International Journal of Computational Intelligence Systems (2023) 16:26
https://doi.org/10.1007/s44196-023-00200-1

RESEARCH ARTICLE

Non-parametric Nearest Neighbor Classification Based on Global Variance Difference

Shaobo Deng[1] · Lei Wang[1] · Sujie Guan[1] · Min Li[1] · Lei Wang[1]

Received: 10 March 2022 / Accepted: 7 February 2023

Abstract
As technology improves, how to extract information from vast datasets is becoming more urgent. As is well known, k-nearest neighbor classifiers are simple to implement and conceptually simple to implement. It is not without its shortcomings, however, as follows: (1) there is still a sensitivity to the choice of *k*-values even when representative attributes are not considered in each class; (2) in some cases, the proximity between test samples and nearest neighbor samples cannot be reflected accurately due to proximity measurements, etc. Here, we propose a non-parametric nearest neighbor classification method based on global variance differences. First, the difference in variance is calculated before and after adding the sample to be the subject, then the difference is divided by the variance before adding the sample to be tested, and the resulting quotient serves as the objective function. In the final step, the samples to be tested are classified into the class with the smallest objective function. Here, we discuss the theoretical aspects of this function. Using the Lagrange method, it can be shown that the objective function can be optimal when the sample centers of each class are averaged. Twelve real datasets from the University of California, Irvine are used to compare the proposed algorithm with competitors such as the Local mean k-nearest neighbor algorithm and the pseudo-nearest neighbor algorithm. According to a comprehensive experimental study, the average accuracy on 12 datasets is as high as 86.27%, which is far higher than other algorithms. The experimental findings verify that the proposed algorithm produces results that are more dependable than other existing algorithms.

Keywords Global variance difference · Non-parametric · Nearest neighbor · Lagrange method · Mean

1 Introduction

In the field of data mining, classification analysis is an essential component. It learns from and trains a training set of labeled samples, builds a model for classification, and finally uses the constructed model to identify class labels of the test set samples. We can use our established model to predict future trends and receive more valuable data. There have been several popular algorithms for classification in the past, such as naive Bayesian algorithms, support vector machines, decision trees, etc. There are a number of factors contributing to the increase in data, which also increases the requirements for classification algorithms. As data processing becomes more complex, classical algorithms are no longer able to cope. The demand for classifiers with improved classification accuracy and lower error rates is pressing. It is among the top ten algorithms in data mining because of its value in pattern classification, and it can also be used in other directions. As an example, in reference [1], a new overlapping community detection algorithm (NOCD) is proposed based on an improved KNN algorithm aimed at detecting overlapping communities in large networks. In addition, in reference [2], the improved KNN algorithm is also applied to discover the DDoS attack. A recent discussion on how to resolve the issue of KNN algorithms not

Lei Wang, Sujie Guan, Min Li and Lei Wang have contributed equally to this work.

Lei Wang (second author) 2020 Postgraduate of Nanchang Institute of Engineering; Lei Wang (fifth author) Teacher of Nanchang Institute of Engineering.

✉ Lei Wang
862519205@qq.com

Shaobo Deng
houjiyuan@nit.edu.cn

[1] School of Information Engineering, Nanchang Institute of Technology, No. 289 Tianxiang Road, Nanchang 330099, Jiangxi, China

Published online: 02 March 2023

being accurate and effective due to proximity measurement attracted many experts and scholars.

We have developed in [3] an effective method of weighting different nearest neighbor samples, Dudani [3] proposed a classical distance weighted k-nearest neighbor (WKNN) classification algorithm. As a result of this algorithm, the closest neighbor samples are used to weigh the voting process of the algorithm, rather than the k value, eliminating the influence of the k value on algorithm accuracy; Wangmeng Zhou et al. [4] presented a kernel difference weighted k-nearest neighbor classification algorithm, which defines the weighted KNN rule as a constrained optimization problem, and a novel approach is proposed for calculating different nearest neighbor weights; Jianping Gou et al. [5] proposed a KNN classification algorithm based on double-weighted voting rules (Dual Weighted voting for k-nearest Neighbor rule, DWKNN); the algorithm establishes a double-weighted voting function which is used for voting, effectively overcoming the problem of how to select nearest neighbors based on size k, and improves the quality of the classification. Local mean k-nearest neighbor algorithm (LMKNN) algorithm is a classifier proposed by Mitani. Y and Hamamoto. Y [6]. To accomplish this task, it is necessary to find the k-nearest neighbors of the samples to be classified in each training set class, obtain local mean points for each class, and finally determine the class of test samples based on the shortest distance. Three methods of determining k value are discussed in literature [7, 8], and [9] respectively, but the last one is an adaptive selection method for k value. Yong Zeng et al. [10] presented a classification algorithm based on the pseudo-nearest neighbor rule and referred to it as PNN. It is an extension of the LMKNN algorithm, which uses pseudo-nearest neighbor instead of the real nearest neighbor. Althrough the research on a large number of improved algorithms based on KNN, we find that the algorithms in the above documents have improved the first problem of KNN mentioned in this paper, that is, the accuracy and stability of KNN algorithm can be easily measured by different k values.

Another problem is that the KNN algorithm selects the closest neighbor samples based on Euclidean distance measurements, ignores correlations and redundancy, and any errors in the similarity distance measurement would affect the accuracy of classification. Zhibin pan et al. [11] proposed a locally adaptive k-nearest neighbor algorithm based on discriminant class. In a study by Weinberger et al. [12], similarity was measured using Manhattan distance, which makes it so the similarity between classes is large, whereas the dissimilarity between classes is small; Xiao X et al. [13] developed the k-nearest neighbor algorithm using attribute information entropy. It is proposed that the similarity measure is calculated as the average information entropy of sample values of the same attribute. According to average information entropy, a set of k-nearest neighbors similar to the sample to be tested is selected, and finally, the class label of the sample to be tested is calculated according to the reliability. Despite this, when the class reliability of the samples to be tested in each category is very close, it is not possible to achieve the proper classification effect. T. Denoeux [14] proposed a k-nearest neighbor classification rule based on Dempster Shafer. The related algorithms involved in this paragraph have solved the second problem of KNN algorithm proposed in the abstract, which is instructive to the algorithm proposed in this paper.

Inspired by the concept of non-parametric [8, 15–19] and the way of seeking sample center [20–31], a non-parametric nearest neighbor classification algorithm based on global variance difference (VKNN) rule is proposed in this paper. As an extension of KNN rule, the purpose of our VKNN is to use global variance differences, thereby solving the problem that KNN nearest neighbors contribute equally and the effectiveness is affected by proximity measurements without affecting the accuracy. When considering the problem, variance has been added in this paper, as opposed to the previous methods. In probability theory and statistics, variance is a measure of the dispersion of random variables. In many practical situations, it is of great importance. As a first step, we calculate the difference of variance before and after adding the sample to be tested in order to obtain an optimal distance metric. After that, we divide the difference by the variance before adding the sample to be tested, and we take the minimum value of the quotient as the objective function, so as to analyze the problem more intuitively and effectively. In experiments using real data, the proposed algorithm effectively resolves the problem that KNN accuracy and effectiveness are somewhat affected by proximity measurement, and improves the accuracy of the classification algorithm significantly.

