Natural Neighborhood-Based Classification Algorithm Without Parameter $k$

Ji Feng*, Yan Wei, and Qingsheng Zhu

Abstract: Various kinds of $k$-Nearest Neighbor (KNN) based classification methods are the bases of many well-established and high-performance pattern recognition techniques. However, such methods are vulnerable to parameter choice. Essentially, the challenge is to detect the neighborhood of various datasets while ignoring the data characteristics. This article introduces a new supervised classification algorithm, Natural Neighborhood Based Classification Algorithm (NNBCA). Findings indicate that this new algorithm provides a good classification result without artificially selecting the neighborhood parameter. Unlike the original KNN-based method, which needs a prior $k$, NNBCA predicts different $k$ for different samples. Therefore, NNBCA is able to learn more from flexible neighbor information both in the training and testing stages. Thus, NNBCA provides a better classification result than other methods.

Key words: nearest neighbor; classification; self-adaptive neighborhood

1 Introduction

With the continuous expansion of data availability in many areas of engineering and science, identifying patterns from vast amounts of data and identifying members of a predefined class, which is called classification, have become critical tasks. Therefore, classification is a fundamental problem, especially in pattern recognition and data mining. Several effective algorithms have been successfully applied in many real-world applications. In setting a classification, two kinds of classifiers exist, namely, parametric and nonparametric classifiers. With the advent of the Big Data era\cite{1,2}, nonparametric classifiers have received particular attention because the data distributions of many classification problems are either unknown or difficult to obtain in practice. $k$-Nearest Neighbor (KNN) classifiers, as a typical representative of nonparametric classifiers, are basic classifiers that are used to classify a query object into the category as its nearest example\cite{3,4}. Additionally, analogous to KNN, Reverse $k$-Nearest Neighbor (RkNN) and the Mutual $k$-Nearest Neighbor (MkNN) capture the connectivity of adjacent regions from other perspectives, and most of the NN classifiers can also be attributed to the above three neighbor methods.

The NN classifier has attracted considerable research effort\cite{5} and is applied in various domains\cite{6-8}. Its application extends to considerable applications in many different disciplines\cite{9-14}. However, two main problems still exist, thereby limiting the efficiency of the NN classifier.

(1) The efficiency of NN classification heavily depends on the type of distance measure, especially in a large-scale and high-dimensional database. In some applications, the data structure is so complicated that the corresponding distance measure is computationally expensive\cite{15,16}. The NN classifier simply compares the data structure with all other examples in the database for each query. Such comparison may be impractical...
with a large database and frequent queries.

(2) Traditional KNN adopts a fixed $k$ for all query samples regardless of their geometric location and related specialties. Furthermore, those $k$-nearest neighbors may not distribute symmetrically around the query sample if the neighborhood in the training set is not spatially homogeneous. The geometrical placement might be more important than the actual distance to depict a query sample’s neighborhood.

Many other algorithms have been proposed to improve neighborhood based classifiers. Recently, on the basis of the mutual $k$-nearest neighborhood method, Tang and He\cite{17} proposed an Extended Nearest Neighbor (ENN) method for classification, which uses the two-way communication style. Unlike the classic KNN rule which considers only the nearest neighbors of a test sample to make a classification decision, ENN method considers not only who the nearest neighbors of the test sample are but also who considers the test sample their nearest neighbors.

This two-way communication style is both an advantage and disadvantage of the ENN method. The advantage is that the classification decision of ENN method depends on all variable training data. The disadvantage is that the problem of parameter selection such as the selection of $k$ in KNN still exists. ENN has a parameter of $k$, on the basis of which the size and shape of the graph can be modified. This feature makes additional optimization possible; however, the problem of parameter selection continues to persist.

To overcome the above limitations of classification algorithms, we propose a new method called the Natural Neighborhood-Based Classification Algorithm (NNBCA). With the help of our previously proposed Natural Neighbor (NaN) method, the solution of choosing the optimum value of $k$ both in the training and testing stages is presented in this paper.

