Knowledge-Based Systems 39 (2013) 240–247

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Genetic algorithms in feature and instance selection

Chih-Fong Tsai [a,*], William Eberle [b], Chi-Yuan Chu [a]

[a] Department of Information Management, National Central University, Taiwan
[b] Department of Computer Science, Tennessee Technological University, USA

ARTICLE INFO

Article history:
Received 9 May 2012
Received in revised form 12 November 2012
Accepted 18 November 2012
Available online 28 November 2012

Keywords:
Genetic algorithms
Feature selection
Instance selection
Data mining
Data preprocessing

ABSTRACT

Feature selection and instance selection are two important data preprocessing steps in data mining, where the former is aimed at removing some irrelevant and/or redundant features from a given dataset and the latter at discarding the faulty data. Genetic algorithms have been widely used for these tasks in related studies. However, these two data preprocessing tasks are generally considered separately in literature. It is unknown what the performance differences would be when feature and instance selection and feature or instance selection are performed individually. Therefore, the aim of this study is to perform feature selection and instance selection based on genetic algorithms using different priorities to examine the classification performances over different domain datasets. The experimental results obtained from four small and large scale datasets containing various numbers of features and data samples show that performing both feature and instance selection usually make the classifiers (i.e., support vector machines and *k*-nearest neighbor) perform slightly poorer than feature selection or instance selection individually. However, while there is not a significant difference in classification accuracy between these different data preprocessing methods, the combination of feature and instance selection largely reduces the computational effort of training the classifiers, as opposed to performing feature and instance selection individually. Considering both classification effectiveness and efficiency, we demonstrate that performing feature selection first and instance selection second is the optimal solution for data preprocessing in data mining. Both SVM and *k*-NN classifiers provide similar classification accuracy to the baselines (i.e., those without data preprocessing). The decisions regarding which data preprocessing task to perform for different dataset scales are also discussed.

1. Introduction

The process of knowledge discovery in databases (KDD), or data mining, generally involves a number of steps, such as dataset selection, data preprocessing, data analysis, and result interpretation and evaluation [5,16]. Data preprocessing is one of the most important steps with the aim of making the chosen dataset as 'clean' as possible for eventual analysis and evaluation. In other words, quality mining results cannot be obtained if the data quality is low [27,8].

Feature selection (or dimensionality reduction) and *instance selection* (or record reduction) are two of the more active preprocessing problems in data mining. This is because the number of features and data samples selected is usually very large in most real-world data mining problems.

If too many instances are considered, it can result in large memory requirements, high disk access, slow execution speed, and a possible over-sensitivity to noise [55]. In addition, it is often the fact that data are not all equally informative and some data points will be further away from the sample mean than what is deemed reasonable. Similarly record reduction is aimed at discarding faulty data (or outliers), which could be considered as noisy points lying outside a set of defined clusters and could lead to significant performance degradation [1,4]. Data mining tasks such as classification or prediction performance that is carried out without considering the instance selection step will very likely lead to poorer results [47,56].

On the other hand, if too many features are used for data analysis, it can cause the curse of dimensionality problems [36]. Since not all of the pre-chosen features are informative, the objective of feature selection is to select more representative features which have more discriminative power over a given dataset. This is also called dimensionality reduction [26].

In the literature, many related studies have shown promising results for feature selection and instance selection approaches [25,35,42,50,53]. However, up until now, the focus has been on either selecting more representative features or reducing faulty data, as it relates to effective classification or prediction. This leads to the important research question about which step (i.e., feature

* Corresponding author. Tel.: +886 3 422 7151; fax: +886 3 4254604.
E-mail address: cftsai@mgt.ncu.edu.tw (C.-F. Tsai).

0950-7051/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.11.005

selection or instance selection) should be performed first when both steps are critical to improving the mining performance. For many relevant and large scale datasets, both data preprocessing steps need to be performed. This is because in many domain problems there is usually no exact agreed upon number of variables, and all of those collected for a specific domain may not be informative. Furthermore, some data samples in a given large dataset may be regarded as noisy. Therefore, feature selection and instance selection should both be considered in order to develop a more effective model [17,11].

Genetic algorithms (GAs) comprise one of the most widely used techniques for feature and instance selection, and can improve the performance of data mining algorithms [12,14,39,37,46,51,52]. In particular, Cano et al. [7] have shown that better results can be obtained with GAs than with many traditional and non-evolutionary instance selection methods in terms of better instance selection rates and higher classification accuracy. Moreover, GAs have been shown to be suitable for large-scale feature selection problems [33].

However, very few consider feature selection and instance selection *together* using a GA over a given dataset. For example, given a dataset D containing m dimensional features and i data samples, using feature selection and instance selection as the first and second preprocessing steps respectively, will lead to D_1 containing n dimensional features and j data samples (where $0 < n < m$ and $0 < j < i$). On the other hand, if the operations are performed in reverse order, different results can be obtained.

The aim of this study is to perform feature selection and instance selection based on genetic algorithms using different priorities and to examine the classification performances over different domain datasets. In addition, the results will be compared, where a dataset is created without considering both data preprocessing steps, by feature selection only, and a dataset by instance selection only.

The rest of this paper is organized as follows. Section 2 describes the concept of feature selection and instance selection. In addition, genetic algorithms are overviewed in terms of data preprocessing. Section 3 presents the research design and experimental results. Finally, some conclusions are offered in Section 4.

