The Journal of Systems and Software 85 (2012) 1067–1074

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

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

Noisy data elimination using mutual *k*-nearest neighbor for classification mining

Huawen Liu[a,b], Shichao Zhang[c,d,*]

[a] Department of Computer Science, Zhejiang Normal University, China
[b] Key Laboratory of Symbol Computation and Knowledge Engineering of Ministry of Education, China
[c] College of Computer Science and Information Technology, Guangxi Normal University, China
[d] Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia

ARTICLE INFO

Article history:
Received 25 April 2011
Received in revised form
19 September 2011
Accepted 7 December 2011
Available online 16 December 2011

Keywords:
Data mining
Pattern classification
*k*NN
Mutual nearest neighbor
Data reduction

ABSTRACT

k nearest neighbor (*k*NN) is an effective and powerful lazy learning algorithm, notwithstanding its easy-to-implement. However, its performance heavily relies on the quality of training data. Due to many complex real-applications, noises coming from various possible sources are often prevalent in large scale databases. How to eliminate anomalies and improve the quality of data is still a challenge. To alleviate this problem, in this paper we propose a new anomaly removal and learning algorithm under the framework of *k*NN. The primary characteristic of our method is that the evidence of removing anomalies and predicting class labels of unseen instances is mutual nearest neighbors, rather than *k* nearest neighbors. The advantage is that pseudo nearest neighbors can be identified and will not be taken into account during the prediction process. Consequently, the final learning result is more creditable. An extensive comparative experimental analysis carried out on UCI datasets provided empirical evidence of the effectiveness of the proposed method for enhancing the performance of the *k*-NN rule.

© 2011 Elsevier Inc. All rights reserved.

1. Introduction

Data mining refers to the process of identifying potentially useful and interesting patterns from large databases (Duda et al., 2001). During past years, researchers have made great progress in many data mining topics, where pattern classification is arguably one of the most fundamental tasks and an extensively studied research topic in data mining and decision making. Pattern classification aims to predict or mark unknown instances (or observations, samples) with one of predefined labels in the light of models or behaviors induced from historical data. Due to its significant role in exploratory pattern analysis, many outstanding and different kinds of classification learning algorithms, such as Naïve Bayes (NBC), *k* nearest neighbors (*k*NN) and C4.5, have been developed and widely used in many domains, such as decision support, pattern recognition and financial forecasts (Duda et al., 2001). At the ICDM'06 conference, top 10 the most influential mining algorithms were voted by researchers in the data mining community, of which 6 (C4.5, SVM, AdaBoost, *k*NN, NBC and CART) are classification related algorithms (Wu et al., 2008).An important aspect to the successful application of classification learning algorithms is the quality of data (Davidson and Tayi, 2009). In reality, a prevalent underlying philosophy in most learning techniques is to assume the data to process is consistent, complete and error free. However, this is rarely the case in practice as the data is often collected from different heterogeneous data sources, where inconsistent and incomplete situations are often encountered. With the rapid development of information technology, the data collected is becoming larger and larger. It not only provides many opportunities and information but also poses a great challenge to classification learning algorithms. Since the large volume of database may include more noisy, incorrect or inconsistent data, wrong decision inevitably may be made by decision-makers. Meanwhile, the complexity of learning algorithms in constructing classification models will increase. Thus, it is necessary to develop an effective clean technique to remove noisy data from databases and this requirement is becoming compelling. Yang and Wu (2006) pointed out that automatic data pre-processing including cleansing and noise handling is one important topic of 10 challenging data mining problems should be resolved.

Noises commonly exist in reality and may come from various possible sources, such as user entry errors, misspellings, missing information, label errors, data transformation or storage (Chapman, 2005; Pyle, 1999). Moreover, fusing data collected from multiple different or heterogeneous data sources may also raise the inconsistent or incorrect problems. According to the origin of noises, inconsistent or incorrect situations mainly occur at (missing or error) value, (continuous or nominal, relevant or redundant) feature, (imbalance, wrong or missing) class and (duplicate,

* Corresponding author at: College of Computer Science and Information Technology, Guangxi Normal University, China.
E-mail addresses: hwliu@zjnu.edu.cn (H. Liu), zhangsc@it.uts.edu.au (S. Zhang).

0164-1212/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2011.12.019

inconsistent or anomalous) instance. The noisy data should be audited firstly before it has been used to analyze and explore classification models. Currently, lots of techniques and tools have been developed to deal with these issues (Chapman, 2005; Pyle, 1999). For example, the missing value is often replaced with its mean or the commonly used ones, while the continuous feature is discretized into nominal one; the significant features are identified by feature extraction or feature selection techniques (Liu et al., 2009, 2010). Here we mainly place our focus on eliminating inconsistent or anomalous instances under the context of pattern classification.

Due to many complex aspects, abnormal instances are often prevailing in large scale datasets. Generally speaking, the average error rate of a dataset is around 5% or higher (Zhang and Wu, 2010). As a result, the truthfulness of datasets is low so that the classification models built on such unreliable or abnormal datasets have poor performance and robustness. This has been demonstrated by numerous studies in literature (Hulse and Khoshgoftaar, 2009). To alleviate this issue, many data preprocessing techniques to detect anomalies or remove inconsistencies are available and have shown their effectiveness in many scenarios (Chandola et al., 2009; Brighton and Mellish, 2002). For instance, Yu (2011) utilized selective sampling technique to enhance the retrieval performance in facing with large scale data.

It is worthy to mention that noisy instances bring less impact to simple learning algorithms, such as NBC and *k*NN than sophisticated classifiers such as SVM or random forests (Zhang and Wu, 2010). Due to the simplicity and easy-to-implement of *k*NN, this off-the-shelf method has also been applied to eliminate the noisy instances in databases, and can achieve competitive results even compared to the most sophisticated methods (Chapman, 2005; Pyle, 1999; Chandola et al., 2009; Brighton and Mellish, 2002). However, from a methodological perspective *k*NN has several shortcomings which are still challenges to researchers. Generally, the performance of *k*NN heavily relies on the number *k* of nearest neighbors, the size *N* of training instances and the choice of distance metric (Latourrette, 2000). If these three aspects have not been done well, the performance of *k*NN will not be good as one expected.

