Cluster Analysis by K-means and DBSCAN

HaoNan Chen
1024041103

***Abstract*—With the widespread application of data mining and machine learning techniques in various fields, clustering analysis, as an important unsupervised learning method, aims to discover the intrinsic structure and patterns in data. This paper focuses on the classic K-means algorithm. Through specific examples, gradually demonstrate the underlying principles and algorithm steps of K-means algorithm. Meanwhile, experiments were conducted on the same set of data using k-means++ and DBSCAN algorithms to analyze the advantages and disadvantages of different algorithms.**

***Index Terms*—K-means, K-means++, DBSCAN.**

I. Introduction

In today's digital age, data is growing explosively. Behind the vast amounts of data lies rich knowledge and patterns waiting to be mined and utilized. Cluster analysis, as a crucial unsupervised learning tool, can automatically group similar data objects together to form different clusters according to the characteristics of the data themselves, thus helping people understand the distribution patterns of data and playing an indispensable role in many fields such as customer segmentation, image recognition, bioinformatics, and text classification [1].

The K-means algorithm, as a classic algorithm in the field of cluster analysis, has attracted much attention since its inception. Due to its simple and intuitive principle, ease of implementation, and relatively low computational complexity, it shows certain advantages in large-scale data processing scenarios. Its core idea is to iteratively adjust the positions of K cluster centers and assign each sample in the dataset to the cluster corresponding to the nearest cluster center until the cluster centers no longer change significantly and reach a relatively stable clustering state.

However, in the actual application process, the K-means algorithm has gradually revealed some limitations. On the one hand, the algorithm is extremely sensitive to the selection of initial cluster centers. Different initial selections may lead to completely different clustering results and fall into local optimal solutions, making it difficult to guarantee the global optimal clustering effect. On the other hand, it is often not easy to determine the appropriate number of clusters K in advance. Without prior knowledge, the setting of the K value is rather blind, which directly affects the clustering quality. In addition, noise and outliers are often inevitable in real data, and these abnormal data will interfere with the calculation of cluster centers and then undermine the accuracy of clustering results [2].

In view of the wide application foundation of the K-means algorithm and the existing problems, many scholars have devoted a lot of efforts to researching and optimizing it. This paper aims to conduct a comprehensive and in-depth exploration of the K-means algorithm. It not only elaborates on its basic principles in detail but also discusses the key challenges it faces, comprehensively analyzes the existing optimization strategies, and proposes practical improvement methods through rigorous experimental comparisons, with the expectation of further improving the performance of the K-means algorithm, expanding its application scope, and providing strong support for meeting the clustering requirements in complex real-world problems.

II. Overview of the K-means Algorithm

The goal of K-means is to divide the dataset into K clusters, so that each data point belongs to the nearest cluster center. By repeatedly adjusting the position of the cluster center, K-means continuously optimizes the compactness within the cluster, thereby obtaining clusters that are as compact and separated from each other as possible [3].

A. Basic Principles

The core goal of the K-means algorithm is to divide a given dataset X into K non-overlapping subsets, that is. K clusters, in order to minimize the sum of squares error (SSE) of the samples within each cluster. During the algorithm initialization phase, K data points are randomly selected from the dataset as the initial clustering centers. Subsequently, the iterative process begins, and for each sample point in the dataset, the distance between it and each cluster center is calculated, usually using Euclidean distance:

$$d(x_i, \mu_j) = \sqrt{\sum_{m=1}^{d} (x_{im} - \mu_{jm})^2}$$

After all samples have been assigned, the cluster center of each cluster is recalculated, that is, the mean value of all sample points in the cluster in each dimension:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

The above steps of assigning samples and updating cluster centers are repeated until the cluster centers no longer change significantly or the preset maximum number of iterations is reached. At this time, the algorithm converges and the clustering process is completed.

B. Algorithm Flow

The K-Means clustering algorithm mainly consists of three steps:

1) Initialize cluster centers;
2) According to the nearest neighbor principle, divide each sample into clusters with the nearest cluster center;
3) Update the cluster center point of each cluster to the mean point of all sample points in each cluster center;

Repeat steps 2 and 3 until each cluster no longer changes.

C. Evaluation

1) Silhouette Coefficient: The silhouette coefficient is used to measure the clustering quality of each sample and then evaluate the entire clustering result. For sample xi, let a(xi) be the average distance from xi to other samples in the same cluster, representing cohesion [9]; let b(xi) be the minimum of the average distances from xi to all samples in other clusters, representing separation. Then the silhouette coefficient s(xi) of sample xi is defined as:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}}$$

The silhouette coefficient of the entire dataset is the average of the silhouette coefficients of all samples. The value range of the silhouette coefficient is [-1,1]. The closer the value is to 1, the higher the matching degree of the sample with its cluster and the better the separation from other clusters, indicating a better clustering effect.