2. Literature review

2.1. Feature selection

The number of features (or variables) collected in a dataset is usually relatively large (i.e., the curse of dimensionality) and not all of these features are informative or can provide high discriminative power [43]. The aim of feature selection is to remove the irrelevant and/or redundant features from the chosen dataset, thereby improving the performance of the classification and/or clustering algorithms. In addition, for a specific a dataset, feature selection can help analysts understand which features are important as well as how they are related.

Feature selection can be defined as the process of choosing a minimum subset of m features from the original dataset of n features ($m < n$), so that the feature space (i.e. the dimensionality) is optimally reduced according to the following evaluation criteria [10]:

- the classification accuracy does not significantly decrease; and
- the resulting class distribution, given only the values for the selected features, is as close as possible to the original class distribution, given all features.

A feature selection algorithm usually consists of four steps: subset generation, subset evaluation, stopping criterion, and result validation [10]. Subset generation is a search procedure which generates subsets of features for evaluation. Each subset generated is evaluated by some specific evaluation criterion and compared with the previous best one with respect to this criterion. If a new subset is found to be better, then the previous best subset is replaced by the new subset.

The interested reader can refer to Kudo and Sklansky [33] and Guyon and Elisseeff [26] for more information.

2.2. Instance selection

Wilson and Martinez [55] found that one problem with using the original data points is that there may not be any located at the precise points that would make for the most accurate and concise concept description. Therefore, the aim of instance selection, or record reduction, is to reduce the size of a dataset while still maintaining the integrity of the original dataset. In some cases, generalization accuracy can increase when noisy instances are removed and when decision boundaries are smoothed to more closely match the true underlying function.

These instances can also be regarded as outliers (or bad data). Specifically, outliers are those data points which are highly unlikely to occur given a model of the data. One approach to performing this task is to calculate the distances to neighboring data points by implementing a clustering algorithm [21].

Instance selection can be defined as follows. Let X_i be an instance where $X_i = (X_{i1}, X_{i2}, \ldots, X_{im}, X_{ic})$ meaning that X_i is represented by m-dimensional features and X_i belongs to class c given by X_{ic}. Then, assume that there is a training set TR which consists of M instances and a testing set TS composed of N instances. If $S \subseteq TR$ is the subset of selected samples that are produced by some instance selection algorithm, then we can classify a new pattern T from TS over the instances of S.

The interested reader can refer to Reinartz [47], Liu and Motoda [40], Jankowski and Grochowski [30] and Grochowski and Jankowski [24] for more information.

2.3. Genetic algorithms

The main idea behind the evolutionary algorithms (EAs) is derived from Darwin's theory of evolution arising from natural selection, of which genetic algorithms (GA) are one example. The basic idea of a GA is that you have a population of strings (called chromosomes), which encode candidate solutions (called individuals) to an optimization problem. In general, the genetic information (i.e., chromosome) is represented by a bit string (such as binary strings of 0s and 1s), and sets of bits encode the solution. Genetic operators are then applied to the individuals of the population for the next generation (i.e., a new population of individuals). There are two main genetic operators: crossover and mutation. Crossover creates two offspring strings from two parent strings by copying selected bits from each parent, whereas mutation randomly changes the value of a single bit (with small probability). In addition, a fitness function is used to measure the quality of an individual in order to increase the probability that the single bit can survive throughout the evolutionary process. Moreover, a GA can deal with large search spaces efficiently, and hence has less chance to arrive at a local optimal solution than other algorithms [22,15].

GAs have been tested on a number of domains for solving the feature and instance selection problems individually, such as Aydogan et al. [3], Das et al. [9], Pedrycz and Syed Ahmad [41], and Ratta et al. [45] for feature selection and Garcia et al. [18,19], Garcia-Pedrajas and Perez-Rodriguez [20], and Triguero et al. [49] for instance selection.

On the other hand, Fragoudis et al. [17] proposed an approach integrating feature and instance selection for text classification. In their approach, features are sequentially selected where there is a high precision in predicting the target class. Documents that do not contain at least one such feature are dropped from the training set. A search is performed within the subset of the initial dataset for a set of features that tend to predict the complement of the target class. All of the features and documents selected comprise the new training set. However, this approach was only applied to the problem of text classification, and results where only compared against one mutual information based feature selection method.

De Souza et al. [11] proposed a simulated annealing-based simultaneous feature and sample selection (SASFSS) algorithm. In contrast to Fragoudis et al.'s method [17], their instance selection results are primarily based on the results of feature selection. Specifically, two simulated annealing executions are utilized to represent the feature and instance selection tasks, and the combination of the feature and instance selection processes allows for SASFSS to implicitly force an influence of the feature selection over the instance selection process and vice versa. Although several datasets were used for the experiments, the largest contained only 70 attributes and 690 instances. In other words, the number of instances is too small to make a reliable assessment of SASFSS. In addition, only one specific feature and instance selection method was evaluated.

There have also been several studies where GAs are used to perform both feature and instance selection tasks at the same time. Kuncheva and Jain [34] conducted experiments where a GA was employed to simultaneously select suitable instances and features for a *k*-NN classifier. They showed that a GA was an expedient solution compared to four other approaches: Wilson's instance selection technique [54], sequential forward feature selection, Wilson's instance selection followed by sequential forward feature selection, and sequential forward feature selection followed by Wilson's instance selection. However, only two datasets were used, with the largest dataset being composed of 36 features and 6435 data samples only. In addition, aside from comparison with the four chosen approaches, we do not know whether using GA for simultaneous instance and feature selection would offer better performance than for instance selection and feature selection consecutively.