In this paper, we propose a new lazy learning algorithm named Mutual *k*NN Classifier (M*k*NNC) for pattern classification. It is a variant of *k*NN classifier, which takes use of *k* nearest neighbors to predict the class label of new instance. Unlike the classical *k*NN method, M*k*NNC firstly eliminates noisy instances by using a concept called mutual nearest neighbor (MNN), and then makes a prediction for new instance on the basis of its MNNs. The advantage is that the prediction result is more creditable because “fake” neighbors or outliers have been excluded before the prediction procedure. To the best of our knowledge, there is no report concerning both classification and noise elimination with MNNs. The contribution of this work can be summarized as follows:

- M*k*NNC is an instance-based learning method. Its core idea is relatively intuitional and easy to implement. Meanwhile, it is more robust as encounter noises or inconsistent data.
- Anomalies will be firstly detected and removed from databases by the mutual nearest neighbors before constructing classification models. Consequently, the information of noise data will not be taken as determinant conditions during the learning process. Thus, the final prediction results are more creditable.
- M*k*NNC involves classification learning and anomaly detection and elimination. Both of them are fulfilled with MNN, which carries more useful and reliable information than *k*NN in determining the relationship between instances.

The rest of the paper is organized as follows. Related work on *k*NN and noise handling is provided in Section 2. Section 3 gives some fundamental concepts about classification and *k*NN. In Section 4, the concept of mutual nearest neighbor and the framework of our proposed algorithm are presented. Section 5 shows the experimental results of our method conducted on UCI datasets. Conclusions and future work are provides in Section 6.

2. Related work

In this section, we will briefly review the-state-of-the-art of *k*NN learning algorithms and anomaly handling techniques. Interested readers can consult good survey literatures (see, e.g., Chandola et al., 2009; Brighton and Mellish, 2002; Bhatia and Vandana, 2010) to get more details.

2.1. *k*NN

As mentioned above, the performance of *k*NN classifiers heavily relies on the following factors: the sample size *N*, the selection of distance metric and *k*. To cope with these problems, great efforts have been made during past years.

For the distance metric, many methods adopt locally adaptive distances, rather than the Euclidean distance in the traditional *k*NN, to tackle the global optimality problem. Typical examples of this kind include the discriminant adaptive metric in DANN, the adaptive distance in ADAMENN, the weight adjusted metric in WAKNN, the Mahanalobis distance in LMNN and the informative metric in IKNN (Song et al., 2007). Note that many distance metrics are estimated with weighted similarity between instances. For example, Jahromi et al. (2009) assigned each instance with different weight in terms of the leave-one-out classification performance, where the instances with zero weight had been virtually removed from the training set. In addition, probability model has also been adopted to enhance the performance of *k*NN. As an example, Toyama et al. (2010) proposed a probably correct approach of *k*NN on the basis of the marginal distribution of *k*th nearest neighbors in low dimensions. It is noticeable that most of these methods above will encounter different constraints in real-application. For instance, two parameters are required to be optimally tuned to achieve better performance for DANN.

The selection of *k* is very crucial to a successful *k*NN learning algorithm. Usually, the larger value of *k* is, the smoother the decision boundary of classifier becomes, while its efficiency is questionable. Contrastively, *k*NN is sensitive to noisy data if *k* is small. Domeniconi et al. (2002) argued that it is very difficult to predetermine the value of *k* especially when the instances are not uniformly distributed. To end this, an empirical strategy frequently used in practice is to determine a proper value of *k* by using cross validation (CV) or other heuristic techniques (Chen et al., 2007). However, the CV method requires much more computational cost. Although many endeavors have been attempted to determine *k* and several methods have shown their effectiveness, how to choose the optimal value of *k* for different applications at hand is still a challenge.

Given a training dataset, the larger its sample size *N* is, the higher quality of prediction results for new instances. However, a large sample size requires much more computational cost and memory requirement. On the other hand, more noise data may be contained within the dataset. Under this context, many instance reduction or condensed techniques are introduced to remove noises or anomalies from the training set. As a typical example, Fayed and Atiya (2009) recently proposed a template reduction algorithm called TRKNN. It exclusively divides instances into the condensed and removed sets by setting a cutoff distance in a chain of nearest neighbors. In addition, special data structures, such as Ball tree, k-d tree and RD-tree, have also been exploited to further enhance the efficiency of *k*NN (Bhatia and Vandana, 2010). For instance, Chen et al.

(2007) utilized a low bound tree (LB-tree) to accelerate the process of searching k nearest neighbors, where each leaf node represents a training instance and each internal node represents a mean point in a space of smaller dimension. Although this kind of methods can benefit the searching procedure, it needs much more time and space to construct data structures. Moreover, it is improper to those dynamic situations.

Recently, several works resort to the properties of reverse nearest neighbor and shell nearest neighbor to dispose of the problems of kNN. For instance, Zhang (2011) replaced the missing values of instance with its shell neighbors, rather than the nearest neighbors, while Gao et al. (2010) took use of mutual nearest neighbor (MNN) to improve the efficiency of query system for moving objects. In a similar vein, Ding and He (2004) employed k-mutual nearest neighbor consistency to improve the performance of K-means algorithm and discussed its applications in clustering and outlier detection tasks. Gowda and Krishna (1979) obtained a condensed training set based on the notation of mutual nearest neighborhood. Jin et al. (2006) applied a symmetric neighborhood relationship, which takes both the NNs and reverse NNs into consideration, into outlier detection. In our method, MNN will be integrated into both the process of classification learning and noise removal so as to improve the final performance of classifier.

2.2. Noises elimination

From the classification perspective, anomaly is often represented as the form of insignificant feature, missing value, redundant or inconsistent instance. The insignificant features can be figured out and removed by feature extraction (e.g., PCA and ICA) or feature selection (e.g., Relief, mRMR and DMIFS) technique (Liu et al., 2009), while missing values are often handled with statistical trick or discarded directly (Pyle, 1999; Garcia-Laencina et al., 2010). For inconsistent instances, there are three major strategies to tackle with them according to different purposes, i.e., sampling, instance selection and anomaly detection.

Data sampling picks a subset of instances out from database in virtue of statistical means to construct classification model. As a result, the size of database is reduced and the efficiency of learning algorithms can be strengthened (Lindenbaum et al., 2004). Currently, various sampling or bootstrapping techniques, such as selective sampling, active sampling, boosting sampling, random sampling, Gibbs sampling, Monte Carlo sampling and k-folds cross-validation, are available (Zollanvari et al., 2010). For instance, Yu (2011) adopted the selective sampling technique to reduce user interaction in learning ranking function so as to facilitate users formulating a preference query in the data retrieval system. To some extent, the sampling techniques are very effective for large-scale datasets. However, their efficiency is relatively poor.

