***IEEE Xplore* ®**
Notice to Reader

"A Novel Effective Distance Measure and a Relevant Algorithm for Optimizing the Initial Cluster Centroids of K-means"

by Yang Liu, Shuaifeng Ma, and Xinxin Du

published in IEEE *Access* Early Access

Digital Object Identifier: 10.1109/ACCESS.2020.3044069

It is recommended by the Editor-in-Chief of IEEE *Access* that this article will not be published in its final form.

We regret any inconvenience this may have caused.

Derek Abbott
Editor-in-Chief
IEEE *Access*

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017. Doi Number

A Novel Effective Distance Measure and a Relevant Algorithm for Optimizing the Initial Cluster Centroids of K-means

Yang Liu[1], Shuaifeng Ma[2]，Xinxin Du[2]

[1] School of Statistics, Southwestern University of Finance and Economics, Chengdu, 613000 China

[2] Big Data Operation Center, Jingdong Century Trading Co., Ltd., Beijing, 100000 China

Corresponding author: Yang Liu (e-mail: statsyangliu@163.com).

This work was supported in part by the commercial project of Jingdong Century Trading Co., Ltd. under Grant JB34947B.

ABSTRACT The traditional K-means algorithm is very sensitive to the selection of clustering centers and the calculation of distances, so the algorithm easily converges to a locally optimal solution. In addition, the traditional algorithm has slow convergence speed and low clustering accuracy, as well as memory bottleneck problems when processing massive data. Therefore, an improved K-means algorithm is proposed in this paper. In this algorithm, the selection of the initial points in the traditional clustering algorithm is improved first, and then a new global measure, the effective distance measure, is proposed. Its main idea is to calculate the effective distance between two data samples by sparse reconstruction. Finally, on the basis of the MapReduce framework, the efficiency of the algorithm is further improved by adjusting the Hadoop cluster. Based on the real customer data from the JD Mall dataset, this paper introduces the DBI, Rand and other indicators to evaluate the clustering effects of various algorithms. The results show that the proposed algorithm not only has good convergence and accuracy but also achieves better performances than those of other compared algorithms.

INDEX TERMS K-means; clustering center; distance measurement; MapReduce; parallel computing

I. INTRODUCTION

A. Literature review

The big data era has led to the rapid development of machine learning technology. As one of the most commonly used traditional clustering algorithms, K-means has been successfully applied in a wide range of areas due to its simplicity, high practicality and high efficiency. Representative applications include document clustering, market segmentation, image segmentation and feature learning (Dhanachandra et al., 2015; Habib & Zahid, 2018; Siddiqui & Mat Isa, 2011; Tleis et al., 2017). Typically, the K-means algorithm consists of three stages: feature selection, feature extraction, and data clustering based on the calculated similarities between data points. The aim of clustering is to divide data into multiple classes or clusters so that the data in the same cluster possess high similarity and the similarity between data in different clusters is low (Sridharan & Sivakumar, 2018). Generally, clustering algorithms fall into two categories: hierarchical clustering and partitional clustering (A & B, 2017). Hierarchical clustering algorithms build a high-level hierarchy of clusters called a dendrogram according to the similarities between data points. A dendrogram can be constructed by two different approaches: agglomerative clustering (merging clusters bottom-up) and divisive clustering (splitting clusters top-down). On the other hand, partitional clustering algorithms require predefining the number of clusters and the initial cluster centroids. These algorithms divide a dataset into multiple clusters without overlap by minimizing a specific loss function (Tal, 2015; Topchy et al., 2004). Proposed in 1967, K-means clustering is one of the most widely used clustering algorithms. It has been widely employed in a large number of applications due to its simplicity and superior performance compared to other clustering algorithms. However, K-means has some limitations. First, the number of clusters k needs to be predefined. Additionally, the initial cluster centroids of K-means are usually selected randomly. Finally, the performance of K-means can be influenced by outliers in the data. To address the above issues with regard to K-means, researchers in different fields have proposed various improved algorithms (Gu, 2016; Premkumar & Ganesh, 2017; Rodriguez & Laio, 2014). Please see section II for a detailed illustration of the K-means algorithm.

The K-means clustering algorithm is a dynamic hard clustering algorithm based on the similarities between static data objects. Compared with other clustering algorithms in terms of complexity, the K-means clustering algorithm is simple to implement and has low linear time complexity, so it is widely used in data science, industrial application, and other fields. However, the K-means clustering algorithm also has some shortcomings, including its inability to determine the proper number of clusters, the high randomness of its clustering results, and its great dependence on the selection of the initial clustering center. The clustering results are greatly influenced by the initial clustering centers, and this causes the clustering algorithm to fall into the local optimal solution rather than the global optimal solution. Furthermore, pretreatment is too costly in cases of massive data analyses, and this influences the overall performance of the algorithm (Ailon et al., 2009; Cohen-Addad, 2018; Friggstad et al., 2019; Stemmer, 2020).

In view of the shortcomings and defects of K-means, many scholars have improved and optimized the K-means algorithm, and these improved algorithms are widely used in different fields (Frey & Dueck, 2007; J. Shi & Luo, 2010; Xu & Ii, 2005). For the selection of K values, as early as 1998, (Rezaee et al., 1998) proposed that the best K value is within the range of (1, $\sqrt{n}$), where n is the data size. This also provided a direction for later improvements to the traditional K-means algorithm. Based on the relationship between the clustering number K and the sum of squared errors SSE, (Chakraborty & Das, 2017) selected the K value corresponding to the elbow point as the optimal clustering number according to the variation trends of the SSE for different K values. To solve the problem of indistinct "dots" in the relation between K and the SSE, (Celebi et al., 2013) determined the optimal K value by combining parameters such as the exponential function parameter, weight term and bias term. For the problem in which the optimal clustering number K needs to be determined by manually analyzing the decision graph, in combination with a statistical method, (Lei et al., 2016; Rodriguez et al., 2014) used linear regression to fit the points in the decision graph and determined the optimal K value and the initial clustering center according to the differences between the observed values and the actual values.

With regard to the selection of clustering centers, (Xiong et al., 2016) first calculated the densities of all data objects, determined the average density of the dataset, selected the data object with the largest density value as the first initial clustering center by taking the data objects with larger values than the average density as the high-density point set, and selected the remaining clustering centers according to the principle of maximum distance from the previously selected clustering centers. Additionally, based on this density-based improvement method, to avoid taking two high-density points in a cluster as the initial clustering center, (Xin et al., 2017) used the basic idea that the distances between the clustering center point and other center points should be relatively large and selected the center point by combining the relative distance with the high-density point. (Tanir & Nuriyeva, 2017) first proposed selecting the two points in the dataset with the largest distance between them as the initial clustering centers, assigning the remaining data objects to the corresponding clusters according to their distances from the clustering center points, updating the clustering centers, and continuing to find the points farthest from the clustering center as the next center points until the number of clustering center points was K. By combining the minimum spanning tree algorithm in graph theory with the traditional K-means algorithm, (Xiao-bin et al., 2014) first used the Prim algorithm to generate a minimum spanning tree based on the hierarchical K-means algorithm, then divided the minimum spanning tree into m subclusters according to the maximum splitting distance principle, found the k ($k \leqslant m$) clusters with the most data objects from the subclusters, and calculated the mean value of each cluster as the initial clustering center for traditional iterative K-means processing.

Regarding distance measurement, (Visalakshi & Suguna, 2009) proposed a spatial similarity measurement for the K-means algorithm in view of the problem that the traditional K-means algorithm is not effective in classifying non-clustered data, used the spatial density similarity distance measure to compensate for the shortcoming that Euclidean distance cannot accurately express the similarities between flow data objects, and obtained clustering results by combining the new clustering centers with the iterative model. (Fan et al., 2017) applied the K-means algorithm to text clustering; measured and compared the similarities between text data objects by using the Euclidean distance, squared Euclidean distance, Manhattan distance, cosine distance and valley distance measures; and concluded from the clustering results of different methods that Euclidean distance has some limitations in terms of measuring similarities in text, and that different similarity measurement methods should be selected according to the datasets in question. (Yan et al., 2018) calculated the weight values of each feature in a dataset using information gain and feature selection algorithms, took the average value as the final feature weight value, and combined it with the Euclidean distance as the weighted distance for K-means clustering, thereby achieving good results. In addition, (Pawlak, 1994; Qian et al., 2008; Xian-Cai, 2008) proposed rough set theory, which is a soft computing tool for dealing with uncertain and fuzzy knowledge and has unique advantages in processing classified attribute data. (Albanese et al., 2011; C. B. Chen & Wang, 2006; Parmar et al., 2007) used rough set theory for clustering attribute data. (Albanese et al., 2011; Breunig et al., 2000) carried out many studies focusing on the problem of handling outliers when performing clustering.

