataset is used by a classification algorithm to build a set of classifiers which is then used to classify the testing dataset.

4.2 Datasets

The experiments used eight benchmark incomplete datasets chosen from UCI Repository of Machine Learning Databases [12]. Table 1 presents main characteristics of these datasets which include the name, the number of features (R:real/I: integer/N:nominal values), the number of classes, the number of instances and the percentage of incomplete instances which contain at least one missing field.

Table 1. Datasets used in the experiments.

Name	**#Features(R/I/N)**	**#Classes**	**#Instances**	**Incomplete instances (%)**
Breast	9 (0/0/9)	2	286	3.15
Cleveland	13 (13/0/0)	5	303	1.98
Crx	15 (3/3/9)	2	690	5.36
Dermatology	34 (0/34/0)	6	366	2.19
Hepatitis	19 (2/17/0)	2	155	48.39
Mammographic	5 (0/5/0)	2	961	13.63
Marketing	13 (0/13/0)	9	8993	23.54
Wisconsin	9 (0/9/0)	2	699	2.29

These datsets were carefully chosen to represent classification tasks with incomplete data of varying difficulty, dimensionality, feature types and number of classes. These datasets have varying levels of incomplete instances (Cleveland has 1.98% incomplete instances while Hepatitis has 48.39% incomplete instances). These problems also range low and high dimensionality (Mammographic has five features while Dermatology has 34 features), different feature types and binary classification and multi-class classification.

None of the datasets has a specific test set. Furthermore, the number of instances in some datasets is relatively small. Consequently, the ten-fold cross-validation method was applied to measure the performance of the learnt classifiers. In the experiments, with each dataset, the ten-fold cross-validation method was done 30 times. Therefore, with each dataset, 300 pairs of training set and test set were generated to evaluate the performance of the algorithms.

4.3 Classification Algorithms

In the experiments, we used C4.5 [19] and REPTree [21] to classify data and evaluate the quality of feature subsets in the feature selection procedure. Both of the algorithms are able to directly classify incomplete data. The WEKA [8] was used to implement the algorithms by setting its parameters as the default values. Following [16], in the proposed method and the original bagging method, the number of classifiers was set 10.

4.4 PSO Parameter Settings

The experiments used continuous PSO to search feature subsets in the feature selection procedure. The PSO parameters were set as common parameter settings proposed by Clerc and Kennedy [3]. The detailed parameter settings are shown as follows: inertia weight (ω) was set 0.729844, the cognitive parameter (c_1) and the social parameter (c_2) were set 1.49618, population size was set to 30, the maximum iteration was set to 50 and the fully connected topology. The classification accuracy was used to evaluate the quality of particles. The threshold θ was set 0.8 to determine whether or not a feature is selected.

5 Results and Analysis

This section shows the comparison of the proposed method with the other methods on classification accuracy and the complexity of learnt classifiers. This section also mentions some reasons for the experimental results.

5.1 Classification Accuracy

Table 2 shows the average of classification accuracy and standard deviation of the four methods with the two classifiers on the eight incomplete datasets. The average of classification accuracy is the average of accuracies of 30 times performing ten-fold cross-validation on each dataset. In the table, *BaFS* column presents the average of accuracy from the proposed setup shown in Fig.1; *All* column presents the average of accuracy from the setup shown in Fig.2; *FS* column presents the average of accuracy from the setup shown in Fig.3, and *BaAll* column presents the average of accuracy from the setup shown in Fig.4.

To compare the performance of the proposed method with the other methods, the Wilcoxon signed-ranks tests at 95% confidence interval is used to compare the classification accuracy achieved by BaFS with the other methods. “T”columns in Table 2 show significant test of the columns before them against BaFS, where “+”, “=”and “-”mean BaFS is significantly more accurate, not significantly different, and significantly less accurate, respectively. The bold ones mean the best results for each dataset.

Table 2. The classification accuracy of different methods.