The initial motivation of instance selection is to improve both the classification performance and robustness of final models and the training and response time of learning algorithms (Brighton and Mellish, 2002). Generally speaking, it can be fulfilled with two ways, i.e., selecting representative instances or removing noisy instances from dataset. As a typical illustration, Lee et al. (2010) grouped instances into several clusters by using a mixture model, and then took the cluster centers as representative points of the whole dataset to construct classification models.

For the noise elimination, it is often interconnected with kNN because of the nature of kNN. CNN, RNN, ENN, SNN and their variants are all classical examples of this kind (Olvera-Lopez et al., 2010). In these methods, an instance will be considered as noisy one if it is misclassified by its nearest neighbors. Recently, Marchiori (2010) developed a large margin based instance selection algorithm, where the effect of eliminating one instance on the hypothesis-margin of other instances is measured by the concept of class conditional nearest neighbor. Unlike traditional methods, here we adopt the mutual neighbors of an instance to achieve the purpose of data reduction. The underlying reason is that the information of mutual neighbors of instance is more creditable.

Anomaly detection is another hot research topic of handling abnormal data. It aims at identifying patterns in data that do not conform to a well defined notion of normal behavior (Chandola et al., 2009). Since its purpose is slightly distinct from the noise elimination and is not of our interest, here we will not discuss its related techniques. More information about anomaly detection can be consulted related literatures (see, e.g., Chandola et al., 2009). It is noticeable that kNN is also a popular detection method used in practice.

3. Background

Pattern classification mainly provides the solutions for the problem, i.e., how to predict the class label of unseen (i.e., unclassified) instance in terms of historical behaviors of data (Duda et al., 2001). Since it can be applied in many potential situations, classification has attracted a great deal of attention and been extensively studied in data mining community. Generally, it is formally described as follows.

Let x_i be an input instance represented as p-dimensional vector form, i.e., $x_i = (x_{i1},\ldots,x_{ip})$, $C = \{c_1,\ldots,c_m\}$ be a set of class labels. The class label of instance x_i belongs to one of the categories c_i, that is, there is a scalar function f, which assigns a class label, $c_i = f(x_i)$, to every instance. Given a dataset D consisting of n pairs of instance and label, i.e., $D = \{(x_1, c_1),\ldots,(x_n, c_n)\}$. The classification task is to determine the scalar function (i.e., hypothesis) f, on which the class labels of unclassified instances can be predicted precisely.

Starting from the seminal work of Cover and Hart (1967), 1-NN perhaps is the most straightforward learning algorithm. The center idea of 1-NN is to predict the class label of an unclassified instance by taking the labels of its closest previously observed instances into consideration. Specifically, 1-NN firstly stores all of these training instances in the memory. In predicting the class label of a new instance x, it tries to find the nearest neighbor x_0 of x and determines the class label c of x as $f(x_0)$, i.e., $c = f(x_0)$. From this definition, one may observed that 1-NN is a kind of instance-driven method and also known as instance-based learning (IBL).

Center of 1-NN is distance formula measuring the similar or intimate relationship between instances. Given two instances x and y, the Minkowski distance between them is,

$$d(x, y) = \sqrt[k]{\sum_{i=1\ldots p} (|x_i - y_i|)^k}$$

Note that the Minkowski distance is a general metric, whereas it is Manhattan distance as $k = 1$. In a similar vein, this formula is turned out as Euclidean or Chebychev distance as $k = 2$ or $k \to \propto$, respectively. In literature, the Euclidean distance is often taken as the similarity metric between instances in 1-NN. Based on this metric, 1-NN takes the instance x_i in D as the nearest neighbor of an unseen instance x, if,

$$d(x, x_i) = \arg\min d(x, x_j), j \in \{1, \ldots, n\}$$

Since 1-NN only considers the information of nearest neighbor in the prediction stage, it is sensitive to noise and its decision boundary is very sharp. To alleviate this problem, an effective solution is to extend the nearest neighbor to k nearest neighbors (kNN).

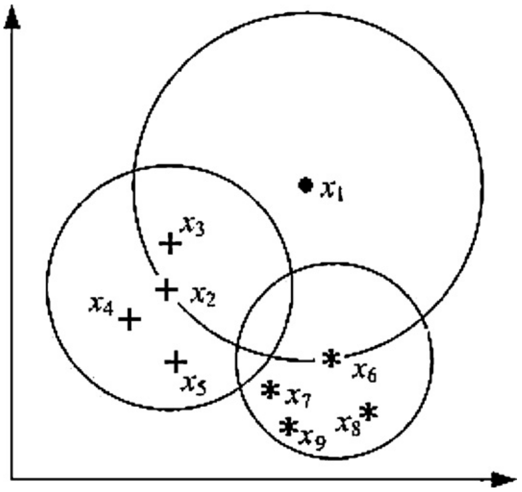

Fig. 1. Instance x_1 and its 3 nearest neighbors.

Let $N_k(x)$ be a set of k nearest neighbors of x, the label c of x is determined by its k nearest neighbors with a majority voting strategy, that is.

$$c = \arg\max_{c_i} \sum_{c_i \in C} \sum_{x_j \in N_k(x)} I(c_j = c_i)$$

In this definition, c_j is the class label of x_j and $I(.)$ is an indicate function, where $I(c_j = c_i) = 1$ as c_j is the same to c_i; otherwise $I(c_j = c_i) = 0$. For example, the nearest neighbors of x_1 in Fig. 1 are $\{x_2, x_3, x_6\}$ if $k = 3$, and its final prediction label is "+".

Despite the simplicity of kNN, this off-the-shelf method can still yield competitive results even compared to the sophisticated machine learning methods. Indeed, the error rate of kNN will be no greater than twice the Bayes error rate, and converges asymptotically to the optimal Bayes error rate as the size N of dataset approaching infinity and k increasing (Cover and Hart, 1967). Another advantage of kNN is that it is particularly effective when the probability distributions of the feature variables are not known. Many simulation experiments on a number of learning tasks have shown its competitive performance (Duda et al., 2001).