The rest of the paper is organized as follows: In the next section, we will survey related works. We will present our proposed method in detail in Section 3, followed by a series of experiments in Section 4. Section 5 concludes this paper with discussions on future works.

2 Related Work

2.1 KNN method

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

Generally, many methods adopt locally adaptive distances rather than the Euclidean distance in the traditional KNN to address the global optimality problem. Typical examples of the KNN method include the discriminant adaptive metric in DANN, the adaptive distance in ADAMENN, the weight adjusted metric in WAKNN, the Mahalanobis distance in LMNN, and the informative metric in IKNN\cite{18}. Various distance metrics are estimated with weighted similarity between instances. For example, Jahromi et al.\cite{19} assigned each instance with a different weight in terms of the leave-one-out classification performance, where the instances with zero weight are virtually removed from the training set. The probability model was also introduced to enhance the performance of KNN. Toyama et al.\cite{20} proposed a probably correct approach of KNN on the basis of the marginal distribution of $k$-th nearest neighbors in low dimensions. Notably, most of these methods encounter unique constraints in real-world applications. For instance, two parameters need to be optimally tuned to achieve improved performance for DANN.

The selection of $k$ is crucial to a successful KNN learning algorithm. Usually, a large value of $k$ corresponds to a smooth decision boundary of the classifier, however, its efficiency is questionable. By contrast, KNN is sensitive to noisy data if $k$ is small. Domeniconi et al.\cite{21} argued that predetermining the value of $k$ is difficult especially when the instances are uniformly distributed. To address this issue, an empirical strategy that is frequently used in practice is to determine a proper value of $k$ by using Cross Validation (CV) or other heuristic techniques\cite{22}. However, the CV method requires considerable computational cost. Although many endeavors have attempted to determine $k$ and several methods have been effective, determining a method of selecting the optimal value of $k$ for different applications remains a challenge.

Given a training dataset, a larger sample size $N$ corresponds to a high quality of prediction results for different instances. However, a large sample size requires a high computational cost and memory requirement. Moreover, more noisy data may be contained within the dataset. Under this context, many instance reduction or condensing techniques are established to remove noises or anomalies from the
training set. As a typical example, Fayed and Atiya\textsuperscript{[23]} recently proposed a template reduction algorithm called TRKNN, which exclusively divides instances into condensed and removed sets by setting a cutoff distance in a chain of nearest neighbors. Special data structures, such as b-tree, k-d tree, and RD-tree, have also been exploited to further enhance the efficiency of KNN\textsuperscript{[24]}. For instance, Chen et al.\textsuperscript{[22]} utilized a low bound 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 method can benefit the searching procedure, it requires considerable time and space to construct data structures. Moreover, this method is unsuitable for dynamic situations.

Recently, several works used the properties of reverse nearest neighbor and shell nearest neighbor to address the problems of KNN. For instance, Zhang\textsuperscript{[25]} replaced the missing values of an instance with its shell neighbors, rather than the nearest neighbors, while Gao et al.\textsuperscript{[26]} used mutual nearest neighbor to improve the efficiency of the query system in moving objects. In a similar vein, Ding and He\textsuperscript{[27]} employed $k$-mutual nearest neighbor consistency to improve the performance of the K-means algorithm and discussed its applications in clustering and outlier detection tasks. Gowda and Krishna\textsuperscript{[28]} obtained a condensed training set based on the notation of mutual nearest neighborhood. Jin et al.\textsuperscript{[29]} applied a symmetric neighborhood relationship, which considers both the NNs and reverse NNs, into outlier detection.

### 2.2 Natural neighbor method

Recently, our team presented a new parameter-free nearest neighbor method called NaN\textsuperscript{[30]}. NaN is inspired by friendships in human society and could be considered to belong to the category of scale free nearest neighbor method. Moreover, it is effective in outlier detection\textsuperscript{[31]}. NaN makes three key contributions to the current state-of-the-art:

1. NaN method can create an applicable neighborhood graph based on the local characteristics of various data sets. This neighborhood graph can identify the basic clusters in the data set, especially manifold clusters and noises.

