Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.026113

ARTICLE

An Adaptive Parameter-Free Optimal Number of Market Segments Estimation Algorithm Based on a New Internal Validity Index

Jianfang Qi[1], Yue Li[1,3], Haibin Jin[1], Jianying Feng[1], Dong Tian[1] and Weisong Mu[1,2,*]

[1]College of Information and Electrical Engineering, China Agricultural University, Beijing, 100083, China

[2]Key Laboratory of Viticulture and Enology, Ministry of Agriculture, Beijing, 100083, China

[3]Institute of Population Research, Peking University, Beijing, 100871, China

*Corresponding Author: Weisong Mu. Email: wsmu@cau.edu.cn

Received: 16 August 2022 Accepted: 16 December 2022

ABSTRACT

An appropriate optimal number of market segments (ONS) estimation is essential for an enterprise to achieve successful market segmentation, but at present, there is a serious lack of attention to this issue in market segmentation. In our study, an independent adaptive ONS estimation method BWCON-NSDK-means++ is proposed by integrating a new internal validity index (IVI) Between-Within-Connectivity (BWCON) and a new stable clustering algorithm Natural-SDK-means++ (NSDK-means++) in a novel way. First, to complete the evaluation dimensions of the existing IVIs, we designed a connectivity formula based on the neighbor relationship and proposed the BWCON by integrating the connectivity with other two commonly considered measures of compactness and separation. Then, considering the stability, number of parameters and clustering performance, we proposed the NSDK-means++ to participate in the integration where the natural neighbor was used to optimize the initial cluster centers (ICCs) determination strategy in the SDK-means++. At last, to ensure the objectivity of the estimated ONS, we designed a BWCON-based ONS estimation framework that does not require the user to set any parameters in advance and integrated the NSDK-means++ into this framework forming a practical ONS estimation tool BWCON-NSDK-means++. The final experimental results show that the proposed BWCON and NSDK-means++ are significantly more suitable than their respective existing models to participate in the integration for determining the ONS, and the proposed BWCON-NSDK-means++ is demonstrably superior to the BWCON-KMA, BWCON-MBK, BWCON-KM++, BWCON-RKM++, BWCON-SDKM++, BWCON-Single linkage, BWCON-Complete linkage, BWCON-Average linkage and BWCON-Ward linkage in terms of the ONS estimation. Moreover, as an independent market segmentation tool, the BWCON-NSDK-means++ also outperforms the existing models with respect to the inter-market differentiation and sub-market size.

KEYWORDS

Optimal number of market segments; internal validity index; cluster connectivity; SDK-means++; market segmentation

1 Introduction

Market segmentation is an important marketing tool that aims to divide a large heterogeneous market into several small homogeneous markets so that consumers located in the same sub-market have similar demands and consumers located in different sub-markets have different demands, and enterprises can focus their resources on the most appropriate target market to maximize their interests [1]. So far, there have been many studies on the market segmentation bases and methods, but few studies on how to determine the optimal number of market segments (ONS) [2–4]; however, in practice, a reasonable ONS is a premise for carrying out a valuable market segmentation, because (1) different ONSs will directly change the composition of the market segmentation results, and the unrealistic sub-markets will mislead the enterprises to make relevant decisions, and (2) most current market segmentation methods need to be provided with the ONS in advance. Obviously, it is practical to strengthen the studies on how to determine an appropriate ONS.

In the state-of-the-art studies, there are two main ways to determine the ONS: (1) the researchers directly specify an ONS or a range based on their prior knowledge [5], and (2) the combination of a validity index and a clustering algorithm is used to determine the ONS due to a fact that the clustering analysis is the current dominant market segmentation technique and most clustering algorithms also require a pre-specified number of clusters (NC), which makes the process of finding the optimal NC (ONC) coincide with the purpose of determining the ONS [6]. Compared with the former, this combination strategy is more objective and user-friendly; and its basic idea is to perform a clustering algorithm many times with different NCs, choose an appropriate validity index to evaluate the clustering results, and determine the ONC (ONS) according to the evaluation values [7,8], in which the selections of the validity index and the clustering algorithm play the very important roles in whether a reasonable ONC can be obtained. And in our study, we also try to propose an effective method for determining the ONC based on such combinatorial idea.

For the validity indices, there are two main types: the external validity index (EVI) and internal validity index (IVI) [9]. Since the former is used with known data labels, while the latter is independent of the data labels, the IVIs can often be combined with the clustering algorithms to determine the ONC, where the Dunn index, Davies–Bouldin (DB) index, Silhouette (Sil) index and Calinski-Harabasz (CH) index are the most commonly used IVIs in the current literature [10]; and all of them evaluate the clustering results from two aspects, compactness and separation. Based on the Sil, Zhou et al. [11] proposed a new IVI CIP that can obtain more accurate ONC than the DB, Sil, Krzanowski-Lai (KL) index, Weighted inter-intra (Wint) index and In-group proportion (IGP) index with different clustering algorithms; particularly, they pointed out that although the CIP still mainly considers the compactness and separability, a valid IVI should consider compactness, separation and connectivity simultaneously and they provided the concept of the cluster connectivity qualitatively. In a recent study, Zhou et al. [7] also proposed a valid IVI BWC and it has been proved to be significantly better than the CH, DB, Sil, KL, Wint, Hartigan (Hart) index, IGP, Dunn and PBM index in terms of the ONC estimation, however, its definition still does not reflect the cluster connectivity. Unlike the compactness and the separation, the cluster connectivity refers to the fact that a sample should be classified into the same cluster with its neighboring samples, and they evaluate the reasonableness of the clustering results from three different perspectives. Thus, it is possible to quantitatively design a definition of the cluster connectivity and integrate it into the IVI to further improve the accuracy of the estimated ONC (EONC).

In terms of the clustering algorithms, the K-means algorithm (KMA) and its variants are commonly used clustering algorithms in combination with the IVIs for choosing the ONC since they are simple to implement and have linear time complexity [7]. For example, He et al. [12] combined the Sil with the KMA to estimate the ONC; Sheikhhosseini et al. [13] also chose the KMA as the basic algorithm from numerous clustering algorithms, and further combined it with the Davies–Bouldin's measure and Chou–Su–Lai's measure together to determine the ONC. Nevertheless, since the KMA is sensitive to the initial cluster centers (ICCs), and different ICCs tend to lead to different clustering results, as well as different clustering results will cause unstable ONCs, the traditional KMA is not the best choice for determining the ONC in combination with the IVIs [14]. In a recent study, Du et al. [15] successfully integrated the DB with a new clustering algorithm SDK-means++ to detect the ONC. Compared with the existing models, the SDK-means++ is stable and does not require setting any parameters, but it still suffers from two major shortcomings: (1) when calculating the first ICC, the setting of the parameter cut-off distance lacks theoretical support; and (2) when determining the remaining ICCs, the noises and edge points are more easily identified as ICCs, thus affecting the number of iterations (NI) of the algorithm and the final clustering performance. Therefore, it is necessary to further optimize the SDK-means++ to make it more suitable for determining the ONC in combination with the IVIs.

In our study, we proposed an adaptive parameter-free BWCON-NSDK-means++ algorithm for determining the ONC by integrating a new IVI and a new stable clustering algorithm. First, since the existing IVIs ignore the cluster connectivity, to complete the evaluation dimensions of the existing IVIs and evaluate a clustering result more comprehensively, we designed a cluster connectivity formula for the first time based on the neighbor relationship, and thus defined a new IVI, Between-Within-Connectivity (BWCON), to evaluate the clustering results from three dimensions of compactness, separation and connectivity. Then, to address the aforementioned problems in the SDK-means++, we chose the sample with the most natural neighbors as the first ICC to solve the parameter problem, and introduced a modification mechanism in the original SDK-means++ to address the noise and the edge point problems, and finally proposed the Natural-SDK-means++ (NSDK-means++) algorithm for combination with the IVI. At last, considering that the existing combination strategies usually take the NC corresponding to the maximum or minimum evaluation value as the ONC, ignoring the fact that such values are easily disturbed by the special points or clusters in the clustering results, we developed a novel ONC estimation framework based on the proposed BWCON, and proposed an independent and objective ONC estimation algorithm BWCON-NSDK-means++ by effectively integrating the NSDK-means++ into this framework. The experimental results obtained show that the BWCON index is able to better estimate the ONC than those previous IVIs such as Dunn, DB, Sil, CH, BWP, CIP and BWC with different clustering algorithms on the seven UCI datasets and five synthetic datasets; on these relatively complex UCI datasets, the proposed NSDK-means++ can achieve the better clustering performance than other five existing clustering algorithms in terms of nine validity indices; moreover, the BWCON-NSDK-means++ is proved to be able to obtain more accurate ONC than the BWCON-KMA, BWCON-MBK, BWCON-KM++, BWCON-RKM++, BWCON-SDKM++, BWCON-Single linkage, BWCON-Complete linkage, BWCON-Average linkage and BWCON-Ward linkage algorithms on all twelve datasets, and can be used as a stand-alone market segmentation tool.

The rest of this paper is organized as follows. The overview of different IVIs and clustering algorithms is presented in Section 2. Section 3 introduces the proposed BWCON-NSDK-means++ algorithm in detail. Experimental results are provided in Sections 4 and 5. Finally, Section 6 summarizes the paper.

2 Related Work

In the following subsections, different IVIs and clustering algorithms are delineated.

2.1 Internal Validity Indices

Many IVIs have been proposed to evaluate the clustering performance and combined with the clustering algorithms to determine the ONC. Dunn [16] proposed the Dunn by combining the within-cluster and between-cluster distances, in which the within-cluster distance is estimated by calculating the maximum distance in any cluster, and the between-cluster distance is measured by calculating the shortest distance between samples in any two clusters; the larger the Dunn, the better the clustering results. In 1979, Davies et al. [17] proposed the DB, which is obtained from the ratio of within-cluster compactness and between-cluster separation; the minimization of the index shows better clustering partitions. In [18], Rousseeuw proposed the well-known Sil. Similar to the previous two indices, the Sil also evaluates the clustering results from two dimensions of compactness and separation; it measures the compactness of a sample by calculating the average distance from this sample to the other samples in this cluster and the separation of a sample by calculating the minimum value of average distance between this sample and samples in every other cluster. The Sil obtains its maximum value when the ONC is achieved. In 1974, Calinski et al. [19] proposed the CH that introduced the within-groups sum of squared and the between-groups sum of squared error to measure the compactness and separation; and similarly, the ONC corresponds to the maximum CH. In 2011, Zhou et al. [20] proposed the BWP index, in which the compactness and separation measures are exactly the same as those of the Sil, and the only difference between them is that their normalization methods are different: the BWP used the clustering distance to normalize itself; theoretical studies and experimental results have shown that the BWP significantly outperforms the DB, KL, Homogeneity-Separation (HS) and IGP in terms of the ONC estimation. To improve the time performance of the Sil and BWP, Zhou et al. [11] proposed a new IVI, CIP, based on the concept of the sample geometry, which measures the compactness of a sample by calculating the distance from this sample to the centroid of this cluster, and the separation of a sample by calculating the minimum distance between this sample and every centroid of other clusters; the obtained results show that the CIP can obtain the accurate NC more quickly than the existing indices. In a recent study, Zhou et al. [7] further improved the Sil and proposed the BWC that still evaluates the clustering results from two aspects of compactness and separation, where it measures the compactness of a cluster by calculating the average distance between every sample of this cluster and its centroid, and the separation of a cluster by calculating the minimum distance between the centroid and every centroid of other clusters; and the ONC is obtained at the maximum BWC under different NCs.

It can be seen that in terms of the ONC estimation, although the validity of all of the above indices has been proved on various datasets with different clustering algorithms, none of them consider the cluster connectivity. The cluster connectivity measures the reasonableness of the clustering results from a neighborhood perspective, and it is of great practical significance to give a quantitative formula for calculating the connectivity and incorporate it into the IVIs to evaluate the clustering results together with the compactness and the separation. In addition, when searching for the ONC, the existing IVIs often determine the ONC by finding their maximum or minimum values under different NCs, however, such values are easily dominated by the locations of the special points or clusters in the clustering results, which will in turn directly affect the final estimates.

2.2 Clustering Algorithms

In the literature, there are two main types of clustering algorithms that are often combined with IVIs to determine the ONC: hierarchical and partitional clustering [8]. For the hierarchical clustering, according to the direction of clustering, there are two types of methods: agglomerative hierarchical clustering (AHC) and divisive hierarchical clustering (DHC) [21], where the former follows the bottom-top strategy, which treats each sample as a complete cluster at the beginning, and then gradually merges them into some larger cluster based on a certain criterion, and on the contrary, the DHC adopts the top-down strategy, which initially regards the entire dataset as a complete cluster and then splits the dataset into some smaller clusters based on a certain criterion [22]. Compared with the DHC, the AHC is more accurate and widely used [23], and in recent years, the classic AHC with single linkage, complete linkage, average linkage and ward linkage are still the most widely used AHC methods [24,25]. In general, the AHC is simple in idea and has the stable clustering performance, but it is not suitable for clustering large datasets due to its high time complexity [24].

While among the partitional clustering algorithms, the KMA is currently the most popular one, and its implementation consists of five main steps: (1) specify the NC m; (2) select m samples randomly as the ICCs; (3) classify each sample into the cluster where its nearest center is located; (4) update the clustering centers by treating the mean of each cluster as a new center; and (5) repeat Steps (3) and (4) until the clustering centers no longer change [26]. In a subsequent study, Sculley [27] improved the KMA and proposed the Mini-batch K-means (MBK) to accelerate the clustering speed by randomly selecting a subset instead of the whole dataset to train the ICCs, but like KMA, its clustering performance is unstable and very sensitive to the ICCs [28]. A representative partition-based clustering algorithm K-means++ proposed by Arthur and Vassilvitskii provided such an effective solution to determine the ICCs, as shown in Algorithm 1, in which the ICCs are determined based on a D^2 weighting method following the principle that the larger the distances among the ICCs, the more reasonable the selection of the ICCs, and it can effectively reduce the possibility of multiple ICCs appearing in the same cluster [29]. However, due to the existence of the line 1 in the Algorithm 1, the K-means++ still has a certain degree of randomness.

Algorithm 1: K-means++

Input: given dataset X, NC m
Output: Clustering results $C = \{C_1, \cdots, C_m\}$
1: Choose a point from X randomly as the first ICC
2: **Repeat:**
3: $Sum = 0$
4: for $x \in S$ do: // S denotes the samples in the X other than the ICCs that have been determined
5: Calculate the distance $D^2(x)$ between x and the nearest cluster center
6: $Sum = Sum + D^2(x)$
7: Choose the next center with largest probability $\frac{D^2(x)}{Sum}$
8: **Until** the m centers are chosen
9: Assign each sample in the S to the cluster where its nearest center is located to obtain the clustering result
$C = \{C_1, \cdots, C_m\}$ //$C_1, \cdots, C_m$ are m mutually disjoint sets, and $\bigcup_i^m C_i = X, i = 1, \ldots, m$
10: Update the clustering centers by treating the mean of each cluster in the C as a new center
11: Repeat lines 9 and 10 until the centers unchanged
12: **Return** Clustering results $C = \{C_1, \cdots, C_m\}$

In [30], an alternative way to determine the ICCs is introduced in the K-means++, called RK-means++ in our study that determines the ICCs by randomly selecting a random value, and then weighting to calculate the ICCs; it is still unstable since (1) the first ICC is generated randomly, and (2) the determination of the remaining ICCs requires first choosing a small random number at random. To completely solve the randomness problem, the SDK-means++ was proposed in 2021 based on the DB, maximum density and the largest sum of distance, as shown in Algorithm 2, in which the IVI DB is used to adaptively determine the ONC to avoid artificially setting the parameter NC, the maximum density is utilized to determine the first ICC to avoid the randomness, and the largest sum of distance is designed to ensure that all ICCs are generated in different clusters, but as mentioned in the introduction, its parameter, noise and edge point issues can be further optimized [15].