B. Motivation

K-means exploits the similarities in data through clustering. It has advantages in several aspects, such as simplicity and high efficiency. However, it also suffers from some shortcomings. First, it is difficult to estimate the number of clusters. The algorithm is easily trapped in local minima with randomly selected initial cluster centroids (Jain, 2010; Jianbin et al., 2013; Xu & Ii, 2005). Many algorithms have been proposed to improve K-means, but the existing algorithms still do not fully address its problems. For example, (Wang et al., 2015) determined the appropriate number of clusters and initial cluster centroids based on ideas from image segmentation. This algorithm segments the original dataset using the data density and watershed algorithm. The centroids of the segmentation regions are used as the initial data centroids, and the number of segmented regions is adopted as the number of clusters k. Although this method can obtain accurate values of k and initial cluster centroids to some extent, the watershed algorithm is sensitive to noise and prone to over-segmentation. Therefore, the performance of this algorithm drops significantly if the dataset contains noise. (Pelleg & Moore, 2002) proposed the X-means algorithm. This algorithm relies on the Bayesian information criterion (BIC) to calculate scores for selecting the local centroids to be further split. Each region corresponding to a selected centroid is then split into two subregions based on some criteria to determine the optimal number of clusters. This method has high computational complexity due to the calculation of variance. Additionally, it cannot achieve good performance for datasets with outliers. Based on the previous discussion on K-means, this paper proposes an improved K-means algorithm, IKM, which is detailed in section II of this paper. IKM includes two stages. The first stage selects the initial cluster centroids and determines the appropriate number of clusters. The second stage performs K-means clustering based on the centroids and number of clusters determined in the first stage. Additionally, the traditional K-means algorithm relies on the Euclidean distance measure to compute the distances between data samples. However, the Euclidean distance only considers the pairwise distance between two samples. It does not take the structure of the global data

distribution into account (D. et al., 2013; Patrick et al., 2001; Rammal et al., 2014; Xiang et al., 2008). Therefore, this paper proposes a novel measure considering the global data distribution, namely, the effective distance measure. The main idea is to calculate the effective distances between data samples using sparse reconstruction. Third, a cluster analysis running on a single machine is usually bottlenecked by RAM and CPU speed when the dataset is large. An effective solution is to parallelize the algorithm and run it distributively on a cluster of multiple machines (X. Chen et al., 2017, 2018; He et al., 2012; Kusuma et al., 2016). Therefore, to address the aforementioned problems, this paper proposes a highly efficient parallel K-means algorithm based on MapReduce for big data environments. The proposed algorithm not only improves the quality of clustering but also enhances the efficiency of clustering in big data environments.

Specifically, IKM has two advantages compared to K-means. First, given a partition of attributes, IKM can determine the grid cell for each sample as well as the number of samples in each grid cell in a single pass. Although the number of potential grid cells could be very large, it is only necessary to create grids for nonempty cells and assign each sample to a cell. The time complexity of this algorithm is O(n), which is much lower than that of K-mean++. Second, the result of the proposed algorithm does not change when the input sequence changes. It is a multiresolution algorithm that relies on densities and grids to determine the correct number of clusters and recognize complex data patterns automatically. It selects the initial cluster centroids that are most likely to lead to fast convergence, thereby improving the overall efficiency of the algorithm. Moreover, K-means usually relies on Euclidean distance to calculate the distances between samples in many situations. However, Euclidean distance only focuses on pairwise distances, and it does not consider the structure of the global data distribution. To take global structural information into account, this paper proposes a new algorithm for the distance measure, EK-means, which is described in section II. The idea of EK-means is to construct a connectivity matrix for data samples. The effective distance is calculated using a ratio-based measure, and the output distance is then used in K-means clustering. Compared to the commonly used Euclidean distance and geodesic distance measures, the proposed effective distance considers the global structural information located between data samples, so it better exploits hidden structures and patterns in data. As a result, replacing the Euclidean distance with the effective distance can be used to better extract relation information from samples, and the effective distance is impervious to factors such as the distributions of samples and the geodesic distances between samples. Third, this paper proposes a highly efficient parallel algorithm for K-means based on MapReduce to address the problems that influence the performance of the traditional K-means algorithm, including slow convergence depending on the selection of initial cluster centroids, low accuracy in clustering, bottlenecks in RAM when handling big data, and high costs for analyzing and preprocessing in applications with big data. The proposed method adjusts Hadoop clusters automatically to further improve the efficiency of the algorithm.

Overall, the traditional K-means algorithm has the following shortcomings. First, the overall robustness is weak due to the random selection of the initial cluster centroids, which also leads to the possibility of the algorithm converging to suboptimal solutions. Additionally, the results can be influenced by outliers due to the random selection of the initial cluster centroids. Next, the selection of initial centroids can lead to slow convergence and low clustering accuracy. Finally, there exists a bottleneck in RAM and high costs for analyzing and preprocessing data in applications with big data. This paper addresses the above problems. The contributions of this study are three-fold. (1) Based on the above discussion on K-means, this paper proposes an improved algorithm to select initial cluster centroids, determine the appropriate number of clusters and recognize complex data patterns; (2) this paper proposes a novel global measure, the effective distance measure, to take the global structural information in data into account; (3) this paper proposes a highly efficient parallel algorithm for K-means based on MapReduce; it adjusts Hadoop clusters automatically to further improve the speed of the algorithm.

This paper performs experiments on the UCI dataset (US Census data from 1990) and JoyBuy dataset, with several evaluation measures, including DBI and Rand. The proposed improved K-means algorithm achieves the best performance among six different clustering algorithms in terms of four evaluation measures. It achieves first place in the average rank comparison, higher than the average score of the DEC algorithm as well as the average scores of the other three clustering algorithms. In terms of speed, the proposed algorithm also shows a significant advantage. Overall, the results demonstrate not only excellent convergence and accuracy but also improved efficiency.

II. Traditional K-means algorithms

Principles of the K-means Algorithm

The K-means algorithm is an unsupervised learning and clustering algorithm based on partitions, and it generally uses Euclidean distance as a measure of similarity between data objects. The similarity is inversely proportional to the distance between the data objects, so the greater the similarity is, the smaller the distance. In the algorithm, it is necessary to specify the numbers of initial clusters K and initial clustering centers k in advance to be able to constantly update the locations of the clustering centers according to the similarities between the data objects and the clustering centers and reduce the sum of squared errors (SSE) between clusters. When the SSE does not change or the objective function converges, the clustering process ends, and the final result is obtained.

The core idea of the K-means algorithm is to randomly select K initial clustering centers $C_i (1 \leqslant i \leqslant k)$ from the dataset first, calculate the Euclidean distances between the other data objects and the clustering center C_i, determine the nearest clustering center to the target data object C_i, assign the target data object to the correct clustering center, calculate the average value of the data objects in each cluster as the new clustering center, and carry out the succeeding iterations until the clustering center no longer changes or the maximum number of iterations is reached.

The computational formula for the Euclidean distance between data objects and clustering centers in space is as follows:

$$d(x, C_i) = \sqrt{\sum_{j=1}^{m} \left(x_j - C_{ij}\right)^2} \qquad (1)$$

where x is a data object, C_i is the ith clustering center, m is the dimension of the data object, and x_j and C_{ij} are the j-th attribute values of x and C_i, respectively.

The computational formula for the SSE of the entire

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3044069, IEEE Access

dataset is:

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} \left| d\left(x, C_i\right) \right|^2 \qquad (2)$$

where the size of the SSE represents the quality of the clustering results and k is the number of clusters.

Flow of the K-means Algorithm

Description of K-means algorithm:

Input: cluster number k, dataset D, D =($x_1, x_2, ..., x_n$);

Output: clustered dataset and k clustering centers

Begin

T=1/T represents the number of iterations

Randomly select k sample data points in the dataset as the initial clustering center

Repeat;

for (int i =0; i <n; ii =){

for (int j =0; j <k; jj =){

}

}

$\mathrm{J}_c(t) = \sum_{j=1}^{k} \sum_{x_i \in c_j} \left\| x_i - c_j \right\|^2$;

$T = t + 1$; / / Seek the next cluster center

for(int $j = 0; j < k; j++$){

$Z_j = \frac{1}{n_j} \sum_{i=1}^{n_j} x_i^{(f)}$;

}

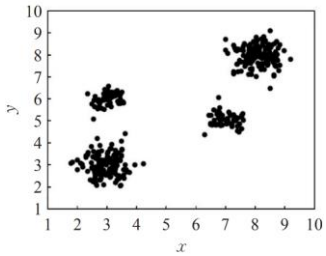
$J_c(T) = \sum_{J=1}^{K} \sum_{x_i \in c_j} \left\| x_i - c_j \right\|^2$

$Until \left| J_c(T) - J_c(t) \right| < \partial$ / / ∂ Tends to infinitesimal

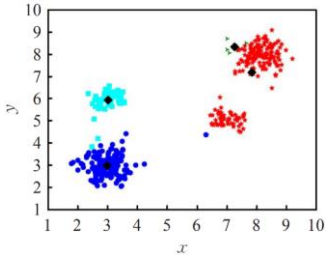
Output clustering results

End

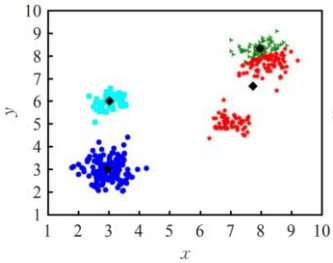
The K-means clustering algorithm is an iterative process(L. Shi et al., 2017). As shown in Figure 1, the original dataset has four clusters, where x and y in the graph represent the horizontal and vertical coordinate values of the data points, respectively. The K-means algorithm was used to cluster the dataset, and the final clustering result was obtained after two iterations of the algorithm.



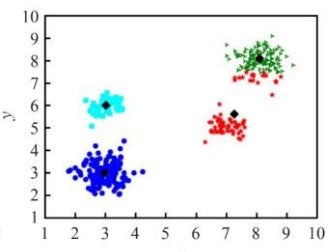
(a) Raw data



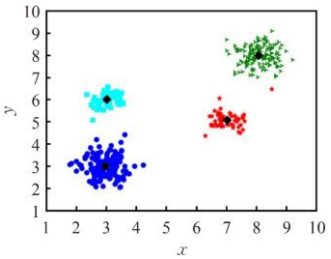
(b) Randomly select initial cluster centers



(c) First iteration



(d) Second iteration



(e) Final clustering result

Figure 1 Iterative Process of the K-means Algorithm

The K-means clustering algorithm is efficient for large datasets, and its algorithmic complexity is O(nmkT). n is the size of the dataset, m is the feature dimension of the data object, k is the number of specified clusters, and T is the total number of iterations, as shown in the flow chart in Figure 2.