With its advantages such as simplicity, effectivity, and competitive performance, KNN classification has become the most attractive classification method in countless practical applications. Nevertheless, as described in Section 2.1, it is susceptible to problems such as sensitive selection of nearest neighbor parameters and the simple maximum voting classification decision-making principle, especially in the case of a small sampling size. It has been clearly known that the selection of neighborhood size plays a very important role in KNN algorithm. When k is too small, the near neighborhood may include noise points and outliers, causing overfitting and incorrect classification rules; when k is too large, the nearest neighbor may include more noise points and outliers, resulting in a reduction in classification accuracy.

Furthermore, the proximity measurement method will also affect the performance of the KNN algorithm. In many improved KNN algorithms, the k-nearest neighbor samples of each test sample are usually selected by the dissimilarity

measure, and the Euclidean distance measure is commonly used to select the nearest neighbor set. The accuracy and effectiveness of the KNN algorithm are affected by proximity measurement. Sometimes the similarity between test samples and nearest neighbor samples cannot be accurately reflected.

A large number of studies have proposed various methods to solve various problems in KNN algorithm, such as adaptive selection of appropriate k value, changing classification decision rules, etc. As far as we know, the difference of ergodic global variance has not attracted the attention of researchers. Therefore, our proposed algorithm pays attention to this breakthrough. The main reasons why the experimental results of this paper are significantly better than other algorithms are as follows: (1) the parameter k is not used. Because according to the previous experimental analysis, the value of k is too large or too small will affect the experimental results. (2) The sample center is selected as the sample mean, and the global variance difference is traversed. The feasibility of this method has been clearly proved in the previous theoretical proof, which is also one of the important reasons why the experimental results of our algorithm are obviously better than other algorithms.

The main contributions of this paper can be summarized as follows: (1) a new scheme of designing classification rules do not involve k-value and other parameters, the VKNN algorithm avoids the influence of k-value and other parameters on the classification effect. (2) A non-parametric nearest neighbor classification based on global variance difference (VKNN) rule is proposed by introducing a global variance difference in the dissimilarity, rather than using the traditional Euclidean distance formula to select the nearest neighbor set. (3) The effectiveness of VKNN is explored by extensive experiments in terms of the classification error and the good classification of VKNN is credibly demonstrated by non-parametric statical tests.

The remainder of this paper is structured as follows. In Sect. 2, we briefly summarize KNN and the classification algorithms used in the comparative experiment. In Sect. 3, we introduce the objective function of the VKNN algorithm and the feasibility of the algorithm. At the same time, we give the pseudocode of the algorithm. In Sect. 4, we conduct experiments for all the competing methods using accuracy and recall on the real UCI datasets. Finally, the conclusions are mentioned in Sect. 5.

2 Related Algorithm

In machine learning, the k-nearest neighbor (KNN) rule has been used extensively in non-parametric classifiers. A variety of KNN-based approaches has also been developed to address the challenges in KNN-based classification, such as the pseudo-nearest neighbor classification (LMPNN) [32], a globally adaptive k-nearest neighbor classifier based on local mean optimization(LMO-GAKNN) [33] and a new locally adaptive k-nearest neighbor algorithm based on discrimination class (DC-LAKNN) [11]. In this section, we briefly review LMPNN, LMO-GAKNN and DC-LAKNN classifiers that motivate our work.

2.1 The Classical KNN Algorithm

In the field of data mining, the KNN algorithm is a classical classification algorithm. This algorithm is the classification algorithm proposed by Cover et al. [34] in 1967. In addition, it has several advantages: (1) its concept is very simple and easy to implement, (2) there is no need to generate additional data to describe the rules, and there will be noise in the data, (3) in category decision-making, it is only related to a very small number of adjacent samples, which allows for better avoiding the imbalance of sample numbers. Its performance, however, is easily affected by the sensitivity of nearest neighbor parameter selection and the maximization of voting in the classification decision, so its performance needs to be further improved.

In the general classification problem, suppose a training set $T = \{y_i = R^d\}_{i=1}^N$ with M classes consists of N training samples in a d-dimensional feature space and the class label of one sample y_i is c_n, where $c_n = \{\omega_1, \omega_2, \ldots, \omega_M\}$. Let $T(\omega_1) = \{y_i^j = R^d\}_{i=1}^N$ denote a class subset of T from the class y_i, with the number of the training samples N_i. Given a query point x, the KNN rule is carried out as follows:

Step 1: Calculate the Euclidean distance from the sample x to $T = \{y_i = R^d\}_{i=1}^N$:

$$d(x, y_i) = \sqrt{(x - y_1)^2 + (x - y_2)^2 + \cdots + (x - y_n)^2} = \sqrt{\sum_{k=1}^{n} (x - y_i)^2} \tag{1}$$

Step 2: Sort all distances in ascending order: $d(x - y_1) < d(x - y_2) < \cdots < d(x - y_N)$

Step 3: Take the first k neighbors: $y_1, y_2, \ldots, y_k$

Step 4: Predict the class label of the sample x that needs to be tested:

$$c = argmax_v \sum_{y_i, \omega_i \epsilon T} I(v = \omega_i) \tag{2}$$

Among them, the class label is expressed as v, and the indicator function is expressed as I(·), as shown in the following formula:

$$I(a, b) = \begin{cases} 1, \ if \ a = b \\ 0, otherwise \end{cases} \tag{3}$$

2.2 Pseudo-Nearest Neighbor Algorithm was Based on Local Mean (LMPNN)

LMPNN is another successful KNN-based classifier. Motivated by the ideas of the LMKNN [6] and PNN [10] rules, it is a pseudo-nearest neighbor algorithm based on each test sample, which makes extensive use of the local information of the sample and minimizes the impact of outliers on the classification accuracy. The advantage of this algorithm is straightforward, efficient, and easy to execute. It still widely uses in the field of classification algorithms.

Given a query point x and a training set T, the class label of x in the LMPNN rule is determined through the following steps:

Step 1: Calculate the Euclidean distance from the sample x to $T_{\omega_j}(j = 1, 2, \ldots, L)$:

$$d(x, y_i) = \sqrt{(x - y_1)^2 + (x - y_2)^2 + \cdots + (x - y_n)^2} = \sqrt{\sum_{k=1}^{n} (x - y_i)^2} \tag{4}$$

Step 2: Arrange the Euclidean distances in the category ω_j in ascending order, and take the first k-nearest neighbors: $y_1^j, y_2^j, \ldots, y_k^j$

Step 3: Calculate the local mean vector of the first i nearest neighbors of the sample x in the category ω_j:

$$\bar{y}_i^j = \frac{1}{i} \sum_{l=1}^{i} y_l^j \tag{5}$$

Use $\bar{T}_{\omega_j} = \{\bar{y}_i^j \epsilon R^d\}_{i=1}^k$ to denote the set of local mean vectors in ω_j. $d(x, \bar{y}_i^j)$ represents the Euclidean distance from the sample x to the local mean in the category ω_j.