Algorithm 2: SDK-means++

Input: given dataset X
Output: Clustering results $R[m]$
1: Initial an empty list R // R is used to store the clustering results corresponding to different NCs
2: Initial an empty list D // D is used to store the DB values corresponding to different NCs
3: **for** $k = 2$ to N **do**: // N is the number of samples in the X
4: Choose the first ICC c_1 from X based on the maximum density
5: Initial an empty list Cen // Cen is used to store the ICCs
6: Store c_1 into the $Cen[1]$
7: **for** $i = 2$ to k do:
8: Choose the next center c_i based on the largest sum of the distance
9: Store the c_i into the $Cen[i]$
10: **end for**
11: Assign each sample to the cluster where its nearest center is located to obtain the clustering result
$C = \{C_1, \cdots, C_k\}$ //$C_1, \cdots, C_k$ are k mutually disjoint sets, and $\bigcup_i^k C_i = X, i = 1, \ldots, k$
12: Update the clustering centers by treating the mean of each cluster in the C as a new center
13: Repeat lines 11 and 12 until the centers unchanged
14: Output the clustering results $C = \{C_1, \cdots, C_k\}$
15: Store the clustering results into the $R[k]$
16: Evaluate the clustering results based on the DB
17: Store the DB value into the $D[k]$
18: $k = k + 1$
19: **end for**
20: Determine the ONC m based on the D
21: **Return** Clustering results $R[m]$

3 Proposed Work

In all the following works, the Euclidean distance is used to evaluate the dissimilarity.

3.1 The IVI: The BWCON Index

Definition 1 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the centroid μ_k of the cluster k is defined as the mean value of all samples in the cluster k, i.e.,

$$\mu_k = \frac{1}{n_k} \sum\nolimits_{p=1}^{n_k} x_p^{(k)} \tag{1}$$

where k represents the cluster label, $x_p^{(k)}$ represents the pth sample in the cluster k, and n_k represents the number of samples in the cluster k.

Definition 2 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the within-cluster distance wc(j) of cluster j is defined as the average distance between every point in the cluster j and the centroid of cluster j, i.e.,

$$\mathrm{wc(j)} = \frac{1}{\mathrm{n_j}} \sum\nolimits_{\mathrm{i=1}}^{\mathrm{n_j}} \left\| \mathrm{X_i^{(j)}} - \mu_{\mathrm{j}} \right\| \tag{2}$$

where j represents the cluster label, $x_i^{(j)}$ represents the ith sample in the cluster j, $\|.\|$ represents the Euclidean distance, and n_j represents the number of samples in the cluster j.

Definition 3 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the between-cluster distance bc(j) of cluster j is defined as the average distance between the centroid of cluster j and every centroid of other clusters, i.e.,

$$\mathrm{bc(j)} = \frac{1}{\mathrm{m} - 1} \sum\nolimits_{1 \le \mathrm{k} \le \mathrm{m}, \mathrm{k} \ne \mathrm{j}} \left\| \mu_{\mathrm{j}} - \mu_{\mathrm{k}} \right\| \tag{3}$$

where j and k represent the cluster label.

Definition 4 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the connectivity $\mathrm{con(j, i)}$ of $x_i^{(j)}$ is defined as the ratio of the length of the intersection of the $KNN_{\mathrm{T}}(x_i^{(j)})$ and $KNN_{\mathrm{T}}(x_i^{(j)})'$ to T, i.e.,

$$\mathrm{con(j, i)} = \frac{length(KNN_{\mathrm{T}}(x_i^{(j)}) \cap KNN_{\mathrm{T}}(x_i^{(j)})')}{\mathrm{T}} \tag{4}$$

where the connectivity $\mathrm{con(j, i)}$ of sample $x_i^{(j)}$ is the ratio of the number of samples that are classified into the j among all T nearest neighbors of $x_i^{(j)}$ to T. Specifically, T is the number of neighboring samples determined using the Algorithm 1 in [31] that is an adaptive method for determining the number of nearest neighbors for a dataset to be clustered, $KNN_{\mathrm{T}}(x_i^{(j)})$ represents the T nearest samples to $x_i^{(j)}$ in the whole dataset and $KNN_{\mathrm{T}}(x_i^{(j)})'$ denotes the T nearest samples to $x_i^{(j)}$ in the cluster j. Moreover, when $n_j \le \mathrm{T}$, $\mathrm{KNN_T}(x_i^{(j)})'$ are all samples in the cluster j.

Definition 5 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the connectivity con(j) of cluster j is defined as the mean of the connectivity values of all samples in the cluster j, i.e.,

$$\mathrm{con(j)} = \frac{1}{\mathrm{n_j}} \sum\nolimits_{\mathrm{i=1}}^{\mathrm{n_j}} \mathrm{con(j, i)} \tag{5}$$

where $\mathrm{n_j}$ represents the number of samples in the cluster j.

Definition 6 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the BWCON of cluster j BWCON(j) is as follows:

$$\mathrm{BWCON(j)} = [\mathrm{wc(j)}, \mathrm{bc(j)}, \mathrm{con(j)}] \tag{6}$$

In Definition (6), the BWCON consists of three evaluation dimensions. For the compactness, we use wc(j) defined in (2) to reflect the overall compactness of cluster j, and the smaller value of wc(j) represents the high cluster compactness. For the separation, the bc(j) defined in (3) is used to reflect the between-cluster separation of cluster j; and the large value of bc(j) shows the high cluster separation. Compared with the separation in the BWC, we replace the minimum distance between the centroid of cluster j and the centroids of other clusters in the clustering results with the average distance, which can better reflect the structural relationship between this cluster and other clusters on the whole. As for the cluster connectivity, we use con(j, i) defined in (4) to reflect the connectivity of sample $x_i^{(j)}$, and use con(j) defined in (5) to reflect the connectivity of cluster j. Extremely, if the T nearest neighbors of a sample in its cluster are exactly the same as its T nearest neighbors in the entire dataset, we consider that this sample is correctly classified in the dimension of connectivity; obviously, the larger the con(j) is, the more reasonable the clustering results are.

To understand the BWCON intuitively, we illustrate the BWCON by providing a distribution diagram of Fig. 1.

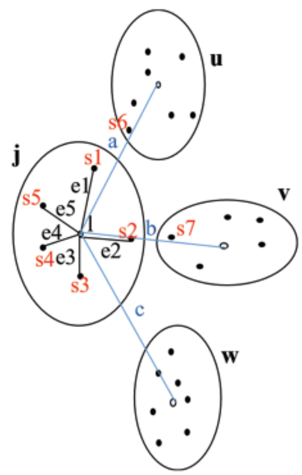


Figure 1: Distribution diagram of clustering structure for the BWCON index

In Fig. 1, the dataset consists of four clusters: j, u, v and w, the centroid of j is marked by l, the centroids of the other three clusters are represented by the hollow circles, and since the dataset is small, the T is directly specified as 1. According to Definitions 2 and 3, we can obtain that $\mathrm{wc(j)} = \dfrac{e_1 + e_2 + e_3 + e_4 + e_5}{5}$ and $\mathrm{bc(j)} = \dfrac{a + b + c}{3}$; according to Definitions 4 and 5, we can get that $\mathrm{con(j,s1)} = \dfrac{|KNN_1(s1) \cap KNN_1(s1)'|}{1} = |\{s6\} \cap \{s5\}| = 0$, $\mathrm{con(j,s2)} = \dfrac{|KNN_1(s2) \cap KNN_1(s2)'|}{1} = |\{s7\} \cap \{s3\}| = 0$, $\mathrm{con(j,s3)} = \dfrac{|KNN_1(s3) \cap KNN_1(s3)'|}{1} = |\{s4\} \cap \{s4\}| = 1$, $\mathrm{con(j,s4)} = \dfrac{|KNN_1(s4) \cap KNN_1(s4)'|}{1} = |\{s5\} \cap \{s5\}| = 1$, $\mathrm{con(j,s5)} = \dfrac{|KNN_1(s5) \cap KNN_1(s5)'|}{1} = |\{s4\} \cap \{s4\}| = 1$, and $\mathrm{con(j)} = \dfrac{0+0+1+1+1}{5} = 0.6$; and ultimately, according to Eq. (6), the BWCON of j is $\left[\dfrac{e_1 + e_2 + e_3 + e_4 + e_5}{5}, \dfrac{a + b + c}{3}, 0.6\right]$.

Furthermore, to evaluate the clustering validity of a whole dataset, the avgBWCON(m) function is defined.

Definition 7 Let dataset $X = \{x_1, x_2, \ldots, x_N\}$, and x_i is the ith sample. Assuming that N samples are clustered into m clusters, the avgBWCON(m) is defined as the average BWCON values of all clusters in the clustering results, i.e.,

$$\mathrm{avgBWCON(m)} = \left[\frac{1}{\mathrm{m}}\sum\nolimits_{\mathrm{j=1}}^{\mathrm{m}} \mathrm{wc(j)}, \frac{1}{\mathrm{m}}\sum\nolimits_{\mathrm{j=1}}^{\mathrm{m}} \mathrm{bc(j)}, \frac{1}{\mathrm{m}}\sum\nolimits_{\mathrm{j=1}}^{\mathrm{m}} \mathrm{con(j)}\right] \tag{7}$$

3.2 The BWCON-Based ONC Estimation Framework

The main steps for estimating the ONC based on the BWCON are presented as follows:

***Step 1*:** Cluster the input dataset under different NCs.

Let N be the number of samples, k is the NC, and the range of k is $\left[2, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor\right]$ as in [7,8,11]. The object of this step is to get different clustering results corresponding to different NCs under a certain clustering algorithm.

***Step 2*:** Evaluate the above clustering results according to the BWCON.

In this Step, a BWCON matrix is generated, written as

$$\begin{pmatrix} \mathrm{avgBWCON(2)} \\ \mathrm{avgBWCON(3)} \\ \vdots \\ \mathrm{avgBWCON}(\lfloor\sqrt{\mathrm{N}}\rfloor) \end{pmatrix} = \begin{pmatrix} \left[\frac{1}{2}\sum\nolimits_{\mathrm{j=1}}^{2} \mathrm{wc(j)}, \frac{1}{2}\sum\nolimits_{\mathrm{j=1}}^{2} \mathrm{bc(j)}, \frac{1}{2}\sum\nolimits_{\mathrm{j=1}}^{2} \mathrm{con(j)}\right] \\ \left[\frac{1}{3}\sum\nolimits_{\mathrm{j=1}}^{3} \mathrm{wc(j)}, \frac{1}{3}\sum\nolimits_{\mathrm{j=1}}^{3} \mathrm{bc(j)}, \frac{1}{3}\sum\nolimits_{\mathrm{j=1}}^{3} \mathrm{con(j)}\right] \\ \vdots \\ \left[\frac{1}{\lfloor\sqrt{\mathrm{N}}\rfloor}\sum\nolimits_{\mathrm{j=1}}^{\lfloor\sqrt{\mathrm{N}}\rfloor} \mathrm{wc(j)}, \frac{1}{\lfloor\sqrt{\mathrm{N}}\rfloor}\sum\nolimits_{\mathrm{j=1}}^{\lfloor\sqrt{\mathrm{N}}\rfloor} \mathrm{bc(j)}, \frac{1}{\lfloor\sqrt{\mathrm{N}}\rfloor}\sum\nolimits_{\mathrm{j=1}}^{\lfloor\sqrt{\mathrm{N}}\rfloor} \mathrm{con(j)}\right] \end{pmatrix} \tag{8}$$

Further, Eq. (8) is simplified as follows:

$$\begin{pmatrix} \mathrm{avgBWCON(2)} \\ \mathrm{avgBWCON(3)} \\ \vdots \\ \mathrm{avgBWCON}\left(\left\lfloor\sqrt{\mathrm{N}}\right\rfloor\right) \end{pmatrix} = \begin{pmatrix} wc_2, bc_2, con_2 \\ wc_3, bc_3, con_3 \\ \vdots \\ wc_{\lfloor\sqrt{\mathrm{N}}\rfloor}, bc_{\lfloor\sqrt{\mathrm{N}}\rfloor}, con_{\lfloor\sqrt{\mathrm{N}}\rfloor} \end{pmatrix}_{(\lfloor\sqrt{\mathrm{N}}\rfloor-1)\times 3} \tag{9}$$

***Step 3*:** Standardize the BWCON matrix.

Since the change trends of the compactness, separation and connectivity are different, they are standardized differently so that the larger the values of the standardized compactness, separation and connectivity, the better the clustering performance. Related calculations are shown in the Eqs. (10)–(12), and (13) is the standardized BWCON matrix.

$$Std_{wc_l} = \frac{max_i wc_i - wc_l}{max_i wc_i - min_i wc_i}, l = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor, i = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor \tag{10}$$

$$Std_{bc_l} = \frac{bc_l - min_i bc_i}{max_i bc_i - min_i bc_i}, l = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor, i = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor \tag{11}$$

$$Std_{con_l} = \frac{con_l - min_i con_i}{max_i con_i - min_i con_i}, l = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor, i = 2, 3, \ldots, \left\lfloor\sqrt{\mathrm{N}}\right\rfloor \tag{12}$$

$$\text{Std}\begin{pmatrix} \text{avgBWCON}(2) \\ \text{avgBWCON}(3) \\ \vdots \\ \text{avgBWCON}\left(\left\lfloor\sqrt{\text{N}}\right\rfloor\right) \end{pmatrix} = \begin{pmatrix} Std_{wc_2}, Std_{bc_2}, Std_{con_2} \\ Std_{wc_3}, Std_{bc_3}, Std_{con_3} \\ \vdots \\ Std_{wc_{\lfloor\sqrt{\text{N}}\rfloor}}, Std_{bc_{\lfloor\sqrt{\text{N}}\rfloor}}, Std_{con_{\lfloor\sqrt{\text{N}}\rfloor}} \end{pmatrix}_{(\lfloor\sqrt{\text{N}}\rfloor-1)\times 3} \tag{13}$$

***Step 4*: Estimate the ONCs under different dimensions in turn.**

In this step, the compactness, separation and connectivity matrices, diff_{com}, diff_{sep} and diff_{con}, are generated as follows:

$$\text{diff}_{\text{com}} = \begin{pmatrix} Std_{wc_3} - Std_{wc_2} \\ Std_{wc_4} - Std_{wc_3} \\ \vdots \\ Std_{wc_{\lfloor\sqrt{\text{N}}\rfloor}} - Std_{wc_{\lfloor\sqrt{\text{N}}\rfloor-1}} \end{pmatrix}_{(\lfloor\sqrt{\text{N}}\rfloor-2)\times 1} \tag{14}$$

$$\text{diff}_{\text{sep}} = \begin{pmatrix} Std_{bc_3} - Std_{bc_2} \\ Std_{bc_4} - Std_{bc_3} \\ \vdots \\ Std_{bc_{\lfloor\sqrt{\text{N}}\rfloor}} - Std_{bc_{\lfloor\sqrt{\text{N}}\rfloor-1}} \end{pmatrix}_{(\lfloor\sqrt{\text{N}}\rfloor-2)\times 1} \tag{15}$$

$$\text{diff}_{\text{con}} = \begin{pmatrix} Std_{con_3} - Std_{con_2} \\ Std_{con_4} - Std_{con_3} \\ \vdots \\ Std_{con_{\lfloor\sqrt{\text{N}}\rfloor}} - Std_{con_{\lfloor\sqrt{\text{N}}\rfloor-1}} \end{pmatrix}_{(\lfloor\sqrt{\text{N}}\rfloor-2)\times 1} \tag{16}$$