Dataset	Algorithm	BaFS	All	T	FS	T	BaAll	T
Breast	C4.5	**96.19**±0.53	94.64±0.43	+	94.40±0.61	+	95.89±0.42	+
	REPTree	**95.98**±0.45	94.42±0.42	+	94.16±0.58	+	95.75±0.39	=
Clevelant	C4.5	**58.67**±1.34	54.45±2.00	+	57.68±1.58	+	57.06±1.56	+
	REPTree	58.68±1.10	56.63±1.49	+	57.60±1.54	+	**59.03**±1.17	=
Crx	C4.5	**85.89**±0.41	84.98±0.80	+	84.62±0.68	+	85.89±0.61	=
	REPTree	**86.03**±0.41	84.55±0.83	+	84.91±0.46	+	84.90±0.63	+
Dermatology	C4.5	96.67±0.72	95.66±0.48	+	92.35±1.20	+	**96.84**±0.56	=
	REPTree	**96.23**±0.67	94.75±0.71	+	92.09±1.50	+	95.67±0.56	+
Hepatitis	C4.5	**83.04**±1.78	78.87±1.89	+	80.58±1.73	+	80.97±1.30	+
	REPTree	**82.71**±1.69	80.21±2.23	+	80.14±1.77	+	81.69±1.96	+
Mammographic	C4.5	**82.84**±0.48	82.08±0.36	+	82.17±0.48	+	82.67±0.45	=
	REPTree	82.60±0.50	82.00±0.68	+	81.76±0.59	+	**82.65**±0.50	=
Marketing	C4.5	**33.14**±0.35	30.86±0.41	+	32.04±0.49	+	31.54±0.28	+
	REPTree	**33.66**±0.31	32.75±0.42	+	32.16±0.59	+	33.04±0.35	+
Wisconsin	C4.5	**96.19**±0.32	94.61±0.49	+	94.42±0.51	+	95.76±0.44	+
	REPTree	**96.03**±0.42	94.45±0.48	+	94.00±0.67	+	95.68±0.45	+

Bold values indicate the best results for each dataset

It is clear from Table 2 that in most cases, the proposed method can achieve the best classification accuracy. In all cases, the proposed method achieves significantly better classification accuracy than using all features as shown in Fig.2. In all cases, the proposed method also achieves significantly better classification accuracy than using selected features as shown in Fig.3. Compared to bagging as shown in Fig.4, in 16 cases, the proposed method achieves significantly better classification accuracy than bagging in 10 cases, similar classification accuracy to bagging in 6 cases and never significantly worse classification accuracy than bagging.

In summary, the combination of bagging and feature selection is able to help significantly improve accuracy of classification with incomplete data.

5.2 Classifier Size

Table 3 shows the average size of decision trees(the number of nodes in the trees) of the four methods with two classifiers on the eight incomplete datasets. The average size of decision trees is the average size of 30 times performing ten-fold cross-validation on each dataset. In the table, *BaFS* column presents the average size from the proposed setup shown in Fig.1; *All* column presents the average size from the setup shown in Fig.2; *FS* column presents the average size from the setup shown in Fig.3, and *BaAll* column presents the average size from the setup shown in Fig.4.

Table 3. The tree size of different methods.

Dataset	Algorithm	BaFS	All	FS	BaAll
Breast	C4.5	18.67	22.96	14.89	22.81
	REPTree	14.54	13.35	13.12	14.86
Clevelant	C4.5	71.33	79.19	18.13	73.36
	REPTree	30.33	17.42	11.61	32.24
Crx	C4.5	38.08	29.00	9.45	48.85
	REPTree	30.76	22.76	13.10	46.76
Dermatology	C4.5	20.88	15.20	17.28	17.43
	REPTree	15.66	15.66	13.85	15.24
Hepatitis	C4.5	11.40	17.46	7.04	15.17
	REPTree	8.55	6.42	7.18	8.91
Mammographic	C4.5	15.88	10.49	8.78	31.75
	REPTree	22.75	13.99	11.08	35.54
Marketing	C4.5	1031.61	1372.32	186.12	1713.21
	REPTree	442.72	361.64	163.12	724.25
Wisconsin	C4.5	18.93	22.95	15.26	23.09
	REPTree	14.50	13.12	12.51	14.87

Fig. 5 summaries Table 3 by showing the average of tree size ratio between the other methods and the proposed method with C4.5 and REPTree (bigger than one means bigger tree, otherwise equal or smaller tree). It can be seen from Fig.5 that the feature selection with original data shown in Fig.3 provides the smallest trees. Moreover, it is clear from Fig.5 that with both C4.5 and REPTree,

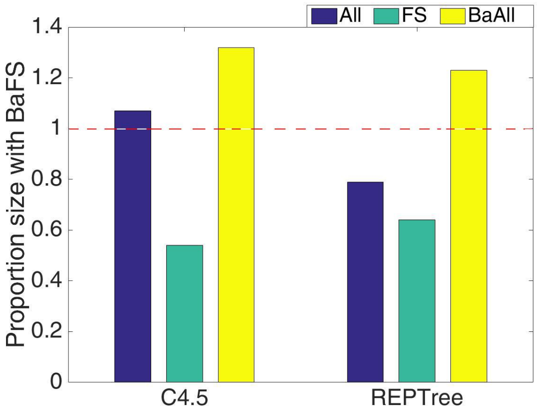

Fig. 5. Tree size ratio between the other methods and BaFS

the average of tree size generated by bagging with all features is bigger than the combination of bagging and feature selection. In other words, the combination of bagging and feature selection helps reduce the complexity of learnt classifiers from bagging.