Step 4: Assign different weights to the local mean vector in each category. In the class ω_j, the weight of the *ith* local mean vector is

$$W_i = \frac{1}{i}(i = 1, 2, \ldots, k) \tag{6}$$

Step 5: Calculate the pseudo-nearest neighbors in each category ω_j:

$$d(x, y_{PNN}^j) = W_1 \times d(x, \bar{y}_1^j) + W_2 \times d(x, \bar{y}_2^j) + \cdots + W_k \times d(x, \bar{y}_k^j) \tag{7}$$

Step 6: Predict the class label c of the sample x that need to be tested:

$$c = argmin_{\omega_j} d(x, y_{PNN}^j) \tag{8}$$

2.3 A Globally Adaptive k-Nearest Neighbor Classifier Based on Local Mean Optimization (LMO-GAKNN)

In the LMKNN classifier, the unreliable nearest neighbor selection rule and the single local average vector strategy have serious defects that have a negative impact on its classification performance. Considering these problems in LMKNN, a globally adaptive k-nearest neighbor classifier based on local mean optimization [33], which has been proposed in 2021, utilizes the globally adaptive nearest neighbor selection strategy and the implementation of local mean optimization to obtain more convincing and reliable local mean vectors.

Given a query point x and a training set T, the class label of x in the LMO-GAKNN rule is determined through the following steps:

Step 1: Calculate the Euclidean distance from the sample x to $T = \{y_i = R^d\}_{i=1}^N$:

$$d(x, y_i) = \sqrt{(x - y_1)^2 + (x - y_2)^2 + \cdots + (x - y_n)^2} = \sqrt{\sum_{k=1}^{n} (x - y_i)^2} \tag{9}$$

Step 2: Sort all distances in ascending order: $d(x - y_1) < d(x - y_2) < \cdots < d(x - y_N)$.

Step 3: Select the first r samples as r nearest neighbors for class ω_j based on sorted $d(x, y_i)$.

Step 4: Find k-nearest neighbors of x in the ascending order of sorted Euclidean distances to x, where $k = r \times M$.

Step 5: Select r_j nearest neighbors for class ω_j in the initial k-nearest neighbors, where $0 \leqslant r_j \leqslant k$ and $\sum_{j=1}^{M} r_j = k$.

Step 6: Compute $MLMV_j^{r_j}$ for class ω_j according to equation as follows:

$$m_j^s = \frac{1}{s} \sum_{i=1}^{s} y_i, y_i \epsilon NN_j^r \tag{10}$$

Step 7: Select optimal local mean vector m_j^* in $MLMV_j^{r_j}$ according to the equation as follows:

$$m_j^* = argmin_s d(x, m_j^s), m_j^s \epsilon MLMV_j^{r_j} \tag{11}$$

Step 8: Classify x into the class ω_c based on minimum Euclidean distance between x and optimal local mean vectors of all classes according to the equation as follows:

$$\omega_c = argmin_{\omega_j} d(x, m_j^*) \tag{12}$$

2.4 A New Locally Adaptive k-Nearest Neighbor Algorithm Based on Discrimination Class (DC-LAKNN)

A new locally adaptive k-nearest neighbor algorithm based on discrimination class(DC-LAKNN) [11] has been proposed in 2020. In this method, the role of the second majority class in classification is for the first time considered. Firstly, the discrimination classes at different values of k are selected from the majority class and the second majority class in the k-neighborhood of the query. Then, the adaptive k value and the final classification result are obtained according to the quantity and distribution information on the neighbors in the discrimination classes at each value of k.

Given a query point x and a training set T, the class label of x in the DC-LAKNN rule is determined through the following steps:

Step 1: Select its k-nearest neighbors $NN_k(x) = (y_i^{NN}, c_i^{NN})_{i=1}^{k}$ in the training set T according to the ascending order of Euclidean distances between x and all training instances.

Step 2: Find the majority classes $(\omega_{1st,i}^k)_{i=1}^{num_{1st}{}^k}$ and second majority classes $(\omega_{2nd,i}^k)_{i=1}^{num_{2nd}{}^k}$ in the k-neighborhood of the query x consisting of $NN_k(x)$, where num_{1st}^k and num_{2nd}^k are the number of the majority classes and second majority classes, respectively. Record the number of the majority class neighbors and the second majority class neighbors r_{1st}^k and num_{2nd}^k.

Step 3: Compute the centroids of nearest neighbors in different majority classes and different second majority classes $(cent_{1st,i}^k)_{i=1}^{num_{1st}{}^k}$,$(cent_{2nd,i}^k)_{i=1}^{num_{2nd}{}^k}$.

Step 4: Select discrimination class ω_{disc}^k from the majority classes $(\omega_{1st,i}^k)_{i=1}^{num_{1st}{}^k}$ and second majority classes $(\omega_{2nd,i}^k)_{i=1}^{num_{2nd}{}^k}$. Specifically, the selection process is divided into the following cases:

Case 1: If $num_{1st}{}^k = 1$, then compute $\delta = r_{1st}^k / r_{2nd}^k$;

Case 1.1: If $\delta = r_{1st}^k / r_{2nd}^k > \theta$, then this sole majority class is selected as the discrimination class, that is,

$$\omega_{disc}^k = \omega_{1st,1}^k \tag{13}$$

Case 1.2: If $\delta = r_{1st}^k / r_{2nd}^k \leqslant \theta$, then the class with the closest centroid to the query x among the sole majority class and the num_{2nd}^k second majority classes is selected as the discrimination class, that is,

$$\omega_{disc}^k = argmin(d(x, cent_{1st,1}^k), d(x, cent_{2nd,i}^k)) \tag{14}$$

Case 2: If $num_{1st}{}^k > 1$, then the majority class with the closest centroid to the query x is selected as the discrimination class, that is,

$$\omega_{disc}^k = argmind(x, cent_{1st,i}^k) \tag{15}$$

Step 5: Record the number and centroid of nearest neighbors in the discrimination class, r_{disc}^k and $cent_{disc}^k$, based on the above results.

Step 6: Compute the ratio of the number of nearest neighbors in the discrimination class to the number of all nearest neighbors, $ratio^k = r_{disc}^k / k$, and the distance between the centroid of nearest neighbors in the discrimination class and the query x, $d^k = d(x, cent_{disc}^k), k = 1, 2, \ldots, kmax$.

Step 7: Generate two rankings $rank_{ratio} = (rank_{ratio}^k)_{k=1}^{kmax}$ and $rank_d = (rank_d^k)_{k=1}^{kmax}$ of the discrimination classes corresponding to different values of k according to the descending order of $ratio^k$ and the ascending order of $d^k, k = 1, 2, \ldots, kmax$.

Step 8: Compute the average ranking of the discrimination classes for each value of k:

$$rank_{ave}^k = \left(\frac{rank_{ratio}^k}{max_{k=1,2,\ldots,kmax} rank_{ratio}^k} + \frac{rank_d^k}{max_{k=1,2,\ldots,kmax} rank_d^k} \right) / 2 \tag{16}$$

$$k = 1, 2, \ldots, kmax \tag{17}$$

Step 9: Select the discrimination class with the minimum average ranking as the classification result of the query x, and the corresponding value of k is the adaptive k value for x, that is,

$$k_{adap} = argmin_{k=2,\ldots,kmax}(rank_{ave}^k) \tag{18}$$

$$\omega_c = \omega_{disc}^{k_{adap}} \tag{19}$$

3 The VKNN Algorithm

As a further improvement to KNN-based classification, we present the non-parametric nearest neighbor classification based on global variance difference (VKNN) in this section. Using the VKNN approach, we further improve KNN-based classification accuracy.

