Research and application of K-means algorithm

1st Zhixu Wang
College of Computer Science
Nanjing University of Posts and Telecommunications

I. INTRODUCTION

In the era of big data, the ability to store, process, and analyze massive datasets has become one of the central challenges across various domains. With the exponential growth in data volume and complexity, traditional data analysis techniques are often insufficient to extract meaningful insights. Data mining and machine learning techniques have emerged as powerful tools to uncover hidden patterns and trends within data. Among these techniques, clustering analysis, which is a form of unsupervised learning, plays a critical role in exploring datasets and grouping similar items together. It is widely applied in diverse fields, including image processing, customer segmentation, social network analysis, and medical diagnostics.

One of the most popular and well-established clustering algorithms is the K-means algorithm.[1] The K-means algorithm is favored for its simplicity, computational efficiency, and the ability to handle large datasets. The algorithm works by dividing a given dataset into K clusters based on the similarity between data points. Each data point is assigned to the nearest cluster center, and the center of each cluster is updated iteratively to minimize the within-cluster variance. Despite its straightforward approach, the K-means algorithm is not without its challenges. The choice of the number of clusters (K) often remains arbitrary and can significantly impact the clustering results. Additionally, the algorithm is sensitive to the initial placement of centroids and may converge to local minima, especially when the dataset contains noise or outliers.

Over the years, numerous research efforts have focused on improving and optimizing the K-means algorithm to address these limitations.[2] Variants of K-means, such as K-means++, have been proposed to enhance the initialization process and reduce sensitivity to initial centroid placements. Moreover, hybrid approaches combining K-means with other clustering or dimensionality reduction techniques have been explored to handle more complex and high-dimensional data. Despite its limitations, K-means continues to be a fundamental tool in data analysis, and its applications remain widespread.

This paper aims to provide an in-depth review of the K-means algorithm, exploring its fundamental principles, applications, and the various modifications that have been introduced to improve its performance. In particular, we will discuss the challenges associated with selecting the optimal K value and the influence of initial centroid placement.

The K-means algorithm divides a dataset into K clusters and performs clustering based on the similarity of data points within each cluster. Although the core concept of the K-means algorithm is straightforward, there are still many challenges in its practical application, such as the selection of the number of clusters (K), the impact of initial centroids, and its sensitivity to outliers. As a result, research on the K-means algorithm extends beyond its basic principles to include algorithm optimization, improvements, and combinations with other algorithms.

This paper will discuss the fundamental principles of the K-means algorithm, analyze its current applications and development trends, and explore some common improvement methods.

II. RELATED WORKS

The K-means algorithm, introduced by MacQueen in 1967[3], has long been a cornerstone of clustering techniques in machine learning and data mining. Over the years, a substantial body of research has been devoted to refining and enhancing the algorithm, addressing its limitations, and extending its applications.

This section reviews key research contributions and developments in the K-means algorithm, focusing on its theoretical advancements, practical modifications, and real-world applications.

A. Improvement and Optimization of K-means

One of the significant challenges in the K-means algorithm is the selection of the number of clusters (K), which directly impacts the results of clustering. Several studies have explored methods to address this issue:

1) Elbow Method and Silhouette Score

Early research focused on heuristic techniques such as the elbow method[4], where the sum of squared distances within clusters is plotted against different K values. This approach helps determine an optimal K, though it remains subjective. The silhouette score[5], proposed by Rousseeuw (1987), is another metric that evaluates the quality of clustering by measuring how similar an object is to its own cluster compared to other clusters.

These methods, while effective in some cases, often do not provide definitive solutions for selecting the ideal K in more complex datasets.

2) K-means++

K-To overcome the problem of poor initialization, Arthur and Vassilvitskii (2007) introduced the K-means++ algorithm[6]. K-means++ optimizes the initialization of centroids by selecting them in a way that maximizes their spread, improving the chances of finding a global minimum and speeding up convergence. This modification has become widely adopted due to its enhanced stability and performance.