Further, the ONCs, n_{com}, n_{sep} and n_{con}, corresponding to the compactness, separability and connectivity, are calculated separately according to Eqs. (17)–(19).

$$\text{n}_{\text{com}} = \begin{cases} \text{v}+1, \left|Std_{wc_{\text{v}+1}} - Std_{wc_\text{v}}\right| = \max(|\text{diff}_{\text{com}}|), Std_{wc_{\text{v}+1}} - Std_{wc_\text{v}} > 0 \\ \text{v}, \left|Std_{wc_{\text{v}+1}} - Std_{wc_\text{v}}\right| = \max(|\text{diff}_{\text{com}}|), Std_{wc_{\text{v}+1}} - Std_{wc_\text{v}} \le 0 \end{cases} \tag{17}$$

$$\text{n}_{\text{sep}} = \begin{cases} \text{v}+1, \left|Std_{bc_{\text{v}+1}} - Std_{bc_\text{v}}\right| = \max(\left|\text{diff}_{\text{sep}}\right|), Std_{bc_{\text{v}+1}} - Std_{bc_\text{v}} > 0 \\ \text{v}, \left|Std_{bc_{\text{v}+1}} - Std_{bc_\text{v}}\right| = \max(\left|\text{diff}_{\text{sep}}\right|), Std_{bc_{\text{v}+1}} - Std_{bc_\text{v}} \le 0 \end{cases} \tag{18}$$

$$\text{n}_{\text{con}} = \begin{cases} \text{v}+1, \left|Std_{con_{\text{v}+1}} - Std_{con_\text{v}}\right| = \max(|\text{diff}_{\text{con}}|), Std_{con_{\text{v}+1}} - Std_{con_\text{v}} > 0 \\ \text{v}, \left|Std_{con_{\text{v}+1}} - Std_{con_\text{v}}\right| = \max(|\text{diff}_{\text{con}}|), Std_{con_{\text{v}+1}} - Std_{con_\text{v}} \le 0 \end{cases} \tag{19}$$

where $2 \le v < \left\lfloor\sqrt{\text{N}}\right\rfloor$, and $|\cdot|$ means taking the absolute value.

***Step 5*: Determine the ONC.**

The ONC, $\text{n}_{\text{optimal}}$, is defined as

$$\text{n}_{\text{optimal}} = \text{round}\left(\frac{\text{n}_{\text{com}} + \text{n}_{\text{sep}} + \text{n}_{\text{con}}}{3}\right) \tag{20}$$

In Step 5, considering that the NC should be an integer, we perform a round operation; and compared with rounding up or rounding down, round can better fetch the NC that occurs more frequently in the [n_{com}, n_{sep}, n_{con}]. For example, when [n_{com}, n_{sep}, n_{con}] = [3, 2, 3], we obtain that round $(3+2+3/3) = 3$, $\lfloor 3+2+3/3 \rfloor = 2$, $\lceil 3+2+3/3 \rceil = 3$, and 3 with the highest number of occurrences is obtained using the round and the rounding up; and when [n_{com}, n_{sep}, n_{con}] = [2, 2, 3], we can obtain that round $(2+2+3/3) = 2$, $\lfloor 2+2+3/3 \rfloor = 2$,$\lceil 2+2+3/3 \rceil = 3$, and 2 is obtained using the round and the rounding down.

Here, to facilitate understanding, we use the commonly used Seeds from the UCI repository as an example and choose the stable AHC with the average linkage as our clustering algorithm to illustrate the above steps in more detail.

Step 1: Cluster the Seeds under different k, where, N is 210, the range of k is [2, 14].

Step 2: According to Eq. (8), the BWCON matrix is shown below:

$$\begin{pmatrix} \text{avgBWCON}(2) \\ \text{avgBWCON}(3) \\ \vdots \\ \text{avgBWCON}(14) \end{pmatrix} = \begin{pmatrix} [1.94, 6.07, 0.98] \\ [1.51, 5.41, 0.96] \\ \vdots \\ [0.73, 5.27, 0.65] \end{pmatrix}$$

Step 3: According to Eqs. (10)–(12), the standardized BWCON matrix is as follows:

$$\text{Std}\begin{pmatrix} \text{avgBWCON}(2) \\ \text{avgBWCON}(3) \\ \vdots \\ \text{avgBWCON}(14) \end{pmatrix} = \begin{pmatrix} 0, 1, 1 \\ 0.35, 0.39, 0.93 \\ \vdots \\ 1, 0.26, 0 \end{pmatrix}$$

Step 4: The n_{com}, n_{sep} and n_{con} are estimated according to Eqs. (14)–(19).

To facilitate the analysis, we visualize the diff_{com}, diff_{sep} and diff_{con} in Fig. 2.

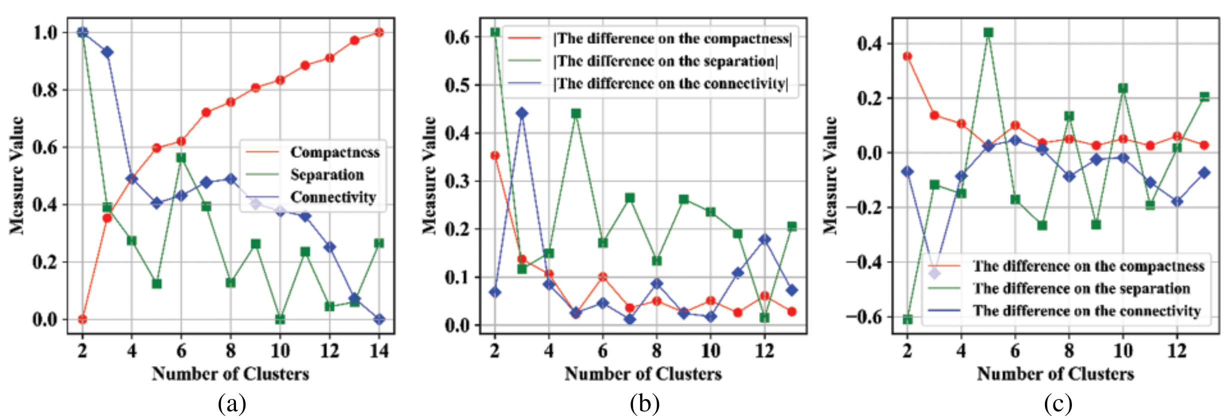


Figure 2: Schematic diagrams for the final ONC estimation

Fig. 2a shows the change trend of the compactness, separation and connectivity under different NCs, Fig. 2b presents the values of $|\text{diff}_{com}|$, $|\text{diff}_{sep}|$ and $|\text{diff}_{con}|$, and Fig. 2c represents the values of diff_{com}, diff_{sep} and diff_{con}. In Fig. 2a, as the NC increases, the overall trend is upward for the red line and downward for the blue line, this is because the compactness and the connectivity of a cluster are only related to the distribution of the cluster itself, and with the increase of the NC, the sample size of each cluster in the clustering result becomes smaller and smaller, which means that the average distance from each point in the cluster to its centroid tends to become smaller and each sample is less and less likely to be grouped into the same cluster with its T nearest neighbors; obviously, it is unreliable to simply take the NCs corresponding to the maximum compactness or the maximum connectivity as the final determined ONCs. As for the separation, there is no such general rule because the separation of a cluster is related not only to the distribution of the cluster itself, but also to the distributions of other clusters in the clustering result. Thus, in our work, we only focus on the points with the largest fluctuations under different NCs as in Eqs. (17)–(19) to determine the corresponding ONCs since small fluctuations of these three measures are likely to be caused by some special points or clusters in the clustering results, and it is also unreliable to judge the reasonableness of the clustering result on this basis. Specifically, for the compactness, the point with the largest fluctuation is searched in Fig. 2a, and we can find that the fluctuation is maximum when the NC changes from 2 to 3 (as the first point on the red line in Fig. 2b), as well as the fluctuation is positive (as the first point on the red line in Fig. 2c), i.e., the clustering result has a higher compactness as the NC increases by 1; thus, according to Eq. (17), $\mathrm{n_{com}} = 3$. Similarly, $\mathrm{n_{sep}} = 2$ and $\mathrm{n_{con}} = 3$.

***Step 5*:** According to Eq. (20), $\mathrm{n_{optimal}} = 3$, which is consistent with the true NC (TNC).

3.3 The Adaptive Parameter-Free ONC Estimation Algorithm: BWCON-NSDK-Means++

From the illustration in Section 3.2, an IVI and a clustering algorithm are indispensable for determining the ONC. Here, we optimize the SDK-means++ algorithm and propose the NSDK-means++ to integrate with the above BWCON-based ONC estimation framework. And the process of the NSDK-means++ is described as follows:

***Step 1*:** Input a dataset $X = \{x_1, x_2, \ldots, x_N\}$, and the NC is m.

***Step 2*:** Choose the first ICC c_1 from X based on the fact: the ICCs should be distributed in the center of each cluster and surrounded by many samples.

Definition 8 Natural neighbor of a sample x_i is defined as follows:

$$x_j \in NN(x_i) \Leftrightarrow x_i \in KNN_\lambda(x_j) \wedge x_j \in KNN_\lambda(x_i) \tag{21}$$

In this step, the c_1 is determined based on the concept of natural neighbor that is derived from [31]; where $NN(x_i)$ denotes a set of natural neighbors of point x_i, the point x_j is a natural neighbor of x_i, λ is the number of neighboring samples, $KNN_\lambda(x_j)$ is a set of λ nearest neighbors of x_j, and $KNN_\lambda(x_i)$ is a set of λ nearest neighbors of x_i. It can be seen that when and only when x_i belongs to one of λ nearest samples from x_j and x_j also belongs to one of λ nearest samples from x_i, x_j is called a natural neighbor of x_i, that is to say, when x_j is a natural neighbor of x_i, these two samples are close to each other. Obviously, the more natural neighbors a point has, the more points it is surrounded by, and the more likely it is to be an ICC, thus in our study, we consider the point with the most natural neighbors as the c_1. Meanwhile, to avoid parameter setting, we also determine the λ using the Algorithm 1 in [31] that is an adaptive method for determining the number of nearest neighbors for a dataset to be clustered.

***Step 3*:** Choose the next center c_i based on the largest sum of the distance.

To ensure that the distances among the ICCs are as large as possible and the obtained ICCs can be located in different clusters, we adopt the largest sum of the distance as in [15] to determine the next center c_i, i.e.,

$$c_i = \{s_h | max\{\mathrm{w}(s_1), \cdots, \mathrm{w}(s_n)\}\} \tag{22}$$

$$\mathrm{w}\left(s_j\right) = \sum\nolimits_{t=1}^{h} \mathrm{d}\left(s_j, c_t\right), s_j \in S, c_t \in C \tag{23}$$

where C denotes a set of ICCs that have been identified, S is a set of samples in the X other than the samples in the C, c_t is an ICC in the C, s_j is a sample in the S, $d(\cdot,\cdot)$ represents the Euclidean distance, h is the number of samples in the C, n is the number of samples in the S, and the sample with the largest $\mathrm{w}(s_j), j \in [1, n]$, is identified as the next ICC. It can be seen that the ICC selection method based on the largest sum of the distance links each new ICC with all existing ICCs, which ensures not only that there are the large differences among the ICCs, but also that the different ICCs are located in different clusters.

Step 4: Repeat Step 3 until the m centers $c = \{c_1, c_2, \cdots, c_m\}$ are chosen.

Step 5: Modify m centers in the c based on their respective λ neighboring samples.

In this step, to ensure that the final ICCs are located in the center of each cluster as much as possible and reduce the number of iterations of the ICCs, the obtained m centers are further modified, i.e.,

$$KNN_\lambda(c_i) = \bigcup_{r=1}^{\lambda} \{findKNN(c_i, r)\} c_i \in c \tag{24}$$

$$c_i' = \frac{Sum(KNN_\lambda(c_i))}{\lambda} \tag{25}$$

where c_i is an ICC in the c, λ is the number of neighboring samples that is equal to λ in the Step 2 to avoid parameter setting, $findKNN(c_i, r)$ returns the rth nearest neighbor of c_i, $KNN_\lambda(c_i)$ is a set of λ nearest neighbors of c_i, the mean value c_i' of the λ nearest neighbors of the c_i is the finalized ICC corresponding to the c_i, and in our work, the modified m ICCs are denoted as $c' = \{c_1', c_2', \cdots, c_m'\}$. The rationality of using the neighboring samples to modify the c is that a sample should be grouped in the same cluster as its neighboring samples, and by interacting with the neighboring samples as in Eq. (25), an ICC located at the edge can be moved closer to the inner part of the corresponding cluster.

Step 6: For each sample x_i, calculate the distance to the m centers in the c' and assign it to the cluster with the smallest Euclidean distance; and the obtained clustering results are denoted as $C = \{C_1, \cdots, C_m\}$, where $\bigcup_i^m C_i = X, i = 1, \ldots, m$, and $C_i \cap C_j = \phi, \forall i, j, i \neq j, j = 1, \ldots, m$.

Step 7: Update the c' by using the mean value of each cluster as the new center of the corresponding cluster.

Step 8: Repeat Steps 6 and 7 until the centers are unchanged.

Step 9: Output the clustering results $C = \{C_1, \cdots, C_m\}$.

To visually illustrate the superiority of the NSDK-means++, Fig. 3 compares the ICCs obtained by the SDK-means++ with those obtained by the NSDK-means++.

In Fig. 3, the SDK-means++ estimates the density of each sample by counting the number of samples falling within the cut-off distance dc around each sample, with more samples indicating a higher density at that point, and in turn using the point with the highest density as the first ICC. In the figure on the left, we set the dc to be the radius of the gray circle, thus the point m is determined as the first ICC, and based on the Eqs. (22) and (23), the remaining two ICCs are, in order, b and g that are located at the edges of the clusters because the largest sum of the distance always determines the next ICC based on the furthest distance from the identified ICCs. For the NSDK-means++, because the dataset is small, we directly take λ as 1; for the point a in the Cluster 1, $\mathrm{KNN}_1(a) = \{d\}$ and $\mathrm{KNN}_1(d) = \{a\}$, thus the number of natural neighbors of a is 1; similarly, the numbers of the natural neighbors of b to r are: 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 3, 1, 1, 1 and 1, thus point n is the first ICC determined, and the remaining ICCs are further determined as b and g based on the largest sum of the distance; moreover, the initial positions of these three ICCs are modified using their respective one neighboring sample (i.e., p, a and h) according to Eqs. (24) and (25), and ultimately, their positions moved from the edges of the clusters to the green dots along the orange arrows, which are closer to the inner parts of the clusters, and still distributed in the different clusters. Note that the orange arrows in the above diagram only qualitatively indicate the movement trend of the ICCs; and for the example above, the advantage of the Step 5 would be even more apparent if λ were to be increased slightly.