3.1 The VKNN Classification Rule

The VKNN method, as an extension of the KNN rule, is presented in this subsection. The idea of the VKNN algorithm is: first, the training samples are divided into categories, and the mean value of each sample is taken as the sample center; Then, according to the sample center, the variance before and after adding the sample to be tested in each class is calculated respectively, and the obtained variance is processed as the difference, and then the difference is divided by

the variance before adding the sample to be tested, and the quotient obtained is used as the objective function; finally, the samples are classified into the class with the smallest objective function.

Take the mean value of N_i samples in each class as the sample center and record it as M. In the proposed VKNN rule, the class label of a query point x is yielded by the following steps:

Step 1: Suppose a training dataset $T = \{y_i = R^d\}_{i=1}^{N}$ is given;

Step 2: Select samples from the training samples, and divide the samples according to classes, and the corresponding classes are marked as set $T_{\omega_j} = \{y_i^j = R^d\}_{i=1}^{N^j}$;

Step 3: For each class, select the sample center x_{ij} of N_i a sample and record it as M, then we have

$$M_i = \frac{\sum_{j=1}^{N_i} x_{ij}}{N_i} \tag{20}$$

Step 4: According to sample center, calculate the global variance between the sample point to be tested and the sample center of each category, then we have the objective function $D_i(x)$, that is

$$D_i(x) = \frac{\sum_{i=1}^{n+1}(x_i - M_i)^2 - \sum_{i=1}^{n}(x_i - M_i)^2}{\sum_{i=1}^{n}(x_i - M_i)^2} \tag{21}$$

Step 5: Finally find the minimum value of the objective function, and classify the sample to be tested into the class with the minimum objective function $D_i(x)$:

$$c = argmin_{\omega_i} D_i(x) \tag{22}$$

3.2 The Pseudo Codes of VKNN

As discussed in this section, the proposed VKNN method is summarized in Algorithm 1 by means of pseudocodes.

Algorithm 1 The VKNN algorithm

Require: : x:a query ,$T_{\omega_j} = \{y_i^j = R^d\}_{i=1}^{N^j}$:a training subset from class ω_j.

Ensure: : Predict the class label of a query by the minimum of the global variance difference among classes.

1: N_i is the number of samples in T
2: **for i:=1 to N_i do**
3: $M_i = \frac{\sum_{j=1}^{N_i} x_{ij}}{N_i}$
4: end for
5: **for i:=1 to n+1 do**
6: $D_i(x) = \frac{\sum_{i=1}^{n+1}(x_i - M_i)^2 - \sum_{i=1}^{n}(x_i - M_i)^2}{\sum_{i=1}^{n}(x_i - M_i)^2}$
7: Traverse all categories until the global variance change minimum is found
8: end for

3.3 Theoretical Proofs

The objective function is mainly composed of the variance before and after adding the sample to be tested:

$$\begin{aligned} var_1 &= \sum_{i=1}^{n}(x_i - M_i)^2 = \sum_{i=1}^{n}(x_i - \tfrac{x_1+x_2+\cdots+x_n}{n})^2 \\ var_2 &= \sum_{i=1}^{n+1}(x_i - M_i)^2 = \sum_{i=1}^{n+1}(x_i - \tfrac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1})^2 \end{aligned} \tag{23}$$

If the two are treated as differences, there are

$$\begin{aligned} var_2 - var_1 &= \sum_{i=1}^{n+1}(x_i - \tfrac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1})^2 - \sum_{i=1}^{n}(x_i - \tfrac{x_1+x_2+\cdots+x_n+x_n}{n})^2 \\ &= (x_{n+1} - \tfrac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1}) \\ &+ \sum_{i=1}^{n}(x_i - \tfrac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1})^2 - \sum_{i=1}^{n}(x_i - \tfrac{x_1+x_2+\cdots+x_n+x_n}{n})^2 \end{aligned} \tag{24}$$

Then, find the partial derivative of x_{n+1} for the above formula:

$$\begin{aligned} \frac{\partial D_i(x)}{\partial x_{n+1}} &= \sum_{i=1}^{n} 2(x_i - \frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1})(-\frac{1}{n+1}) \\ &+ 2(x_{n+1} - \frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1})(1 - \frac{1}{n+1}) \end{aligned} \tag{25}$$

After simplification, the following formula can be obtained:

$$\sum_{i=1}^{n+1}\left(\frac{x_i}{n}-\frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n}+\frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1}\right) = x_{n+1}-\frac{x_1+x_2+\cdots+x_n}{n} \tag{26}$$

where

$$\sum_{i=1}^{n+1}\left(\frac{x_i}{n}-\frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n}\right)=-(x_1+x_2+\cdots+x_n+x_{n+1}) \tag{27}$$

In addition,

$$\sum_{i=1}^{n+1}\frac{x_1+x_2+\cdots+x_{n+1}}{n+1}=(n+1)\frac{x_1+x_2+\cdots+x_{n+1}}{n}=x_1+x_2+\cdots+x_{n+1} \tag{28}$$

Therefore, there are

$$\sum_{i=1}^{n+1}\left(\frac{x_i}{n}-\frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n}+\frac{x_1+x_2+\cdots+x_n+x_{n+1}}{n+1}\right)=0 \tag{29}$$

That is

$$x_{n+1}-\frac{x_1+x_2+\cdots+x_n}{n}=0 \tag{30}$$

The final result is

$$x_{n+1}=\frac{x_1+x_2+\cdots+x_n}{n} \tag{31}$$

The objective function obtained in this paper is obtained by calculating the difference between the variance of the sample to be tested and the variance of all sample centers. As shown in Formula 14, the difference between the variance of the sample to be tested and the variance of the sample center as a whole is a variable. This section also analyzes theoretically, and gives the variances of the sample to be tested and all sample centers respectively, and makes a difference between them. The final results show that this method is completely feasible in theory. In this way, it is proved theoretically that when the sample center of each sample is the sample mean, the value of the objective function $D_i(x)$ can be the minimum, and it is most appropriate to divide the samples to be tested into the class with the minimum value of the objective function.

4 Experimental

To validate the proposed VKNN on the classification performance, we compare VKNN with competing classifiers: KNN, LMKNN, LMPNN, LMO-GAKNN, DC-LAKNN. During the experiment, real datasets were used to verify the accuracy of this algorithm. Moreover, we use recall to further verify superiority of the proposed method on all real datasets, since non-parametric statistical tests play an important role in comparing the performance of the competing classifiers over multiple datasets so as to demonstrate the effectiveness of a new method in machine learning. The comparison between algorithms shows the performance of each algorithm. In the next section, we will discuss dataset selection, experimental setting, evaluation index selection, and comparison with other classification algorithms. The results and analysis of the experiments will be described in detail.