In summary, the combination of bagging and feature selection not only help improve classification accuracy of bagging, but also helps reduce the complexity of the learn classifiers compared with the standard bagging method.

5.3 Further Analysis

To understand how the combination of bagging and feature selection can achieve better classification and smaller trees than bagging with all features, we looked carefully at the trees generated by C4.5 bagging with all features and bagging with selected features on Hepatitis dataset which has 19 features (*Age, Sex, Steroid, Antivirals, Fatigue, Malaise, Anorexia, LiverBig, LiverFirm, SpleenPalpable, Spiders, Ascites, Varices, Bilirubin, AlkPhosphate, Sgot, AlbuMin, ProTime, Histology*). The Hepatitis dataset was chosen since the trees generated on Hepatitis are not too big to analyse. Figs. 6 and 7 present two typical pattern trees we observed.

It is clear from Figs. 6 and 7 that the combination of bagging and feature selection can generate more accurate and less complex trees than bagging with all features. The reason might be that classifiers like C4.5 are greedy algorithms which make the locally optimal choice at each stage. Therefore, they may provide locally optimal solutions. The purpose of feature selection is to search for more suitable feature subsets. Therefore, feature selection can help reduce the limitation of greedy algorithms. For example, in Fig.6, with a training resampled data, when C4.5 bagging uses all features, the information gain of feature Spiders and feature Sex are higher than the information gain of feature Age, so feature

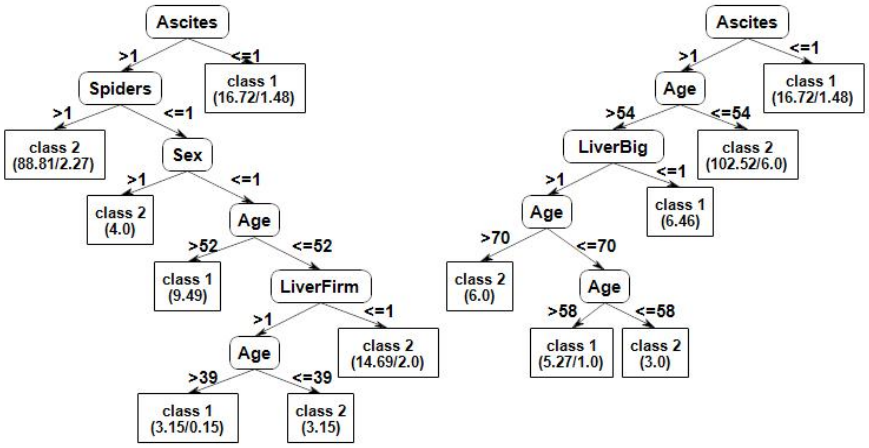

Fig. 6. Left tree with 90.0% of accuracy generated by C4.5 bagging with all features and right tree with 92.14% of accuracy generated by C4.5 bagging with selected features

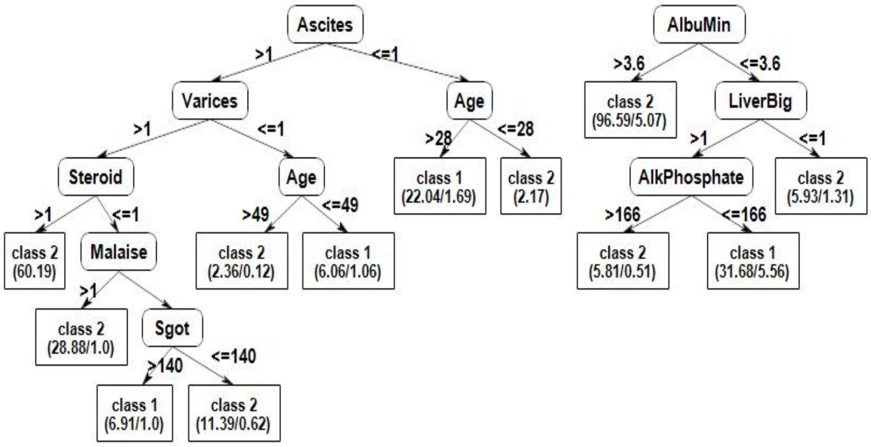

Fig. 7. Left tree with 86.42.0% of accuracy generated by C4.5 bagging with all features and right tree with 91.42% of accuracy generated by C4.5 bagging with selected features

Spiders and feature Sex are chosen to build the right tree before choosing feature Age. When feature selection is applied to the training resampled data, only three features Age, Ascite and LiverBig are selected. Consequently, feature Age is chosen to develop the right tree instead of feature Spiders or feature Sex. As a result, bagging with selected features generates more accurate and less complex trees than bagging with all features.