4. M*k*NN-based learning algorithm

4.1. Mutual nearest neighbor

As illustrated in Fig. 1, instances x_2, x_3 and x_6 are the nearest neighbors of x_1 as $k = 3$. Similarly, the nearest neighbors of x_2, x_3 and x_6 are $\{x_3, x_4, x_5\}$, $\{x_2, x_4, x_5\}$ and $\{x_7, x_8, x_9\}$, respectively. One may observe an interesting fact that x_1 is not one of the nearest neighbors of x_2, x_3 and x_6 as $k = 3$, notwithstanding it treats all of them as its nearest neighbors. The reason is that the instance x_1 is far from others. To some extent, x_2, x_3 and x_6 are the pseudo nearest neighbors of x_1.

From an optimal perspective, it is not a good idea to fix a globally optimal value of k for kNN in some real applications. As a matter of fact, the instance x_1 in Fig. 1 is often considered as an outlier in bank fraud, network analysis or intrusion detection, which will bring serious effects to economical investment or security. Analogically, x_1, x_2, x_4 and x_5 shown in Fig. 2 are usually regarded as noisy data in the task of classification or object recognition, where data is always expected to having good quality. This kind of anomalous data like x_1 should be discarded during the pre-processing phase, because the performance of classifiers will be significantly degraded if the anomalies have been used to build classifiers. To end this, a concept call mutual nearest neighbor (MNN) is introduced as follows.

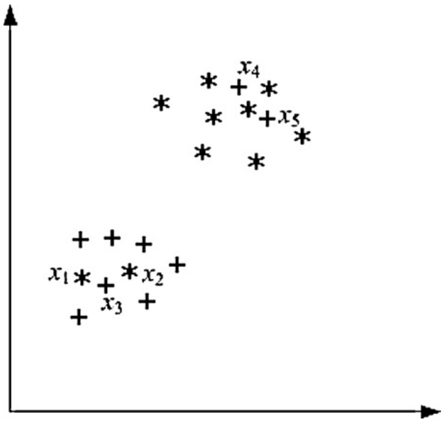

Fig. 2. Noisy instances x_1, x_2, x_4 and x_5.

Definition 1. Given a dataset D, two parameters k_1 and k_2, the mutual k nearest neighbors (MkNN) of an instance x is,

$$M_{k_1,k_2}(x) = \{x_i \in D | x_i \in N_{k_1}(x) \wedge x \in N_{k_2}(x_i)\},$$

where $N_k(x)$ is a set of k nearest neighbors (kNN) of x.

According to the above definition, an instance x_1 is a mutual nearest neighbor of x_2, if x_1 belongs to the k_1 nearest neighbors of another instance x_2, and x_2 is also one of the k_2 nearest neighbors of x_1 at the same time. For the sake of simplicity, hereafter k_1 and k_2 are assigned to the same value k and $M_{k1,k2}(x)$ is denoted as $M_k(x)$. For example, in Fig. 1, $M_3(x_2) = \{x_3, x_4, x_5\}$ and $M_3(x_8) = \{x_6, x_7\}$. Generally speaking, MNN has the following properties.

Lemma 1. For any value of k and instance x, $M_k(x) \subseteq N_k(x)$.

Proof *(.).* This lemma can be proved in a straightforward way. Let y is a MkNN of x, i.e., $y \in M_k(x)$. According to the definition of MkNN, we have the fact that y is also one of k nearest neighbors of x. On the other hand, for any nearest neighbor y of x, i.e., $y \in N_k(x)$, we have either $x \notin N_k(y)$ or $x \in N_k(y)$. The former indicates that $y \notin M_k(x)$ holds on. Therefore, $M_k(x) \subseteq N_k(x)$ □

This property shows a fact that not all nearest neighbors of x are also mutual neighbors. As an example, $M_3(x_1) = \varnothing$ and $N_3(x_1) = \{x_2, x_3, x_6\}$ in Fig. 1. For the instance x_2, its 3NN and M3NN are the same, i.e., $M_3(x_2) = N_3(x_2) = \{x_3, x_4, x_5\}$.

Lemma 2. Let x be an instance in a dataset D and $k_1 < k_2$, $M_{k_1}(x) \subseteq M_{k_2}(x)$.

Proof. Similarly, this lemma can be intuitively understood. Assume $y \in M_{k_1}(x)$, we know that $y \in N_{k_1}(x) \subseteq N_{k_2}(x)$ and $x \in N_{k_1}(y) \subseteq N_{k_2}(y)$. Thus, $y \in M_{k_2}(x)$. On the contrary, let $y \in M_{k_2}(x)$ and $k_1 < k \leq k_2$. If y is the k nearest neighbor of x, we have $y \notin N_{k_1}(x)$. In other words, y does not belong to MNN of x definitely, i.e., $y \notin M_{k_1}(x)$. □

From this lemma, one may notice that for the same instance x, its MkNNs $M_k(x)$ varies from the different value of k. If $k = 1$, the MkNN $M_k(x)$ of x is empty or its nearest neighbor. Contrastively, $M_k(x) = M_k(x)$ as $k = N - 1$, where N is the number of instances in the dataset D. In this case, the MkNN is degenerated into kNN. In Fig. 1, the MkNNs of x_1 are empty and $\{x_6, x_3\}$ as $k = 3$ and $k = 5$, respectively. Although $M_k(x)$ may be different, our empirical experiments show that the classification performance is good as $k = 3$.

Lemma 3. For any two instances x and y, if $x \in M_k(y)$, then $y \in M_k(x)$.

This lemma can also be straightforwardly proved according to the definition of MkNN. It implies that unlike traditional nearest neighbors, MkNN meets with symmetric property. This fact accords with our intuitive understanding about similarity or intimacy between instances. That is to say, if x is similar to y, then y should also be similar to x. However, kNN is an unsymmetrical one and can not represent this kind of meaning.

4.2. MkNN classification algorithm

It is well known that a major hurdle to the successful application of classification learning algorithms is the quality of data. High quality of data often leads to robust classifiers generated from it, while noisy data deteriorates the performance of final classification models. A common assumption to most learning techniques is that the data to process is complete, consistent and error free. However, most real-world situations are always complicated, so that noise information resulted from various aspects is pervasive. Therefore, it is necessary to require anomalies should be properly handled before the learning algorithms' application. Currently, how to improve the quality of data is still an open issue in data mining community.