4.1 Selection of Datasets

In this subsection, we briefly give the information of all the datasets used in our experience. The 12 real datasets are selected from the UCI machine learning repository. For short, among these datasets, the abbreviations of 'wine', 'seeds', 'sonar', 'wrd', 'haber', 'glass', 'trans', 'iris', 'aus', 'iono' 'ozone' and 'segment'. The name and relevant information of the data are shown in Table 1. The main characteristics of these 12 datasets are introduced in detail, including the number of samples, the number of attributes, the number of categories, and the test sets.

As shown in Table 1, there are seven two-class classification problems among all the real datasets, while the others are the multi-class classification problems. In these datasets, the largest and lowest numbers of the samples are 2536 and 146, and the largest and lowest numbers of dimensionality are 73 and 3.

4.2 Experimental on Real UCI Datasets

Our algorithm is designed for the problem of KNN nearest neighbors having the same contribution and the effectiveness being affected by proximity measurement. For changing the traditional Euclidean distance formula to select the nearest

Table 1 Detailed information about datasets

Datasets	Samples	Attributes	Classes	Testing sets
wine	178	13	3	59
seeds	210	7	3	60
sonar	208	60	2	66
wrd	1571	11	4	523
haber	306	3	2	140
glass	146	9	2	53
trans	748	4	2	160
iris	150	4	3	55
aus	690	14	2	212
iono	351	34	2	128
ozone	2536	73	2	920
segment	2310	18	7	832

neighbors, an improved KNN algorithm is proposed. These performances are performed by 10-fold cross validation by means of the classification accuracy and racall. As a comparative experiment, we chose a variety of KNN algorithms, including classical KNN algorithm, LMKNN [6], LMPNN [32], LMO-GAKNN and DC-LAKNN. The four comparison algorithms adopt the method of identifying and verifying the k value one by one, the specific settings are as follows: firstly, the value range of k is $1\ \sqrt{n}$ (n indicating the number of samples); next, perform the m-fold cross validation five times to obtain the average accuracy for each k value, and select the k value with the highest average accuracy as the optimal k value. For the comparative experiment, we choose the best k value. In Fig. 1, we show the optimal k values for four comparison algorithms for 12 datasets. The best classification of each method with highest accuracy is obtained in the interval of k on each data set.

4.3 Experimental Result

We use two performance indicators to measure the classification ability of different algorithms. One is the accuracy rate (Acc), the other is the recall rate (Recall). The following is a detailed introduction of the two performance indicators:

a. Accuracy (Acc): indicates the percentage of correctly classified samples. The formula is as following:

$$Acc = \frac{\sum_{i=1}^{m} \delta(r_i, map(p_i))}{m} \tag{32}$$

where m is the total amount of data objects in a given dataset; p_i and r_i labels, respectively, representing the classification of data objects and the real situation; $\delta(x, y)$is Dirac function, when $x = y$, $\delta(x, y) = 1$, otherwise, $\delta(x, y) = 0$;$map(p_i)$ is a permutation mapping function, which maps the classification division p_i obtained by classification to the equivalent real label. In addition, the value ranges of Acc being [0, 1]. The greater the classification accuracy, the better the classification effect.

b. Recall: refers to the ratio of the number of samples correctly identified as this category to the total number of samples belonging to this category in the original sample set. The specific definition is as following:

$$Recall = \frac{n_k^m}{n_m} \tag{33}$$

where n_k^m represents the number of data objects shared by the kth class in the classification result and the mth class in the real classification result, n_m represents the number of data objects in the mth class in the real classification result. The larger the value of the Recall, the better the classification effect.

Tables 2 and 3 show the best accuracy and recall rate of each method respectively, bold values are shows that the classification ability of corresponding classifier performs the best among all comparative classifier. There are also corresponding standard deviations in brackets in the table. Note that in the competitive method, the best value for each dataset is expressed in bold.

We draw the results of Tables 2 and 3 into the histogram of Figs. 2 and 3, which can more clearly compare the performance of several algorithms.

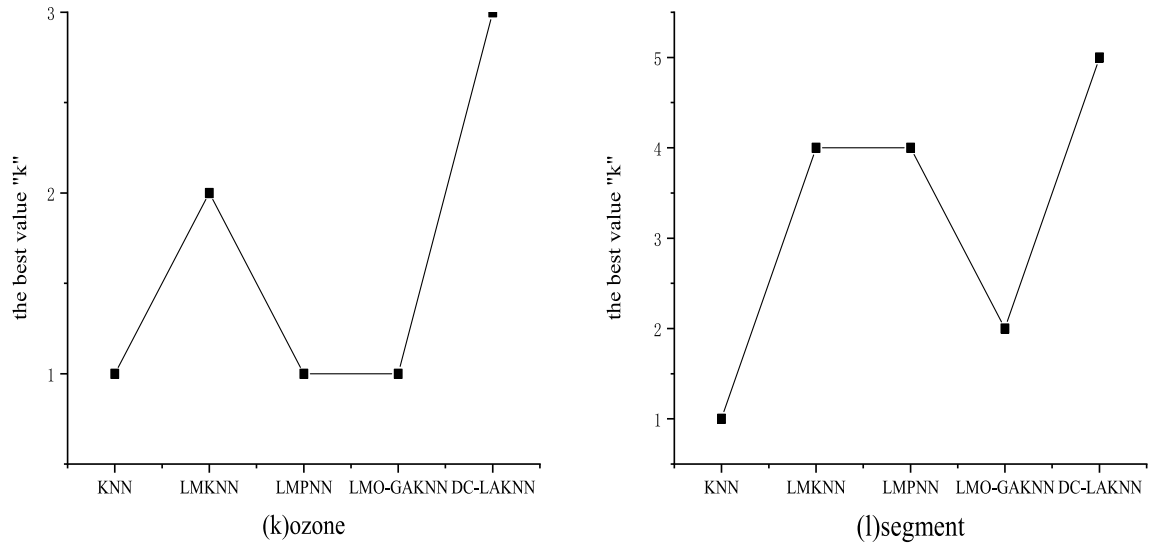

Fig. 1 The best value k of each method on different datasets

Table 2 Accuracy on different datasets

Data	KNN	LMKNN	LMPNN	LMO-GAKNN	DC-LAKNN	VKNN
wine	83.22±5.79	87.46±2.90	87.80±2.08	80.58±1.34	88.33±2.17	**91.87 ± 4.74**
seeds	90.50±5.67	92.13±4.01	92.13±4.38	91.32±1.56	90.88±2.13	**92.67 ± 3.99**
sonar	80.30±4.23	81.36±3.30	80.38±2.15	82.14±3.12	**83.03±3.70**	81.61±3.23
wrd	58.37±1.24	58.37±1.24	60.67±1.70	59.26±3.65	60.13±1.39	**60.89 ±5.45**
haber	73.86±1.27	70.79±2.72	70.21±2.08	72.45±3.16	70.88±5.12	**76.43 ±3.57**
glass	80.75±6.15	81.51±5.82	83.02±4.87	**87.94±3.20**	83.25±1.51	87.55±4.90
trans	80.19±1.25	80.12±1.97	79.62±1.75	81.02±4.10	79.98±0.27	**81.38±5.63**
iris	95.73±2.48	93.68±2.31	94.21±1.37	94.31±1.26	95.21±0.39	**96.51±3.49**
aus	81.32±1.56	83.35±1.97	84.64±1.66	84.75±1.24	83.97±1.22	**84.96±2.34**
iono	80.05±4.76	88.71±4.13	89.41±3.64	87.95±2.33	88.12±2.56	**89.82±3.94**
ozone	96.88±2.44	94.00±3.01	96.81±2.72	94.88±2.15	96.32±5.13	**97.08±1.49**
segment	93.40±0.81	93.41±1.36	93.91±0.87	91.23±4.12	93.52±3.21	**94.45±1.86**
Average	82.88±3.22	83.74±2.87	84.62±2.57	83.99±2.08	84.47±2.40	**86.27±3.71**