Ahn and Kim [2] used a GA method to simultaneously optimize feature weighting (i.e., feature selection) and instance selection for case-based reasoning in the bankruptcy prediction problem. However, the dataset they used was only composed of 2670 data samples. Similarly, Ros et al. [48] proposed a hybrid genetic approach, which treats feature and instance selection problems as a single optimization problem. Specifically, seven datasets were used for experiments, in which the datasets contain 166 features (in 566 data samples) and the largest number of data samples is 768 (using eight features). Ramirez-Cruz et al. [44] combined GA and evolution strategies (i.e., one of the EA models) to select instances and weight the features for the *k*-NN classifier. Again, the datasets used for their experiments are very small, with the largest containing only nine features and 958 data samples.

Ho et al. [28] designed an intelligent genetic algorithm (IGA) to tackle both instance and feature selection problems simultaneously by introducing a special orthogonal cross operator. They show that IGA performs better than the method developed by Kuncheva and Jain [34].

A more comprehensive study was conducted by Derrac et al. [13] who proposed an evolutionary model based on cooperative coevolution to perform feature and instance selection in *k*-NN classification. This approach performs better than other evolutionary feature and instance selection methods over a wide range of datasets. However, the dataset containing the largest number of data samples is 6435, where the number of features is 36, and the dataset containing the largest number features is 90, where the number of data samples is 360. In addition, the datasets only range from 2 to 15-class problems.

In short, performing both feature and instance selection at the same time is an important research problem, and there have been several studies focused on this issue appearing in the literature. However, some limitations of related studies are that (1) much related work only considers one specific classifier to assess the selected result of the GA, e.g. *k*-NN; (2) there has been no comprehensive study to examine the performance obtained when both tasks are executed individually or in certain orders with the GA, which is the weakness of Kuncheva and Jain's method [34]; (3) since both feature and instance selection problems need to be solved, the datasets should be large enough to more accurately reflect real-world domains. The following work attempts to address these issues.

3. Experiments

3.1. Experimental setup

3.1.1. The datasets

There are eight different datasets used in this study. They are divided into small and large scale datasets. The reason for doing this is to examine whether the findings will be different findings depending on the scale of the dataset. In this paper, we define a small scale dataset as containing either small numbers of features or data samples, or both (e.g., the Balance-scale dataset contains only four features and 625 data samples, which is a very small dataset), and a large scale dataset contains either a certain large number of features or data samples or both (e.g., the KDD Cup'99 and '01 datasets contain 494,020 data samples and 2197 features respectively).

We choose four small datasets from the UCI Machine Learning Repository.[1] This repository has been widely used in related studies, such as De Souza et al. [11], Derrac et al. [13], Ramirez-Cruz et al. [44], and Ros et al. [48]. In addition, we choose four datasets from four different sources that contain a very large number of features and data samples. The reason for choosing these datasets is the diversity in their numbers of features and data samples. The information regarding these datasets is listed in Table 1.

3.1.2. The experimental process

Five different procedures are utilized for constructing the classifier given a specified training dataset, as shown in Fig. 1. Note that the training samples used for feature and instance selection are the same as those used for training the classifier. A procedure consists of one of the following:

- *Procedure 1*: the baseline. This process is used to develop a baseline classifier based on the original dataset without any data preprocessing steps for comparison. That is, the comparative results allow us to understand whether the preprocessed datasets can improve the classifier without a preprocessing step.
- *Procedure 2*: feature selection. In this process, a GA is only used to select important features from the original dataset. The new dataset containing the selected features is then used to develop a classifier. As a result, we can understand whether performing feature selection can result in better classification performances.
- *Procedure 3*: instance selection. In this procedure, a GA is only used to select instances from the original dataset. The new dataset containing the selected instances is then used to develop a

[1] http://archive.ics.uci.edu/ml/.

Table 1
Datasets.

Datasets	No. of features	No. of samples	No. of classes
Small scale datasets			
Abalone	8	4177	28
Balance-scale	4	625	3
German credit	20	1000	2
Ionosphere	34	351	2
Large scale datasets			
KDD Cup'99[a]	41	494,020	5
KDD Cup'01[b] (Genes)	2197	779	15
UCSD Competition[c]	334	130,475	2
Covertype[d]	54	581,012	7

[a] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
[b] http://pages.cs.wisc.edu/~dpage/kddcup2001/.
[c] http://mill.ucsd.edu/.
[d] http://archive.ics.uci.edu/ml/datasets/Covertype.

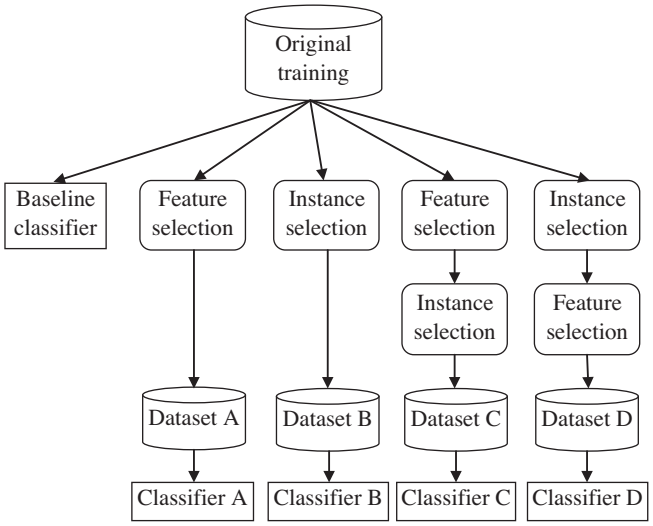

Fig. 1. The experimental process.

classifier. The results can be used to compare if the classifier followed by the instance selection step can provide better performances than the classifier only followed by the feature selection step.