As discussed above, given an instance x, the traditional kNN classifiers often identify its pseudo neighbors. If the false information of these neighbors is taken into account in predicting the class label of x, the final prediction result is questionable and untrustworthy in some way. Therefore, it is preferable to taking only those real neighbors into consideration for the lazy learning algorithm of kNN, and putting pseudo neighbors aside throughout the whole learning and prediction process.

The definition and property of MkNN tell us that the pseudo neighbors in kNN can be figured out to some extent if it has been considered. Intuitively, an instance x locates in the center of its neighbors when all of its neighbors are also taken it as one of their neighbors. In other words, x is not an isolated point (or noisy data) if the k nearest neighbors of x also take it as one of their k nearest neighbors. This releases a signal that x is not a noisy data if it has MkNNs and the larger $M_k(x)$ of x, the more center to its k nearest neighbors. On the contrary, x will be considered as an anomaly if it does not have any mutual k nearest neighbors, i.e., $M_k(x) = \Phi$. Under this context, the purpose of data reduction can be achieved in the light of the quantity of mutual nearest neighbors of instance. Algorithm 1 presents the details of our anomaly removal method.

Algorithm 1. Anomalies removal method

```
Input: A dataset $D$ and the number of nearest neighbors $k$;
Output: The reduction of $D$;
(1) For each instance $x$ in $D$, perform the next two steps;
(1.1) Obtain the M$k$NNs of $x$, i.e., $M_k(x)$;
(1.2) Remove $x$ from $D$ if $M_k(x) = \Phi$; Otherwise, retain it in $D$.
(2) Return $D$ as the final reduction.
```

From the properties of MkNN, one may observe that the prediction result of instance x is more creditable if its mutual nearest neighbors $M_k(x)$, rather than nearest neighbors, are involved. The underlying reason is that the pseudo neighbors have been excluded outside the learning phase, and the rest have more truthful information. This, however, is very like the relationship among friends in our real-world social community, where we always believe the words said by our closest friends and doubt others by unfamiliar ones. Thus, it is more desirable if the mutual neighbors of instances were taken as the prediction evidence than k nearest neighbors.

Based on the analysis above, we propose a new lazy classification algorithm called MkNNC. It is an extension of kNN by taking mutual nearest neighbor into account. The framework of MkNNC is elaborately given as follows (Algorithm 2).

MkNNC is simple and works in a straightforward way. At the beginning, it initializes the set of candidate class labels $C(x)$, which is used to predict the label of x. The second step aims to obtain k nearest neighbors of x by using traditional search technique. Subsequently, the algorithm (Step 3 to 5) identifies the mutual nearest neighbors of x and collects their corresponding label information. More specifically, for each nearest neighbor y of x, its k neighbors are also searched from D in order to determine whether y is also one of the k nearest neighbors of x at the same time. If it is true, y is one of mutual neighbors of x and its label will become a candidate class label of x. Finally, the class label $c(x)$ of x is determined on the basis of $C(x)$ by using the majority voting strategy.

Algorithm 2. MkNNC: MkNN based Classification Algorithm

```
Input: A dataset $D$, an instance $x$ and the number of nearest neighbors $k$;
Output: the prediction label $c(x)$ of $x$;
(1) Initialize relative parameters, e.g., $C(x) = \varnothing$;
(2) Obtain $k$ nearest neighbors $N_k(x)$ of $x$ from $D$;
(3) For each neighbors $y \in N_k(x)$ of $x$, perform the following two steps
(4) Obtain $k$ nearest neighbors $N_k(y)$ of $y$ from $D$;
(5) If $x$ is also one of $k$ nearest neighbors $N_k(y)$ of $y$, then
Add the label information of $y$ into $C(x)$;
(6) The class label $c(x)$ of $x$ is determined as follows:
$c(x) = \arg\max_c \sum_{c_i \in C(x)} I(c_y = c)$, where $I(.)$ is an indication function.
(7) Return $c(x)$ as the final result.
```

In comparison with the basic form of kNN method, the proposed algorithm adds several steps into the search process to distinguish pseudo neighbors from $N_k(x)$. To further its efficiency, several tricks are available, such as using special data structures or obtaining nearest neighbors for all instances in advance.

It is worthy to mention that most kNN algorithms take use of the Euclidean distance to measure the similarity between instances in literature. Without loss of generality, the Euclidean distance is also our first choice. However, this metric is only effective to numerical data. To scale the distance between mixed numerical and nominal values, here we adopt Heterogeneous Euclidean-Overlap Metric function (HEOM) (Wilson and Martinez, 1997), which is defined as:

$$\mathrm{HVDM}(x, y) = \sqrt{\sum_{i=1\ldots p} d_i^2(x_i, y_i)}$$

where

$$d_i(x, y) = \begin{cases} 1, & \text{if either } x \text{ or } y \text{ is unknown;} \\ x - y, & \text{if the } i\text{th feature is numerical;} \\ \mathrm{overlap}(x, y), & \text{if the } i\text{th feature is nominal.} \end{cases}$$

and overlap$(x, y) = 1$ if the value of x is the same to y; otherwise *overlap*$(x, y) = 0$.

5. Simulation experiments

In order to validate the effectiveness of the proposed learning method, three group experiments were carried out on 13 UCI benchmark datasets. This section firstly gives brief description of benchmark datasets and experimental settings. Then experimental results are presented and discussed.

5.1. Experimental datasets and settings

Classification performance is an effective and straight guide line to validate learning algorithms. There is no exception to our experiment evaluation. To serve for this purpose, thirteen benchmark datasets with different type and size were chosen in our simulation experiments. All of them can be freely accessed at the UCI Machine Learning Repository (Asuncion and Newman, 2007). These datasets are frequently used to verify the performance of classifiers. Table 1 summarizes some general description information about these datasets.

The second column in Table 1 is the dataset names. The quantities of instances, features and class labels for each dataset are

Table 1
General information of datasets used in experiments.

No	Datasets	Instances	Features	Classes
1	Artificial	5109	7	10
2	Auto-mpg	398	8	3
3	Musk Clean1	476	167	2
4	Echocardiogram	132	8	2
5	Glass	214	10	7
6	Ionosphere	351	35	2
7	Letter	20,000	17	26
8	Machine	209	8	7
9	Page-blocks	5473	10	5
10	Segment	2310	10	7
11	Sonar	208	61	2
12	Vehicle	846	18	4
13	Wine	178	14	3

presented in the last three columns. For instance, the 7th line denotes the "*Letter*" dataset represented with 17 features has 20,000 instances, which are tagged with 26 different labels. Note that these benchmark datasets have different number of class labels and differ greatly in the sample size (range from 178 to 20,000) and the number of attributes (from 8 to 167). Hence, they can provide a comprehensive test in validating learning algorithms under different conditions.