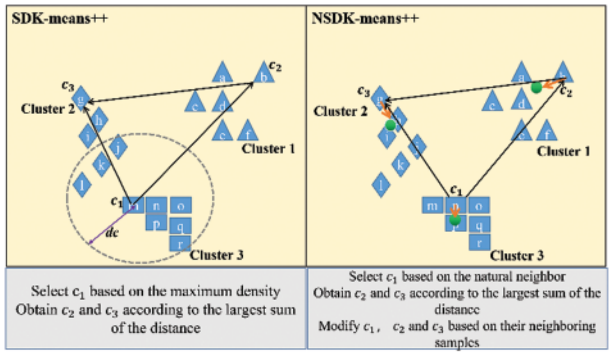


Figure 3: Schematic diagrams of the processes of determining the ICCs

As a result, the BWCON-NSDK-means++ that can independently determine the ONC is proposed in Algorithm 3.

Algorithm 3: BWCON-NSDK-means++

Input: given dataset X
Output: ONC m, clustering results $R[m]$
1: Determine the NC k in the search range of $\left[2, \left\lfloor \sqrt{N} \right\rfloor\right]$; // N is the number of samples in the X
2: Initial an empty list R // R is used to store the clustering results corresponding to different NCs
3: **Repeat:**
4: Use the NSDK-means++ to cluster X into k clusters $\{C_1, \cdots, C_k\}$;
//$C_1, \cdots, C_k$ are k mutually disjoint sets, and $\bigcup_i^k C_i = X, i = 1, \ldots, k$
5: Store the k clusters into the $R[k]$
6: Use Eq. (7) to calculate the BWCON of the $R[k]$;
7: $k = k + 1$
8: **Until** an empty cluster exists in the k clusters or $k > \left\lfloor \sqrt{N} \right\rfloor$
9: Use Eqs. (8) to construct a BWCON matrix;
10: Use Eqs. (10)–(12) to standardize the BWCON matrix;
11: Use Eqs. (17)–(19) to determine the $[n_{com}, n_{sep}, n_{con}]$;
12: Use Eq. (20) to calculate the $n_{optimal}$ m;
13: **Return** ONC m, clustering results $R[m]$

---

4 Experimental Studies

In this section, to demonstrate the effectiveness of the combination "BWCON+NSDK-means++", we conduct three experiments: (1) to show that the BWCON is a more suitable IVI for determining the ONC than those already existing IVIs, we combine the BWCON and seven other

IVIs with different clustering algorithms and compare their abilities to estimate the ONC; (2) to illustrate that the NSDK-means++ is a more appropriate algorithm than the KMA and its variants for determining the ONC in combination with IVIs, the clustering performance of the NSDK-means++ and other five partition-based clustering algorithms is compared on seven UCI datasets; and finally (3) to prove the superiority of the BWCON-NSDK-means++, we integrate several representative clustering algorithms into our proposed BWCON-based ONC estimation framework and compare their ONC accuracies. All works of this research are implemented using python 3.8, running on an 11th Gen Intel(R) Core (TM) i7-1165G7@2.80 GHz CPU with 16.0 GB RAM and windows 10-64-bit operating system; and the SPSS is used for the Kruskal–Wallis test and Friedman test.

4.1 Data Acquisition

For validation purpose, the TNCs are known for all datasets used in this section, and the details of the selected datasets are shown in Table 1. Among them, Seeds, Vehicle, Cleveland, Balance, Haberman, Thyroid and Wine are the most commonly used public real-world datasets from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/); S1, S2 and S3 are three 2-D synthetic datasets generated by using the "multivariate_normal" function from the numpy, and each cluster in these datasets is Gaussian-distributed; and S4 and S5 are generated using the "make_blobs" function of the sklearn. From Table 1, it can be seen that these datasets have different sample sizes (100–1800), features (2–18), NCs (2–5), and cluster sizes.

Table 1: The description of the datasets for testing

Dataset	Number of observations	Cluster size	Number of features	TNC
Seeds	210	70/70/70	7	3
Vehicle	846	212/218/199/217	18	4
Cleveland	297	160/54/35/35/13	13	5
Balance	625	49/288/288	4	3
Haberman	306	225/81	3	2
Thyroid	215	150/35/30	5	3
Wine	178	59/71/48	13	3
S1	1500	250/1000/250	2	3
S2	1800	600/600/600	2	3
S3	800	200/200/200/200	2	4
S4	1000	——	2	3
S5	100	——	2	3

4.2 Superiority Evaluation of the BWCON-Based ONC Estimation Framework

To show the performance of the BWCON-based ONC estimation framework, like the existing studies [7,11], we choose the AHC with single linkage algorithm in combination with different IVIs to determine the ONC. In addition, the other four stable clustering algorithms are also adopted. In our study, all the above combinations are run within a search range $\left[2, \left\lfloor \sqrt{N} \right\rfloor\right]$. The experimental results are shown in Tables 2–6, where the bold numbers indicate the correct NCs.

Table 2: Experimental results of ONCs using the AHC with the single linkage

Datasets	TNC	EONC							
		Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	3	2	**3**	2	2	2	2	13	**3**
Vehicle	4	2	11	2	2	2	2	18	**4**
Cleveland	5	2	**5**	2	2	2	2	2	3
Balance	3	2	2	2	2	2	2	25	2
Haberman	2	**2**	**2**	**2**	**2**	**2**	**2**	6	**2**
Thyroid	3	2	**3**	2	4	2	2	4	**3**
Wine	3	2	2	2	4	2	2	2	**3**
S1	3	36	**3**	2	36	2	2	10	**3**
S2	3	4	4	4	4	4	4	41	**3**
S3	4	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
S4	3	2	**3**	2	2	2	2	31	4
S5	3	2	**3**	**3**	**3**	**3**	**3**	9	**3**

Table 3: Experimental results of ONCs using the AHC with the complete linkage

Datasets	TNC	EONC							
		Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	3	13	2	2	**3**	2	2	2	**3**
Vehicle	4	29	3	3	5	3	3	3	**4**
Cleveland	5	2	2	2	3	2	2	2	4
Balance	3	22	16	2	2	15	15	16	**3**
Haberman	2	**2**	5	**2**	3	**2**	**2**	12	3
Thyroid	3	10	5	**3**	**3**	**3**	**3**	5	**3**
Wine	3	12	8	**3**	**3**	**3**	**3**	8	4
S1	3	35	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S2	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S3	4	**4**	**4**	**4**	**4**	**4**	**4**	**4**	7
S4	3	2	2	2	**3**	2	2	2	4
S5	3	2	**3**	**3**	**3**	**3**	**3**	**3**	**3**

Table 4: Experimental results of ONCs using the AHC with the average linkage

Datasets	TNC	EONC							
		Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	3	8	2	2	**3**	2	2	2	**3**
Vehicle	4	2	2	3	3	3	3	2	**4**
Cleveland	5	2	3	2	4	2	2	3	3
Balance	3	19	16	15	2	15	15	16	**3**
Haberman	2	**2**	3	**2**	7	**2**	**2**	4	**2**
Thyroid	3	2	**3**	2	6	2	**3**	**3**	**3**
Wine	3	2	2	2	8	2	2	**3**	4
S1	3	**3**	**3**	**3**	**3**	**3**	**3**	5	4
S2	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S3	4	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
S4	3	2	2	2	**3**	2	2	2	**3**
S5	3	2	**3**	**3**	**3**	**3**	**3**	**3**	**3**

Table 5: Experimental results of ONCs using the AHC with the ward linkage

Datasets	TNC	EONC							
		Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	3	10	2	2	**3**	2	2	2	**3**
Vehicle	4	27	2	2	2	2	2	2	5
Cleveland	5	17	2	2	2	2	2	2	**5**
Balance	3	23	16	15	**3**	15	15	16	**3**
Haberman	2	6	6	**2**	**2**	**2**	**2**	6	5
Thyroid	3	**3**	2	2	**3**	2	2	2	4
Wine	3	11	2	2	**3**	**3**	2	2	**3**
S1	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S2	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S3	4	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
S4	3	2	2	2	**3**	2	2	2	**3**
S5	3	2	**3**	**3**	**3**	**3**	**3**	**3**	**3**

Table 6: Experimental results of ONCs using the NSDK-means++ algorithm

Datasets	TNC	EONC							
		Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	3	4	2	2	**3**	2	2	2	**3**
Vehicle	4	**4**	2	2	2	2	2	2	3
Cleveland	5	**5**	2	2	2	2	2	2	4
Balance	3	8	13	8	8	**3**	**3**	**3**	**3**
Haberman	2	**2**	4	**2**	**2**	**2**	**2**	8	**2**
Thyroid	3	**3**	4	**3**	**3**	**3**	**3**	4	**3**
Wine	3	2	**3**	**3**	2	**3**	**3**	**3**	**3**
S1	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
S2	3	**3**	**3**	**3**	**3**	**3**	**3**	**3**	4
S3	4	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**
S4	3	2	2	2	**3**	2	2	2	**3**
S5	3	2	**3**	**3**	**3**	**3**	**3**	**3**	**3**

From Table 2, we can find that the Dunn is valid for the Haberman and S3, the DB is valid for all the datasets except Vehicle, Balance, Wine and S2, the Sil, CH, BWP and CIP are valid for the Haberman, S3 and S5, the BWC is valid for the S3, and our proposed BWCON is invalid only for Cleveland, Balance and S4. Meanwhile, it can be also seen that all the above indices are unable to get the correct NC for the Balance, and only BWCON can detect the correct NC for Vehicle, Wine and S2 datasets. Obviously, the combination "BWCON+AHC with the single linkage" has the best NC estimation ability.

In Table 3, the ONCs estimated by the "BWCON+AHC with the complete linkage", the "Sil +AHC with the complete linkage" and the "CH+AHC with the complete linkage" are more accurate among all combinations and their estimated biased ONCs are closer to the TNCs. Specifically, for the above 12 datasets, the maximum deviations of the TNCs and the EONCs obtained based on the Dunn, DB, Sil, CH, BWP, CIP, BWC and BWCON are: 32, 13, 3, 2, 12, 12, 13, and 3, respectively. Furthermore, all the above indices are invalid for Cleveland, but the EONC obtained by our index is the closest to the TNC; and only BWCON can obtain the correct NC for Vehicle and Balance datasets.

Based on the AHC with the average linkage, the Dunn cannot get the correct NC for six UCI datasets and two synthetic datasets; the DB, Sil and BWP cannot get the correct NC for six UCI datasets and a synthetic dataset; the CH cannot get the correct NC for six UCI datasets; the CIP cannot get the correct NC for five UCI datasets and a synthetic dataset; the BWC cannot get the correct NC for five UCI datasets and two synthetic datasets; and the BWCON cannot get the correct NC only for two UCI datasets and a synthetic dataset. And in Table 4, only BWCON is valid for the Vehicle and Balance, and all the indices are invalid for Cleveland.

Table 5 shows that the Dunn cannot choose the correct NC for all the twelve datasets except Thyroid, S1, S2 and S3; the DB and BWC are invalid for all the seven UCI datasets and a synthetic dataset; the Sil and CIP are invalid for six UCI datasets and a synthetic dataset; the CH is invalid for Vehicle and Cleveland; the BWP is invalid for six datasets; and the BWCON is invalid for Vehicle, Haberman and Thyroid. It can be seen that combining the BWCON and the CH with the AHC with the ward linkage yield significantly better NC estimation ability than all other combinations. In addition, all the indices cannot obtain the correct NC for Vehicle, and only BWCON can detect the correct NC for Cleveland.

As can be seen from Table 6, the Dunn is invalid for three UCI datasets and two synthetic datasets; the DB cannot get the TNCs for six UCI datasets and one synthetic dataset; the Sil cannot get the TNCs on four UCI datasets and one synthetic dataset; the CH is invalid on four UCI datasets; the BWP and CIP are invalid for three UCI datasets and one synthetic dataset; the BWC is invalid for five UCI datasets and one synthetic dataset; and our proposed BWCON is invalid only on Vehicle, Cleveland and S2. Not only that, further observing the maximum deviations of the TNCs and the EONCs under different indices, it can be seen that for the above 12 datasets, the BWCON has a minimum deviation of only 1, and the other indices, Dunn, DB, Sil, CH, BWP, CIP and BWC, correspond to deviations of 5, 10, 5, 5, 3, 3 and 6, respectively. It is evident that the "BWCON+NSDK-means++" has better NC estimation ability, and the BWCON is more suitable for estimating the NC in combination with our proposed NSDK-means++.

Furthermore, the average running times of eight IVIs on the CPU are presented in Table 7, where to ensure the objectivity of the runtime, we take the average runtime within the search range of each index as the final evaluation value under the NSDK-means++. As shown in Table 7, the time performance of the BWCON is acceptable and it is much faster than the BWP and Dunn indices.

Table 7: Average running times of eight IVIs (s)

Datasets	Dunn	DB	Sil	CH	BWP	CIP	BWC	BWCON
Seeds	0.335	0.001	0.002	0.0003	0.232	0.008	0.002	0.079
Vehicle	3.459	0.002	0.011	0.0005	5.005	0.018	0.007	0.921
Cleveland	0.425	0.001	0.003	0.0004	0.580	0.011	0.002	0.398
Balance	9.703	0.003	0.006	0.001	2.142	0.045	0.006	1.233
Haberman	0.441	0.001	0.003	0.0004	0.531	0.009	0.002	0.263
Thyroid	0.207	0.001	0.002	0.0003	0.267	0.007	0.002	0.225
Wine	0.103	0.001	0.002	0.0002	0.208	0.004	0.002	0.079
S1	135.154	0.005	0.032	0.001	14.902	0.134	0.012	0.931
S2	34.638	0.001	0.043	0.001	26.800	0.057	0.010	1.968
S3	8.428	0.001	0.009	0.0004	5.588	0.030	0.005	0.553
S4	40.366	0.003	0.015	0.001	8.252	0.084	0.010	0.860
S5	0.054	0.001	0.001	0.0003	0.053	0.004	0.001	0.063

To further support the effectiveness of the BWCON from a statistical point of view, the Friedman test and the Nemenyi test are used [32]. First, to show whether the observed differences of the ability for detecting the TNC among the Dunn, DB, Sil, CH, BWP, CIP, BWC and BWCON are statistically significant, the study conducts the Friedman test for the proposed BWCON and other seven indices on all the experimental datasets in terms of the AHC with the single linkage, complete linkage, average linkage, and ward linkage, and the NSDK-means++ algorithms. For every experimental dataset, the absolute value of the difference between the EONC and the TNC under an index is taken as the final evaluation value; the smaller the evaluation value is, the closer the EONC is to the TNC.

Finally, the resulting p-values are 0.000, 0.000, 0.081, 0.000 and 0.162, respectively, revealing that the NC estimation performance of the above eight indices is significantly different under the AHC with the single linkage, complete linkage and ward linkage and there are no significant differences in the NC estimation ability under all indices based on the NSDK-means++ and the AHC with the average linkage, which is likely related to the excellent clustering performance of these two algorithms themselves because the clustering algorithm and the IVI together determine the NC estimation ability, and a good clustering result tends to be more favorable for various IVIs to find the correct NC (the level of test significance is set to 0.05). Then, to further distinguish the performance of each index, the Nemenyi test is performed (the significance level is 0.1) and the obtained critical distance (CD) is 2.78. Where, if the difference between the average order values of the two indices exceeds the CD, the hypothesis that these two indices have the same estimation performance will be rejected; in addition, the smaller the average order value, the better the overall NC estimation ability of the corresponding index on all datasets. Fig. 4 is the Nemenyi test diagram.