- *Procedure 4*: feature selection + instance selection. This procedure is based on performing feature selection first using a GA after which the dataset is further 'processed' for the instance selection task using the GA. Therefore, a new dataset containing smaller numbers of features and instances is used to train a classifier.
- *Procedure 5*: instance selection + feature selection. Instance selection is performed first using a GA. The reduced dataset is then further processed by feature selection using the GA. This can be compared to Procedure 4 to determine which preprocessing step should be performed first in order to construct an effective classifier.

The original datasets are divided into training and testing sets, and 10-fold cross validation is used to train and test a classifier to avoid sample variability which may affect the performance of these classifiers. In particular, the dataset is divided into ten equal parts, with 90% of the dataset to perform model training, and the other 10% is used for model testing. This means that every subset will be trained and tested ten times, and the average prediction performance can be obtained consequently [32].

Table 2
Parameter settings of GA in related work.

Work	Population size	Crossover rate	Mutation rate
Ahn and Kim [2]	100	0.7	0.1
Grefenstette [23]	30	0.9	0.01
Ho et al. [28]	10	1.0	0.1
Kim and Han [31]	20	0.6	0.033
Kuncheva and Jain [34]	10	1.0	0.1
Ramirez-Cruz et al. [44]	50	0.8	0.01
Ros et al. [48]	100	0.5	0.05

3.1.3. The parameters for genetic algorithms

The WEKA (Waikato Environment for Knowledge Analysis) suite is used to perform feature and instance selection using a GA [29]. The fitness function used for the genetic search process is based on the Bayesian network learning algorithm and the coding method is based on binary encoding. Moreover, three parameters need to be adjusted: population size, crossover rate, and mutation rate. However, there are no general agreed-upon parameter settings for the GAs, as shown in Table 2. Therefore, in this paper, the range for the population size is set from 20, 30, 40 and 50, and the crossover rate is set from 0.6, 0.7, 0.8, 0.9 and 1. There are four different mutation rates, 0.001, 0.005, 0.01, and 0.05. Other related parameters are based on the default values of WEKA.

3.1.4. Classifier design

Classifiers are constructed using the five procedures outlined in the previous section (c.f. Fig. 1) based on support vector machines (SVM) and the *k*-nearest neighbor (*k*-NN). In related work, *k-NN* is the most widely used classifier to assess the feature and instance selection performances (c.f. Section 2.3). On the other hand, SVMs have shown their effectiveness in many pattern classification problems and provide better generalizations than many other techniques [6]. The radial basis function (RBF) kernel function is used for the SVM, in which different gamma values are examined, i.e. $\gamma = 0$, 0.1, 0.5, 1, and 2. For the *k*-NN algorithm, the *k* values are set from 1, 3, 5, 7, 9 and 11.

3.2. Results

3.2.1. Results on small scale datasets

Table 3 shows the average classification accuracy (from 10-fold cross validation) of the five procedures (c.f. Fig. 1), i.e., the four different preprocessing steps and one baseline. Note that feature selection, instance selection, feature selection + instance selection, instance selection + feature selection are abbreviated FS, IS, FSIS, and ISFS, respectively. In addition, the number in parentheses, after the classification accuracy, represents the ranking of the five procedures obtained by a specific classifier. Moreover, the numbers of selected features and samples in the reduced datasets created by the four different preprocessing steps are also presented.

It is interesting to note that performing instance selection alone appears to make an SVM perform better than performing feature selection alone, whereas for *k*-NN, performing feature selection only is better. This corresponds to Li et al. [38] where SVMs do not depend explicitly on the dimensionality of the problem.

One also notices that with the combination of features and instance selection, performing feature selection first and instance selection second (i.e., FSIS) results in both SVM and *k*-NN providing a higher accuracy than performing instance selection first and feature selection second (i.e., ISFS). In addition, the results show that with the combination of feature and instance selection, the classifiers perform slightly more poorly than feature selection or instance selection individually.

Table 3
Classification accuracy of the five procedures over small scale datasets (the numbers in the brackets mean their performance rankings.).

Datasets	No. of features	No. of samples	SVM	*k*-NN
Abalone				
Non-preprocessing	8	4177	26.87% (1) ± 1.66%	25.26% (1) ± 1.5%
FS	3	4177	26.25% (2) ± 1.14%	22.51% (2) ± 1.15%
IS	8	530	21.47% (5) ± 1.32%	21.61% (4) ± 2.01%
FSIS	3	3758	25.87% (3) ± 0.67%	21.85% (3) ± 0.62%
ISFS	3	530	21.92 (4) ± 0.46%	20.44% (5) ± 0.55%
Balance-scale				
Non-preprocessing	4	625	90.55% (1) ± 1.45%	90.22% (1) ± 0.64%
FS	3	625	76.92% (3) ± 1.28%	78.37% (2) ± 2.13%
IS	4	28	78.15% (2) ± 0.49%	77.04% (3) ± 2.02%
FSIS	3	68	72.70% (4) ± 0.32%	73.99% (4) ± 1.61%
ISFS	2	28	62.38% (5) ± 1.1%	65.28% (5) ± 2.71%
German				
Non-preprocessing	20	1000	69.97% (2) ± 0.3%	74.37% (1) ± 1.3%
FS	8	1000	69.87% (4) ± 0.27%	69.67% (2) ± 1.1%
IS	20	210	70.00% (1) ± 0.3%	68.00% (4) ± 0.7%
FSIS	8	55	69.90% (3) ± 0.1%	69.20% (3) ± 0.5%
ISFS	6	210	69.90% (3) ± 0.6%	67.20% (5) ± 0.9%
Ionosphere				
Non-preprocessing	34	351	94.29% (1) ± 0.58%	86.86% (2) ± 1.72%
FS	4	351	90.29% (3) ± 0.29%	88.86% (1) ± 0.29%
IS	34	112	92.01% (2) ± 1.13%	86.29% (3) ± 2%
FSIS	4	95	89.44% (4) ± 0.57%	82.30% (4) ± 0.29%
ISFS	7	112	86.91% (5) ± 1.41%	81.17% (5) ± 0.26%