2. This method can provide a numeric result named NaN Eigenvalue (NaNE) to replace the parameter $k$ in the traditional KNN method, and the number of NaNE is dynamically chosen for different data sets.

(3) The NaN number of each point is flexible, and this value is a dynamic number ranging from 0 to NaNE. The center point of the cluster has more neighbors, and the neighbor number of noise is equal to 0.

The NaN method is defined as follows: Given a set of data points $x_1, x_2, x_3, \cdots, x_n$ and some notion of similarity $s_{ij}$ between all pairs of data points $x_i$ and $x_j$, the goal is to find the natural neighbors of the points in the data set. The notion of similarity can be given as prior knowledge, or it can be calculated and stored in a distance matrix. One of the most popular choices to measure this distance is known as Euclidean distance.

In the following, we assume that $X$ is a set of points, $s_{ij}$ is the similarity between two points $x_i$ and $x_j$. With the help of a comparison of the similarity, let $\text{findKNN}(x_i, r)$ denote the function of KNN searching, which returns the $r$-th nearest neighbor of point $x_i$. $\text{KNN}_r(x_i)$ is a subset of $X$ and is defined as follows:

$$\text{KNN}_r(x_i) = \bigcup_{n=1}^{r} \{\text{findKNN}(x_i, n)\} \quad (1)$$

**Definition 1 (Stable searching state)** The NaN searching process achieves the stable searching state only if

\[(\forall x_i)(\exists x_j)(r \in \mathbb{N}) \land (x_i \neq x_j) \rightarrow (x_i \in \text{KNN}_r(x_j)) \land (x_j \in \text{KNN}_r(x_i))\] (2)

when the searching round $r$ increases from 1 to $\lambda$.

The RkNN method is used to restrict the KNN searching progress, and the value of the searching round is close to the optimum value of the KNN method. But if there is a noise point at infinity, the noise can significantly enlarge the value of the searching round. Thus, stable searching state focuses on the normal points, and the definition of noise is presented after the definition of NaN.

**Definition 2 (NaN)** When the algorithm reaches the stable searching state, NaN $\lambda$ is equal to the searching round $r$:

$$\lambda \triangleq \inf_{r \in \mathbb{N}} \{r | (\forall x_i)(\exists x_j)(r \in \mathbb{N}) \land (x_i \neq x_j) \rightarrow (x_i \in \text{KNN}_r(x_j)) \land (x_j \in \text{KNN}_r(x_i))\}$$ (3)

**Definition 3 (NaN)** The natural neighbor of $x_j$ is defined as follows:

$$x_j \in \text{NN}(x_i) \iff (x_j \in \text{KNN}_k(x_i)) \land (x_j \in \text{KNN}_k(x_i))$$ (4)

A significant difference between the traditional neighbor and the natural neighbor is the number of the neighbor. Thus, we may extract more interesting information from a data set by using this natural way.
Definition 4 (Noise) A data point is noise only if all points except noise reach the stable searching state and it cannot have only a natural neighbor after \( \sqrt{x} \) more rounds.

Definition 5 (Natural Neighborhood Graph, NaNG) When the algorithm reaches the stable searching state, the neighborhood structure of the data set constitutes an NaNG. Each vertex \( v_i \) in this graph represents a data point \( x_i \). Two vertices \( x_i \) and \( x_j \) are connected if the data point \( x_i \) is a natural neighbor of \( x_j \).

The NaNG is a good way of representing the relationship of the data set. In addition, the graph can have different forms depending on its applications, e.g., weighted natural neighborhood graph in NaNE calculation and maximum neighborhood graph in outlier detection.

Algorithm 1 is the description of the NaN algorithm.