Table 3 Recall on different datasets

Data	KNN	LMKNN	LMPNN	LMO-GAKNN	DC-LAKNN	VKNN
wine	60.61±1.91	59.13±3.80	60.79±1.19	60.68±0.23	75.32±4.03	**89.76±5.54**
seeds	96.32±2.31	96.81±0.66	98.42±0.85	97.25±4.19	96.31±1.80	**98.67±1.13**
sonar	87.38±3.88	**92.83±2.35**	88.18±4.77	84.58±1.23	87.62±3.11	86.20±3.39
wrd	45.11±2.78	58.01±5.75	53.83±0.93	**54.26±0.78**	52.12±1.26	50.11±1.04
haber	68.65±3.23	69.84±1.86	74.57±0.82	74.59±2.48	72.39±0.33	**75.03±2.59**
glass	95.53±2.17	94.88±2.82	94.77±2.82	97.69±3.22	96.83±2.96	**98.56±0.78**
trans	69.76±5.16	63.03±1.57	65.44±1.59	71.96±2.89	71.35±0.69	**72.24±1.02**
iris	94.89±4.55	94.94±2.43	94.87±1.87	96.31±1.20	97.32±1.56	**97.78±1.63**
aus	83.57±5.28	83.51±1.41	81.34±2.73	83.69±1.78	82.97±3.10	**84.25±3.16**
iono	83.16±1.05	88.81±1.57	90.43±1.08	92.57±3.54	89.71±3.36	**94.58±2.77**
ozone	98.07±2.06	88.69±2.57	90.52±0.63	92.12±3.12	90.38±3.69	**98.71±1.25**
segment	94.71±1.61	90.55±2.56	93.35±1.47	94.25±1.02	90.11±0.23	**94.79±2.03**
Average	81.48±2.99	81.75±2.45	82.21±1.73	83.33±2.14	83.54±2.18	**86.72±2.19**

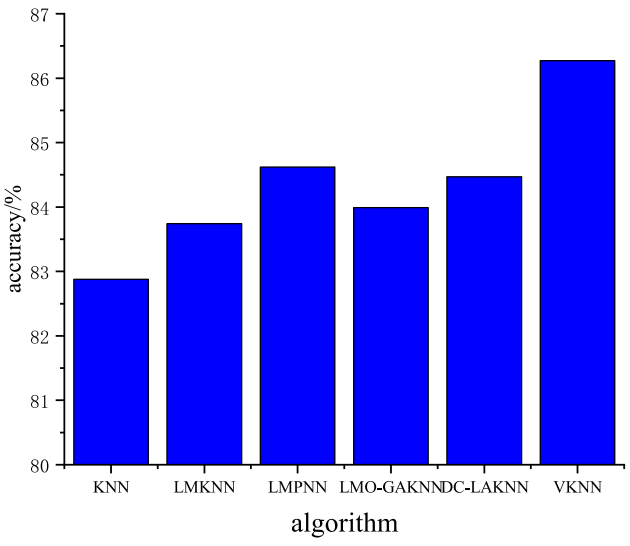

Fig. 2 Average accuracy of different algorithms

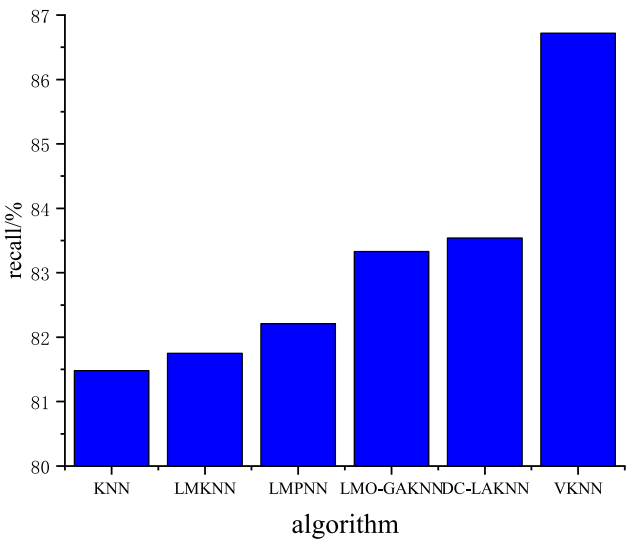

Fig. 3 Average recall of different algorithms

5 Discussion

5.1 Analysis of Experimental Results

The experimental results in Tables 2 and 3 clearly illustrate that our proposed VKNN is almost as effective as the other five competitive methods in these real UCI datasets. The average classification accuracy of the VKNN algorithm is higher than that of the other five classification algorithms on the 12 datasets in Table 2. The classification accuracy of this algorithm for sonar dataset is 81.61%, which is lower than 83.03% of DC-LAKNN algorithm, the difference is 1.42, but still better than 80.30% of KNN, 81.06% of LMKNN and 80.38% of LMPNN, and the difference is acceptable. In addition, in the glass dataset, the accuracy of VKNN is lower than that of LMO-GAKNN, but it is significantly higher than that of the other four compared algorithms. However, in terms of classification accuracy, the comparison between competing classifiers is not statistically convincing. Here, we further use recall to compare the performance of classifiers. It can be seen from Table 3 that the recall rate of 10 datasets is significantly better than that of other algorithms. Especially in the sonar dataset, the recall rate of VKNN algorithm is the lowest, which is 86.20%. The difference between VKNN algorithm and the other four algorithms is 1.18%, 6.63%, 1.98% and 1.42% respectively, and the difference between VKNN algorithm and LMKNN algorithm is the largest. In the wrd dataset, the recall rate of VKNN is 50.11%, significantly higher than 45.11% of KNN, but also lower than 58.01% of LMKNN, 53.83% of LMPNN, 54.26% of LMO-GAKNN and 52.12% of DC-LAKNN. In general, the average of the two evaluation indexes of the algorithm in 12 datasets is significantly better than the other three algorithms, indicating that the algorithm improves the classification performance on the whole. This also means that our algorithm can be better applied to data classification (Table 4).