The experiments were grouped three categories. Since our algorithm is an extension of kNN, we took the kNN classifier as the baseline in the first group, where both MkNNC and kNN were implemented with VC++ 6.0. The second group made a comparison of MkNNC with three popular classifiers, i.e., NBC (Naive Bayes Classifier), C4.5 and RIPPER (Repeated Incremental Pruning to Produce Error Reduction) (Cohen, 1995), in the aspect of classification performance. The reason of choosing them is that these classifiers represent quite different learning types and are often used in literature because of their relatively high efficiency and performance. Furthermore, they are all off-the-shelf in the WEKA[1] (Waikato environment for knowledge analysis) (Witten and Frank, 2005) software toolkit. In this group, the datasets were firstly reduced by MNN, and then fed into Weka. The purpose of the last group is to validate the effects of different k values on MkNNC.

To achieve impartial results, ten 10-fold cross-validations had been adopted for each algorithm-dataset combination in verifying classification capability. That is to say, for each dataset we run every classification algorithm ten times. At each time, a 10-fold cross-validation was used to obtain classification accuracy. The average value over these ten times was the final result.

5.2. Experimental results

5.2.1. Experimental comparison between MkNNC and kNN

As mentioned in the previous section, our proposed algorithm is an extension of kNN, where the prediction decision of MkNNC is made by using mutual nearest neighbors. Thus, we took the kNN classifier as the baseline and made a comparison between kNN and MkNNC. In order to demonstrate the effectiveness of noise removal, kNN was also conducted over the reduction of datasets (as shown kNN* in Table 2). The experimental results on datasets are given in Table 2, where k was assigned to three, five and seven, respectively.

From the results in Table 2, one can observe that MkNNC can significantly improve the performance of kNN on most datasets. For example, in the case of $k = 3$, the performance of MkNNC is distinctly superior to kNN. There are only one over thirteen datasets (i.e., *Echocardiogram*) as $k = 5$, on which MkNNC performs worse than kNN. Indeed, the instances in *Echocardiogram* are sparsely scattered and their distances are relatively large. As a result, some instances had been taken as anomalies. The similar situation can be found as $k = 7$.

In Section 4, we declared that noisy data can be distinguished from the normal ones in terms of the number of mutual nearest neighbors. To demonstrate that, additional experiments were carried out, where noisy data was firstly eliminated by using mutual neighbors and then the kNN classifier was performed on these new datasets. The experimental results are also presented in Table 2 as the kNN* columns. The results tell us that the kNN classifier has higher classification capability on new datasets than the original ones. That is to say, our anomaly removal method by mutual neighbors is an effective solution. It is capable of strengthening the generalization capability of kNN to some extent.

One may observe that the performance of kNN* was not the same to MkNNC. It is normal, for in the MkNNC classifier, instances will not be eliminated from the original ones. However, the instances which do not belong to the mutual ones in kNN* were removed and then the rest were fed into kNN. This means that information was lost in kNN* and it will be increasing as k decreasing. As a result, the performance of kNN* is lower than MkNNC.

5.2.2. Experimental comparison between MkNNC and other classifiers

Apart from the kNN classifier, three popular classifiers, i.e., C4.5, NBC and RIPPER, were also adopted to be made a comparison with MkNNC. The comparison of classification performance among these four classifiers is described as Table 3, where $k = 3$ and other parameters in these classifiers were set to default values as recommended in the Weka toolkit.

As shown in Table 3, there are nine over thirteen cases, where MkNNC significantly outperforms other three classifiers. However, MkNNC is slightly inferior to C4.5 on the *Artificial*, *Auto-mpg* and *Vehicle* datasets, and works better than the rest classifiers. For the *Echocardiogram* dataset, the performance of MkNNC is the worst one. As discussed in the previous subsection, some normal instances had been eliminated as being taken as anomalies and this situation was worsened by the fact that the quantity of instances in the dataset is relatively less.

5.2.3. Experimental results of MNN with different values of k

The value of k is very important to the kNN classifiers, because it directly affects the final performance and generalization capability. There is no exception to the MkNNC classifier. In a similar vein, the choice of k also brings great effects to MkNNC. To verify this, a group of experiments was conducted on datasets by choosing different values of k for MkNNC. Specifically, the MkNNC classifier in this group experiment was carried out on the datasets with $k = 3$, 5 and 7, respectively. The comparison of classification performance of MkNNC with different values of k is illustrated as a bar graph (see, Fig. 3), where the baseline is the accurate rate of MkNNC with $k = 7$.

In Fig. 3, A bar above the zero line implies that the performance of classifier with corresponding value of k is better than the one with $k = 7$; Otherwise the corresponding accuracy is worse than the one with $k = 7$. For example, on the "*Musk Clean1*″ dataset (i.e., the 3rd dataset), the performance of MkNNC with $k = 3$ was 91.67%, which was higher 2.31% than $k = 7$. Contrastively, the performance of MkNNC with $k = 5$ was lower 0.11% in comparing with $k = 7$ under the same condition.

From this Figure, we know that the performance of MkNNC is the best when $k = 3$, and along with increasing of k, the classification capability incline to be stable. It is reasonable and conforms to our intuitive meanings because the larger the value of k is, the more mutual neighbors of instances. Indeed, given a dataset, all instances become mutual neighbors when k is equal to the sample size of

[1] Weka is freely available at: http://www.cs.waikato.ac.nz/∼ml.