Comparison of the four reduced datasets with their original dataset, shows that the GA filters out many features and instances, but the reduced datasets do not degrade the performances of the SVM and *k*-NN except when performing instance selection (i.e., IS and ISFS) over the Abalone. Possible future work could include a study of the effect of the number of original features on the performance of instance selection in certain datasets.

3.2.2. Results on large scale datasets

For large scale datasets, Table 4 shows the results of SVM and *k*-NN based on the four different preprocessing steps and one baseline. For feature selection vs. instance selection, one will note that when the number of features in the dataset is relatively large (e.g., 2197 in KDD Cup'01), SVM and *k*-NN, based on performing feature selection alone, does not provide better classification results than the ones based on performing instance selection alone. This happens when the number of data samples in the dataset is relatively large (e.g., 494,020 in KDD Cup'99), instance selection alone does not make SVM and *k*-NN outperform the methods performing feature selection alone.

However, it is difficult to identify the priority when feature and instance selection are combined that will allow SVM and *k*-NN to provide higher classification accuracy. Unlike the findings based on small scale datasets, the classifiers followed by combining feature and instance selection sometimes perform better than those followed by feature selection and instance selection individually.

Table 4
Classification accuracy of the five procedures over large scale datasets.

Datasets	No. of features	No. of samples	SVM	*k*-NN
KDD Cup'99				
Non-preprocessing	41	494,020	99.71% (2) ± 0.98%	99.95% (2) ± 0.73%
FS	21	494,020	99.74% (1) ± 0.75%	99.93% (3) ± 0.8%
IS	41	244,389	94.05% (5) ± 0.67%	98.19% (4) ± 0.52%
FSIS	21	233,168	99.08% (3) ± 0.61%	98.96% (1) ± 0.43%
ISFS	15	244,389	95.42% (4) ± 0.39%	98.17% (5) ± 0.52%
KDD Cup'01 (Genes)				
Non-preprocessing	2197	779	46.67% (1) ± 1.02%	42.77% (1) ± 1.42%
FS	108	779	38.82% (5) ± 0.64%	41.13% (4) ± 0.51%
IS	2197	215	42.84% (3) ± 1.21%	41.28% (3) ± 1.56%
FSIS	108	612	42.97% (2) ± 0.77%	39% (5) ± 0.56%
ISFS	106	215	41.28% (4) ± 0.39%	41.8% (2) ± 0.95%
UCSD				
Non-preprocessing	334	130,475	90.68% (2) ± 0.58	84.65% (5) ± 0.71
FS	45	130,475	90.06% (3) ± 0.37	86.74% (4) ± 0.44
IS	334	37,601	90.7% (1) ± 0.51	90.79% (2) ± 0.97%
FSIS	45	28,281	89.8% (5) ± 0.13%	90.8% (1) ± 0.6%
ISFS	40	37,601	89.84% (4) ± 0.4%	90.67% (3) ± 0.66%
Covertype				
Non-preprocessing	54	581,012	52.15% (2) ± 0.65	64.97% (1) ± 0.73
FS	19	581,012	58.69% (1) ± 0.53	56.21% (3) ± 0.87
IS	54	162,845	48.76% (3) ± 0.44	60.7% (2) ± 1.09%
FSIS	19	173,160	46.12% (5) ± 0.62%	44.65% (5) ± 0.95%
ISFS	23	162,845	48.38% (4) ± 0.74%	46.83% (4) ± 0.87%

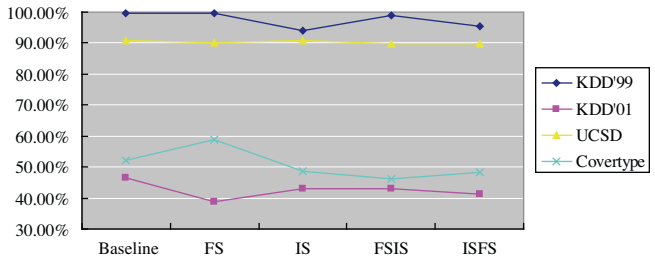

Fig. 2. The results of SVM based on the five procedures.

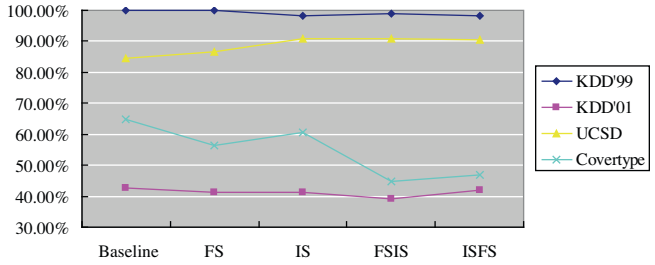

Fig. 3. The results of *k*-NN based on the five procedures.

Figs. 2 and 3 show the results of SVM and *k*-NN based on the four preprocessing steps and one baseline respectively. We can observe that the performance deviations between these five procedures over these datasets are very small, except for the Covertype dataset. This may be because it contains the largest number of data samples among these four datasets.