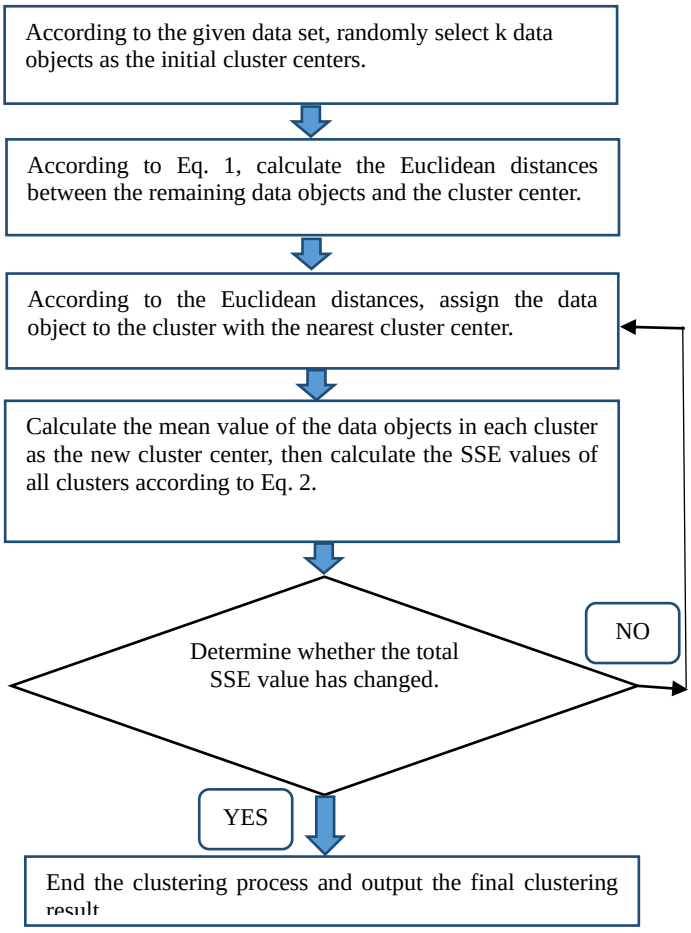


Figure 2 Flowchart of the K-means Clustering Algorithm

III. Improvement of the K-means Algorithm

A. Improved K-means Algorithmic principle

1. Methods

This study proposes an improved K-means (IKM) algorithm based on the previous discussion of the K-means algorithm. The IKM algorithm is divided into two stages:① select the initial clustering centers and determine the number of clusters; ② execute the original K-means algorithm based on the results of ①. The flow of the IKM algorithm is described below; the execution process of the IKM algorithm is shown in Figure 3.

Suppose there is a sample dataset, $S = \{x_1, x_2, \cdots, x_n\}$, and k initial cluster centers: $z_1, z_2, \cdots, z_k$.

Definition 1: The Euclidean distance between two data objects is as follows:

$$d\left(x_i, x_j\right)=\left(\left|x_{i1}-x_{j1}\right|^2+\left|x_{i2}-x_{j2}\right|^2+\cdots+\left|x_{ip}-x_{jp}\right|^2\right)^{\frac{1}{2}} \quad (3)$$

where $x_i=\left(x_{i1}, x_{i2}, \cdots, x_{ip}\right)$ and $x_j=\left(x_{j1}, x_{j2}, \cdots, x_{jp}\right)$ are two p-dimensional data objects.

Definition 2: The average distance between sample points is as follows:

$$\text{AvgDist}=\frac{1}{C_n^2}\times\sum d\left(x_i, x_j\right) \quad (4)$$

where n is the total number of sample points, C_n^2 is the number of possible combinations of two points out of the n total points, and $d\left(x_i, x_j\right)$ is the distance between data objects x_i and x_j.

Definition 3: For any data object point p in the space, the number of data objects in the area with point p as the center and AvgDist as the radius is called the density parameter of point p, which is recorded as density $(p,\ \text{AvgDist})$.

$$\begin{cases} u(x)=\begin{cases}1, x\geqslant 0\\ 0, \text{Otherwise}\end{cases} \\ \text{density}\ (p,\ \text{AvgDist})=\sum_{i=1}^{n} u\left(\ \text{AvgDist}\ -\left|p_i-p\right|\right)\end{cases} \quad (5)$$

where u(x) is a defined function and $\left|p_i-p\right|$ is the Euclidean distance between p_i and p.

Definition 4: The neighborhood distance of the data object is:

$$\text{DIS}_i=\sum_{j=1}^{n} d\left(x_i, x_j\right) \quad (6)$$

where X_j is a set of data objects in an area centered about X_i with a radius of AvgDist.

In the traditional K-means algorithm, the initial cluster centers are randomly selected, and the similarity between each pair of data objects is measured by Euclidean distance. The smaller the distance is, the more similar the objects, and the larger the distance is, the greater the difference. AvgDist is the average distance between the samples of the data object. If the data points p in the space are distributed in a space with AvgDist as the radius, the more distributed the points are, the greater the density parameter density $(p,\ \text{AvgDist})$, which means that point p is in a high-density distribution area. Taking point p as the cluster center is most conducive to the convergence of the objective function. According to the above principles, k points with the largest density parameters in the data object distribution are chosen as the initial clustering center points. The specific steps of this process are shown in Figure 3 below:

Step 1. Calculate the distance between two data objects according to Eq. 3. $d\left(x_i, x_j\right), i, j=1,2,\cdots,n$.

Step 2. Calculate the average distance AvgDist between data objects according to Eq. 4.

Step 3. Calculate the density $(p,\ \text{AvgDist})$ of all data objects according to Eq. 5 to form a set D.

Step 4. Scan the density parameter set D of the data object to find the set with the largest density parameter. If the set is not unique, calculate the field distance of the corresponding object in the set according to Eq. 6, and select the data object with the smallest distance in the neighborhood as the cluster center. Data objects whose distances from the cluster center are less than AvgDist are deleted from the sample dataset.

Step 5. Repeat steps (2)-(4) until k cluster centers are found.

Figure 3 IKM Algorithm Flowchart

2. Comparison of the IKM algorithm and K-means algorithm

Assume there are n input data points, the dimension of each data point is d, the total number of grids is g, and the number of determined clusters is k. The time complexity of the algorithm is analyzed as follows:

1) Calculate the distance from each data point to the clustering center and assign the point to the nearest cluster. The time complexity is O(nkd).

2) Calculate the minimum value of each cluster and reconfigure the new clustering center. The time complexity is O(nd).

3) Determine whether the distortion value converges; if so, stop the clustering analysis. The time complexity is O(nd).

Through the above analysis, we know that when the dataset approaches infinity, the overall time complexity of the IKM algorithm is O(n).

Let n be the number of inputs. Let d be the dimension of each data point. The computational cost of a single iteration of K-means consists of three components. First, step 2 and step 3 of K-means have computational complexities of O(nkd). Next, the computational complexity of step 4 is O(nd). Finally, the complexity is O(nd) for the calculation of the loss value to determine convergence and decide if the clustering process should be stopped.

The above comparisons between IKM and K-means demonstrate two advantages of IKM. First, given a partition of attributes, IKM can determine the grid cell for each sample as well as the number of samples in a grid cell for a single pass. Although the number of potential grid cells could be very large, it is only necessary to create grids for nonempty cells and assign each sample to a cell. The time complexity of this algorithm is O(n), which is much higher than that of K-mean++. Second, the result of the proposed algorithm does not change when the input sequence changes. It is a multiresolution algorithm that relies on density and grids to determine the appropriate number of clusters and recognize complex data patterns automatically. It selects the initial cluster centroids that are most likely to lead to fast convergence, thereby improving the overall efficiency of the algorithm.

3. Experiments

The clustering performance is determined by four measures: accuracy (AC), precision (PE), recall (RE) and the number of iterations. The definition of AC, PE and RE are as follows:

$$AC = \frac{\sum_{i=1}^{k} a_i}{n}; \quad PE = \frac{\sum_{i=1}^{k} \frac{a_i}{a_i + b_i}}{k}; \quad RE = \frac{\sum_{i=1}^{k} \frac{a_i}{a_i + c_i}}{k} \quad (7)$$

where n denotes the number of samples in the dataset, a_i denotes the number of true positives for class i, b_i denotes the number of false positives for class i, c_i denotes the number of false negatives for class i, and k is the number of clusters.

To examine the effectiveness of the proposed algorithm, three subdatasets are selected from the UCI dataset ("Drug Review Dataset", "Diabetes 130-US hospitals for years 1999-2008", and "Mushroom") to make comparisons between the proposed algorithm, K-means and K-mean++ (Arthur & Vassilvitskii, 2007). Tab. 1 describes the selected data.

TABLE 1
DESCRIPTION OF DATASETS

Dataset	Samples	Attributes	Classes	
			I	II
Drug Review Dataset	435	16	267	168
Diabetes 130-US hospitals for years 1999-2008	699	16	458	241
Mushroom	8124	22	3916	4208

The results of K-means depend on the selection of the initial cluster centroids. In other words, the results change with different selections of cluster centroids. Therefore, 100 initial centroids are randomly selected for the Drug Review Dataset, Diabetes 130-US hospitals for years 1999-2008 and Mushroom datasets, and each algorithm is run 500 times. The effectiveness of the algorithms is evaluated by their mean clustering performances. Tab. 2-4 shows comparisons between the performances of different algorithms on different datasets.