In contrast to the traditional neighbor methods, NaN is parameter-free, and it finds multiple-valued neighbors of each data point by considering the characteristics of the data set. This method can improve the performance in handling noise and manifold data, and it can also be used in other areas such as clustering, image segmentation, and face recognition, instead of the KNN method and achieve better performance. Figure 1 shows the basic framework of the NaN method.

3 Proposed Approach: NNBCA

3.1 Concepts of NNBCA

We assume \( X_{\text{training}} \) is a set of points \( x_1, x_2, x_3, \ldots, x_m \). Then, we define the generalized class-wise statistic \( T_i \), for class \( i \) as follows:

Definition 6 (Generalized class-wise statistic)

\[
T_i = \frac{1}{n_i} \sum_{x \in X_i} \left\{ \sum_{r=1}^{\text{num}(x)} I_r(x, X_{\text{training}}) / \text{num}(x) \right\} \quad (5)
\]

where \( X_i \) denotes the points in class \( i \), \( x \) denotes one single sample in \( X_{\text{training}} \), \( n_i \) is the number of samples in \( X_i \), and \( \text{num}(x) \) is the neighborhood parameter, which is equal to the number of point \( x \)'s natural neighbors.

\( I_r(x, X_{\text{training}}) \) indicates whether both the point \( x \) and its natural neighbor belongs to the same class, which is defined as follows:

Definition 7

\[
I_r(x, X_{\text{training}}) = \begin{cases} 
1, & \text{if } x \in X_i \text{ and } \text{NaN}_{r}(x, X) \in X_i; \\
0, & \text{otherwise}
\end{cases} \quad (6)
\]

where \( \text{NaN}_{r}(x, X_{\text{training}}) \) denotes the \( r \)-th natural neighbor of \( x \) in \( X \).

The generalized class-wise statistic \( T_i \) has \( 0 \leq T_i \leq 1 \) with \( T_i = 1 \) when all the natural neighbors of class \( i \) points are also from the same class \( i \), and \( T_i = 0 \) when all the natural neighbors of class \( i \) points are also from other classes. On the basis of this discussion, we can use \( T_i \) to represent the data distribution across multiple classes. Therefore, we use the concept of intra-class coherence, which is defined as follows:
**Definition 8**

\[ \Theta = \sum_{i=1}^{c} T_i \]  

(7)

where \( c \) denotes the number of classes in \( X \).

### 3.2 Construction of an efficient classification algorithm with natural neighborhood

Our goal is to construct an efficient classification algorithm for nearest neighbor classification, with the help of NaN methods. This NNBCA consists of the training stage and the testing stage, and the steps are explained below.

#### 3.2.1 Training stage

In the training stage, the algorithm deals with training data, and the NaN method uses parameter \( k \) in the generalized class-wise statistic calculation step. NaN can then be used to calculate the distance and natural neighbor efficiently for a given test sample. Therefore, in the generalized class-wise statistic calculation step, the algorithm stores NaN and transfers it to the next stage. Algorithm 2 is the description of the NNBCA classification algorithm in the training stage.

The time complexity of Algorithm 2 is \( O(m \cdot \log(m)) \), \( m \) is the size of the training data set. This algorithm first creates a \( k-d \) tree of the data set, and the time complexity of this step is \( O(m \cdot \log(m)) \). Afterwards, the complexity of \( \text{NaN}_n(x_i) \) calculation is \( O(l \cdot m \cdot \log(m)) \), same as the NaNG calculation. Here \( l \) is the value of NaN, and the range of \( \lambda \) must be \( 2 \leq \lambda < m \), this value is generally 6 or 7. For high-dimensional or irregular data sets, \( \lambda \) will be more than 20 but less than 30. Thus in these steps, the complexity is \( O(m \cdot \log(m)) \).

Finally, we perform generalized class-wise statistic computation with only computational complexity of only \( O(m) \).