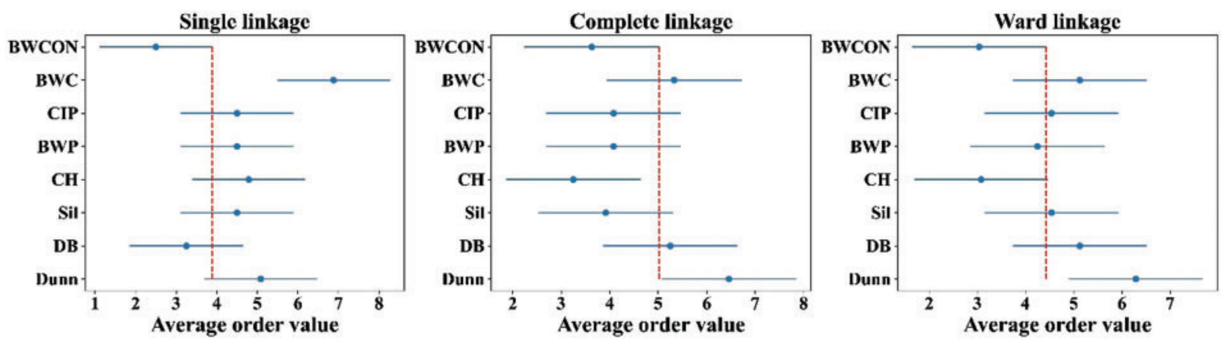


Figure 4: Nemenyi test diagrams under the AHC with the single linkage, complete linkage and ward linkage

In Fig. 4, "·" is the mean ranking score, and the line segment size is CD. For the AHC with the single linkage, the difference between the mean ranking scores of the BWCON and the BWC exceeds the CD, indicating that our proposed BWCON is significantly superior to the BWC in terms of the NC estimation; in addition, compared with all the comparative works, the distribution of "·" corresponding to the BWCON is on the left, which shows that the overall NC estimation ability of the BWCON is better. Similarly, for the AHC with the complete linkage, the NC estimation ability of the BWCON significantly outperforms that of the Dunn, and from the overall NC estimation performance on all datasets, the BWCON beats all indices except the CH but is very close to the CH; and for the AHC with the ward linkage, the BWCON also significantly outperforms the Dunn, and is better than all the comparative indices on the whole as respect to their NC estimation abilities. In general, compared with the other seven popular IVIs, the BWCON has more stable NC estimation ability, and it is easier to be integrated with different clustering algorithms to obtain an accurate NC.

4.3 Performance Evaluation of the BWCON-NSDK-Means++ Algorithm

In this section, we conduct experiments to demonstrate the effectiveness of the BWCON-NSDK-means++ in detail from two aspects: (1) to show that the NSDK-means++ itself is a valid clustering algorithm, we focus on comparing the NSDK-means++ and the SDK-means++ since Du et al. [15] have proved that the SDK-means++ is significantly better than the KMA and K-means++ as respect to their clustering performance and speed; and (2) we compare the average ONC accuracies of the

"BWCON+NSDK-means++" and nine other combinations to justify that the BWCON-NSDK-means++ is indeed an excellent ONC estimation method. For the first experiment, the results are shown in Tables 8–10; and the second experimental results are shown in Table 11. In addition, since the UCI datasets are usually more complex than those 2-D synthetic datasets, we conduct the first experiment on only UCI datasets; and in terms of the evaluation metrics, to ensure the reliability of the conclusions, we finally adopt three IVIs (i.e., Sil, CH and DB) and six EVIs (i.e., Adjusted Rand Index (ARI), Adjusted Mutual Information (AMI), Homogeneity Score (HOM), Compactness Score (COM), V_measure (V) and Fowlkes and Mallows Index (FMI)) to evaluate the clustering results, which are described in [10,33,34]; except for the DB, all the other indices are found to have the better clustering performance with larger measure values.

Table 8: Results for the ICSDK-means++ and SDK-means++ algorithms on the UCI datasets

Datasets	Algorithm	Sil	CH	DB	ARI	AMI	HOM	COM	V	FMI	NI
Seeds	SDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	6
	ICSDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	6
Vehicle	SDK-means++	0.306	402.315	1.150	0.076	0.111	0.105	0.128	0.115	0.342	11
	ICSDK-means++	0.311	402.719	1.152	0.078	0.112	0.106	0.128	0.116	0.342	26
Cleveland	SDK-means++	0.253	148.200	0.974	0.026	0.029	0.049	0.046	0.047	0.329	11
	ICSDK-means++	0.253	148.200	0.974	0.026	0.029	0.049	0.046	0.047	0.329	10
Balance	SDK-means++	0.165	128.208	1.737	0.074	0.056	0.064	0.054	0.059	0.427	6
	ICSDK-means++	0.159	119.866	1.824	0.000	0.001	0.004	0.003	0.004	0.380	6
Haberman	SDK-means++	0.387	238.553	1.016	−0.004	−0.002	0.001	0.001	0.001	0.551	10
	ICSDK-means++	0.381	236.424	1.029	−0.001	−0.002	0.001	0.001	0.001	0.551	8
Thyroid	SDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	6
	ICSDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	5
Wine	SDK-means++	0.299	83.160	1.315	0.802	0.793	0.799	0.792	0.795	0.869	7
	ICSDK-means++	0.299	83.160	1.315	0.802	0.793	0.799	0.792	0.795	0.869	8

Table 9: Results for the MCSDK-means++ and SDK-means++ algorithms on the UCI datasets

Datasets	Algorithm	Sil	CH	DB	ARI	AMI	HOM	COM	V	FMI	NI
Seeds	SDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	6
	MCSDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	7
Vehicle	SDK-means++	0.306	402.315	1.150	0.076	0.111	0.105	0.128	0.115	0.342	11
	MCSDK-means++	0.255	445.524	1.460	0.075	0.096	0.100	0.100	0.100	0.307	34
Cleveland	SDK-means++	0.253	148.200	0.974	0.026	0.029	0.049	0.046	0.047	0.329	11
	MCSDK-means++	0.263	158.068	1.095	0.035	0.038	0.061	0.056	0.059	0.330	12
Balance	SDK-means++	0.165	128.208	1.737	0.074	0.056	0.064	0.054	0.059	0.427	6
	MCSDK-means++	0.175	136.738	1.723	0.140	0.106	0.120	0.100	0.109	0.468	22
Haberman	SDK-means++	0.387	238.553	1.016	−0.004	−0.002	0.001	0.001	0.001	0.551	10
	MCSDK-means++	0.387	238.553	1.016	−0.004	−0.002	0.001	0.001	0.001	0.551	5
Thyroid	SDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	6
	MCSDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	2
Wine	SDK-means++	0.299	83.160	1.315	0.802	0.793	0.799	0.792	0.795	0.869	7
	MCSDK-means++	0.299	83.317	1.310	0.837	0.814	0.820	0.811	0.815	0.891	5

Table 10: Results for the different algorithms on the UCI datasets

Datasets	Algorithm	Sil	CH	DB	ARI	AMI	HOM	COM	V	FMI	NI
Seeds	KMA	0.422	314.661	0.876	0.701	0.668	0.670	0.672	0.671	0.800	8
	MBK	0.418	309.472	0.879	**0.722**	**0.689**	**0.690**	**0.692**	**0.691**	**0.814**	5
	K-means++	0.422	314.651	0.877	0.698	0.666	0.668	**0.700**	0.669	0.798	7
	RK-means++	0.422	314.655	0.876	0.699	0.667	0.669	0.671	**0.700**	0.799	8
	SDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	6
	NSDK-means++	0.422	314.675	0.876	0.705	0.671	0.673	0.675	0.674	0.803	5
Vehicle	KMA	**0.260**	**439.549**	1.423	0.083	0.112	0.114	0.117	0.116	0.317	23
	MBK	0.254	**440.620**	1.464	0.074	0.095	0.098	0.099	0.099	0.307	9
	K-means++	**0.309**	402.569	**1.151**	0.078	0.112	0.105	0.128	0.116	0.342	15
	RK-means++	**0.261**	**438.235**	1.421	0.082	0.111	0.113	0.117	0.115	0.317	24
	SDK-means++	**0.306**	402.315	**1.150**	0.076	0.111	0.105	0.128	0.115	0.342	11
	NSDK-means++	0.257	414.833	1.254	0.096	0.150	0.143	0.165	0.153	0.350	9
Cleveland	KMA	0.255	153.383	1.161	**0.050**	0.029	0.053	0.046	0.049	0.324	17
	MBK	0.234	140.786	1.188	**0.050**	0.024	0.047	0.042	0.045	**0.336**	12
	K-means++	0.259	153.062	**0.990**	**0.042**	0.016	0.036	0.034	0.035	**0.338**	14
	RK-means++	0.253	153.139	1.151	**0.050**	0.029	0.053	0.046	0.049	0.326	15
	SDK-means++	0.253	148.200	**0.974**	0.026	0.029	0.049	0.046	0.047	0.329	**11**
	NSDK-means++	0.263	158.068	1.095	0.035	0.038	0.061	0.056	0.059	0.330	12
Balance	KMA	0.171	133.929	1.718	**0.130**	**0.108**	**0.122**	**0.102**	**0.111**	**0.462**	**14**
	MBK	0.163	126.873	1.758	**0.132**	**0.110**	**0.123**	**0.104**	**0.113**	**0.464**	20
	K-means++	0.163	124.027	1.789	**0.139**	**0.130**	**0.145**	**0.121**	**0.132**	**0.467**	**10**
	RK-means++	0.171	134.282	1.714	**0.132**	**0.111**	**0.125**	**0.104**	**0.114**	**0.463**	**15**
	SDK-means++	0.165	128.208	1.737	0.074	0.056	0.064	0.054	0.059	0.427	**6**
	NSDK-means++	0.174	136.827	1.691	0.114	0.098	0.110	0.092	0.100	0.453	20
Haberman	KMA	0.384	237.636	1.022	**−0.003**	−0.002	0.001	0.001	0.001	0.551	6
	MBK	0.383	236.226	1.025	**−0.002**	**−0.001**	**0.002**	0.001	0.001	0.551	6
	K-means++	0.385	238.000	1.019	**−0.003**	−0.002	0.001	0.001	0.001	0.551	9
	RK-means++	0.383	237.419	1.023	**−0.002**	−0.002	0.001	0.001	0.001	0.551	7
	SDK-means++	0.387	238.553	1.016	−0.004	−0.002	0.001	0.001	0.001	0.551	10
	NSDK-means++	0.387	238.553	1.016	−0.004	−0.002	0.001	0.001	0.001	0.551	6
Thyroid	KMA	0.562	141.008	0.847	0.626	0.589	0.521	0.695	0.595	0.854	10
	MBK	0.225	68.450	1.330	0.288	0.370	0.392	0.365	0.378	0.638	3
	K-means++	0.559	135.625	0.850	0.611	0.579	0.509	0.692	0.585	0.850	6
	RK-means++	0.559	138.331	0.849	0.619	0.585	0.516	0.694	0.591	0.852	9
	SDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	6
	NSDK-means++	0.562	141.895	0.847	0.628	0.591	0.523	0.695	0.597	0.855	3
Wine	KMA	**0.300**	83.292	1.314	**0.845**	**0.828**	**0.834**	**0.826**	**0.830**	**0.897**	7
	MBK	0.292	81.243	1.339	0.802	0.793	0.797	0.792	0.795	0.869	5
	K-means++	**0.300**	83.274	1.316	**0.850**	**0.833**	**0.838**	**0.832**	**0.835**	**0.900**	8
	RK-means++	0.297	81.516	1.329	0.828	**0.815**	0.817	**0.817**	**0.817**	0.887	7
	SDK-means++	0.299	83.160	1.315	0.802	0.793	0.799	0.792	0.795	0.869	7
	NSDK-means++	0.299	83.317	1.310	0.837	0.814	0.820	0.811	0.815	0.891	5

Table 11: Average ONC accuracy using various clustering algorithms based on the BWCON index

Datasets	TNC	Average ONC accuracy									
		BWCON-KMA	BWCON-MBK	BWCON-KM++	BWCON-RKM++	BWCON-SDKM++	BWCON-Single linkage	BWCON-Complete linkage	BWCON-Average linkage	BWCON-Ward linkage	BWCON-NSDK-means++
Seeds	3	0.28	0.58	0.59	0.41	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Vehicle	4	0.24	0.94	0.25	0.29	**1.00**	**1.00**	**1.00**	**1.00**	0.00	0.00
Cleveland	5	0.38	0.26	0.27	0.23	0.00	0.00	0.00	0.00	**1.00**	0.00
Balance	3	**1.00**	0.97	**1.00**	**1.00**	**1.00**	0.00	**1.00**	**1.00**	**1.00**	**1.00**
Haberman	2	0.07	0.00	0.00	0.00	0.00	**1.00**	0.00	**1.00**	0.00	**1.00**
Thyroid	3	0.66	0.20	0.89	0.56	**1.00**	**1.00**	**1.00**	**1.00**	0.00	**1.00**
Wine	3	0.14	0.20	0.05	0.20	0.00	**1.00**	0.00	0.00	**1.00**	**1.00**
S1	3	0.12	0.01	0.23	0.06	0.00	**1.00**	**1.00**	0.00	**1.00**	**1.00**
S2	3	0.01	0.07	0.27	0.08	0.00	**1.00**	**1.00**	**1.00**	**1.00**	0.00
S3	4	0.22	0.00	0.33	0.05	0.00	**1.00**	0.00	**1.00**	**1.00**	**1.00**
S4	3	0.09	0.27	0.04	0.16	**1.00**	0.00	0.00	**1.00**	**1.00**	**1.00**
S5	3	0.60	0.98	0.66	0.70	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

As shown in Table 8, to illustrate that it is feasible to determine the first ICC based on the natural neighbor, we replace the corresponding step in the SDK-means++ and the new algorithm is called ICSDK-means++. From Table 8, we can find that the ICSDK-means++ achieves the comparable or even better clustering performance than the SDK-means++ for all the UCI datasets except Balance, that is to say, the strategy proposed in this study to determine the first ICC without any parameters to be specified is effective. Specifically, on the Seeds, Cleveland, Thyroid and Wine, the ICSDK-means++ has the same clustering performance as the SDK-means++ on nine indices, and the similar NIs; on the Vehicle, although the NI of the ICSDK-means++ is fifteen times more than that of the SDK-means++, the ICSDK-means++ outperforms the SDK-means++ on six indices, achieves the same clustering performance as the SDK-means++ on two indices, and only slightly underperforms the SDK-means++ on one index; on the Haberman, the ICSDK-means++ outperforms the SDK-means++ on one index, achieves the same clustering performance as the SDK-means++ on five indices, and is inferior to the SDK-means++ on three indices, but has two fewer iterations than the SDK-means++.

Furthermore, to validate the necessity of modifying the positions of the ICCs in the SDK-means++, we add the modification step to the original SDK-means++, and the new algorithm is denoted as MCSDK-means++. As can be seen from Table 9, except for the Vehicle, the MCSDK-means++ can obtain better clustering results on the remaining six datasets. That is, it is necessary to add the modification step to the SDK-means++. Specifically, on the Seeds, Haberman and Thyroid, the MCSDK-means++ algorithm achieves the same clustering performance as the SDK-means++ on all the indices; for their NIs, except for the Seeds, where the MCSDK-means++ iterates once more than the SDK-means++, in the remaining two datasets, the MCSDK-means++ has fewer iterations than the SDK-means++. On the Clever, the MCSDK-means++ outperforms the SDK-means++ on eight indices; on the Balance, the MCSDK-means++ outperforms the SDK-means++ on all nine indices; and on the Wine, the MCSDK-means++ outperforms the SDK-means++ on eight indices, has the same clustering performance on one index, and has two fewer iterations than the MCSDK-means++.