B. Handling High-Dimensional Data

High-dimensional data, common in fields such as bioinformatics, text mining, and image processing, poses challenges for K-means due to the "curse of dimensionality." In high dimensions, distance measures like Euclidean distance become less effective in distinguishing between clusters.

1) Dimensionality Reduction Techniques

Research has focused on integrating K-means with dimensionality reduction methods such as Principal Component Analysis (PCA)[7] and t-SNE to alleviate the curse of dimensionality[8]. By reducing the number of features while preserving the variance of the data, these techniques can enhance K-means' ability to find meaningful clusters in high-dimensional spaces.

2) Subspace Clustering

Another approach is subspace clustering[9], where the K-means algorithm is applied in lower-dimensional subspaces, allowing for more effective clustering in high-dimensional datasets.

C. Handling Noise and Outliers

K-means is sensitive to outliers and noise, as these points can skew the positions of cluster centroids. Several works have addressed this issue by integrating K-means with other techniques:

1) K-means with Outlier Detection

Research has explored methods to incorporate outlier detection into the K-means framework, such as the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm[10]. DBSCAN identifies core points, border points, and noise, and is particularly effective when the data contains a lot of noise or irregularly shaped clusters.

2) Robust K-means

Researchers have proposed robust K-means algorithms, such as the L1-norm based K-means, which replace the Euclidean distance with the Manhattan distance (L1-norm) to make the algorithm more resilient to outliers[11].

D. Applications of K-means

K-means has found applications across a wide range of domains:

1) Image Compression and Segmentation

K-means is extensively used in image processing for tasks like color quantization[12], where it reduces the number of distinct colors in an image, and image segmentation, where it groups pixels into clusters to identify objects or regions of interest. Research has shown that K-means can significantly reduce the size of image data while retaining important features for analysis.

2) Customer Segmentation in Marketing

In marketing, K-means is applied to segment customers based on purchasing behavior, demographics, or online interactions[13]. This helps businesses tailor their marketing strategies to specific customer groups, improving customer satisfaction and targeting.

III. PROBLEM STATEMENT

In this project, the goal is to apply the K-means clustering algorithm to a dataset that contains multiple two-dimensional coordinates, in order to group these points into K distinct clusters based on their spatial proximity. Each coordinate represents a point in a two-dimensional space with two numerical values: the X and Y coordinates. The aim is to automatically divide these points into meaningful clusters, revealing underlying structures or patterns within the data.

The dataset is provided in the form of a CSV file, where each entry consists of two values representing the X and Y coordinates of a point. The K-means algorithm will iteratively assign all the data points to K clusters, with each cluster having a centroid (center). The assignment of points is based on their proximity to the centroids, and the centroids are updated by calculating the mean of all points within each cluster until they stabilize.

A. Primary Goals

1) Clustering Analysis

Apply the K-means algorithm to automatically assign two-dimensional coordinates to different clusters based on spatial proximity, in order to uncover the underlying structure of the data.

2) Visualization

Visualize the clustering results through graphical representations to better understand the distribution of the clusters and assess the quality of the clustering.

3) Cluster Quality Evaluation

Analyze and evaluate the clustering results for different values of K (the number of clusters), ensuring that the clustering is of high quality, with compact intra-cluster similarity and distinct inter-cluster separation.

B. Key Challenges

1) Selection of K

Determining the optimal number of clusters (K) is one of the most challenging aspects of the K-means algorithm. A suitable K value helps generate meaningful clusters, avoiding both excessively fine clusters and overly broad clusters. Traditional methods, such as the elbow method and silhouette score, can assist in selecting K, but these approaches can sometimes be subjective or not fully effective for complex datasets. Therefore, selecting the best K is a crucial aspect of this task.

2) Impact of Centroid Initialization

The K-means algorithm relies on the initial selection of centroids, and the choice of initial centroids can greatly affect the final clustering results. Poor initialization can lead the algorithm to converge to a local minimum, resulting in suboptimal clusters. To address this, the K-means++ initialization method is often used, which selects initial centroids that are as far apart as possible, increasing the likelihood of finding the global optimal solution.