**Algorithm 2** NNBCA: Training stage

**Require:** training data set \( X_{\text{training}} \); 
**Ensure:** 
\( \text{NaN}_n, G = \langle V, E > \); generalized class-wise statistic for each class, \( T = \{T_1, T_2, \ldots, T_c\} \);

1. Create a \( k-d \) tree \( \text{Tree} \) from training data set \( X_{\text{training}} \); 
2. \( \forall x_i \in X, \text{NaN}_n(x_i) = 0 \); 
3. Calculate \( \text{NaN}_n(x_i) \) of training data set \( X_{\text{training}} \) by using \( k-d \) tree \( \text{Tree} \); 
4. Build NaNG \( G \) by using \( k-d \) tree \( \text{Tree} \); 
5. for all class of training data set \( X_{\text{training}} \) do 
6. Calculate the generalized class-wise statistic \( T_i \) by using NaNG \( G \) and natural neighbor number \( \text{NaN}_n(x_i) \); 
7. end for 
8. Return: \( G = \langle V, E >, T = \{T_1, T_2, \ldots, T_c\} \)

#### 3.2.2 Testing stage

In the testing stage, the goal of the algorithm is to determine to which class the sample in the testing data set belongs. Taking into account the diversity of samples, we use the number of samples’ natural neighbors as the neighborhood parameters to calculate the generalized class-wise statistic in \( X_{\text{training}} \cup \{\text{sample}\} \). Here we use classification prediction \( f \) to make a classification decision for sample \( x_i \), which is defined as follows:

\[ f = \arg \max_{j=1,2,\ldots,c} \sum_{i=1}^{c} \Theta^j. \]

Algorithm 3 is the description of the NNBCA classification algorithm in the testing stage.

The time complexity of Algorithm 3 is \( O(m \cdot n) \), \( m \) is the size of the training data set, and \( n \) is the size of the testing data set. As a result of the existence of NaN, for any sample in the testing data set, the number of its natural neighbors in the training data set needs to be calculated with a computational complexity of \( O(m) \). Similarly, the time complexity of the generalized class-wise statistic calculation step in this algorithm is \( O(m) \).

### 4 Experimental Evaluation

#### 4.1 Dynamic natural neighbor

To illustrate the dynamic natural neighbor, we draw the NaNGs of the artificial data sets. The three data sets have clusters with various shapes (circular, elongated, etc.) and intracluster and intercluster density

**Algorithm 3** NNBCA: Testing stage

**Require:** 
training data set \( X_{\text{training}} \); testing data set \( X_{\text{testing}} \); 
NaN, \( G = \langle V, E > \); generalized class-wise statistic for each class, \( T = \{T_1, T_2, \ldots, T_c\} \); 
**Ensure:** 
label set of testing data:

1. for all sample \( x_i \) in testing data set \( X_{\text{testing}} \) do 
2. Calculate the number of its natural neighbor \( \text{numNaN}(x_i) \) in data set \( X_{\text{training}} \cup \{x_i\} \) by using NaNG \( G \) 
3. for all class of training data set \( X_{\text{training}} \) do 
4. Assume sample \( x_i \) belongs to current class 
5. Calculate the generalized class-wise statistic \( T_i \) by using NaNG \( G \) 
6. end for 
7. Calculate classification prediction \( f \) 
8. Make classification decision of sample \( x_i \) 
9. end for 
10. Combine all samples’ classification decisions as label set of testing data 
11. Return: label set of testing data
variations\cite{32}. In Fig. 2, the figures of the first line show the data sets, the figures of the second line show the KNN graph with $k = n \cdot 1\%$, the figures of the third line show the KNN graph with $k = n \cdot 2\%$, and the figures of the last line show the NaNG of each data set. Figure 2 shows that NaNG achieved higher performance in representing the relationship of the data sets than the KNN graphs. Traditional KNN graphs can achieve the same effect if they choose a good parameter under some circumstances, and sometimes their deficiency is unavoidable.

To better show the dynamic value of natural neighbors, this experiment searches the natural neighbor of all points in these artificial data sets and identifies the number of natural neighbors of each point. The results are illustrated in Fig. 3 which clearly show the flexibility of natural neighbor method.