In addition, Table 10 records the experimental results of the NSDK-means++ and other clustering algorithms; and any evaluation value with better performance than the NSDK-means++ is presented in bold. Besides, for the unstable clustering algorithms (KMA, MBK, K-means++ and RK-means++), we run them 100 times repeatedly and take their average clustering performance as their final evaluation values; particularly, for the MBK, the parameter *iteration* is equal to the NI of the SDK-means++ on the corresponding dataset, and the *mini-batch size* is obtained by dividing the dataset size by the NI.

As indicated in Table 10, the proposed NSDK-means++ outperforms the other clustering algorithms on most of the datasets. To better justify our improvements, we focus on the performance differences between the NSDK-means++ and SDK-means++: (1) on the Seeds, the NSDK-means++ has the same clustering performance as the SDK-means++ on nine indices, but in terms of the NI, the NSDK-means++ has one less iteration than the SDK-means++; (2) on the Haberman, the NSDK-means++ has the same clustering performance as the SDK-means++ on nine indices, but the NSDK-means++ has four fewer iterations than the SDK-means++; (3) on the Thyroid, the NSDK-means++ has the same clustering performance as the SDK-means++ on nine indices, but the NSDK-means++ has three fewer iterations than the SDK-means++; (4) on the Vehicle, the NSDK-means++ is superior to the SDK-means++ on seven indices, and iterates two times less than the SDK-Means++; (5) on the Cleveland, although the NSDK-means++ has one more iteration than the SDK-Means++, the NSDK-means++ is superior to the SDK-means++ on eight indices; (6) on the Balance, although the NSDK-means++ iterates fourteen times more than the SDK-means++, the NSDK-means++ is superior to the SDK-means++ on all nine indices; and (7) on the Wine, the NSDK-means++ is superior to the SDK-means++ on eight indices, is comparable to its clustering performance on one index, and has two fewer iterations than the SDK-means++. In particular, our proposed NSDK-means++ achieves good results on the Balance compared with the ICSDK-means++, and also achieves the better results on the Vehicle compared with the MCSDK-means++, which indicate that it is necessary to combine the step of determining the first ICC based on the natural neighbor, the step of determining the remaining ICCs based on the largest sum of distance and the step of modifying the positions of all the ICCs, which can make the obtained ICCs more reasonable and thus improve the overall clustering performance of the SDK-means++. Moreover, shows the average running time of running them 100 times, and the running time of the NSDK-means++ is acceptable.

Table 12: The average running times of the concurrent algorithms (s)

Algorithm	Seeds	Vehicle	Cleveland	Balance	Haberman	Thyroid	Wine
K-means	0.033	0.471	0.143	0.158	0.025	0.039	0.022
Mini-batch K-means	0.003	0.009	0.004	0.009	0.003	0.002	0.002
K-means++	0.130	1.210	0.532	0.413	0.185	0.128	0.144
RK-means++	0.137	1.718	0.557	0.572	0.140	0.174	0.122
SDK-means++	0.322	11.525	1.464	2.048	0.448	0.274	0.389
ICSDK-means++	0.111	1.697	0.480	1.228	0.325	0.143	0.185
MCSDK-means++	0.418	14.199	1.325	4.195	0.702	0.385	0.520
NSDK-means++	0.109	1.166	0.453	1.427	0.309	0.141	0.170

Not only that, we also further verify the performance differences of the above algorithms from a statistical point of view. Here the resulting p-values using the Friedman test are 0.005, 0.001, 0.003, 0.818, 0.900, 0.713, 0.677, 0.555 and 0.576 respectively as respect to the SC, CH, DB, ARI, AMI, HOM, COM, V and FMI, which show that the performance of the above algorithms is significantly different on the SC, CH and DB (the level of test significance is set to 0.05). Further from Fig. 5, we can find that the NSDK-means++ are significantly superior to the MBK on three indices; moreover, observing the positions of "·", except that the MCSDK-means++ outperforms the NSDK-means++ on the CH, the NSDK-means++ is not worse than all other algorithms on the whole. Particularly, the performance of the above algorithms does not differ significantly on the six EVIs, which indicates that the NSDK-means++ can achieve a clustering performance comparable to that of those existing algorithms in one run. Here the significance level in the Nemenyi test is 0.1 and CD is 3.640.

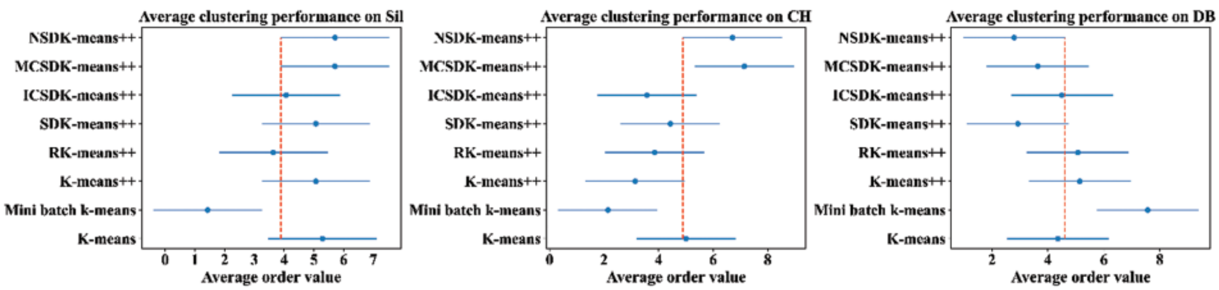


Figure 5: Nemenyi test diagrams for the clustering algorithms

At last, to demonstrate the advantage of the "BWCON+NSDK-means++", we integrate the KMA, MBK, K-means++, RK-means++, SDK-means++, AHC with the single linkage, AHC with the complete linkage, AHC with the average linkage, AHC with the ward linkage and NSDK-means++ into the proposed BWCON-based ONC estimation framework, respectively, to form the BWCON-KMA, BWCON-MBK, BWCON-KM++, BWCON-RKM++, BWCON-SDKM++, BWCON-Single linkage, BWCON-Complete linkage, BWCON-Average linkage, BWCON-Ward linkage and BWCON-NSDK-means++ to compare their average ONC accuracies (see Eq. (26)), where those unstable clustering algorithms are run 100 times as in [8]. Table 11 lists the results and bold indicates the best performance.

$$\text{Average ONC accuracy} = \frac{\text{Number of correct determination of ONC based on acombination}}{\text{Number of runs}} \tag{26}$$

From Table 11, the BWCON-KMA, BWCON-KM++ and BWCON-RKM++ can achieve an accuracy of 1 only on the Balance; the BWCON-MBK cannot achieve 1 on all datasets, and can only achieve 0.98 on the S5; the BWCON-SDKM++ can obtain an accuracy of 1 on four datasets; the BWCON-Complete linkage can achieve 1 on seven datasets; the BWCON-Single linkage, BWCON-Average linkage, BWCON-Ward linkage and BWCON-NSDK-means++ can achieve an accuracy of 1 on nine datasets. Compared with those existing partition-based clustering algorithms, the NSDK-means++ performs better since it can obtain appropriate ICCs. In addition, although the BWCON-Single linkage, BWCON-Average linkage, BWCON-Ward linkage and BWCON-NSDK-means++ have similar ONC estimation performance in Table 11, the NCs estimated by the algorithm in this paper are closer to the TNCs as shown in Tables 2 and 4–6.

5 Application of the BWCON-NSDK-Means++ in the Market Segmentation

The exact estimation of the ONS serves to achieve an effective market segmentation; and if the obtained sub-markets are not differentiated under a so-called ONS, such market segmentation is not meaningful for an enterprise. From this perspective, we verify the usefulness of the BWCON-NSDK-means++ by analyzing the reasonableness of the corresponding market segmentation results under our ONS. Here, it should be noted that the experiments in this section can be conducted in relation to the fact that the BWCON-NSDK-means++ can also be used as a stand-alone market segmentation tool, which is validated later on the Chinese wine market dataset in 2020.

5.1 Data Collection and Preprocessing

The Chinese wine market dataset in this work is provided by the Chinese Grape Industry Technology System, with a total of 2747 effective samples, including two parts: the factors that affect consumers' decision-making and the social demographic characteristics as in [5]. The basic information is listed in Table 13 and the first part is presented through the design of Likert five-category attitude scale. Since the wine consumers tend to be younger and more feminine at present, the dataset focuses more on young and well-educated consumers; specifically, the wine consumers under the age of 46 account for 85.7%, and those with a bachelor's degree or above account for 86.8%.

Table 13: The basic information of the Chinese wine market data

	Feature name	Feature description	Feature type
	a. Brand	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	b. Vintage	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	c. Producing area	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	d. Package	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	e. Price	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
Purchasing decision factors	f. Sale promotion	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5;	Numerical

(Continued)

Table 13 (continued)

	Feature name	Feature description	Feature type
	g. Recommendations from relatives and friends	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	h. Advertisements	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	i. Public praise (positive reviews)	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
	j. Function	Unimportant-1; Slightly important-2; Generally important-3; Very important-4; Especially important-5	Numerical
Social demographic characteristics	Gender	Male-1; Female-2	Categorical
	Age	18-25-1; 26-45-2; >45-3	Categorical
	Marital status	Unmarried-1; Married-2	Categorical
	Occupation	Student-1; Farmer-2; Freelancer-3; Unemployment and retirement-4; State-owned enterprise-5; Foreign or private-owned enterprise-6; Party and government organ and institution-7; Education and research institution-8; Other-9	Categorical
	Per capita disposable monthly income	<2000 RMB-1; 2001-7000 RMB-2; >7000 RMB-3	Categorical
	Education	Junior high school or below-1; Senior high school-2; Bachelor degree-3; Master's degree or above-4	Categorical

In our study, to facilitate the analysis of the practical implications of the market segmentation results, we choose the first part as the market segmentation bases (SB). Moreover, to remove the relevance among the SB, we adopt the Exploratory Factor Analysis (EFA) [35] to deal with them, and after statistical analysis, the KMO value is 0.840, the approximate Chi-Square value in the Bartlett's sphere test is 6479.893, and it is significant at the level of 0.000, which verifies that the dataset is suitable for the EFA. Thus ultimately, we identify four SB to participate in this segmentation task based on the following principles: the cumulative variance contribution rate is greater than 60%, the extraction degrees of the factors are all greater than 0.5, the factor loading values are all greater than 0.6 and each common factor represents at least two factors, as shown in Fig. 6.

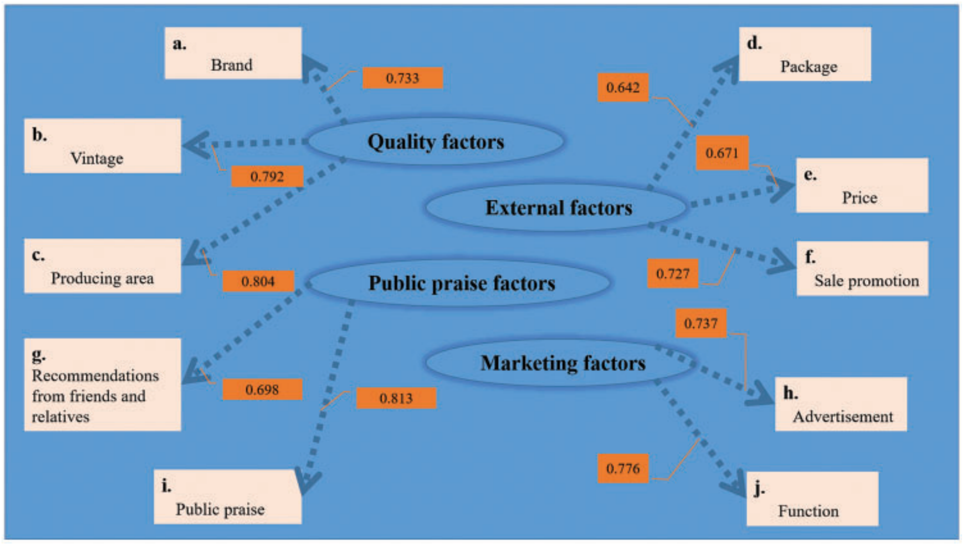


Figure 6: Factor loading diagram of four SB

5.2 *Determination of the ONS*

In this section, based on the above four SB, the BWCON-NSDK-means++ is used to detect the ONS, as shown in Fig. 7. We can find that for the compactness, when the NC changes from 3 to 4, the fluctuation is maximum and positive, thus according to Eq. (17), $n_{com} = 4$; for the separation, when the NC changes from 2 to 3, a positive maximum fluctuation can be obtained, so according to Eq. (18), $n_{sep} = 3$; and finally, for the connectivity, a negative maximum fluctuation can be found when the NC changes from 2 to 3, thus according to Eq. (19), $n_{con} = 2$. Eventually, we can obtain that the ONS is 3 based on the Eq. (20), which is following the finding of [36].

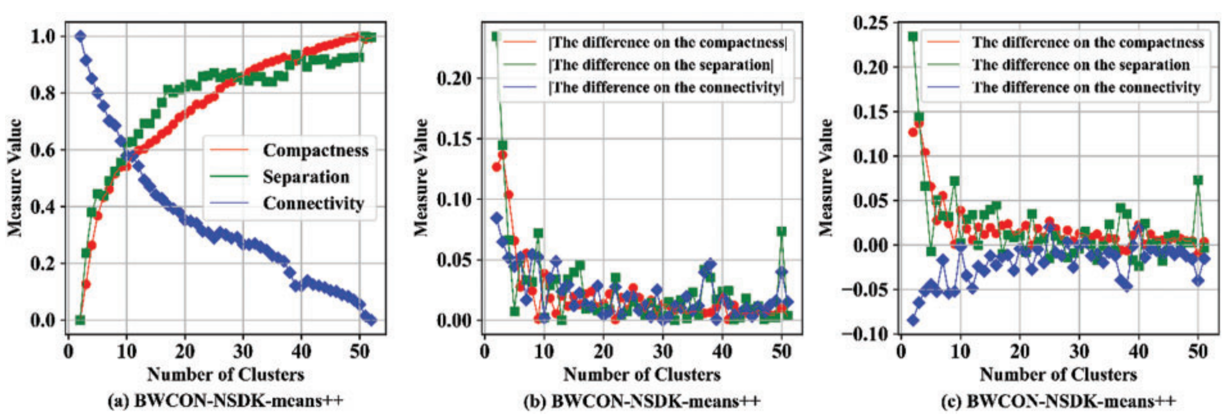


Figure 7: The determination of the ONS based on the BWCON-NSDK-means++

Furthermore, to verify the reliability of the ONS obtained above, we further detect the corresponding ONSs using the BWCON-Single linkage, BWCON-Average linkage and BWCON-Ward linkage mentioned in Section 4.3, as shown in Fig. 8. It can be found that all these algorithms consider segmenting the Chinese wine market into three sub-markets as the best, which indicates that the ONS determined based on the BWCON-NSDK-means++ is indeed reliable.