TABLE 2
COMPARISON OF ALGORITHMS ON THE DRUG REVIEW DATASET

Validation Measure	K-means	K-mean++	IKM
AC	0.821	0.871	**0.894**
PE	0.812	0.826	**0.847**
RE	0.832	0.858	**0.880**
Iterations	3.69	3.94	**3.47**

TABLE 3
COMPARISON OF ALGORITHMS ON THE DIABETES 130-US HOSPITALS FOR YEARS 1999-2008 DATASET

Validation Measure	K-means	K-mean++	IKM
AC	0.858	0.861	**0.872**
PE	0.789	0.805	**0.822**
RE	0.622	0.686	**0.690**
Iterations	3.72	3.85	**3.94**

TABLE 4
COMPARISON OF ALGORITHMS ON THE MUSHROOM DATASET

Validation Measure	K-means	K-mean++	IKM
AC	0.715	0.752	**0.778**
PE	0.682	0.691	**0.694**
RE	0.776	0.794	**0.802**
Iterations	5.82	6.42	**5.98**

Based on the data in Tab. 2-4, IKM with a novel method for selecting initial cluster centroids achieves the best performance on the Drug Review Dataset, Diabetes 130-US hospitals for years 1999-2008 and Mushroom datasets. Its performance is better than that of K-mean++. The experimental results demonstrate the effectiveness of the proposed method for selecting initial cluster centroids in IKM, which achieves higher accuracy than the other algorithms with fewer iterations.

B. Improvement of Distance Measurement -- Clustering Algorithm Based on Effective Distance

1. Concept of effective distance

(D. et al., 2013) proposed a nontraditional effective distance measurement method in a study of the factors affecting the spread of infectious diseases. Their distance measure is based on the population mobility ratio between two places due to air transport. The effective distance between a location and the origin of the pathogen is fixed and proportional to the time when the pathogen reaches the corresponding location. Moreover, their simulation of the transmission of the H1N1 and SARS viruses using this effective distance measure perfectly matched the real situation. Taking real pathogen transmission as an example, the effective distance method can also successfully determine the transmission source of infectious bacteria and predict the future transmission state. In contrast, the simple Euclidean distance measure cannot achieve satisfactory results in these cases. These facts show that the distance between two objects in the real world actually depends on more than the Euclidean distance or geographical distance between their geographical coordinates. When measuring the distance between two objects, in addition to considering the relationship between them, it is also necessary to consider the influences of other related objects, that is, to consider the global structural information of the data.

Therefore, in this paper, an effective distance measure is proposed to reflect the global structural information between samples in terms of probability. A specific diagram demonstrating the effective distance measure is shown in Figure 4. Assuming that there are 4 data sample points A, B, C, and D, Figure 4 (a) shows the directed relationships between the four sample points, and the weights of all sides are equal in Figure 4(a). In Figure 4(b), the weight of each side between the sample points in the directed graph is obtained by calculating the probability value P (n |m), and the wider an edge is, the greater the weight. The probability value P (n |m) represents the ratio of the number of direct paths from point m to the number of direct paths from point n. For example, the probability P (B |A)=1/4 means that the probability from A to B is 1/4, where 4 refers to a total of 4 paths from point A and 1 refers to 1 of these paths reaching point B directly. In addition, Figure 4(b) shows that the probability of arriving at point D from point B (e.g., P (D |B)=1) is significantly greater than the probability of arriving at point D from the point of departure (e.g., P (D|C)= 1/5). According to the idea of effective distance put forward by Brockmann et al., the smaller the probability value P (n |m) is, the greater the effective distance from point m to point n, and vice versa. Compared with the common Euclidean distance or geographical distance, because effective distance considers the global structural information between data samples, it can reflect the hidden pattern information between data samples. Therefore, replacing the Euclidean distance measure with the effective distance measure leads to a more comprehensive demonstration of the correlation information between samples, and effective distance is completely unaffected by the sample distribution, geographical distance and other factors.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3044069, IEEE Access

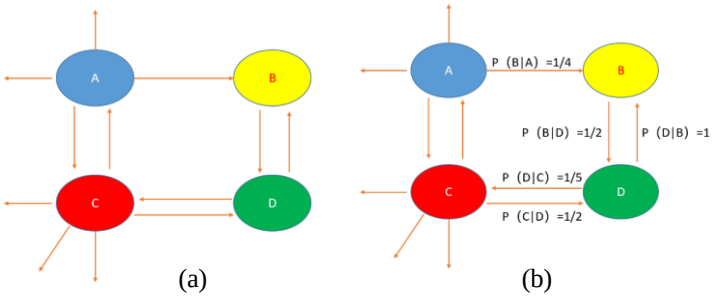


Figure 4 Directed Relational Graph

2. Methods

Traditional K-means clustering algorithms generally use Euclidean distance to evaluate the similarity between two data samples. Considering the idea of effective distance, an EK-means clustering algorithm based on effective distance is proposed in this paper.

(1) The construction of effective distance

As mentioned before, the proposed effective distance takes the global information in the data into account. Specifically, sparse representation can effectively express the global characteristics of data. Therefore, this paper proposes a method for calculating the effective distance based on sparse representation. The detailed steps are as follows:

Step 1. Construct a directed graph based on the coefficient weights of data samples in the sparse representation:

$$\min_{w_i} \left\|x_i - Bw_i\right\|_2^2 + \lambda \left\|w_i\right\|_1 \text{ s.t. } w_i \geqslant 0 \qquad (8)$$

where $\mathbf{B} = \left[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_n\right]^{\mathrm{T}}$ represents a dataset X with $\mathbf{x}_i$ removed.

A coefficient weight matrix $W = \left[w_1, w_2, ..., w_n\right]^T$ can be obtained by minimizing Eq. 8. W_{ij} is the weight of sample $\mathbf{x}_i$ in the sparse representation of x_j; λ is a regularization parameter. $\lambda \in (0,1]$. The larger λ is, the sparser the coefficients.

Step 2. Normalize the coefficient weights.

$$P_{ij} = \frac{W_{ij}}{\sum_{i=1}^{n} W_{ij}} \qquad (9)$$

A normalized coefficient weight matrix P can be obtained by Eq. 9. The higher P_{ij} is, the larger the weight of $\mathbf{x}_i$ in the sparse representation of x_j. A higher P_{ij} also illustrates a higher ranking of $\mathbf{x}_i$ among the nearest neighbors of x_j, a higher similarity between $\mathbf{x}_i$ and x_j, and a smaller effective distance between the two samples.

Step 3. Calculate the effective distances between samples:

$$ED_{ij} = 1 - \ln P_{ij} \qquad (10)$$

A matrix of effective distances, ED, can be obtained by Eq. 10. Since $0 \le P_{ij} \le 1$, $\ln P_{ij} \leqslant 0, ED_{ij} \geqslant 1$.

(2) EK-means clustering algorithm

The objective function of the EK-means clustering algorithm is:

$$\min_{|v_v, v_2, \cdots, v_c|} J_{11}(\mathbf{U}, \mathbf{V}) = \sum_{j=1}^{c} \sum_{i=1}^{n} u_{ji} \mathbf{D}_{ij} \qquad (11)$$

where D_{ij} is the effective distance from a data sample x_i to the clustering center v_j, and

$$D_{ij} = f\left(\mathbf{x}_i, \mathbf{v}_j\right) = \frac{1}{m} \sum_{q=1}^{m} ED_{i,r_{jq}} \qquad (12)$$

In Eq. 12, m represents the number of data samples contained in the cluster, and r_{jq} represents the number of data samples belonging to the q-th cluster in the original data sample.

The specific flow of the EK-means algorithm is as follows.

Input: sample set $X = [x_1, x_2, ..., x_n]^T$, number of categories c from pre-clustering

Output: c sample classes

1. Calculate the effective distance between each pair of raw data samples by the sparse representation method and construct the effective distance matrix $ED \in R_{n \times n}$;

2. Randomly initialize the clustering centers $v_1, v_2, ..., v_c$;

3. Calculate the effective distance between each data sample x_i and each clustering center $\mathbf{v}_k$ according to Eq. 12, and then assign the data sample to the cluster nearest to it;

4. Recalculate the clustering center of each cluster;

5. Continue until the central point of clustering does not change or the number of iterations exceeds 100, and then stop;

6. Return $v_1, v_2, ..., v_c$.

The section uses an example to demonstrate how to calculate effective distances, as well as the implementation steps of EK-means.

Input: Assume we have five sample points in three-dimensional space: $x_1 = [3,1,2]^{\mathrm{T}}$, $x_2 = [5,2,4]^{\mathrm{T}}$, $x_3 = [2,6,7]^{\mathrm{T}}$, $x_4 = [4,6,8]^{\mathrm{T}}$ and $x_5 = [5,7,2]^{\mathrm{T}}$. The random cluster centroids are selected as $\mathbf{v}_1 = x_1 = [3,1,2]^{\mathrm{T}}$ and $\mathbf{v}_2 = x_2 = [5,2,4]^{\mathrm{T}}$.

Output: two sample clusters

Step 1. Construct the effective distance matrix.

1. Calculate the coefficient weight matrix according to Eq. 8.

Taking w_1 as an example, the objective function to minimize is as follows:

$$\min_{w_1} \left\|x_1 - Bw_1\right\|_2^2 + \lambda \left\|w_1\right\|_1 \text{ s.t. } w_i \geqslant 0$$

This is equivalent to

$$\min_{w_{11}, w_{12}, w_{13},} \left\| \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} - \begin{bmatrix} 5 & 2 & 4 & 5 \\ 2 & 6 & 6 & 7 \\ 4 & 3 & 8 & 2 \end{bmatrix} \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \end{bmatrix} \right\|_2^2 + \lambda \left\| \begin{bmatrix} w_{11} \\ w_{12} \\ w_{13} \end{bmatrix} \right\|_1 \text{ s.t. } w_{11}, w_{12}, w_{13} \geqslant 0$$

w_2, w_3, and w_4 can be calculated using a similar method, leading to the coefficient weight matrix $W = \left[w_1, w_2, w_3, w_4\right]^T$.