4.2 Real world data sets

To demonstrate the superiority of the proposed algorithm over other classification algorithms, we compare the KNN and ENN algorithms with our NNBCA. The best $k$ in the KNN algorithm is searched in the range $1, 3, 5^{(33)}$ and $\sqrt{n}^{(34)}$ by using 10-fold cross-validation on the training set, and the neighborhood parameter of ENN algorithm is $k = 3^{(17)}$.

Fig. 2 Neighborhood graphs of KNN with different $k$ and NaN with no parameter (line 4). NaN can maintain the basic shape of data, and the neighbor number of each point is the variable.
All algorithms and experiments are run on MATLAB R2014a.

All 10 real-world data sets are obtained from the University of California, Irvine (UCI) machine learning repository. Table 1 shows the properties of these data sets.

We evaluate their performance in terms of classification accuracy. The results of all 15 UCI data sets are presented in Table 2.

Table 2 shows that our algorithm performs best in 6 out of 10 cases over all other choices, and its performance is very close to the best one in most situations. Overall, NNBCA achieves the highest accuracy among all other parameter selections with all KNN and ENN algorithms. Recall that, the proposed algorithm is completely neighborhood parameter-free, which means that we no longer need to choose the adaptive $k$ for each data set, this advantage is even more important than classification accuracy.

For every data set, we can see from Table 2 that the classification accuracy of KNN algorithm is significantly related to the neighborhood parameter $k$. For example, the accuracy of data set “DIABETES” is 54.22% when $k = 1$, but it increases to 98.98% when the neighborhood parameter is equal to $\sqrt{n}$. By contrast, the accuracy of the data set “LETTER” is 29.24% when $k=1$, but it decreases to 7.44% when the neighborhood parameter is equal to $\sqrt{n}$. Therefore, the parameter choice problem clearly exists in the KNN algorithm, and a poor selection of $k$ will also sharply decrease the classification accuracy. Moreover, a good parameter choice of one data set may be a poor choice for another data set.

Table 1  UCI data sets.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Data size</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIABETES</td>
<td>768</td>
<td>8</td>
</tr>
<tr>
<td>ECOLI</td>
<td>336</td>
<td>7</td>
</tr>
<tr>
<td>HABERMAN</td>
<td>306</td>
<td>3</td>
</tr>
<tr>
<td>SEGEMENT,TRAIN</td>
<td>210</td>
<td>13</td>
</tr>
<tr>
<td>WINE</td>
<td>178</td>
<td>13</td>
</tr>
<tr>
<td>IRIS</td>
<td>150</td>
<td>4</td>
</tr>
<tr>
<td>VEHICLE</td>
<td>846</td>
<td>18</td>
</tr>
<tr>
<td>CANCER</td>
<td>569</td>
<td>30</td>
</tr>
<tr>
<td>LETTER</td>
<td>2000</td>
<td>16</td>
</tr>
<tr>
<td>PAGEBLOCKS</td>
<td>5473</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 2  Classification accuracies of KNN algorithms over a range of $k$, ENN algorithm ($k=3$), and NNBCA algorithm with no neighborhood parameter. (%)