3) Convergence and Stability

The K-means algorithm iteratively adjusts the cluster centers to optimize the clustering results. Ideally, the algorithm converges when the centroids stabilize, meaning that the points in each cluster do not change. However, in certain cases, especially with noisy data or poor centroid initialization, the algorithm may not converge to an ideal result. Ensuring the algorithm converges properly and that the final results are stable is a key challenge.

C. *Expected Outputs*

1) Cluster Assignment for Data Points

Based on the K-means algorithm, each two-dimensional coordinate will be assigned to a specific cluster, and the clusters should be clearly separated.

2) Visualization of Results

The clustering results will be visualized using scatter plots, cluster assignment diagrams, or other visualization techniques to provide a clear representation of the clusters and their distribution.

3) Clustering Evaluation Report

A report evaluating the quality of the clustering, including metrics such as intra-cluster variance, inter-cluster distance, and silhouette score, to assess how well the clustering process has grouped the data.

D. *Implementation Steps*

1) Data Preprocessing

Load the CSV file, check the data for completeness and correct formatting, and handle missing values or outliers.

2) means Algorithm Implementation

L-Apply the standard K-means algorithm to the data, using the K-means++ initialization method to select initial centroids.

3) Clustering Quality Evaluation

Use the elbow method, silhouette score, and other techniques to evaluate clustering results for different values of K, and select the optimal K.

4) Result Visualization

Visualize the clustering results using scatter plots, cluster assignment plots, etc., to interpret the clustering patterns.

5) Analysis and Conclusion

Analyze the clustering results, summarize findings, and suggest potential improvements or further analysis to better understand the data structure.

IV. ALGORITHMS

A. *K-means Algorithm*

The K-means algorithm is a widely used clustering method that partitions the data into K clusters by minimizing the intra-cluster variance. The basic idea is to assign data points to the nearest cluster center (centroid) and iteratively update the centroids until convergence.

1) Algorithm Steps

Initialization: Randomly select K data points as initial centroids.

Assignment Step: Assign each data point x_i to the nearest centroid c_k. Specifically, for each data point x_i, compute the Euclidean distance $d(x_i,c_k)$ to each centroid and assign it to the cluster with the closest centroid:

$$ci = arg\ \underset{k}{min}d(x_i, c_k)$$

$$d(x_i, c_k) = \sqrt{(x_{i1} - c_{k1})^2 + (x_{i2} - c_{k2})^2 + \cdots} ci$$

Update Step: Recalculate each centroid c_k by computing the mean of all data points in cluster C_k

$$c_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$$

Where C_k represents the set of data points assigned to cluster k and $|C_k|$ is the number of data points in C_k.

Repeat: Repeat the assignment and update steps until the centroids no longer change.

2) Convergence Condition

The algorithm converges when the centroids stop changing or the changes are smaller than a specified threshold. Alternatively, the algorithm can stop after a predefined maximum number of iterations.

3) Objective Function

K-means minimizes the Sum of Squared Errors (SSE)[14], which is the sum of the squared Euclidean distances between each data point and its assigned centroid:

$$SSE = \sum_{k=1}^{K} \sum_{x_i \in C_k} ||x_i - \mathrm{c}_k||^2$$

where $||x_i - \mathrm{c}_k||^2$ is the squared Euclidean distance between x_i data point and centroid c_k.

B. *K-means++ Algorithm*

K-means++ is an improved version of the K-means algorithm that optimizes the initialization of centroids. By choosing centroids more carefully, K-means++ helps avoid poor initializations that could lead to suboptimal clustering results and speeds up convergence.

1) Algorithm Steps

Select the first centroid: Randomly select one data point as the first centroid.

Select subsequent centroids: For each remaining data point x_i, compute its minimum distance $D(x_i)$ to any of the already selected centroids, and choose the next centroid with probability proportional to $D(x_i)^2$ Specifically:

$$P(x_i) = \frac{D(x_i)^2}{\sum_{x_j \in X} D(x_j)^2}$$

Repeat until K centroids are selected.

K-means Clustering: Once the centroids are initialized, perform the regular K-means algorithm.

2) Objective Function

The objective function for K-means++ is the same as K-means, minimizing the Sum of Squared Errors (SSE).