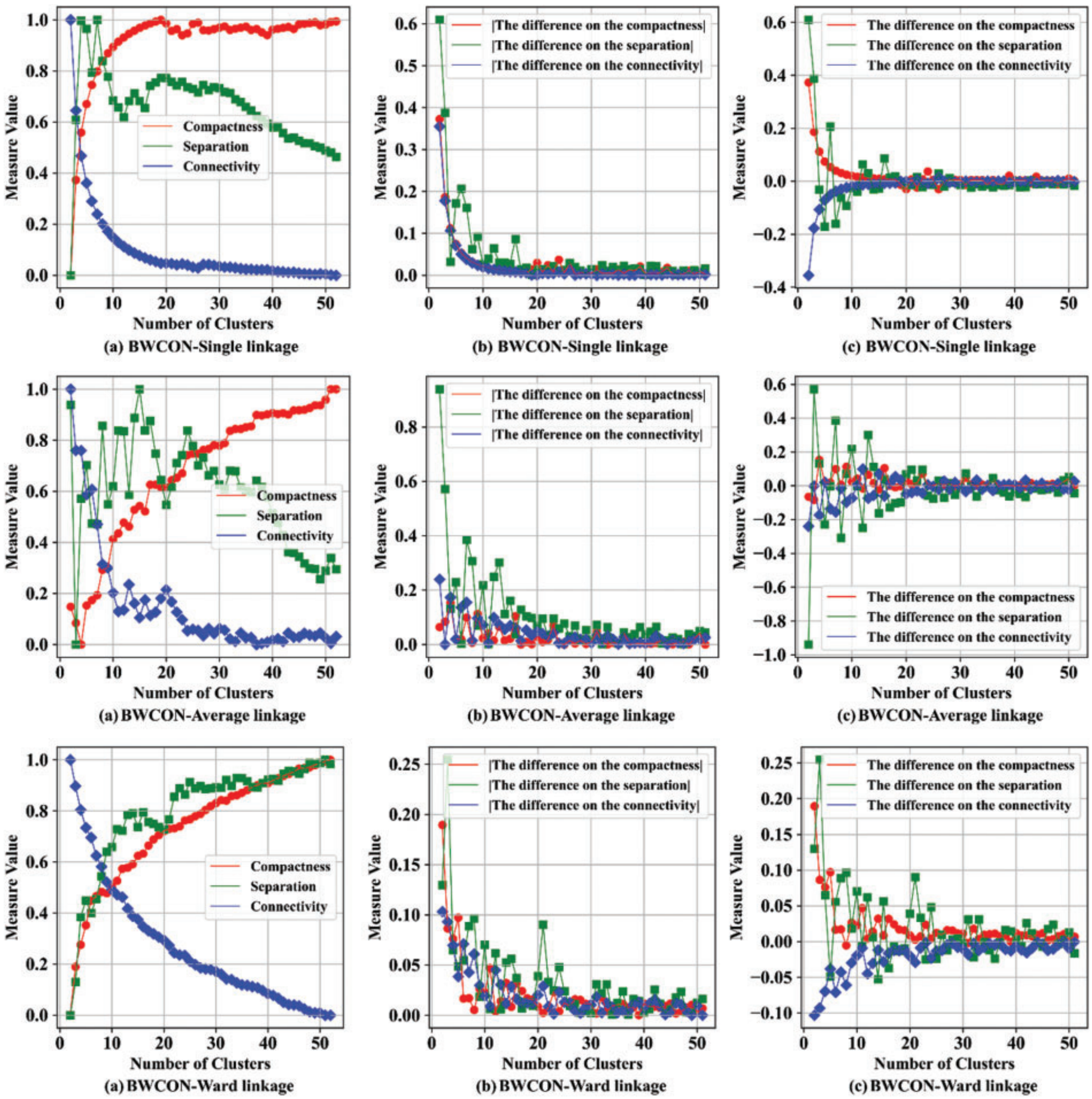


Figure 8: The determination of the ONS based on the BWCON-Single linkage, BWCON-Average linkage and BWCON-Ward linkage algorithms

5.3 The Practicality Analysis of the BWCON-NSDK-Means++ Algorithm

To check the rationality of the market segmentation results under our obtained ONS and the effectiveness of the BWCON-NSDK-means++ as an independent market segmentation tool, we compare the market segmentation results of the BWCON-Single linkage, BWCON-Average linkage, BWCON-Ward linkage and BWCON-NSDK-means++ as respect to the inter-market differentiation, sub-market size and consumer group characteristics, as shown in Tables 14 and 15, and Figs. 9 and 10.

Table 14: The difference test of the SB

Algorithm	SB	Kruskal–Wallis test	
		Chi-Square	Significance ($p < 0.05$)
BWCON-Single linkage	Quality factors	5.667	0.059
	External factors	4.818	0.090
	Public praise factors	5.816	0.055
	Marketing factors	5.812	0.055
BWCON-Average linkage	Quality factors	54.673	0.000
	External factors	55.344	0.000
	Public praise factors	93.026	0.000
	Marketing factors	34.785	0.000
BWCON-Ward linkage	Quality factors	1271.314	0.000
	External factors	475.627	0.000
	Public praise factors	78.183	0.000
	Marketing factors	920.170	0.000
BWCON-NSDK-means++	Quality factors	1274.006	0.000
	External factors	295.092	0.000
	Public praise factors	208.821	0.000
	Marketing factors	1468.264	0.000

Table 15: The distribution of the sub-markets

Algorithm	Distribution		
	Segment 1	Segment 2	Segment 3
BWCON-Average linkage	2711	8	28
BWCON-Ward linkage	827	782	1138
BWCON-NSDK-means++	632	865	1250

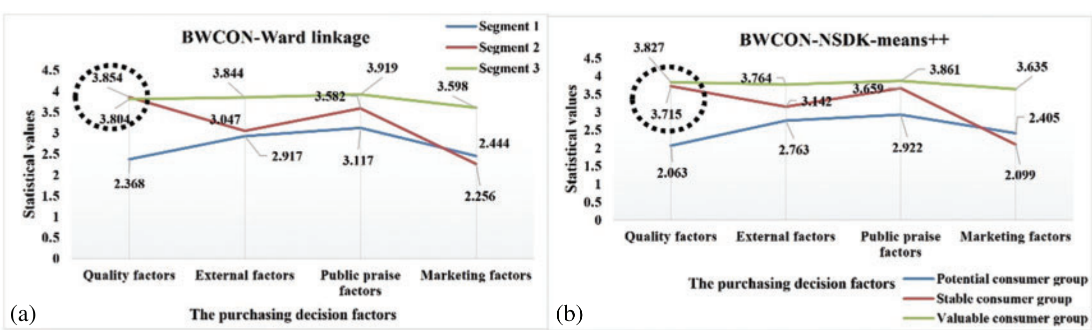


Figure 9: Visualization of segments on the SB

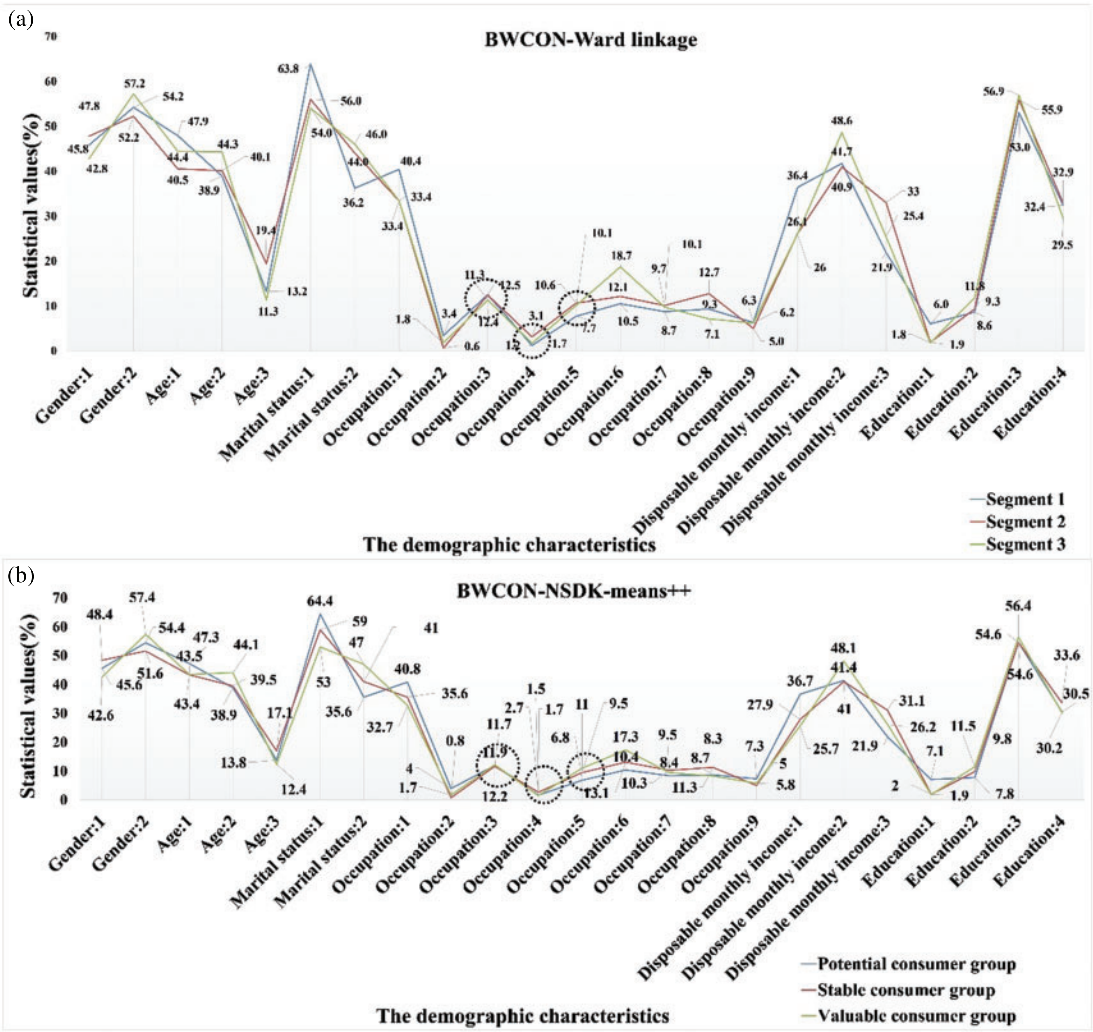


Figure 10: Visualization of segments on the demographic characteristics

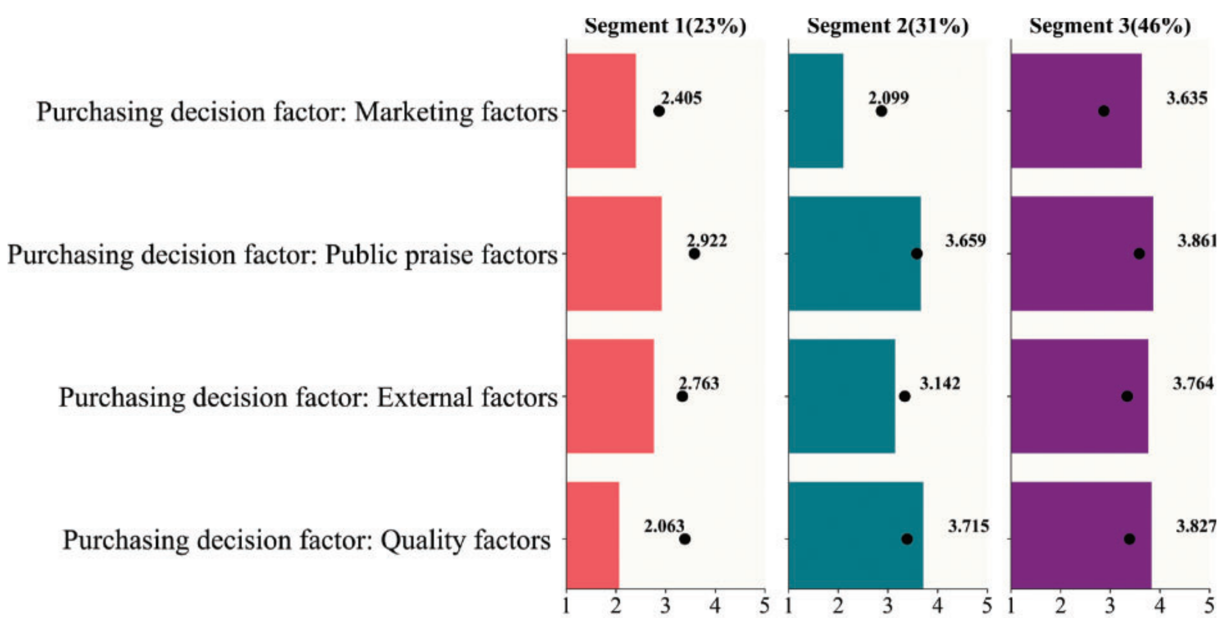
Table 14 tabulates the results of the Kruskal-Wallis test. Except for the BWCON-Single linkage, the p-values corresponding to the remaining three algorithms are all less than 0.05, which indicate that there are significant differences on the SB among the three sub-markets obtained by the BWCON-Average linkage, BWCON-Ward linkage and BWCON-NSDK-means++. Based on this, we further summarize the sub-market sizes corresponding to these three algorithms in Table 15; and for the distribution, we adopt the view of Kotler et al. [37], i.e., any given sub-market is worthy of further analysis only if it contains at least 5% of all the consumers, and it is 138 here. Thus it can be seen that for the BWCON-Average linkage, although there are differences among the different segments, the sub-market sizes of both Segments 2 and 3 are significantly smaller than 138, which are of no research value to the enterprises; on the contrary, the sub-market scales of the BWCON-Ward linkage and BWCON-NSDK-means++ are both larger than 138, and their distributions are similar, which may have something to do with the fact that the AHC with the ward linkage itself is an excellent market segmentation tool [38].

Given the informative value of the market segmentation results derived from the BWCON-Ward linkage, we compare the consumption preferences and consumer characteristics of all the sub-markets obtained using the BWCON-Ward linkage and BWCON-NSDK-means++ to illustrate the robustness of the conclusions obtained by the BWCON-NSDK-means++ in Figs. 9 and 10, respectively, where for the SB, the mean value of each feature is taken as the statistical value, and for the demographic characteristics, the frequency share of each feature is taken as the final statistical value.

As shown in Fig. 9, the wine consumers in the "Segment 1" and "Potential consumer group" have completely similar consumption psychologies: Public praise factors > External factors > Marketing factors > Quality factors; similarly, both "Segment 2" and "Stable consumer group" are characterized by the same consumption tendencies: Quality factors > Public praise factors > External factors > Marketing factors; and as for the remaining two sub-markets, their consumer preferences are also only slightly different, e.g., in the "Segment 3", the relationship among different SB is Public praise factors > External factors > Quality factors > Marketing factors, and in the "Valuable consumer group", the relationship is Public praise factors > Quality factors > External factors > Marketing factors. That is to say, the purchasing psychologies derived using these two algorithms are basically identical. Not only that, from Fig. 10, except for the Occupations 1, 3, 4 and 5, and Education 3, the distributions of the demographic characteristics in our "Potential consumer group", "Stable Consumer Group", and "Valuable consumer group" are fully consistent with those in the "Segment 1", "Segment 2" and "Segment 3", respectively. Therefore, the conclusions provided by our method are robust and reliable, and the BWCON-NSDK-means++ is also indeed a qualified market segmentation tool.