The results, except for the Covertype dataset, indicate that although SVM and *k*-NN without any data preprocessing perform slightly better than with feature and/or instance selection, there is not a significant difference.

3.2.3. Comparisons of computational cost

Since the results for four different data preprocessing procedures are similar for both SVM and *k*-NN to those without data-preprocessing, it is necessary to analyze the tradeoff between non-preprocessing and preprocessing. Tables 5 and 6 show the time complexity of data preprocessing and classifier training obtained for the four different preprocessing steps, as well as the baseline experiments over the four large scale datasets.[2]

According to Table 5, there is a large reduction in the time required for performing the data preprocessing and training in the SVM when compared with the baseline SVM without data preprocessing. The KDD Cup'01 dataset is the exception. This may be because the number of data samples is much smaller than in the other three datasets, too small to reveal the efficiency of performing data preprocessing. In other words, the size of this dataset can be regarded as a 2197×779 matrix. This suggests that if SVM is chosen as the classifier for pattern classification, it is appropriate to perform data preprocessing for enhancing the computational complexity. In addition, when the chosen dataset is very large, such as can be found in the KDD Cup'09, UCSD, and Covertype datasets, SVM data preprocessing can perform similarly to or better than the baseline SVM without data preprocessing.

However, for the *k*-NN classifier, which is a lazy learner (i.e., there is no need for offline training). Data preprocessing cannot significantly reduce the time complexity. The KDD Cup'99 dataset is the exception.

Although the computational complexity cannot be reduced by *k*-NN with data preprocessing, the classification accuracy is better than or similar to the baseline *k*-NN without data preprocessing over the UCSD and KDD Cup'99 datasets. Therefore, this indicates that when the chosen dataset is very large, such as with the UCSD and KDD Cup'99 datasets, it is better to perform data preprocessing for the *k*-NN to provide reasonably good performance.

In addition, the strategy of performing feature selection first followed by instance selection provides similar classification accuracies for both SVM and *k*-NN.

Table 5
Time complexity of data preprocessing and SVM training.

	Preprocessing	SVM	Total
KDD Cup'99			
Non-preprocessing	0	384.5 h	384.5 h
FS	2 min	42.5 h	42.5 h
IS	25 h	43.4 h	68.4 h
FSIS	15 h	20 h	35 h
ISFS	10 min	19.7 h	19.86 h
KDD Cup'01 (Genes)			
Non-preprocessing	0	5 min	5 min
FS	5 min	1 min	6 min
IS	3 min	10 s	3.16 min
FSIS	6 min	2.5 s	6 min
ISFS	4.7 min	1 s	4.7 min
UCSD			
Non-preprocessing	0	332.5 h	332.5 h
FS	20 min	103.5 h	103.8 h
IS	66.7 h	21.7 h	88.4 h
FSIS	7 h	3.6 h	10.6 h
ISFS	67.3 h	3.8 h	71.1 h
Covertype			
Non-preprocessing	0	840.5 h	840.5 h
FS	6 min	523.3 h	523.3 h
IS	35.5 h	149 h	184.5 h
FSIS	17.6 h	88.2 h	105.8 h
ISFS	10 min	73.5 h	73.67 h

Table 6
Time complexity of data preprocessing and *k*-NN training.

	Preprocessing	*k*-NN	Total
KDD Cup'99			
Non-preprocessing	0	85 h	85 h
FS	2 min	24.5 h	24.53 h
IS	25 h	35.5 h	60.5 h
FSIS	15 h	18.6 h	33.6 h
ISFS	10 min	14 h	14.67 h
KDD Cup'01 (Genes)			
Non-preprocessing	0	2 min	2 min
FS	5 min	1 min	6 min
IS	3 min	7 s	3.1 min
FSIS	6 min	1.9 s	6 min
ISFS	4.7 min	0.7 s	4.7 min
UCSD			
Non-preprocessing	0	3.5 h	3.5 h
FS	20 min	1.5 h	1.83 h
IS	66.7 h	2.5 h	69.17 h
FSIS	7 h	1.2 h	8.2 h
ISFS	67.3 h	1.3 h	68.6 h
Covertype			
Non-preprocessing	0	20 h	20 h
FS	6 min	16.7 h	16.8 h
IS	35.5 h	18.7 h	54.2 h
FSIS	17.6 h	17.7 h	35.3 h
ISFS	10 min	14.4 h	14.57 h

[2] The Matlab 7 software runs on an Intel Pentium 4, with a 3.4 Ghz CPU, and 1.5 GB RAM.

```
Begin
    If (Dataset feature size or instance size > 2000)
        If (feature size > instance size)
            Execute instance selection
        Else
            Execute feature selection
    Else
        If (two data preprocessing steps are required)
            Execute feature selection → instance selection
        Else
            Execute instance selection
End
```

Fig. 4. The strategy of performing data preprocessing for SVM.

```
Begin
    If (Dataset feature size or instance size > 3000)
        If (instance size > feature size)
            Execute feature selection
        Else
            data preprocessing is unnecessary
    Else
        If (two data preprocessing steps are required)
            Execute feature selection→instance selection
        Else
            Execute feature selection
End
```

Fig. 5. The strategy of performing data preprocessing for *k*-NN.

3.3. Discussion

Based on the experimental results, there are two strategies for performing the data preprocessing task – one for SVM and one for *k*-NN. Given a particular dataset, users can follow the defined decision rules to appropriately execute feature and/or instance selection before classifier training and testing. Figs. 4 and 5 show the respective pseudo codes for the decision rules when SVM and *k*-NN are used for pattern classification.