<table>
<thead>
<tr>
<th>Data set</th>
<th>$k = 1$</th>
<th>$k = 3$</th>
<th>$k = 5$</th>
<th>$k = \sqrt{n}$</th>
<th>ENN</th>
<th>NNBCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIABETES</td>
<td>54.22</td>
<td>83.31</td>
<td>93.05</td>
<td>98.98</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>ECOLI</td>
<td>72.27</td>
<td>73.77</td>
<td>74.97</td>
<td>73.45</td>
<td>74.09</td>
<td>77.38</td>
</tr>
<tr>
<td>HABERMAN</td>
<td>69.47</td>
<td>71.20</td>
<td>72.15</td>
<td>75.41</td>
<td>60.68</td>
<td>72.46</td>
</tr>
<tr>
<td>SEGEMENT,TRAIN</td>
<td>65.24</td>
<td>56.19</td>
<td>50.48</td>
<td>43.81</td>
<td>65.24</td>
<td>65.71</td>
</tr>
<tr>
<td>WINE</td>
<td>69.41</td>
<td>66.08</td>
<td>61.54</td>
<td>55.39</td>
<td>67.32</td>
<td>70.65</td>
</tr>
<tr>
<td>IRIS</td>
<td>96.00</td>
<td>94.67</td>
<td>93.33</td>
<td>93.33</td>
<td>95.33</td>
<td>95.33</td>
</tr>
<tr>
<td>VEHICLE</td>
<td>64.30</td>
<td>62.52</td>
<td>60.98</td>
<td>57.91</td>
<td>63.46</td>
<td>64.53</td>
</tr>
<tr>
<td>CANCER</td>
<td>90.52</td>
<td>91.39</td>
<td>91.75</td>
<td>91.75</td>
<td>91.56</td>
<td>92.80</td>
</tr>
<tr>
<td>LETTER</td>
<td>29.24</td>
<td>27.17</td>
<td>25.62</td>
<td>7.44</td>
<td>30.31</td>
<td>24.51</td>
</tr>
<tr>
<td>PAGEBLOCKS</td>
<td>81.14</td>
<td>84.85</td>
<td>86.86</td>
<td>89.76</td>
<td>73.76</td>
<td>87.61</td>
</tr>
</tbody>
</table>

Overall 69.18 71.11 71.07 68.72 72.17 75.10
For most data sets, the accuracy of the ENN algorithm is better than the average accuracy of the KNN algorithm, but the parameter selection problem still exists. The proposed NNBCA algorithm performs best in 8 out of 10 cases among all other choices and solves the parameter selection problem completely.

5 Conclusion

This paper proposes a novel parameter free classification algorithm called NNBCA, and the problem of parameter $k$ selection in the training and testing stages is solved perfectly by using the NaN method. Experimental results show that our classification result intuitively reflects the characteristics of data sets. Moreover, compared with the KNN and ENN algorithms, our algorithm increases the accuracy of classification results, adapts well to different kinds of data sets, and solves the neighborhood selection problem, which means that the value of parameter $k$ is no longer needed.

NaN method is a new concept of the nearest neighbor method, and we have used this method in clustering, outlier detection, prototype reduction, and classification. The improved algorithms obtained excellent results by using the toolkits of the NaN method. In the proposed algorithm, we use the toolkit of NaNG to achieve improved classification accuracy, and further work should address the reduction of the complexity of the proposed algorithm and NaN-based incorporation of better application areas.

Acknowledgment

This work was supported by the Foundation of Chongqing Normal University (No. 17XLB003) and the Natural Science Foundation Project of CQ CSTC (No. cstc2016jcyjA1362).

References


Ji Feng received the PhD degree in computer science from Chongqing University, China, in 2016. He is currently a lecturer at the Computer and Information Science College of Chongqing Normal University, China. His research interests include data mining and pattern recognition and applications.

Yan Wei is a professor of Computer Science at Chongqing Normal University. He received the PhD degree in control theory and control engineering from Chongqing University, China, in 2010. He is a director of the National Institute of Computer Education for Colleges and Universities, a committee member of the Technical Committee Educational of China Computer Federation, an executive committee member of the CCF Chongqing Membership Center, a director of the Chongqing Institute of Artificial Intelligence. His main research interests are machine learning, support vector machine theory and data mining.

Qingsheng Zhu is a professor of Computer Science at Chongqing University and a director of Chongqing Key Laboratory of Software Theory and Technology, China. He received the bachelor, master, and PhD degree from Chongqing University in 1982, 1985, and 2000, respectively. He was a visiting scholar with University of London, UK from 1993 to 1994 and a visiting professor with the University of Illinois in Chicago, USA from 2001 to 2002. His main research interests are service-oriented software and data mining.