5.2 Algorithm Complexity Analysis

In the process of classifier design, complexity can be used as an index to evaluate the classifier. Suppose the size of the dataset is n, the data dimension is d, the number of nearest neighbors is k, the number of found nearest neighbors is r, the number of categories is c, num^k_{1st} and num^k_{2nd} are the number of the majority classes and second majority classes, respectively. The complexity of several comparison algorithms used in the experiment is as follows:

(1) KNN: In reference [32], we can clearly get that the complexity of KNN and WKNN are $O(nd + nk + k)$ and $O(nd + nk + 2k)$.

(2) LMKNN: The complexity of LMKNN algorithm comes from four aspects: (a) calculate the distance from the test sample to each class, and the complexity is $O(n_1 d + n_2 d + \cdots + n_c d)$; (b)find k-nearest neighbors from each class, the complexity is $O(n_1 k + n_2 k + \cdots + n_c k)$; (c) calculate the local average vector of k-nearest neighbors in each class, then the complexity of this step is $O(ckd)$;(d) assign the samples to be classified to the class

Table 4 Abbreviations

Notations	Descriptions
$T = \{y_i = R^d\}_{i=1}^N$	The training datasets
N	The number of samples in T
ω	The class label
ω_j	The jth class label
x	The query
y	The training samples
c	The label of y
k	The number of found nearest neighbors
d	Data dimension
M	The sample center for each class
r	The number of found nearest neighbors
r_j	The number of found nearest neighbors for class
ω_c	The classification result of x
NN_j^r	The nearest neighbors set for class ω_j with r nearest neighbors
m_j^r	The local mean vector for class ω_j
$MLMV_j^{r_j}$	The multi-local mean vector set for class ω_j, which has r local mean vectors
var_1	Variance before adding the sample to be tested
var_2	Variance after adding the sample to be tested

with the nearest local average vector, and the complexity is $O(cd + c)$. Therefore, in general, the complexity of LMKNN is $O(nd + nk + ckd + cd + c)$.

(3) LMPNN: On the basis of LMKNN algorithm, calculate the distance from the test sample to each class, and find k-nearest neighbors from each class, then the complexity is $O(nd + nk)$. The other three complexities come from the following three aspects: (a) calculate k local average vectors corresponding to k-nearest neighbors, and calculate the distance between the sample to be tested and k local average vectors, the complexity is $O(2ckd)$; (b) the weight is assigned to the local mean vector of each class to find out the pseudo-nearest neighbor of each class, the complexity is $O(3ck)$; (c) determine the type of sample to be tested, with a complexity of $O(c)$. Therefore, in general, the complexity of LMPNN is $O(nd + nk + 2ckd + 3ck + c)$.

(4) LMO-GAKNN: In reference [33], we can clearly get that LMO-GAKNN can be summarized into three aspects. Therefore, the time complexity of the algorithm can be summarized as: $O(nk + 2Mr)$.

(5) DC-LAKNN: According to the discussion in reference [11], we can accurately obtain that the complexity of DC-LAKNN can be summarized as: $O(num^k_{1st} + num^k_{2nd} + k_{max})$.

(6) VKNN: The steps of VKNN algorithm can be summarized into two aspects: (a) according to sample center, calculate the global variance between the sample point to be tested and the sample center of each category; (b) classify the sample to be tested into the class. Then, the complexity of VKNN is $O(n^2 + cd)$.

By analyzing the complexity of the above algorithms, it is not difficult to see that the complexity of VKNN algorithm is greater than others. However, considering that the VKNN algorithm avoids the influence of k value, these increased calculations are acceptable.

6 Conclusion

In this paper, we propose a non-parametric nearest neighbor classifier to solve the problem that KNN classifier depends on nearest neighbor measurement and is sensitive to k-value selection. Distinct from the aforementioned improved methods, in the existing improved algorithms, the nearest neighbor of each test sample is typically selected by the traditional distance measurement, which will drastically reduce the effect of classification. The improved method of this paper is to introduce variance, divide the difference between the variance before and after additional samples, take the minimum value of the result as the objective function, and determine the rules of the algorithm by optimizing the objective function for classification. It is also demonstrated here that the central sample point of each nearest neighbor class may be replaced by the mean value of all sample points within it. To some extent, the sensitivity of k-value selection and the influence of proximity have been solved. Finally, experimental results on 12 real datasets show that the improved algorithm overcomes the problems of k-value sensitivity and proximity measurement of KNN algorithm to a certain extent.

The advantages of VKNN algorithm is that its classification rules do not involve k-value and other parameters, effectively avoiding the influence of k-value and other parameters on the classification effect. Furthermore, the traditional Euclidean distance formula is not used to select the nearest neighbor set in the dissimilarity measurement of VKNN, but rather a global variance difference is introduced. Thus, it avoids to an extent the impact of k value and the nearest neighbor measurements on the classification performance, compared to other improved classification algorithms based on KNN.

However, as the algorithm complexity analysis in 5.2 shows, the time complexity of VKNN algorithm is relatively complex. In the next step, the proposed algorithm must be applied to complex practical problems. It is necessary to reduce the algorithm complexity without reducing the accuracy of the algorithm on the basis of existing algorithms. In addition, the most important thing of the classifier is to use and practice. How to combine the classifier with the actual situation, and what kind of dataset is applicable to, are also issues we need to consider in the future.

Acknowledgements The project is funded in part by natural Science fund project of department of Jiangxi Province Science and Technology (No. 20224BAB202014), the Science and Technology Project of Jiangxi Provincial Education Department (Nos. GJJ211921, GJJ201917 and GJJ190941), and the National Science Foundation of China (Nos. 61763032 and 61562061).

Author Contributions Paper writing and experiment implementation, LW (second author); methodology, SD and LW (second author); revising and reviewing, LW (fifth author); supervision, ML and SG; funding acquisition, SD.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Availability of Data and Materials Due to the large number of data and materials, the data and materials generated during this study are not public, but can be obtained from the corresponding authors according to reasonable requirements.

Declarations

Conflict of Interest The authors have no relevant financial or non-financial interests to disclose.

Ethics Approval and Consent to Participate I warrant, on behalf of myself and my co-authors, that: (1) the article is original, has not been formally published in any other peer-reviewed journal, is not under consideration by any other journal and does not infringe any existing copyright or any other third party rights; (2) I am/we are the sole author(s) of the article and have full authority to enter into this agree-

ment and in granting rights to Springer are not in breach of any other obligation; (3) the article contains nothing that is unlawful, libelous, or which would, if published, constitute a breach of contract or of confidence or of commitment given to secrecy; (4) I/we have taken due care to ensure the integrity of the article. To my/our—and currently accepted scientific—knowledge, all statements contained in it purporting to be facts are true and any formula or instruction contained in the article will not, if followed accurately, cause any injury, illness or damage to the user. Signature: WangLei Date:2022.3.8.

Consent for Publication I, and all the co-authors, agree that the article, if editorially accepted for publication, shall be licensed under the Creative Commons Attribution License 4.0. If the law requires that the article be published in the public domain, I/we will notify Springer at the time of submission, and in such cases the article shall be released under the Creative Commons 1.0 Public Domain Dedication waiver. For the avoidance of doubt, it is stated that sections 1 and 2 of this license agreement shall apply and prevail regardless of whether the article is published under Creative Commons Attribution License 4.0 or the Creative Commons 1.0 Public Domain Dedication waiver. I, and all co-authors, agree that, if the article is editorially accepted for publication in Chemistry Central Journal, Chemical and Biological Technologies in Agriculture, Geochemical Transactions, Heritage Science, Journal of Cheminformatics, or Sustainable Chemical Processes, data included in the article shall be made available under the Creative Commons 1.0 Public Domain Dedication waiver, unless otherwise stated. Signature: WangLei Date: 2020.3.8.