C. Elbow Method

The Elbow Method is a heuristic used to determine the optimal number of clusters K. It works by plotting the total Sum of Squared Errors (SSE) for different values of K and identifying the point where the rate of decrease in SSE sharply slows down (the "elbow").

1) Algorithm Steps:

Run K-means for a range of K values .

For each K, compute the SSE.

Plot the relationship between K and SSE. Typically, the SSE decreases as K increases.

Identify the "elbow" in the graph, where the rate of decrease in SSE slows down. The K at this point is considered the optimal number of clusters.

D. Silhouette Score

The Silhouette Score is a metric that measures the quality of clustering by evaluating both the tightness of clusters (intra-cluster similarity) and the separation between clusters (inter-cluster dissimilarity). The score ranges from -1 to 1, where a score closer to 1 indicates well-defined clusters.

For each data point x_i, the silhouette score $s(x_i)$ is defined as:

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

where:

$a(x_i)$ is the average distance from x_i to all other points in the same cluster:

$$a(x_i) = \frac{1}{|\,\mathrm{C}_{x_i}\,|\;1} \sum\nolimits_{x_j \in C_{x_i}} |\, x_i - \mathrm{x}_j \,|$$

$b(x_i)$ is the average distance from x_i to all points in the nearest neighboring cluster:

$$b(x_i) = \min_{C_k \neq C_{x_i}} \frac{1}{|C_k|} \sum\nolimits_{x_j \in C_k} |\, x_i - \mathrm{x}_j \,|$$

V. EXPERIMENT

A. Data Preparation

The dataset used in this experiment comes from the xclara.csv file, which contains several two-dimensional coordinates. The goal is to perform clustering on these points. The data was read into a NumPy array and passed to the clustering algorithm for processing.

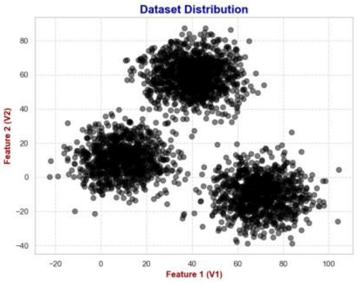

Fig. 1. Distribution of data

B. Experimental Setup

In this experiment, we compare the classic K-means clustering algorithm with the improved K-means++ algorithm to analyze their performance on a two-dimensional dataset.

K-means Clustering: The classic K-means clustering algorithm randomly selects initial centroids and iteratively optimizes the allocation of data points to the nearest centroid. The centroids are updated by computing the mean of the points assigned to each cluster until convergence, i.e., when the centroids no longer change.

K-means++ Clustering: To address the instability of results caused by random initialization of centroids in K-means, K-means++ uses a smarter initialization strategy to choose the initial centroids. This method significantly improves the clustering quality and reduces the dependence on the initial centroid selection.

C. Choosing the Number of Clusters

In this experiment, we tested different numbers of clusters (K values), ranging from 2 to 5, to observe the clustering effect. We compared both clustering algorithms to assess the quality of the results. To determine the optimal number of clusters, we used evaluation metrics such as inertia and silhouette score.

D. Evaluation Metrics

Inertia: Inertia is a commonly used metric in K-means clustering, representing the sum of squared distances from data points to their corresponding cluster centroids. A smaller inertia typically indicates better clustering. We calculated inertia for different K values and applied the Elbow Method to identify the optimal number of clusters.

Silhouette Score: The silhouette score is another important metric for evaluating clustering quality. It considers both the tightness within clusters and the separation between clusters. A higher silhouette score indicates better clustering. We calculated the silhouette score for each K value to further evaluate the performance of different clustering algorithms.

E. Experimental Process

In the experiment, we first performed clustering using both the classic K-means and K-means++ algorithms. Each run involved randomly initializing the centroids and iteratively updating the cluster labels and centroids until the algorithm converged. The maximum number of iterations was set to 100.

We then tested different K values (from 2 to 5) and visualized the clustering results. For each K value, we plotted the data points and indicated the corresponding centroids, allowing for a visual comparison of the clustering effects.