2. Normalizing the coefficient weights using Eq. 9 leads to:

$$P = \begin{bmatrix} 0.0552 & 0.3794 & 0.0624 & 0.0588 & 0.0215 \\ 0.7276 & 0.2765 & 0.4002 & 0.9039 & 0.2749 \\ 0.2171 & 0.3440 & 0.5373 & 0.0372 & 0.7036 \end{bmatrix}.$$

3. Calculating the effective distances between samples according to Eq. 10 yields:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3044069, IEEE Access

$$ED = \begin{bmatrix} 3.8968 & 1.9691 & 3.7742 & 3.8336 & 2.7846 \\ 1.3180 & 2.2855 & 1.9158 & 1.0103 & 2.1984 \\ 2.5274 & 2.0671 & 1.6211 & 4.2914 & 3.4749 \end{bmatrix}.$$

Step 2. Choose random cluster centroids as $\mathbf{v}_1 = x_1 = [3,1,2]^{\mathrm{T}}$ and $\mathbf{v}_2 = x_2 = [5,2,4]^{\mathrm{T}}$.

Step 3. Calculate the distances between each sample $\mathbf{x}_i$ and the cluster centroids $\mathbf{v}_1, \mathbf{v}_2$ according to Eq. 12, and assign the sample to the nearest cluster. The effective distances between x_3 and the centroids $\mathbf{v}_1, \mathbf{v}_2$ are 2.226 and 3.748, respectively. Therefore, x_2 is assigned to cluster $\mathbf{v}_1$, x_4 is assigned to cluster $\mathbf{v}_1$, and x_5 is assigned to cluster $\mathbf{v}_2$.

Step 4. Recalculate the centroids for each cluster and repeat the above steps until convergence is achieved. Thus, the sample points are divided into two clusters.

3. A comparison between EK-means and K-means

In the K-means algorithm, Euclidean distance is usually used to measure the distance between two data objects. However, the Euclidean distance only represents the distance between each pair of samples. It does not consider the global distribution of the data. In contrast, this paper proposes a global measure, effective distance, to model global structural information. The idea is to construct a connectivity matrix for the data samples, calculate the effectives distance between samples based on ratios, and use the obtained results in K-means. Compared to the commonly used Euclidean distance and geodesic distance, the effective distance takes the global structural information in data into account, so it better describes the hidden patterns and structures in data. Therefore, using the effective distance to replace the Euclidean distance enables the utilization of global information to better exploit the relations between samples. Additionally, the result of this method is not influenced by factors such as the sample distribution and geodesic distances.

The following uses the same UCI dataset and discrimination criteria as in section III.A.3 to conduct experiments for the purpose of comparing the performance of the EK-means algorithm with those of the K-means and K-mean++ algorithms. Tab. 5-7 are based on the performance comparisons between various algorithms on different datasets.

TABLE 5
COMPARISON OF ALGORITHMS ON THE DRUG REVIEW DATASET

Validation Measure	K-means	K-mean++	EK-means
AC	0.715	0.741	**0.790**
PE	0.662	0.681	**0.694**
RE	0.794	0.815	**0.820**
Iterations	3.75	3.94	**4.08**

TABLE 6.
COMPARISON OF ALGORITHMS ON THE DIABETES 130-US HOSPITALS FOR YEARS 1999-2008 DATASET

Validation Measure	K-means	K-mean++	EK-means
AC	0.802	0.851	**0.880**
PE	0.751	0.774	**0.794**
RE	0.821	0.858	**0.872**
Iterations	4.19	4.52	**4.96**

TABLE 7
COMPARISON OF ALGORITHMS ON THE MUSHROOM DATASET

Validation Measure	K-means	K-mean++	EK-means
AC	0.712	0.715	**0.728**
PE	0.786	0.806	**0.821**
RE	0.801	0.826	**0.849**
Iterations	3.64	3.81	**3.67**

Based on the data in Tab. 5-7, EK-means with a novel distance measurement achieves the best performance on the Drug Review Dataset, Diabetes 130-US hospitals for years 1999-2008 and Mushroom datasets. Its performance is better than that of K-mean++. The experimental results demonstrate the effectiveness of the method, which achieves higher accuracy than other algorithms with fewer iterations.

C. Design of an Efficient Parallel Algorithm for K-means

1. Method

The K selection sorting algorithm is a heap sort algorithm that selects the first K elements of a set to establish a binary tree. When a new element is added, it is compared with the parent node of the binary tree; if the new element is greater, the parent node is replaced, and the tree is readjusted until all the elements are processed.

The most common sampling method is line-by-line scan sampling, which can retain the original data format by traversing the global data. When a small number of data samples are taken, this method is simple and feasible, and the data samples can be obtained quickly. However, as the number of samples increases, the method increases the power consumption of the system, and the running time increases linearly; this is not suitable for sampling and computations with big data.

With regard to the defects of progressive scan sampling, based on the MapReduce framework, this paper uses the K selection sorting algorithm for parallel random sampling. The specific process is as follows:

Input: sample size K, random number range X(0<X<K), Reduce number R_n.

Output: K data samples.

Step 1: In the Map stage, randomly generate an integer from 0 to X for each data point as its key; the data content is the value, forming a <key, value> output.

Step 2: Merge the randomly generated key data to form <key, list <value1,value2,…,valuen>> and carry out internal sorting according to the keys.

Step 3: each Reduce stage outputs the first K/R*n* data.

The core code of the algorithm is as follows:

```
Map stage:
Random rd =new Random();
int tmp=rd.nextInt (X);
Context.write (new IntWritable (key), new Text (value.toString));
Reduce phase:
for (Text value: value){
i=0;
if <K/R in){
// Internal ranking
Context.write (null, new Text (val.toString));
i++;}
}
```

In the algorithm proposed in this paper, we first determine the formula according to the sample size, use the K selection sorting algorithm to sample randomly on the basis of the MapReduce framework, and save the collected sample data to a sample file. Then, we select the initial clustering values from the sample files through the sample preprocessing strategy to carry out pre-clustering. Then, in the iterative

process, we start MapReduce and perform the MapReduce task once per iteration, and we determine the new initial value by using the iterative replacement method. The perturbation of the clustering results of a single point is reduced by mean iteration. When the initial clustering value meets the set deviation threshold, the iterative process ends, and the clustering results are saved to the final clustering file.

IV. Results and discussion

A. Experimental data

This paper evaluates the performance of the improved K-means algorithm on the UCI dataset and Jingdong dataset. The UCI dataset contains US census data from 1990. It has 68 dimensions and 2 458 285 samples. Please see (UCI, 2010) for more details. Additionally, this paper uses 150 million user data points between 1 Jan 2014 and 31 Jun 2020 from JoyBuy. The reasons for the selection of these two datasets are as follows. First, it is highly convenient to evaluate the performances of algorithms using data from two different areas with different structures that were captured at different times and in different countries. Second, compared to the UCI dataset, JoyBuy data have larger values and higher time dependencies. Moreover, the structure of JoyBuy data is more conducive for clustering compared to the UCI dataset. Therefore, this paper performs clustering on two datasets to evaluate the performance of the improved K-means algorithm.

For the Jingdong Mall user dataset, because of the massive amount of consumption data, high level of data privacy, and company reasons for keeping information confidential, the 150000000 pieces of data selected contain basic consumption data for a given user, such as the customer number, the purchase amount (actual payment after discount), the number of purchases in the time period, the number of days from the last purchase date to the end of the data collection period, and other variables.

It is necessary to check the quality of the data before modeling. First, we check whether there are any missing data. After checking the set, we find that there are 49,372 missing values. There are methods, such as average interpolation and linear interpolation, for dealing with missing values. Because the amount of data in this case is too large, we delete the missing values directly. Then, we check all abnormal values. In terms of sales, due to the large volume of data and the large differences in purchase amounts in actual situations, we ignore any discrepancies; in terms of the number of purchases, considering the actual situation, we delete data points where the number of purchases is listed as more than 400; in terms of time intervals, in view of the actual situation, we delete data larger than 400. After processing, there are no missing values, unreasonable data, or impossible abnormal values. Upon completing the data processing step, we carry out the subsequent analysis.

1. Descriptive analysis

After completing the preprocessing of the data, we perform data integration. We select several fields involving customer consumption from the database, including customer number, sales volume, cumulative purchase number, time from last purchase to the end of the period, etc. During the course of the analysis, we find that sales are highly correlated with purchase times, so we calculate the average purchase amount, that is, the total sales of a customer divided by the number of cumulative purchases; thus, the fields used in the model are actually number of customers, number of cumulative purchases, average purchase amount, and purchase interval. To remove the sensitivity of business data, we recode the customer numbers and change their purchases in equal quantities. After the data reduction process is completed, to obtain a general understanding of the data in terms of several of its fields, a descriptive analysis of the number of cumulative purchases, the average purchase amount, and the purchase time interval is carried out, as shown in Table 8:

TABLE 8
DESCRIPTIVE ANALYSIS

	N	Max	Min	Mean	Std	Median	Skewness	Kurtosis
Cumulative purchases	150000000	109	1	31.08	23.21	25	0.79	-0.4
Average consumption	150000000	536	1	153.35	113.48	126	0.75	-0.44
Time interval	150000000	300	10	106.82	86.27	75	0.76	-0.75
Valid N	150000000							

The general data characteristics can be seen from the above statistics. In terms of the numbers of cumulative purchase by customers, the difference between the maximum and the minimum values is 100, indicating that customer loyalty is polarized. The average value is not different from the median, and the absolute skewness does not exceed one, indicating that the data are approximately left and right symmetric. A similar situation occurs with regard to the average purchase amount, for which the data differ greatly and form a left-right symmetric structure. The same situation is also present in the time interval data. The kurtosis of the data for each of the three fields is close to a normal distribution.

B. Experimental settings

In this experiment, considering that the initialized clustering center is different each time an algorithm is run, we repeat each algorithm 50 times and then take the average value of the 50 runs as the final clustering result. To perform parallel computing, the experiment is conducted on 15 PCs, one of which is responsible for resource scheduling and allocation as a master node; the remaining 14 PCs are slave nodes. The 15 computers used for running the tasks have the same configurations: 1 4-core Pentium (R) Dual-Core E6600 CPU with a main frequency of 2.9 GHZ, 4G of memory, a 500G hard drive, , the Ubuntu 14.04 LTS operating system, JDK of 1.7.0, and Hadoop 2.2.0 for cluster building.