2) Calinski-Harabasz Index: The essence of the Calinski Harabasz index is the ratio of inter cluster distance to intra cluster distance, and the overall calculation process is similar to the variance calculation method, so it is also called the variance ratio criterion [9].

The larger the index, the better the clustering effect. A larger value indicates better clustering performance, with strong separation between clusters and more concentrated sample points within clusters. A smaller value indicates poor clustering performance, possibly due to overlapping between clusters or scattered distribution of points within clusters.

$$s = \frac{tr(B_k)(N-K)}{tr(W_k)(K-1)}$$

Among them, Bk is the covariance matrix between classes, and Wk is the covariance matrix of data within classes.

D. Experimental Result

	V1	V2
1	2.072345	-3.241693
2	17.93671	15.78481
3	1.083576	7.319176
4	11.12067	14.40678
5	23.71155	2.557729
6	24.169929	32.02478
7	21.665779	4.892855
8	4.6936839	12.34217
9	19.21191	-1.121366
10	4.230391	-4.441536
11	9.1271300	23.60572
12	0.4075031	15.29705
13	7.3148460	3.3093120000000003
14	-3.438403	-12.02527
15	17.63935	-3.212345
16	4.415292	22.81555
17	11.94122	8.122487
18	0.7258532	1.806819
19	8.185273	28.1326
20	-5.773587	1.0248
21	18.76943	24.16946
22	7.752016	-3.334991
23	10.61132	28.44378
24	2.02165	-4.687122
25	5.145525	11.186
26	6.24331	19.47716
27	-11.70671	8.073185

Fig. 1. Data

First convert the above data into a two-dimensional array, with the first column V1 as the x-coordinate and the second column V2 as the y-coordinate, so that 3000 pairs of data can be regarded as points on the plane. Plot these points :

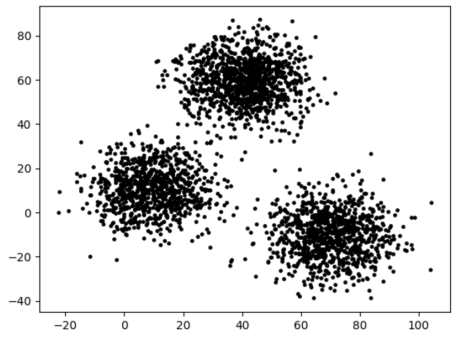


Fig. 2. Initialized Data

Determine the number of cluster K, and set K=3 here, so that the samples in the cluster are distributed together as closely as possible and the distance between clusters is as large as possible:

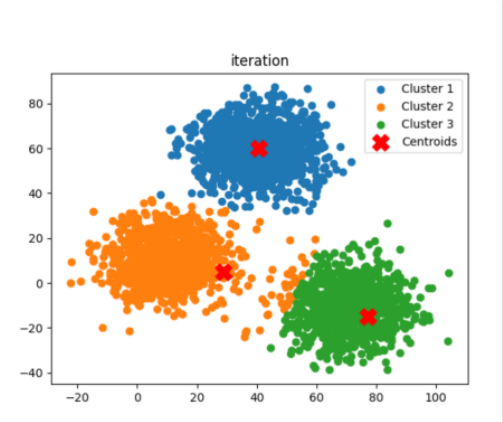


Fig. 3. The first iteration

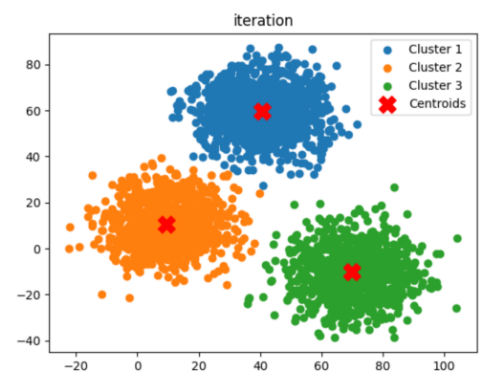


Fig. 4. The last iteration

When k=3, the corresponding Silhouette coefficient and Calinski-Harabasz index are:

Silhouette coefficient: 0.696

Calinski-Harabasz index: 10826.601

If set K incorrectly at the beginning, such as 2, 4, etc., Different k values will result in different clustering effects as shown in the following example:

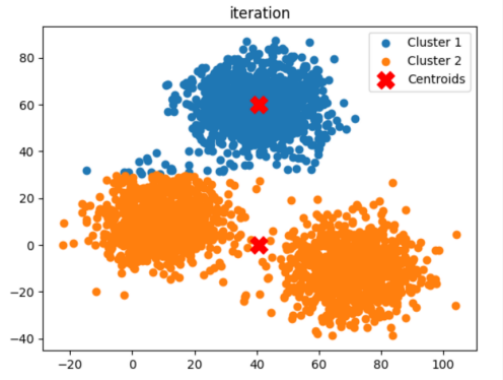


Fig. 5. K = 2

When k=2, the corresponding Silhouette coefficient and Calinski-Harabasz index are:

Silhouette coefficient: 0.509

Calinski-Harabasz index: 3055.988

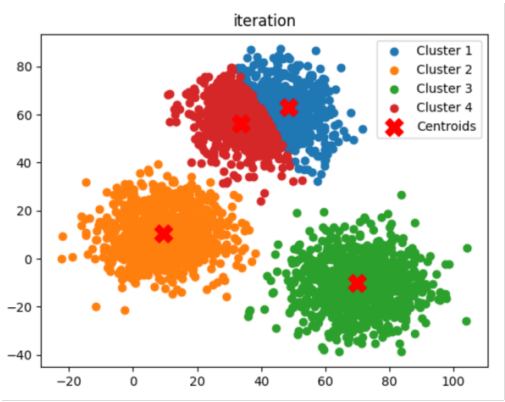


Fig. 6. K = 4

When k=4, the corresponding Silhouette coefficient and Calinski-Harabasz index are:

Silhouette coefficient: 0.536

Calinski-Harabasz index: 8355.659

By comparing different values of K, it was found that the iteration effect is best when K=3.

E. Advantages and Limitations of the Algorithm

The Algorithm's principle is simple and intuitive, easy to understand and implement, and even non-professionals can quickly master the basic usage and the computational complexity is relatively low.

However, it is highly sensitive to the selection of initial cluster centers. Different random initial values may cause the algorithm to converge to completely different local optimal solutions, and it is impossible to guarantee finding the globally optimal clustering results. At the same time, it has poor resistance to noise. Since the algorithm assigns samples based on distance measurement, noise points and outliers can easily change the positions of cluster centers, resulting in misclassification of normal samples and a reduction in clustering accuracy.Next, we will attempt to conduct experiments using different methods [4].

III. Comparison Algorithms

The optimized K-means algorithm is compared with the traditional K-means algorithm and DBSCAN algorithms.

A. K-Means++ algorithm

K-Means++ is an improved initialization algorithm for K-Means, aimed at solving the problem of slow convergence or falling into local optima caused by randomly initializing cluster centers in the original K-Means [4].

1) Algorithm steps:

- Randomly select the first center point: Randomly select a point from the dataset as the first cluster center.
- Calculate the distance from each point to the selected center point: For each point x in the dataset, calculate its distance D(x) to the nearest selected center point

- Select the next center point: Using D(x)2 as the probability distribution, select the next center point from the data:

$$P(x) = \frac{D(x)^2}{\sum_{x' \in X} D(x')^2}$$

- Repeat steps 2 and 3 until k initial center points are selected.
- Start standard K-Means clustering: Use these initial points as cluster centers and perform regular iterations (assign points to clusters and update center points).

2) Experimental result: We conclude from experimental comparison that, the K-means++ algorithm improves the overall performance of K-Means clustering by optimizing initialization and reducing the number of iterations. At the same time, it can select initial points more reasonably and avoid adverse partitioning caused by randomness [6].

The evaluation result of K-means++ algorithm is:

Silhouette Coefficient: 0.6945587736089913

Calinski Harabasz Coefficient: 10826.600579461161

Fig. 7. The first iteration

Fig. 8. The last iteration

B. DBSCAN algorithm

DBSCAN is a density based clustering algorithm that aims to discover clusters of arbitrary shapes and has robustness to noise points. It finds high-density regions in the data space, treats these regions as clusters, and classifies isolated points (low-density points) as noise [7].

1) Algorithm steps:

- Select any point p from the dataset and determine whether it is the core point (i.e., whether the neighborhood contains at least minPts points).
- If p is the core point, start a new cluster, add p and its neighboring points to the cluster, and continuously expand the neighborhood of the new core point.
- If a point is neither in any cluster nor meets the conditions to become a core point, it is marked as a noise point.
- Continue checking all unvisited points until all points have been accessed.