4. Conclusion

Feature selection and instance selection are two important data preprocessing steps in the data mining process. The main goal of conducting each of these two steps is to make a given dataset 'cleaner' and/or 'more representative' by filtering out irrelevant features and noisy data samples in order to obtain good quality mining results. This is the first attempt to assess the performance of using genetic algorithms to perform feature and instance selection steps of different priorities over different domain problems. In particular, there are four different data preprocessing approaches: instance selection, feature selection + instance selection, and instance selection + feature selection.

The small-scale experimental results show that performing feature selection first and instance selection second can make the classifiers provide slightly better classification results than performing instance selection first and feature selection second. However, the classifiers utilizing a combination of feature and instance selection perform slightly more poorly than the ones using feature selection or instance selection individually.

On the other hand, in the large-scale experiments, the classifiers sometimes perform better based on a combination of feature and instance selection than those based on feature and instance selection alone. However, it is hard to figure out the winner of these four different data preprocessing steps since there is not a big difference between them. Consequently, the computational cost of training classifiers becomes another important indicator to assess these data preprocessing methods. The time complexity analysis shows that the combination of feature and instance selection greatly reduces the computational cost of training classifiers. As a result, it can be seen that the combination of feature and instance selection is a suitable solution for data preprocessing on large data sets. More specifically, performing feature selection first and instance selection second will allow the SVM and *k*-NN classifiers to provide similar classification accuracies to the ones without data preprocessing.

Our research findings provide guidelines for implementing suitable baselines for performing feature and instance selection. Several issues can be considered in the future. First of all, the baselines can be compared with existing algorithms performing feature and instance selection simultaneously in terms of classification accuracy and computational complexity. Secondly, it would be useful to examine the performance of combining different feature and instance selection algorithms as a hybrid approach. For example, the feature selection task can be based on the genetic algorithm, principal component analysis, or information gain, etc., whereas the instance selection task can be approached by methods such as DROP 3 [55,18,19].

Acknowledgement

This work was supported in part by the National Science Council of Taiwan under Grant NSC 99-2410-H-008-034-.

References

[1] C.C. Aggarwal, P.S. Yu, Outlier detection for high dimensional data, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Santa Barbara, California, 2001. pp. 37–46.
[2] H. Ahn, K.-J. Kim, Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach, Applied Soft Computing 9 (2) (2009) 599–607.
[3] E.K. Aydogan, I. Karaoglan, P.M. Pardalos, HGA: hybrid genetic algorithm in fuzzy rule-based classification systems for high-dimensional problems, Applied Soft Computing 12 (2) (2012) 800–806.
[4] V. Barnett, T. Lewis, Outliers in Statistical Data, John Wiley & Son, New York, 1994.
[5] I. Bose, R.K. Mahapatra, Business data mining – a machine learning perspective, Information & Management 39 (3) (2001) 221–225.
[6] H. Byun, S.-W. Lee, A survey on pattern recognition applications of support vector machines, International Journal of Pattern Recognition and Artificial Intelligence 17 (3) (2003) 459–486.
[7] J.R. Cano, F. Herrera, M. Lozano, Using evolutionary algorithms as instance selection for data reduction: an experimental study, IEEE Transactions on Evolutionary Computation 7 (6) (2003) 561–575.
[8] S.F. Crone, S. Lessmann, R. Stahlbock, The impact of preprocessing on data mining: an evaluation of classifier sensitivity in direct marketing, European Journal of Operational Research 173 (3) (2006) 781–800.
[9] N. Das, R. Sarkar, S. Basu, M. Kundu, M. Nasipuri, D.K. Basu, A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application, Applied Soft Computing 12 (5) (2012) 1592–1606.
[10] M. Dash, H. Liu, Feature selection methods for classifications, Intelligent Data Analysis 1 (3) (1997) 131–156.
[11] De Souza, J.T., Do Carmo, R.A.F., and De Campos, G.A.L., 2008. A novel approach for integrating feature and instance selection. In: Proceedings of the International Conference on Machine Learning and Cybernetics, pp. 374–379.
[12] J. Derrac, S. García, F. Herrera, A survey on evolutionary instance selection and generation, International Journal of Applied Metaheuristic Computing 1 (1) (2010) 60–92.
[13] J. Derrac, S. García, F. Herrera, IFS-CoCo: instance and feature selection based on cooperative coevolution with nearest neighbor rule, Pattern Recognition 43 (2010) 2082–2105.
[14] M.E. ElAlmi, A novel image retrieval model based on the most relevant features, Knowledge-Based Systems 24 (1) (2011) 23–32.