C. Evaluation indicators

Two kinds of criteria are used to evaluate the data division accuracy from the perspectives of unsupervised and supervised learning. For data with missing class information, the improved DBI index is used to evaluate the clustering effects of the algorithms.

(1) The DBI (Davies-Bouldin index) is the Davies-Bouldin clustering validity measurement function (Davies & Bouldin, 1979). Since the DBI is independent of the setting of the initial class number K, it can be used to evaluate the validity of the data partitions. For a rough clustering algorithm, the samples in the upper and lower approximation classes have different effects in terms of cohesion; therefore, (Mitra et al., 2005) improved the DBI function and proposed the RDB evaluation index. The smaller the RDB index is, the better the clustering effect.

$$RDB = \frac{1}{K}\sum_{k=1}^{K}\max_{l\neq k}\left\{\frac{S_r(C_k)+S_r(C_l)}{d(C_b,C_l)}\right\} \qquad (13)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3044069, IEEE Access

where $S_r(C_k)$ is the weighted mean square error of the approximation of the kth class and the sample/class center in the boundary region to characterize the degree of cohesion of the class. For data with class information, the classification accuracy can be used to evaluate the clustering accuracy in addition to the unsupervised clustering evaluation index.

(2) The accuracy indicator Rand evaluates the accuracy of clustering results for data partitioning using class labels of the known samples. For datasets with N samples:

$$\text{Rand} = \left(\sum_{k=1}^{K} |R_k| / N \right) \cdot 100\% \qquad (14)$$

where R_k is a sample set that is correctly partitioned with regard to class k. For any subset in the clustering (or coverage) results, if class k contains the largest number of samples, it is said that the set represents the distribution of class k data.

In addition, learning from some mature clustering evaluation methods, in this study, we use normalized mutual information (NMI) (Strehl & Ghosh, 2002), accuracy (ACC) (Peng et al., 2016), and other indicators.

D. Discussion of the results of the improved K-means algorithm

In the experiments in this section, we compare the performance of the proposed improved K-means algorithm with those of six other clustering algorithms. Tab. 9 reports the scores of the compared algorithms on the UCI dataset, while Tab. 10 reports the scores of the compared algorithms on the JoyBuy dataset. The compared algorithms include traditional K-means, KCC (Liu et al., 2017), SEC (Dong et al., 2017), LWGP (Junyuan Xie et al., 2015), DEC (Wu et al., 2014) and IDEC (Guo et al., 2017). Among the compared algorithms, KCC, SEC and LWGP are integrative clustering algorithms, and DEC and IDEC are deep clustering algorithms. The parameters of each algorithm are set as follows. For SEC, the connectivity matrix is constructed with the parameter "nearest_neighbors"; the kernel function parameter, gamma, is set to the polynomial kernel "poly", and n_neighbors is set to 10 by default. For KCC, the maximum number of iterations is set to 100, and the convergence tolerance of the objective function is set to 10^{-5}. For LWGP, the distance between two samples is calculated by "Euclidean", and the criterion to merge the sample points is set to "ward", i.e., each cluster itself is a set. For DEC, the number of initial cluster centroids is set to be the same as that in K-means. This optimal number of clusters is selected as the one that minimizes the BIC. For IDEC, the clustering loss γ is set to 0.1. The initial learning rate is λ=0.001, and the convergence threshold is set to δ=0.1%. Each algorithm is run 500 times, and each run is evaluated using the four measures mentioned above. The 500 evaluations are then averaged to obtain the final scores of the algorithms. The results are shown in Tab. 9 and Tab. 10.

As shown in Tab. 9 and Tab. 10, the improved K-means algorithm proposed in this paper achieves the highest scores among the seven different clustering algorithms for all four evaluation criteria. It is worth mentioning that in this experiment, for each dataset, the hyperparameters of the DEC algorithm are adjusted manually to their optimal values, while the improved K-means algorithm proposed in this paper does not require dataset-specific tuning. Even so, the improved algorithm can achieve a score that is equal to or higher than that of DEC (with optimal hyperparameters) on the selected real dataset. Furthermore, comparisons using the other discriminative criteria also demonstrate the significant advantages of the improved K-means algorithm proposed in this paper on the real dataset.

Additionally, we provide the average score and average rank of each algorithm in the last two rows of the two tables. The average score represents the average of the RDB/Rand/NMI/ACC scores of each algorithm; the average rank represents the average rank of each algorithm across multiple criteria. As shown in Tab. 9 and Tab. 10, the proposed improved K-means algorithm obtains the highest score among the seven different clustering algorithms for all four discriminative criteria; its average rank is 1, which is higher than the average rank of DEC (with optimal hyperparameters) and higher than the average ranks of the other three clustering algorithms, thereby showing the significant advantages of the improved K-means algorithm proposed in this paper on the real dataset.

TABLE 9
COMPARISON OF THE SCORES OF DIFFERENT CLUSTERING ALGORITHMS ON THE UCI DATASET
(HIGHEST SCORE IS IN BOLD)

	K-means	SEC	IDEC	KCC	LWGP	DEC	Improved K-means
RDB	42.48±1.22	43.82±2.18	48.38±3.12	52.81±2.15	56.12±4.15	68.16±2.18	**88.26±5.49**
Rand	40.61±2.64	42.52±3.15	44.81±5.18	49.31±2.05	55.47±2.78	66.35±2.57	**72.64±2.16**
NMI	44.85±3.49	48.62±1.64	55.16±4.48	58.35±5.16	64.52±2.65	74.35±5.64	**81.65±2.34**
ACC	50.63±4.52	52.71±2.46	61.35±1.58	66.24±5.15	71.28±5.23	80.56±5.78	**88.65±5.82**
Avg. score	24.62	26.78	34.05	42.46	48.35	55.84	**59.34**
Avg. rank	7.04	5.02	4.52	4.20	2.99	1.82	**1.26**

TABLE 10
COMPARISON OF THE SCORES OF DIFFERENT CLUSTERING ALGORITHMS ON THE JINGDONG MALL DATASET
(HIGHEST SCORE IN BOLD)

	K-means	SEC	IDEC	KCC	LWGP	DEC	Improved K-means
RDB	40.56±2.18	48.82±4.89	58.67±6.76	57.59±3.14	74.68±1.18	79.18±2.41	**91.47±6.17**
Rand	39.18±3.41	47.22±6.42	57.63±7.18	66.86±4.18	70.18±0.52	75.61±4.18	**82.81±2.16**
NMI	47.61±4.18	52.47±3.18	64.17±3.46	74.26±5.16	86.17±1.15	84.67±3.27	**94.38±5.15**
ACC	56.28±5.16	67.16±1.27	75.34±4.18	78.67±4.25	84.18±2.14	88.67±4.26	**92.76±4.49**
Avg. score	22.18	24.51	38.49	44.71	47.58	49.52	**51.49**
Avg. rank	6.28	4.40	5.52	5.00	2.94	1.91	**1.00**

E. Efficient MapReduce Parallel Computing Results and Discussion

1. Data sampling test results

For the same dataset, we test the efficiency of the two text sampling methods to test the differences in sampling times between progressive scan sampling and parallel sampling when the number of samples changes. The test results are shown in Fig. 5 and Tab. 11.

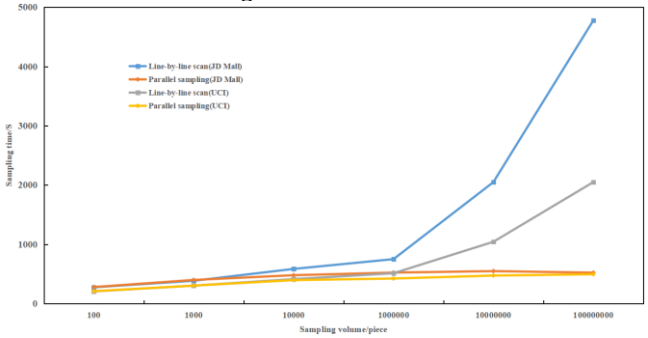


Figure 5 Data Sampling Comparison

TABLE 11
DATA SAMPLING TIME COMPARISON (TIME: S)

Number of samples	Line-by-line scan (JD Mall)	Parallel sampling (JD Mall)	Line-by-line scan (UCI)	Parallel sampling (UCI)
100	274	278	205	208
1000	384	398	302	302
10000	584	478	412	395
1000000	749	522	511	422
10000000	2049	547	1042	472
100000000	4777	521	2049	494

It can be seen from Figure 5 that when the sample size is small, progressive scanning sampling has the highest efficiency, but its efficiency decreases as the sample size increases, while the time required for parallel sampling tends to be stable. Therefore, the parallel sampling method used in this paper is more suitable than progressive sampling for big data environments.

2. Parallel computation performance test

This experiment is carried out on 15 PCs, one of which is responsible for the scheduling and allocation of resources as the master node; the other 14 PCs are slave nodes that participate in the calculations in sequence. A comparison of the efficiency and acceleration ratios obtained with different numbers of clusters for the data with a fixed number of samples are shown in Tab. 12.

TABLE 12
EXPERIMENTAL DATA OF PARALLEL COMPUTATION

Dataset size/GB	Number of data rows	Data dimension	Number of categories
A(JD)-0.74	47387236	3	4
B (JD)-1.07	86164783	3	4
C (JD)-1.94	143843877	3	4
A (UCI)-0.11	526842	68	52
B (UCI)-0.25	1062845	68	52
C (UCI)-0.58	2458285	68	52

A comparison of the times required to run the algorithm on different large-scale datasets as number of clusters increases is performed, and the results are detailed in Tab. 13 and Fig. 6.