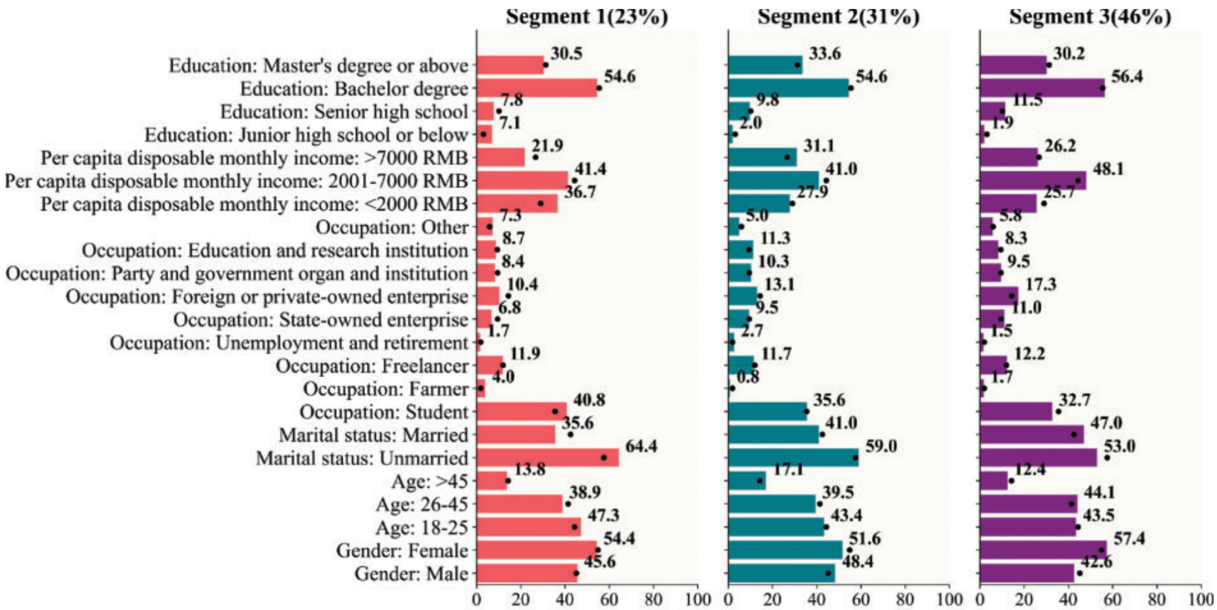
5.4 Results of the Chinese Wine Market Segmentation

In this section, to have a better understanding of the practical meanings of our obtained market segmentation results based on the BWCON-NSDK-means++, we visualize each sub-market in Fig. 11, in which the black dots represent the characteristics of the entire Chinese wine market, and for each sub-market, the numerical variables have the mean as the statistical object, and the categorical variable has the frequency share as the statistical object.



(a)

Figure 11: (Continued)



(b)

Figure 11: Visualization of segments based on the BWCON-NSDK-means++: (a) Differences on the SB; (b) Differences on the demographic characteristics

Segment 1: Potential consumer group (23%). This is the smallest of the segments. Compared with the other two consumer groups, this group has the largest share of young unmarried consumers (47.3% are 18–25 years old) and more students, farmers and other occupations. In particular, the students account for 40.8% of the total consumer group, making this consumer group the one with the lowest per capita disposable monthly income of the three consumer groups (i.e., the disposable income per month of 36.7% consumers is less than 2000 RMB). In term of education, maybe since the percentage of farmers is slightly higher than the other two groups, there are slightly more consumers with junior high school education or below, but within the group, 54.6% consumers have a bachelor's degree. For this young group, the concern for wine quality, external factors, public praise factors and marketing factors is lower than the average level of the whole market, but inside the sub-market, this group cares more about the public praise and external factors of the wine. Therefore, if an enterprise plans to choose this sub-market as its target market, it can improve its public praise or change the price, package, etc. to increase consumers' satisfaction.

Segment 2: Stable consumer group (31%). This is the second largest segment. Among the three consumer groups, this group has the largest proportion of elderly male consumers (17.1% of consumers are over 46 years old), and because of this, the proportion of the leavers and retirees in this group is higher than the other two groups (i.e., 2.7% of consumers have left their jobs or retired). In addition, for the occupation, the shares of employees come from Party and government organ (10.3%) and in education and research institutions (11.3%) are also higher than them in the other two groups. At the same time, this is a high-income and highly educated group; 31.1% of consumers have a per capita disposable monthly income above 7000 RMB, and 33.6% of consumers are with master's degree or above. Within the market, the consumers care most about the quality of the wine when purchasing wine; therefore, improving the quality of wine can increase the satisfaction of this consumer group.

Segment 3: Valuable consumer group (46%). This is the largest segment. In this consumer group, there are more married middle-aged female consumers than in the other two groups (i.e., 57.4% of consumers are female, 44.1% are 26–45 years old); and freelancers, state, foreign and private employees account for a higher proportion than the other two groups. This is a middle-income consumer group (i.e., 48.1% of consumers have a per capita disposable monthly income of 2001–7000 RMB), and consumers with senior high school or bachelor degrees are higher than the other two groups. This group is more concerned with the quality, external factors (i.e., price, package, etc.), public praise and marketing factors; in particular, if an enterprise chooses this sub-market as its target market, improving the public praise of the wine can significantly increase consumers' satisfaction.

6 Conclusions and Future Work

It is a premise to set a reasonable ONS to carry out a successful market segmentation, however, in the current research, there is a serious lack of attention to this issue. In our study, we propose such a method called BWCON-NSDK-means++ to adaptively determine the ONS by effectively integrating a new IVI and a valid clustering algorithm into a novel ONS estimation framework. Specifically, inspired by the neighboring samples, a connectivity formula is quantitatively defined for the first time, and thus the BWCON is designed to comprehensively evaluate the market segmentation results from three perspectives: compactness, separation and connectivity. Then, a BWCON-based ONS estimation framework is innovatively constructed by elegantly trade-off the ONCs from the three evaluation dimensions. Finally, the BWCON-NSDK-means++ is obtained by integrating our improved NSDK-means++ into the aforementioned ONS estimation framework. The final experimental results show that compared with those existing models, the BWCON and NSDK-means++ are more suitable to be combined to determine the ONC. Moreover, the experiments on the Chinese wine market dataset in particular prove that the BWCON-NSDK-means++ is not only an effective ONS estimation method, but also a qualified market segmentation tool without extra parameters; and it can help relevant practitioners understand a market more objectively and thus make more correct and valuable decisions.

Experiments have demonstrated the power of the BWCON-NSDK-means++, but there are still some shortcomings to be further optimized, such as the choice of the clustering algorithm used in the combination. In the present study, we choose the partition-based NSDK-means++ to integrate with the BWCON mainly due to its stability and the fact that only one parameter NC needs to be set, but we ignore its limitation that the partitioning clustering itself is more suitable for spherical datasets. Therefore, as a future direction, we will further optimize the NSDK-means++ to make the combination "BWCON+NSDK-means++" more widely applicable.

Funding Statement: This study was supported by the earmarked fund for CARS-29 and the open funds of the Key Laboratory of Viticulture and Enology, Ministry of Agriculture, China.

Availability of Data and Materials: The Seeds, Vehicle, Cleveland, Balance, Haberman, Thyroid and Wine are the public datasets, and they are available in the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/). The S1, S2 and S3 are generated by using the "multivariate_normal" function from the numpy. The S4 and S5 are generated using the "make_blobs" function of the sklearn. And the Chinese wine market dataset is available on request from the corresponding author.

Conflicts of Interest: The authors declare that we have no conflicts of interest to report regarding the present study.

References

1. Chang, Y. T., Fan, N. H. (2022). A novel approach to market segmentation selection using artificial intelligence techniques. *The Journal of Supercomputing, 22(2),* 79. https://doi.org/10.1007/s11227-022-04666-2
2. Casas-Rosal, J. C., Segura, M., Maroto, C. (2021). Food market segmentation based on consumer preferences using outranking multicriteria approaches. *International Transactions in Operational Research, 30(3),* 1537–1566. https://doi.org/10.1111/itor.12956
3. Seo, H. (2021). Dual' labour market? Patterns of segmentation in European labour markets and the varieties of precariousness. *Transfer: European Review of Labour and Research, 27(4),* 485–503. https://doi.org/10.1177/10242589211061070
4. Wang, H. J. (2022). Market segmentation of online reviews: A network analysis approach. *International Journal of Market Research, 64(5),* 652–671. https://doi.org/10.1177/14707853211059076
5. Qi, J., Li, Y., Jin, H., Feng, J., Mu, W. (2022). User value identification based on an improved consumer value segmentation algorithm. *Kybernetes, 13(3),* 233. https://doi.org/10.1108/K-01-2022-0049
6. Abbasimehr, H., Bahrini, A. (2022). An analytical framework based on the recency, frequency, and monetary model and time series clustering techniques for dynamic segmentation. *Expert Systems with Applications, 192(4),* 116373. https://doi.org/10.1016/j.eswa.2021.116373
7. Zhou, S., Liu, F., Song, W. (2021). Estimating the optimal number of clusters via internal validity index. *Neural Processing Letters, 53(2),* 1013–1034. https://doi.org/10.1007/s11063-021-10427-8
8. Abdalameer, A. K., Alswaitti, M., Alsudani, A. A., Isa, N. A. M. (2022). A new validity clustering index-based on finding new centroid positions using the mean of clustered data to determine the optimum number of clusters. *Expert Systems with Applications, 191,* 116329. https://doi.org/10.1016/j.eswa.2021.116329
9. Tavakkol, B., Choi, J., Jeong, M. K., Albin, S. L. (2022). Object-based cluster validation with densities. *Pattern Recognition, 121,* 108223. https://doi.org/10.1016/j.patcog.2021.108223
10. Naderipour, M., Zarandi, M. H. F., Bastani, S. (2022). A fuzzy cluster-validity index based on the topology structure and node attribute in complex networks. *Expert Systems with Applications, 187(1),* 115913. https://doi.org/10.1016/j.eswa.2021.115913
11. Zhou, S., Liu, F. (2022). A novel internal cluster validity index. *Journal of Intelligent & Fuzzy Systems, 38(4),* 4559–4571. https://doi.org/10.3233/JIFS-191361
12. He, H., Zhao, Z., Luo, W., Zhang, J. (2021). Community detection in aviation network based on K-means and complex network. *Computer Systems Science and Engineering, 39(2),* 251–264. https://doi.org/10.32604/csse.2021.017296
13. Sheikhhosseini, Z., Mirzaei, N., Heidari, R., Monkaresi, H. (2021). Delineation of potential seismic sources using weighted K-means cluster analysis and particle swarm optimization (PSO). *Acta Geophysica, 69(6),* 2161–2172. https://doi.org/10.1007/s11600-021-00683-6
14. Wu, G., Zhang, D., Fan, S. (2022). Application of an improved K-means clustering algorithm in power user grouping. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 35(4).* https://doi.org/10.1002/jnm.2990
15. Du, G., Li, X., Zhang, L., Liu, L., Zhao, C. (2021). Novel automated K-means++ algorithm for financial data sets. *Mathematical Problems in Engineering, 2021,* 1–12. https://doi.org/10.1155/2021/5521119
16. Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics, 3(3),* 32–57. https://doi.org/10.1080/01969727308546046
17. Davies, D. L., Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1(2),* 224–227. https://doi.org/10.1109/TPAMI.1979.4766909
18. Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20(87),* 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

19. Calinski, T., Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-Simulation and Computation, 3(1),* 1–27. https://doi.org/10.1080/03610917408548446
20. Zhou, S., Xu, Z., Tang, X. (2011). Method for determining optimal number of clusters based on affinity propagation clustering. *Control and Decision, 26(8),* 7.
21. Dogan, A., Birant, D. (2022). K-centroid link: A novel hierarchical clustering linkage method. *Applied Intelligence, 52(5),* 5537–5560. https://doi.org/10.1007/s10489-021-02624-8
22. Mahdi, M. A., Hosny, K. M., Elhenawy, I. (2021). Scalable clustering algorithms for big data: A review. *IEEE Access, 9,* 80015–80027. https://doi.org/10.1109/access.2021.3084057
23. Chan, C., Al-Bashabsheh, A., Zhou, Q. (2021). Agglomerative info-clustering: Maximizing normalized total correlation. *IEEE Transactions on Information Theory, 67(3),* 2001–2011. https://doi.org/10.1109/TIT.2020.3040492
24. Banerjee, P., Chakrabarti, A., Ballabh, T. K. (2021). Accelerated single linkage algorithm using the farthest neighbour principle. *Sādhanā, 46(1),* 45. https://doi.org/10.1007/s12046-020-01544-6
25. Tokuda, E. K., Comin, C. H., Costa, L. (2022). Revisiting agglomerative clustering. *Physica A: Statistical Mechanics and its Applications, 585(3),* 126433. https://doi.org/10.1016/j.physa.2021.126433
26. Sinaga, K. P., Yang, M. S. (2020). Unsupervised K-means clustering algorithm. *IEEE Access, 8,* 80716–80727. https://doi.org/10.1109/ACCESS.2020.2988796
27. Sculley, D. (2010). Web-scale k-means clustering. *Proceedings of the 19th International Conference on World Wide Web-WWW '10*. https://doi.org/10.1145/1772690.1772862
28. Fahim, A. (2021). K and starting means for K-means algorithm. *Journal of Computational Science, 55(1),* 1011445. https://doi.org/10.1016/j.jocs.2021.101445
29. Arthur, D., Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035. New Orleans, Louisiana.
30. Zhang, M., Duan, K. F. (2015). Improved research to K-means initial cluster centers. *Ninth International Conference on Frontier of Computer Science and Technology*, pp. 349–353. https://doi.org/10.1109/fcst.2015.61
31. Zhu, Q., Feng, J., Huang, J. (2016). Natural neighbor: A self-adaptive neighborhood method without parameter K. *Pattern Recognition Letters, 80(1),* 30–36. https://doi.org/10.1016/j.patrec.2016.05.007
32. Li, Y., Chu, X., Tian, D., Feng, J., Mu, W. (2021). Customer segmentation using K-means clustering and the adaptive particle swarm optimization algorithm. *Applied Soft Computing, 113(1),* 107924. https://doi.org/10.1016/j.asoc.2021.107924
33. Qaddoura, R., Faris, H., Aljarah, I. (2020). An efficient clustering algorithm based on the k-nearest neighbors with an indexing ratio. *International Journal of Machine Learning and Cybernetics, 11(3),* 675–714. https://doi.org/10.1007/s13042-019-01027-z
34. Shi, J., Ye, L., Li, Z., Zhan, D. (2022). Unsupervised binary protocol clustering based on maximum sequential patterns. *Computer Modeling in Engineering & Sciences, 130(1),* 483–498. https://doi.org/10.32604/cmes.2022.017467
35. Maciejewski, G., Mokrysz, S., Wróblewski, Ł. (2019). Segmentation of coffee consumers using sustainable values: Cluster analysis on the polish coffee market. *Sustainability, 11(3),* 613. https://doi.org/10.3390/su11030613
36. Mu, W., Zhu, H., Tian, D., Feng, J. (2017). Profiling wine consumers by price segment: A case study in Beijing, China. *Italian Journal of Food Science, 29(3),* 377–397.
37. Kotler, P., Stewart, A., Linden, B., Armstrong, G. (2001). *Principles of marketing*. Frenchs Forest: Pearson Education Australia.
38. Kaliji, S. A., Imami, D., Gjonbalaj, M., Canavari, M., Gjokaj, E. (2022). Fruit-related lifestyles as a segmentation tool for fruit consumers. *British Food Journal, 124(13),* 126–142. https://doi.org/10.1108/BFJ-09-2021-1001