[15] P.G. Espejo, S. Ventura, F. Herrera, A survey on the application of genetic programming to classification, IEEE Transactions on Systems, Many, and Cybernetics – Part C: Applications and Reviews 40 (2) (2010) 121–144.
[16] U. Fayyad, S.G. Piatetsky, P. Smyth, Advances in Knowledge Discovery and Data Mining, The MIT Press, 1996.
[17] Fragoudis, D., Meretakis, D., Likothanassis, S., 2002. Integrating feature and instance selection for text classification. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 501–506.
[18] S. Garcia, J. Derrac, J.R. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 417–435.
[19] S. Garcia, J. Derrac, I. Triguero, C.J. Carmona, F. Herrera, Evolutionary-based selection of generalized instances for imbalanced classification, Knowledge-Based Systems 25 (1) (2012) 3–12.
[20] N. Garcia-Pedrajas, J. Perez-Rodriguez, Multi-selection of instances: a straightforward way to improve evolutionary instance selection, Applied Soft Computing 12 (11) (2012) 3590–3602.
[21] A. Ghosting, S. Parthasarathy, M.E. Otey, Fast mining of distance-based outliers in high-dimensional datasets, Data Mining and Knowledge Discovery 16 (2008) 349–364.
[22] D.E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addition Wesley, 1989.
[23] J.J. Grefenstette, Optimization of control parameters of genetic algorithms, IEEE Transactions on Systems, Man and Cybernetics 16 (1) (1986) 122–128.
[24] Grochowski, M., Jankowski, N., 2004. Comparison of instances selection algorithms II: results and comments. In: Proceedings of the International Conference on Artificial Intelligence and Soft Computing, pp. 580–585.
[25] S. Gunal, R. Edizkan, Subspace based feature selection for pattern recognition, Information Sciences 178 (2008) 3716–3726.
[26] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
[27] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2000.
[28] S.-Y. Ho, C.-C. Liu, S. Liu, Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm, Pattern Recognition Letters 23 (2002) 1495–1503.
[29] W.H. Ian, F. Eibe, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, 1999.
[30] Jankowski, N., Grochowski, M., 2004. Comparison of instances selection algorithms I: algorithms survey. In: Proceedings of the International Conference on Artificial Intelligence and Soft Computing, pp. 598–603.
[31] K.J. Kim, I. Han, Genetic algorithm approach to feature discretization in artificial neural network for the prediction of stock price index, Expert Systems with Applications 19 (2) (2000) 125–132.
[32] Kohavi, R., 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International Joint Conference on Artificial Intelligence, pp. 1137–1143.
[33] M. Kudo, J. Sklansky, Comparison of algorithms that select features for pattern classifiers, Pattern Recognition 33 (2000) 25–41.
[34] L.I. Kuncheva, L.C. Jain, Nearest neighbor classifier: simultaneous editing and feature selection, Pattern Recognition Letters 20 (1999) 1149–1156.
[35] A. Kuri-Morales, F. Rodriguez-Erazo, A search space reduction methodology for data mining in large databases, Engineering Applications of Artificial Intelligence 22 (1) (2009) 57–65.
[36] J. Li, M.T. Manry, P.L. Narasimha, C. Yu, Feature selection using a piecewise linear network, IEEE Transactions on Neural Networks 17 (5) (2006) 1101–1115.
[37] R. Li, J. Lu, Y. Zhang, T. Zhao, Dynamic Adaboost learning with feature selection based on parallel genetic algorithm for image annotation, Knowledge-Based Systems 23 (3) (2010) 195–201.
[38] S. Li, J.T. Kwok, H. Zhu, Y. Wang, Texture classification using support vector machines, Pattern Recognition 36 (12) (2003) 2883–2893.
[39] S. Li, H. Wu, D. Wan, J. Zhu, An effective feature selection method for hyperspectral image classification based on genetic algorithm and support vector machine, Knowledge-Based Systems 24 (1) (2011) 40–48.
[40] H. Liu, H. Motoda, On issues of instance selection, Data Mining and Knowledge Discovery 6 (2002) 115–130.
[41] W. Pedrycz, S.S. Syed Ahmad, Evolutionary feature selection via structure retention, Expert Systems with Applications 39 (15) (2012) 11801–11807.
[42] S. Piramuthu, Evaluating feature selection methods for learning in data mining applications, European Journal of Operational Research 156 (2004) 483–494.
[43] W.B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality, Wiley-Interscience, 2007.
[44] Ramirez-Cruz, J.-F., Alarćon-Aquino, V., Fuentes, O., Garćıa-Banuelos, L., 2006. Instance selection and feature weighting using evolutionary algorithms. In: Proceedings of the International Conference on Computing, pp. 73–79.
[45] G.A. Ratta, J. Vega, A. Murari, JET-EFDA Contributors, Improved feature selection based on genetic algorithms for real time disruption prediction on JET, Fusion Engineering and Design 87 (9) (2012) 1670–1678.
[46] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, A.K. Jain, Dimensionality reduction using genetic algorithms, IEEE Transactions on Evolutionary Computation 4 (2) (2000) 164–171.
[47] T. Reinartz, A unifying view on instance selection, Data Mining and Knowledge Discovery 6 (2) (2002) 191–210.
[48] F. Ros, S. Guillaume, M. Pintore, J.R. Chrétien, Hybrid genetic algorithm for dual selection, Pattern Analysis and Applications 11 (2008) 179–198.
[49] I. Triguero, S. Garcia, F. Herrera, Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification, Pattern Recognition 44 (4) (2011) 901–916.
[50] C.-F. Tsai, Feature selection in bankruptcy prediction, Knowledge-Based Systems 22 (2) (2009) 120–127.
[51] C.-F. Tsai, Y.-H. Lu, D.Y. Yen, Determinants of intangible assets value: the data mining approach, Knowledge-Based Systems 31 (2012) 67–77.
[52] H. Uğuz, A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm, Knowledge-Based Systems 24 (7) (2011) 1024–1032.
[53] J.-S. Wang, J.-C. Chiang, A cluster validity measure with outlier detection for support vector clustering, IEEE Transactions on Systems, Man, and Cybernetics – Part B Cybernetics 38 (1) (2008) 78–89.
[54] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics 2 (1972) 408–421.
[55] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Machine Learning 38 (3) (2000) 257–286.
[56] J. Yang, S. Olafsson, Optimization-based feature selection with adaptive instance sampling, Computers & Operations Research 33 (11) (2006) 3088–3106.