TABLE 13
COMPARISON OF RUNNING TIMES WITH DIFFERENT NUMBERS OF NODES (TIME: S)

Number of nodes	A (JD)	B (JD)	C (JD)	A (UCI)	B (UCI)	C (UCI)
1	101	112	131	68	75	81
2	91	101	104	55	60	65
3	84	91	97	50	51	58
4	76	88	82	31	35	44
5	70	72	65	26	30	38

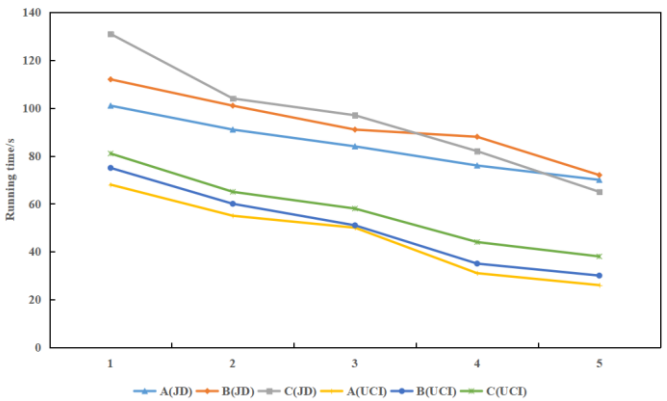


Figure 6 Comparison of the running times of the algorithm with different numbers of nodes

It can be seen from Figure 6 that when processing these three groups of big datasets, the computational efficiency of the algorithm is improved as the number of nodes increases; furthermore, the convergence time is obviously reduced, indicating that the algorithm in this paper is suitable for clustering big data.

The acceleration ratio is the ratio of running time of the single system and to that of the parallel system when processing the same task, and it is used to measure the expansibility and parallelization effects in parallel computing. As the number of nodes increases, the changes in the acceleration ratios of different large datasets are compared, and the comparison results for the selected Jingdong Mall dataset are shown in Figure 7.

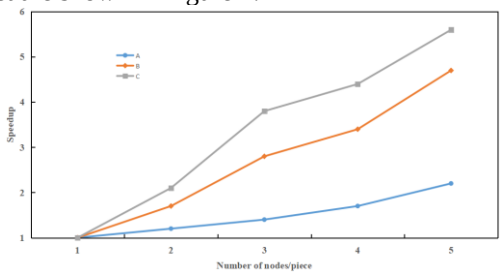


Figure 7 Comparison of Acceleration Ratios

Figure 7 shows that the acceleration ratio for each dataset increases as the number of computational nodes increases. When the dataset is large, increasing the number of computational nodes yield significant improvements in the parallel execution process. As a result, using the Hadoop distributed computing platform can effectively improve the efficiency of the clustering algorithm.

3. Parallel computing tuning

The number of data blocks allocated in the Hadoop cluster (the default size is 64 M) determines the number of map concurrencies. Therefore, the sizes of different data blocks and the number of map concurrencies affect the efficiency of the algorithm. In this experiment, we modify the dfs.block.size setting of hdfs-site.xml to adjust the data block sizes. For the three groups of datasets, we set different sizes of data blocks and adjust the number of map concurrencies. The experimental data in Table 9 are used, the specific allocation is shown in Table 14, and the running times of the algorithm are shown in Table 15.

TABLE 14
DATA BLOCK ALLOCATION

Dataset size/GB	64 M	128 M	256 M	512 M	1024 M
A (JD)-0.74	10	8	7	7	6
B (JD)-1.07	26	22	18	15	11
C (JD)-1.94	38	31	27	22	17
A (UCI)-0.11	5	4	3	2	1
B (UCI)-0.25	9	8	6	4	2
C (UCI)-0.58	15	10	7	5	3

TABLE 15
ALGORITHM RUNNING TIMES UNDER DIFFERENT NUMBERS OF MAP CONCURRENCIES (TIME: S)

Number of concurrencies/M	A (JD)	B (JD)	C (JD)	A (UCI)	B (UCI)	C (UCI)
64 M	101	112	131	68	75	81
128 M	91	101	104	55	60	65
256 M	84	91	97	50	51	58
512 M	76	88	82	31	35	44
1024 M	70	72	65	26	30	38

As seen in Tab. 15, as the number of data blocks increases, the number of map concurrencies decreases, but the running time of the algorithm does not decrease linearly as the number of map concurrencies decreases. Conversely, each dataset has its most suitable number of map concurrencies. Since the experimental configuration uses a 4-core CPU, the parallel

computation of 4 threads is carried out; an excessive number of map concurrencies increases the number of map tasks allocated by each CPU core, and when each CPU core runs the allocated map task, the addressing time also increases, increasing the overhead of the system. As the number of map concurrencies decreases, this overhead also decreases. However, as the number of map concurrencies for the three sets of data decreases to 2, the running time increases. This is because each computer performs double-threaded or single-threaded computing, there is an idle computing resource, and the amount of computations per CPU core increases, so the system cannot fully reflect the advantages of cluster computing. From the above experiment, we can draw a conclusion, namely, when the number of map concurrencies is close to the number of CPU cores, the efficiency of the algorithm can be greatly improved.

V. Conclusions

The K-means algorithm is one of the most commonly used tools for data mining and other information processing applications concerning clustering calculations. Compared with other clustering methods, it has the advantages of simplicity and high efficiency, but there is also some room for improvement and enhancement. Aiming at the shortcomings of the K-means algorithm, according to our previous discussion on the K-means algorithm, this paper first proposed an improved algorithm for selecting the appropriate initial clustering centers, determining the correct number of clusters and identifying complex data; then, considering the global structural information of data, this paper proposed a new global measurement method — an effective distance measure. Finally, this paper proposed an efficient parallel K-means algorithm based on MapReduce, where the efficiency of the algorithm can be improved by adjusting the Hadoop cluster. The results show that:

(1) The improved K-means algorithm proposed in this paper obtained the highest score and has good convergence and accuracy based on an experiment involving seven different clustering algorithms and four different evaluation criteria for the real JD Mall dataset.

(2) In the clustering performance test based on adjusting the number of cluster nodes and calculating the acceleration ratio, the algorithm proved to be suitable for the analysis and processing of big data.

(3) In the cluster tuning experiment based on adjusting the number of map concurrencies and cluster memory, it was determined the performance of this algorithm for big data processing was further improved by using the optimal settings.

Data Availability statement

The data that support the findings of this study are available in the supplementary material of this article.

Conflicts of interest statement

We declare that we have no financial and personal relationships with other people or organizations that could inappropriately influence our work, and there is no professional or other personal interest of any nature or kind in any product.

Author Contributions

Conceptualization: Yang Liu, Shuai Feng Ma, Xinxin Du.
Data curation: Shuai Feng Ma, Xinxin Du.
Formal analysis: Yang Liu.
Funding acquisition: Yang Liu.
Methodology: Yang Liu, Shuai Feng Ma.
Software: Yang Liu, Xinxin Du.
Supervision: Yang Liu.
Visualization: Shuai Feng Ma, Xinxin Du.
Writing—original draft: Yang Liu.
Writing—review & editing: Shuai Feng Ma.

VI. References

A, S. C., & B, S. Das. (2017). k Means clustering with a new divergence-based distance measure: Convergence and performance analysis. *Pattern Recognition Letters*, *100*, 67–73.

Ailon, N., Jaiswal, R., & Monteleoni, C. (2009). Streaming k-means approximation. *International Conference on Neural Information Processing Systems*.

Albanese, A., Pal, S. K., & Petrosino, A. (2011). *A Rough Set Approach to Spatio-temporal Outlier Detection*. Springer Berlin Heidelberg.

Arthur, D., & Vassilvitskii, S. (2007). K-Means++: The Advantages of Careful Seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*.

Breunig, M. M., Kriegel, H. P., Ng, R., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *Acm Sigmod Record*, *29*(2), 93–104.

Celebi, M. E., Kingravi, H. A., & Vela, P. A. (2013). A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications: An International Journal*, *40*.

Chakraborty, S., & Das, S. (2017). k means Clustering with a New divergence-based Distance measure: Convergence and Performance Analysis. *Pattern Recognition Letters*, S0167865517303380.

Chen, C. B., & Wang, L. Y. (2006). Rough Set-Based Clustering with Refinement Using Shannon's Entropy Theory. *Computers & Mathematics with Applications*, *52*(10–11), 1563–1576.

Chen, X., Hong, W., Nie, F., He, D., Yang, M., & Huang, J. Z. (2018). Spectral Clustering of Large-Scale Data by Directly Solving Normalized Cut. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1206–1215. https://doi.org/10.1145/3219819.3220039

Chen, X., Nie, F., Huang, J. Z., & Yang, M. (2017). Scalable Normalized Cut with Improved Spectral Rotation. *Twenty-Sixth International Joint Conference on Artificial Intelligence*.

Cohen-Addad, V. (2018). *Approximation Schemes for Capacitated Clustering in Doubling measures*.

D., Brockmann, D., & Helbing. (2013). The Hidden Geometry of Complex, Network-Driven Contagion Phenomena. *Science*.

Davies, D., & Bouldin, D. (1979). A Cluster Separation Measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions On, PAMI-1*, 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm. *Procedia Computer Science*, *54*, 764–771. https://doi.org/https://doi.org/10.1016/j.procs.2015.06.090

Dong, Huang, Chang-Dong, Wang, Jian-Huang, & Lai. (2017). Locally Weighted Ensemble Clustering. *IEEE Transactions on Cybernetics*.

Fan, Z., Sun, Y., & Luo, H. (2017). Clustering of College Students Based on Improved K-means Algorithm. *Journal of Computers (Taiwan)*, *28*(6), 195–203.