It also can be seen from Figs.6 and 7 that the combination of bagging and feature selection can generate more diverse trees than bagging with all features. For example, the right tree on Fig.6 uses the same feature Ascites in the *first level* as the right tree on Fig.7. However, the left tree on Fig.6 uses different feature in the *first level* from the right tree on Fig.7. By generating more diverse trees, feature selection helps improve the bagging method.

In summary, bagging with selected features can generate more accurate, less complex and more diverse learnt models than bagging with all features. Therefore, the combination of bagging and feature selection helps improve the traditional bagging method.

6 Conclusions and Future Work

This paper proposed a combination of bagging and feature selection method to improve classification with incomplete data. In order to achieve the purpose, bagging is firstly used to construct a set of training resampled data. After that, the set of training resampled data is used by a wrapper-based feature selection to build a set of training selected data which is then used to learn a set of classifiers. The proposed method was compared with three other popular classification methods which can directly work with incomplete data. The experiments on eight incomplete datasets used C4.5 and REPTree, which is able to directly classify incomplete data, as classifiers and PSO as a search technique in feature selection. The results showed that the combination of bagging and feature selection method is more accurate than the other methods. Moreover, the combination of bagging and feature selection is able to reduce the complex learnt models generated by the bagging method.

One of the other most popular ensemble methods is boosting. We already tried to use the same process as the proposed method to combine boosting and feature section for classification with incomplete data. However, experimental results were not promising. Therefore, future work could be to investigate more suitable approaches to combining boosting and feature selection, and check which ensemble methods are more suitable to what kinds of problems.

References

1. G. Chandrashekar and F. Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.
2. H. Chen, Y. Du, and K. Jiang. Classification of incomplete data using classifier ensembles. In *2012 International Conference on Systems and Informatics (ICSAI)*, pages 2229–2232, 2012.

3. M. Clerc and J. Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, 6:58–73, 2002.
4. T. G. Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15, 2000.
5. G. Doquire and M. Verleysen. Feature selection with missing data using mutual information estimators. *Neurocomputing*, 90:3–11, 2012.
6. 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.
7. C. Guerra-Salcedo and D. Whitley. Feature selection mechanisms for ensemble creation: a genetic search perspective. In *Data Mining with Evolutionary Algorithms: Research Directions. Papers from the AAAI Workshop*, 1999.
8. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11:10–18, 2009.
9. J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.
10. J. Kennedy. Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. 2011.
11. S. Krause and R. Polikar. An ensemble of classifiers approach for the missing feature problem. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 1, pages 553–558, 2003.
12. M. Lichman. UCI machine learning repository, 2013.
13. R. J. Little and D. B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.
14. H. Liu and H. Motoda. *Feature selection for knowledge discovery and data mining*, volume 454. Springer Science & Business Media, 2012.
15. L. S. Oliveira, M. Morita, and R. Sabourin. Feature selection for ensembles applied to handwriting recognition. *International Journal of Document Analysis and Recognition (IJDAR)*, 8:262–279, 2006.
16. D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.
17. D. W. Opitz. Feature selection for ensembles. *AAAI/IAAI*, 379:384, 1999.
18. W. Qian and W. Shu. Mutual information criterion for feature selection from incomplete data. *Neurocomputing*, 168:210–220, 2015.
19. J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.
20. M. Saar-Tsechansky and F. Provost. Handling missing values when applying classification models. *Journal of machine learning research*, 8:1623–1657, 2007.
21. J. Su and H. Zhang. A fast decision tree learning algorithm. In *Proceedings of the 21st National Conference on Artificial Intelligence*, volume 1, pages 500–505, 2006.
22. C. T. Tran, M. Zhang, P. Andreae, and B. Xue. Improving performance for classification with incomplete data using wrapper-based feature selection. *Evolutionary Intelligence*, 9:81–94, 2016.
23. C. T. Tran, M. Zhang, P. Andreae, and B. Xue. A wrapper feature selection approach to classification with missing data. In *European Conference on the Applications of Evolutionary Computation*, pages 685–700, 2016.
24. B. Xue, M. Zhang, W. Browne, and X. Yao. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, 20:606–626, 2016.