Table 2
Classification accuracy (%) of M*k*NNC and *k*NN on benchmark datasets, where *k*NN* denotes that *k*NN conducted on datasets without noisy data.

	k=3			*k*=5			*k*=7		
	M*k*NNC	*k*NN	*k*NN*	M*k*NNC	*k*NN	*k*NN*	M*k*NNC	*k*NN	*k*NN*
1	**55.25**	53.85	52.89	**55.00**	53.28	53.00	**55.06**	52.30	52.08
2	**73.58**	67.09	68.75	**71.24**	68.59	69.39	**71.09**	68.59	69.53
3	**91.67**	84.24	86.27	**89.25**	83.82	84.87	**89.36**	82.56	82.55
4	**37.14**	36.36	35.24	36.59	**37.12**	35.77	36.00	**40.15**	36.00
5	**70.39**	66.36	65.92	**67.71**	63.08	59.38	**67.50**	61.68	59.50
6	**97.67**	86.04	93.60	**95.07**	84.90	90.64	**95.87**	83.76	88.53
7	**94.94**	93.53	94.17	**94.64**	93.07	93.38	**94.38**	92.65	92.80
8	**89.07**	87.56	88.52	**87.88**	88.04	88.38	**87.56**	84.69	84.08
9	**96.97**	95.96	96.50	**96.44**	95.93	96.18	**96.37**	95.58	95.76
10	**98.05**	96.10	96.79	**97.67**	95.76	96.05	**97.70**	94.94	95.10
11	**94.01**	83.65	90.00	**89.83**	83.17	85.31	**88.30**	81.73	81.38
12	**71.56**	71.16	70.92	**70.25**	70.09	69.89	70.08	**70.69**	70.44
13	**99.31**	96.07	**99.31**	**98.75**	94.94	97.50	**98.00**	96.07	97.00
	82.26	78.31	79.91	**80.79**	77.83	78.44	**80.56**	77.34	77.29

Bold values indicate the highest one under the same conditions.

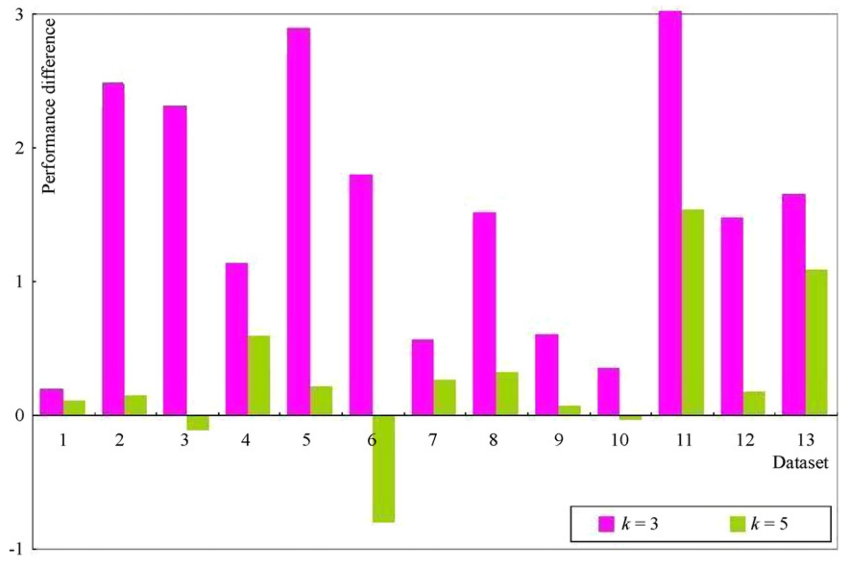

Fig. 3. Performance comparison of M*k*NNC with different values of *k*.

Table 3
Classification accuracy (%) of NBC, C4.5, RIPPER and M*k*NNC on benchmark datasets, where *k* = 3.

	NBC	C45	RIPPER	M*k*NNC
1	29.89	**59.77**	49.68	55.25
2	67.6	**78.40**	77.97	73.58
3	73.6	83.28	75.76	**91.67**
4	**74.52**	72.98	72.26	37.14
5	49.64	68.80	67.73	**70.39**
6	82.63	89.56	88.52	**97.67**
7	64.07	88.01	85.32	**94.94**
8	80.17	88.67	84.35	**89.07**
9	89.86	**96.97**	96.88	**96.97**
10	80.26	96.75	95.47	**98.05**
11	67.35	73.38	74.41	**94.01**
12	45.07	**71.75**	68.79	71.56
13	97.18	93.83	92.49	**99.31**

Bold values indicate the highest one under the same conditions.

dataset. In this case, M*k*NNC is degenerated to the traditional *k*NN classifier.

According to the experimental results above, one may observe that the performance of our proposed method is superior to other popular ones roundly in most cases. However, there are still several situations where M*k*NNC is relatively poor. The underlying reason perhaps is that the instances in those datasets are sparsely scattered and the quantity of training data is not enough. As a matter of fact, for the *Echocardiogram* dataset, the available training data was less after the noisy data had been removed. This indicates that M*k*NNC is preferred in the case that the dataset has enough training instances and they are tightly scattered at the same time.

6. Conclusions

In this paper, we proposed a new noisy removal and classification algorithm called M*k*NNC. The center idea of M*k*NNC is that it eliminates anomalies to improve the quality of data by virtue of the notion of mutual nearest neighbors. Besides, it also predicts the class label of new instance on the basis of the mutual neighbors. The advantage underlying M*k*NNC is that it not only can distinguish noisy data from dataset, but also strengthen the credibility or truthfulness of the final prediction result. More specifically, after obtaining the nearest neighbors of unseen instance, M*k*NNC further identifies its mutual neighbors and utilizes them to determine the class label. An extensive comparative experimental analysis shows that the classification performance achieved by our proposed method is better than the traditional one.

Acknowledgments

This paper is supported in part by the Australian Research Council (ARC) under large grant DP0985456, the Nature Science Foundation (NSF) of China under grants 61100119 and 61170131, the China "1000-Plan" Distinguished Professorship, the China 863 Program under grant XXX, Opening Fund of Key Discipline of Computer Software and Theory of Zhejiang Province at ZJNU (ZSDZZZZXK05), Opening Fund of Key Laboratory of Symbol Computation and Knowledge Engineering of Ministry of Education (93K-17-2010-K02), the Jiangsu Provincial Key Laboratory of E-business at the Nanjing University of Finance and Economics, and the Guangxi NSF (Key) grants.