Code Availability Due to the large number of algorithm codes, the codes generated and analyzed during this study are not public, but can be obtained from the corresponding authors according to reasonable requirements.

References

1. Dong, S., Sarem, M.: Nocd: a new overlapping community detection algorithm based on improved knn. J. Ambient. Intell. Humaniz. Comput. **13**, 3053–3063 (2022)
2. Dong, S., Sarem, M.: Ddos attack detection method based on improved knn with the degree of ddos attack in software-defined networks. IEEE Access **8**, 5039–5048 (2019)
3. Dudani, A, S.: The distance-weighted k-nearest-neighbor rule. IEEE Transactions on Systems, Man, and Cybernetics SMC-6(4), 325–327 (1976)
4. Zuo, Z.D.W., Wang, K.: On kernel difference-weighted k-nearest neighbor classification. Pattern. Anal. Applic. **11**, 247–257 (2008)
5. Gou, J., Xiong, T., Kuang, Y.: A novel weighted voting for k-nearest neighbor rule. J. Comput. **6**(5), 833–840 (2011)
6. Mitani, Y., Hamamoto, Y.: A local mean-based nonparametric classifier. Pattern Recogn. Lett. **27**(10), 1151–1159 (2006)
7. García-Pedrajas, N., Del Castillo, J.A.R., Cerruela-García, G.: A proposal for local *k* values for *k*-nearest neighbor rule. IEEE transactions on neural networks and learning systems **28**(2), 470–475 (2015)
8. Zhang, S., Li, X., Zong, M., Zhu, X., Wang, R.: Efficient knn classification with different numbers of nearest neighbors. IEEE transactions on neural networks and learning systems **29**(5), 1774–1785 (2017)
9. Mullick, S.S., Datta, S., Das, S.: Adaptive learning-based *k*-nearest neighbor classifiers with resilience to class imbalance. IEEE transactions on neural networks and learning systems **29**(11), 5713–5725 (2018)
10. Zeng, Y., Yang, Y., Zhao, L.: Pseudo nearest neighbor rule for pattern classification. Expert Syst. Appl. **36**(2), 3587–3595 (2009)
11. Pan, Z., Wang, Y., Pan, Y.: A new locally adaptive k-nearest neighbor algorithm based on discrimination class. Knowl.-Based Syst. **204**, 106185 (2020)
12. Weinberger, K.Q., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. Advances in neural information processing systems **18** (2005)
13. Xiao, X., Ding, H.: Enhancement of K-nearest Neighbor Algorithm Based on Weighted Entropy of Attribute Value, pp. 1261–1264 (2012)
14. Sharma, K.K., Seal, A.: Spectral Embedded Generalized Mean Based K-nearest Neighbors Clustering with S-distance **169**, 114326 (2021)
15. Tang, M., Pérez-Fernández, R., De Baets, B.: Distance metric learning for augmenting the method of nearest neighbors for ordinal classification with absolute and relative information. Information Fusion **65**, 72–83 (2021)
16. Nguyen, B., Morell, C., De Baets, B.: Distance metric learning for ordinal classification based on triplet constraints. Knowl.-Based Syst. **142**, 17–28 (2018)
17. Syaliman, K., Nababan, E., Sitompul, O.: Improving the accuracy of k-nearest neighbor using local mean based and distance weight **978**(1), 012047 (2018)
18. Anvari, S., Abdollahi Azgomi, M., Ebrahimi Dishabi, M., Maheri, M.: (2205-7400) weighted k-nearest neighbors classification based on whale optimization algorithm. Iranian Journal of Fuzzy Systems (2023)
19. Istiadi, I., Rahman, A.Y., Wisnu, A.D.R.: Identification of tempe fermentation maturity using principal component analysis and k-nearest neighbor. Sinkron: jurnal dan penelitian teknik informatika 8(1), 286–294 (2023)
20. Sharma, K.K., Seal, A.: Clustering Analysis Using an Adaptive Fused Distance **96**, 103928 (2020)
21. Zhang, S., Li, X., Zong, M., Zhu, X., Wang, R.: Efficient knn classification with different numbers of nearest neighbors. IEEE transactions on neural networks and learning systems **29**(5), 1774–1785 (2017)
22. Wagner, T.: Convergence of the nearest neighbor rule. IEEE Trans. Inf. Theory **17**(5), 566–571 (1971)
23. Yu, Z., Chen, H., Liu, J., You, J., Leung, H., Han, G.: Hybrid *k*-nearest neighbor classifier. IEEE transactions on cybernetics **46**(6), 1263–1275 (2015)
24. Ma, H., Gou, J., Wang, X., Ke, J., Zeng, S.: Sparse coefficient-based *k* -nearest neighbor classification. IEEE Access **5**, 16618–16634 (2017)
25. Sánchez, J.S., Pla, F., Ferri, F.J.: On the use of neighbourhood-based non-parametric classifiers. Pattern Recogn. Lett. **18**(11–13), 1179–1186 (1997)
26. Azadifar S, B.K.e.a. Rostami M: Graph-based relevancy-redundancy gene selection method for cancer diagnosis. Computers in Biology and Medicine 147, 105766 (2022)

27. Saberi-Movahed, F., Rostami, M., Berahmand, K., Karami, S., Tiwari, P., Oussalah, M., Band., S.S.: Dual regularized unsupervised feature selection based on matrix factorization and minimum redundancy with application in gene selection. Knowledge-Based Systems 256, 109884 (2022)
28. Biswas, N., Chakraborty, S., Mullick, S.S., Das, S.: A parameter independent fuzzy weighted k-nearest neighbor classifier. Pattern Recogn. Lett. **101**, 80–87 (2018)
29. Karlekar, A., Seal, A., Krejcar, O., Gonzalo-Martin, C.: Fuzzy k-means using non-linear s-distance. IEEE Access **7**, 55121–55131 (2019)
30. Cordón, I., García, S., Fernández, A., Herrera, F.: Imbalance: Oversampling algorithms for imbalanced classification in r. Knowl.-Based Syst. **161**, 329–341 (2018)
31. Tao, X., Wang, R., Chang, R., Li, C.: Density-sensitive fuzzy kernel maximum entropy clustering algorithm. Knowl.-Based Syst. **166**, 42–57 (2019)
32. Gou, J., Zhan, Y., Rao, Y., Shen, X., Wang, X., He, W.: Improved pseudo nearest neighbor classification. Knowl.-Based Syst. **70**, 361–375 (2014)
33. Pan, Z., Pan, Y., Wang, Y., Wang, W.: A new globally adaptive k-nearest neighbor classifier based on local mean optimization. Soft. Comput. **25**(3), 2417–2431 (2021)
34. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)