2) Experimental result: When eps=3 and min samples=4, the corresponding evaluation metric is:

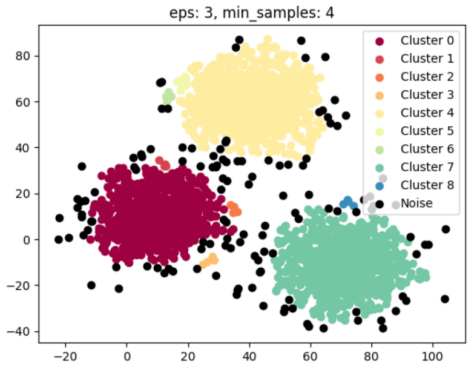
Silhouette Coefficient: 0.20878044305602825

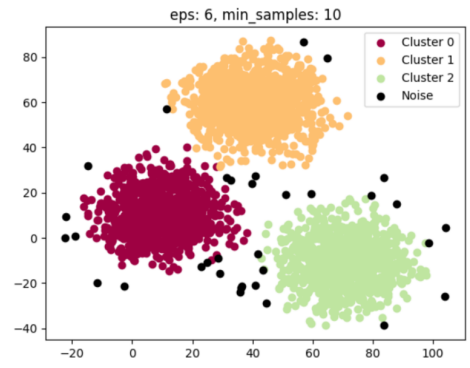
Calinski Harabasz Coefficient: 1937.7262134436978

When eps=6 and min samples=10, the corresponding evaluation metric is:

Silhouette Coefficient: 0.6587411547792584

Calinski Harabasz Coefficient: 6757.854275437878





The DBSCAN algorithm automatically discovers clustering clusters based on the density distribution of data by

setting parameters such as density thresholds, without the need for manually determining the number of clusters in advance, reducing the risk of poor clustering results due to improper selection of clustering numbers.

IV. Conclusion

With the rapid growth of data volume in various fields, clustering algorithms, as an unsupervised learning method, are crucial for mining the inherent structure of data and discovering potential patterns. The K-means algorithm is widely used for its simplicity and efficiency, but it also has limitations such as sensitivity to initial values and difficulty in determining the optimal number of clusters. The K-means++ algorithm aims to improve the initialization process of K-means and reduce its sensitivity to initial values; The DBSCAN algorithm is based on the concept of density connectivity and can automatically determine the number of clusters and effectively identify noise points, forming a complementary comparison with the K-means series algorithms. It is expected to reveal the advantages and applicable scenarios of each algorithm through experiments.

Based on the comprehensive experimental results, the K-means++ algorithm has shown significant effectiveness in overcoming the drawbacks of K-means initialization, improving clustering stability and accuracy, especially in scenarios where the approximate number of clusters is known and the data distribution is relatively regular; The DBSCAN algorithm has unique advantages in handling irregularly distributed and noisy data due to its automatic determination of cluster numbers and anti-noise properties. However, parameter tuning is complex, and the precision of partitioning tightly connected data clusters needs to be improved. In practical applications, algorithms should be carefully selected based on data characteristics and business needs, or considered for combined use, fully leveraging the strengths of each algorithm to achieve efficient and accurate data clustering analysis, and helping various fields extract valuable information from massive data.

References

[1] J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd ed., 2011.

[2] L. Rokach and O. Maimon, “Clustering methods,” Data mining and knowledge discovery handbook, pp. 321–352, 2005.

[3] F. Murtagh, “A survey of recent advances in hierarchical clustering algorithms,” The computer journal, vol. 26, no. 4, pp. 354–359, 1983.

[4] Feng T ,Tian F ,Dapeng M , et al.Unknown Protocol Identification Based on Improved K-Means++ Algorithm[J].Journal of Physics: Conference Series,2020,1646(1):012023-.

[5] A. M K .On the Consistency of k-means++ algorithm[J].Fundamenta Informaticae,2020,172(4):361-377.

[6] Ma Y ,Cheng W .Optimization and Parallelization of Fuzzy Clustering Algorithm Based on the Improved Kmeans++ Clustering[J].IOP Conference Series: Materials Science and Engineering,2020,768(7):072106.

[7] Chen S ,Liu X ,Ma J , et al.Parameter selection algorithm of DBSCAN based on K-means two classification algorithm[J].The Journal of Engineering,2019,2019(23):8676-8679.

[8] Yin L ,Hu H ,Li K , et al.Improvement of DBSCAN Algorithm Based onK -Dist Graph for Adaptive Determining Parameters[J].Electronics,2023,12(15):

[9] Lai H ,Huang T ,Lu B , et al.Silhouette coefficient-based weighting k-means algorithm[J].Neural Computing and Applications,2024,(prepublish):1-15.