References

Asuncion, A., Newman, D.J., 2007. UCI Repository of Ma-chine Learning Databases, Department of Information and Computer Science. University of California, Irvine, Available at: http://www.ics.uci.edu/~mlearn/MLRepository.html.
Bhatia, N., Vandana, 2010. Survey of nearest neighbor techniques. International Journal of Computer Science and Information Security 8 (2), 302–305 arxiv:1007.0085v1.
Brighton, H., Mellish, C., 2002. Advances in instance selection for instance-based learning algorithms. Data Mining and Knowledge Discovery 6, 153–172.
Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: a survey. ACM Computing Surveys 41 (3), 1–58.
Chapman, A.D., (2005). Principles and Methods of Data Cleaning-Primary Species and Species-Occurrence Data, version 1.0. Report for the Global Biodiversity Information Facility, Copenhagen.
Chen, Y.-S., Hung, Y.-P., Yen, T.-F., Fuh, C.-S., 2007. Fast and versatile algorithm for nearest neighbor search based on a lower bound tree. Pattern Recognition 40, 360–375.
Cohen, W.W., 1995. Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning, pp. 115–123.
Cover, T.M., Hart, P.E., 1967. Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13, 21–27.
Davidson, I., Tayi, G., 2009. Data preparation using data quality matrices for classification mining. European Journal of Operational Research 197, 764–772.
Ding, C.H.Q., He, X., 2004. K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization. In: Proceedings of ACM Symposium on Applied Computing (SAC), pp. 584–589.
Domeniconi, C., Peng, J., Gunopulos, D., 2002. Locally adaptive metric nearest-neighbor classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (9), 1281–1285.
Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification, 2nd ed. Wiley, New York.
Fayed, H.A., Atiya, A.F., 2009. A novel template reduction approach for the k-nearest neighbor method. IEEE Transactions on Neural Networks 20 (5), 890–896.
Gao, Y., Zheng, B., Chen, G., Li, Q., Chen, C., Chen, G., 2010. Efficient mutual nearest neighbor query processing for moving object trajectories. Information Sciences 180, 2170–2195.
Garcia-Laencina, P.J., Sancho-Gomez, J.L., Figueiras-Vidal, A.R., 2010. Pattern classification with missing data: a review. Neural Computing & Applications 19, 263–282.
Gowda, K.C., Krishna, G., 1979. The condensed nearest neighbor rule using the concept of mutual nearest neighborhood. IEEE Transactions on Information Theory 25 (4), 488–490.
Hulse, J.V., Khoshgoftaar, T., 2009. Knowledge discovery from imbalanced and noisy data. Data and Knowledge Engineering 68, 1513–1542.
Jahromi, M.Z., Parvinnia, E., John, R., 2009. A method of learning weighted similarity function to improve the performance of nearest neighbor. Information Sciences 179, 2964–2973.
Jin, W., Tung, A.K.H., Han, J., Wang, W., 2006. Ranking outliers using symmetric neighborhood relationship. In: Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 577–593.
Latourrette, M., 2000. Toward an explanatory similarity measure for nearest-neighbor classification. In: Proc. of the 11th European Conference on Machine Learning, London, UK, pp. 238–245.
Lee, H.K.H., Taddy, M., Gray, G.A., 2010. Selection of a representative sample. Journal of Classification 27, 41–53.
Lindenbaum, M., Markovitch, S., Rusakov, D., 2004. Selective sampling for nearest neighbor classifiers. Machine Learning 54, 125–152.
Liu, H., Liu, L., Zhang, H., 2010. Ensemble gene selection for cancer classification. Pattern Recognition 43 (8), 2763–2772.
Liu, H., Sun, J., Liu, L., Zhang, H., 2009. Feature selection with dynamic mutual information. Pattern Recognition 42, 1330–1339.
Marchiori, E., 2010. Class conditional nearest neighbor for large margin instance selection. IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2), 364–370.
Olvera-Lopez, J.A., Carrasco-Ochoa, J.A., Martinez-Trinidad, J.F., Kittler, J., 2010. A review of instance selection methods. Artificial Intelligence Review 34, 133–143.
Pyle, D., 1999. Data Preparation for Data Mining. Morgan Kaufmann Publishers, San Francisco, CA, USA.
Song, Y., Huang, J., Zhou, D., Zha, H., Giles, C.L., 2007. IKNN: informative k-nearest neighbor pattern classification. In: PKDD 2007, LNAI 4702, pp. 248–264.
Toyama, J., Kudo, M., Imai, H., 2010. Probably correct k-nearest neighbor search in high dimensions. Pattern Recognition 43, 1361–1372.
Wilson, D.R., Martinez, T.R., 1997. Improved heterogeneous distance functions. Journal of Artificial Intelligence Research 6, 1–34.
Witten, I.H., Frank, E., 2005. Data Mining-Practical Machine Learning Tools and Techniques with JAVA Implementations, 2nd ed. Morgan Kaufmann Publishers, Los Altos.
Wu, X., Kumar, V., Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLach-lan, G., Ng, A., Liu, B., Yu, P., Zhou, Z., Steinbach, M., Hand, D., Stein-berg, D., 2008. Top 10 algorithms in data mining. Knowledge and Information Systems 14, 1–37.
Yang, Q., Wu, X., 2006. 10 challenging problems in data mining research. International Journal of Information Technology and Decision Making 5 (4), 597–604.
Yu, H., 2011. Selective sampling techniques for feedback-based data retrieval. Data Mining and Knowledge Discovery 22 (1), 1–30.
Zhang, Y., Wu, X., 2010. Integrating induction and deduction for noisy data mining. Information Sciences 180 (14), 2663–2673.
Zhang, S., 2011. Shell-neighbor method and its application in missing data imputation. Applied Intelligence 35 (1), 123–133.
Zollanvari, A., Braga-Neto, U.M., Dougherty, E.R., 2010. Joint sampling distribution between actual and estimated classification errors for linear discriminant analysis. IEEE Transactions on Information Theory 56 (2), 784–804.

Huawen Liu received his PhD and MS degrees in Computer Science from Jilin University, China, in 2010 and 2007, respectively. Then he joined Department of Computer Science of Jilin University as a Lecturer in the same year. Currently, he has wide research interests, mainly including feature selection, machine learning, pattern recognition and data mining.

Shichao Zhang is a China "1000-Plan" National Distinguished Professor and a Vice President of the Guangxi Normal University, China. He holds a PhD degree from the CIAE, China. His research interests include machine learning and information quality. He has published about 60 international journal papers and over 60 international conference papers. He is a CI for 12 competitive nation-level grants, including China NSF, China 863 Program, China 973 Program, and Australia large ARC. He is a senior member of the IEEE, a member of the ACM; and served/ing as an associate editor for IEEE Transactions on Knowledge and Data Engineering, Knowledge and Information Systems, and IEEE Intelligent Informatics Bulletin.