Frey, B. J., & Dueck, D. (2007). Clustering by Passing Messages Between Data Points. *Science*, *315*(5814), 972–976. https://doi.org/10.1126/science.1136800

Friggstad, Z., Khodamoradi, K., & Salavatipour, M. R. (2019). Exact Algorithms and Lower Bounds for Stable Instances of Euclidean *k*-means. *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2958–2972.

Gu, L. (2016). A novel locality sensitive k-means clustering algorithm based on subtractive clustering. *IEEE International Conference on Software Engineering & Service Science*.

Guo, X., Gao, L., Liu, X., & Yin, J. (2017). Improved Deep Embedded Clustering with Local Structure Preservation. *Ijcai*.

Guo, X., Zhu, E., Liu, X., & Yin, J. (2018). *Deep Embedded Clustering with Data Augmentation*.

Habib, S. T., & Zahid, A. (2018). An Analysis of MapReduce Efficiency in Document Clustering using Parallel K-Means Algorithm. *Future Computing & Informatics Journal*, S2314728817300661--.

He, Y., Tan, H., Luo, W., Mao, H., Ma, D., Feng, S., & Fan, J. (2012). MR-DBSCAN: An Efficient Parallel Density-Based Clustering Algorithm Using MapReduce. *IEEE International Conference on Parallel & Distributed Systems*.

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, *31*(8), 651–666.

Jianbin, H., Jianmei, K., & Amp, J. J. (2013). A hierarchical clustering method based on a dynamic synchronization model. *Entia Sinica(Informationis)*.

Kusuma, I., Ma'Sum, M. A., Habibie, N., Jatmiko, W., & Suhartanto, H. (2016). Design of intelligent k-means based on spark for big data clustering. *International Workshop on Big Data & Information Security*.

Lei, J., Jiang, T., Wu, K., Du, H., Zhu, G., & Wang, Z. (2016). Robust K-means algorithm with automatically splitting and merging clusters and its applications for surveillance data. *Multimedia Tools and Applications*, *75*(19), 12043–12059.

Likas, A., Vlassis, N., & Verbeek, J. J. (2003). The global k-means clustering algorithm. *Pattern Recognition*, *36*(2), 451–461.

Liu, H., Wu, J., Liu, T., Tao, D., & Fu, Y. (2017). Spectral Ensemble Clustering via Weighted K-Means: Theoretical and Practical Evidence. *IEEE Transactions on Knowledge & Data Engineering*, *29*(5), 1129–1143.

MacQueen, J. B. (1967). Some Methods for Classification and Analysis of MultiVariate Observations. *Proceedings of 5-Th Berkeley Symposium on Mathematical Statistics and Probability*, *1*, 281–297.

Mitra, S., Banka, H., & Pedrycz, W. (2005). Collaborative Rough Clustering. *Proceedings of the First International Conference on Pattern Recognition and Machine Intelligence*.

Parmar, D., Wu, T., & Blackhurst, J. (2007). *MMR: An algorithm for clustering categorical data using Rough Set Theory*. Elsevier Science Publishers B. V.

Patrick, J., F., Groenen, and, Krzysztof, & Jajuga. (2001). Fuzzy clustering with squared Minkowski distances. *Fuzzy Sets & Systems*.

Pawlak, Z. (1994). *Rough sets-theoretical aspects of reasoning about data*.

Pelleg, D., & Moore, A. (2002). X-means: Extending K-means with Efficient Estimation of the Number of Clusters. *Machine Learning*, *P*.

Peng, X., Xiao, S., Feng, J., Yau, W.-Y., & Yi, Z. (2016). Deep Subspace Clustering with Sparsity Prior. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 1925–1931.

Premkumar, M. S., & Ganesh, S. H. (2017). *A Median Based External Initial Centroid Selection Method for K-Means Clustering*. 143–146.

Qian, Y., Liang, J., & Dang, C. (2008). *Converse approximation and rule extraction from decision tables in rough set theory*. Pergamon Press, Inc.

Rammal, A., Perrin, E., Vrabie, V., Bertrand, I., & Chabbert, B. (2014). Optimal preprocessing and FCM clustering of MIR, NIR and combined MIR-NIR spectra for classification of maize roots. *E-Technologies and Networks for Development (ICeND), 2014 Third International Conference*.

Renjith, S., Sreekumar, A., & Jathavedan, M. (2020). Performance Evaluation of Clustering Algorithms for Varying Cardinality and Dimensionality of Data Sets. *Materials Today: Proceedings*, *27*, 627–633. https://doi.org/10.1016/j.matpr.2020.01.110

Rezaee, M. R., Lelieveldt, B. B. F., & Reiber, J. H. C. (1998). *A new cluster validity index for the fuzzy c-mean*. Elsevier Science Inc.

Rodriguez, A., & Laio, A. (2014). Machine learning. Clustering by fast search and find of density peaks. *Science (New York, N.Y.)*, *344*(6191), 1492–1496. https://doi.org/10.1126/science.1242072

Rodriguez, Alex, Laio, & Alessandro. (2014). Clustering by fast search and find of density peaks. *Science*.

Shi, J., & Luo, Z. (2010). Nonlinear dimensionality reduction of gene expression data for visualization and clustering analysis of cancer tissue samples. *Computers in Biology & Medicine*, *40*(8), 723–732.

Shi, L., Wang, W., Cai, W., Wang, Z., & Zhou, W. (2017). Mobility patterns analysis of Beijing residents based on call detail records. *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*.

Siddiqui, F. U., & Mat Isa, N. A. (2011). Enhanced moving K-means (EMKM) algorithm for image segmentation. *IEEE Transactions on Consumer Electronics*, *57*(2), 833–841.

Sridharan, K., & Sivakumar, P. (2018). A systematic review on techniques of feature selection and classification for text mining. *International Journal of Business Information Systems*, *28*(4), 504–518.

Stemmer, U. (2020). *Locally Private k -Means Clustering*.

Strehl, A., & Ghosh, J. (2002). Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, *3*(3), 583–617.

Tal, G. (2015). dendextend: an R package for visualizing, adjusting and comparing trees of hierarchical clustering. *Bioinformatics*, *22*, 3718–3720.

Tanir, D., & Nuriyeva, F. (2017). On selecting the Initial Cluster Centers in the K-means Algorithm. *2017 IEEE 11th International Conference on Application of Information and Communication Technologies (AICT)*.

Tleis, M., Callieris, R., & Roma, R. (2017). Segmenting the organic food market in Lebanon: an application of K-means cluster analysis. *British Food Journal*, *119*(7), 1423–1441.

Topchy, A. P., Law, M. H. C., Jain, A. K., & Fred, A. L. N. (2004). Analysis of Consensus Partition in Cluster Ensemble. *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK*.

UCI. (2010). *US Census Data (1990) Data Set*. http://archive.ics.uci.edu/ml/datasets/US+Census+Data+%281990%29

Visalakshi, N. K., & Suguna, J. (2009). K-means clustering using Max-min distance measure. *NAFIPS 2009 - 2009 Annual Meeting of the North American Fuzzy Information Processing Society*, 1–6. https://doi.org/10.1109/NAFIPS.2009.5156398

Wang, X., Jiao, Y., & Fei, S. (2015). Estimation of Clusters Number and Initial Centers of K-Means Algorithm Using Watershed Method. *International Symposium on Distributed Computing & Applications for Business Engineering & Science*.

Wu, J., Liu, H., Xiong, H., Cao, J., & Chen, J. (2014). K-Means-Based Consensus Clustering: A Unified View. *IEEE Transactions on Knowledge & Data Engineering*, *27*(1), 155–169.

Xian-Cai, G. (2008). Information quantity and rough entropy in ordered information systems based on dominance relations. *Computer Engineering and Design*.

Xiang, S., Nie, F., & Zhang, C. (2008). Learning a Mahalanobis distance measure for data clustering and classification. *Pattern Recognition*, *41*(12), 3600–3612.

Xiao-bin, Zhi, Jiu-lun, Fan, Feng, & Zhao. (2014). Robust local feature weighting hard c-means clustering algorithm. *Neurocomputing*.

Xie, Juanying, Jiang, S., Xie, W., & Gao, X. (2011). An Efficient Global K-means Clustering Algorithm. *Journal of Computers*, *6*(2), 271–279.

Xie, Junyuan, Girshick, R., & Farhadi, A. (2015). Unsupervised Deep Embedding for Clustering Analysis. *Computer Ence*.

Xin, Du, Ning, Xu, Cailan, Zhou, Shihui, & Xiao. (2017). *A Density-Based Method for Selection of the Initial Clustering Centers of K-means Algorithm*.

Xiong, C., Hua, Z., Lv, K., & Li, X. (2016). An Improved K-means Text Clustering Algorithm by Optimizing Initial Cluster Centers. *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*.

Xu, R., & Ii, D. C. W. (2005). Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, *16*(3), 645–678.

Yan, Xu, Xueliang, Fu, Honghui, Li, Gaifang, Dong, Qing, & Wang. (2018). *A K-means Algorithm Based On Feature Weighting*.

Yang, Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Proc Amia Annu Fall Symp*, *1*(1–2), 358–362.

First author, corresponding author: Liu Yang, male, born in 1991. He is now studying for a PhD in statistics at the School of Statistics, Southwestern University of Finance and Economics. Research content: macroeconomic statistical analysis, income distribution and sustainable development, HDI. A typical paper recently published: Yang Liu (2020): "Does urban spatial structure affect labour income? – research based on 97 cities in China", Economic Research-Ekonomska Istraživanja, DOI:10.1080/1331677X.2020.1798265

Second author: Shuaifeng Ma, male, 37 years old, achieved the Professional Doctorate in Business Management at Northwestern University. He is now at the Jingdong Century Trading Co., Ltd. Department of Consumer Commodity.

Third author: Xinxin Du, male, 34 years old, achieved the Professional Doctorate in Statistics at School of Statistics, Renmin University of China. He is now at the Jingdong Century Trading Co., Ltd. Department of Consumer Commodity.