F. Experimental Results and Analysis

1) Inertia and the Elbow Method

After calculating the inertia for different K values, we plotted the inertia versus K. Using the Elbow Method, we observed that the inertia decreased gradually as K increased, but there was a noticeable "elbow" at K=3, indicating that 3 is a reasonable choice for the number of clusters.

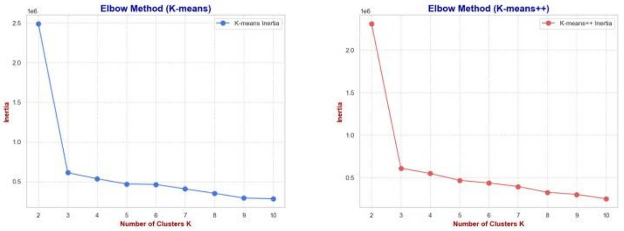

Fig. 2. The elbow method was used to analyze the data, and the clustering effect was better when k=3.

2) Silhouette Score Analysis

We calculated the silhouette scores for each K value and plotted the relationship between K and the silhouette score. A higher silhouette score indicates better separation between clusters and higher clustering quality. By comparing the silhouette scores for different K values, we confirmed that K=3 provides the best clustering result.

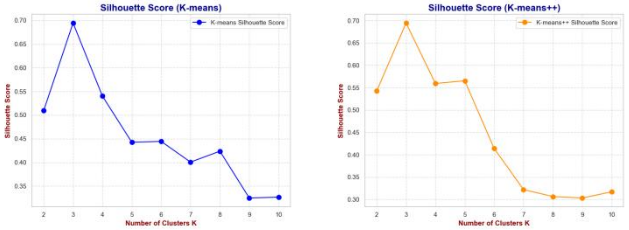

Fig. 3. Among the contour coefficients of the two methods, k=3 has the best effect

3) Clustering Effect Visualization

We visualized the clustering results for both K-means and K-means++ algorithms. Each cluster was assigned a different color, and the centroids were marked with yellow "X" symbols. The visual comparison showed that the K-means++ algorithm selected more balanced initial centroids, resulting in more stable clustering compared to K-means, which could sometimes produce unstable results due to random initialization.

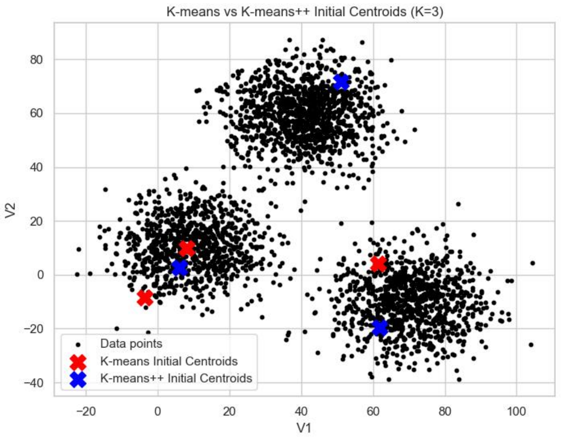

Fig. 4. As shown in the figure, the initial centroid selection at k=3 using the k-means method and the k-means++ method respectively, it can be observed that the initial centroid selected by the k-means++ method is closer to the final centroid position

G. Discussion of Results

By comparing the performance of different K values and algorithms, we found that K-means++ generally produced more stable and higher-quality clustering results. Specifically, at K=3, the K-means++ algorithm showed the best performance, with both the silhouette score and inertia indicating a favorable clustering result.

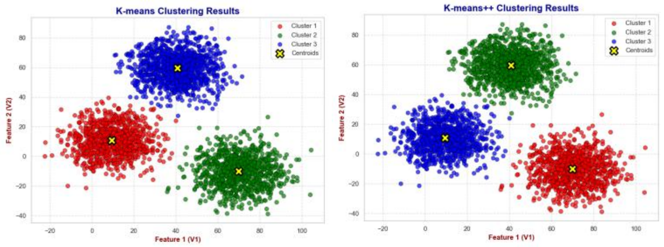

Fig. 5. Clustering end results.

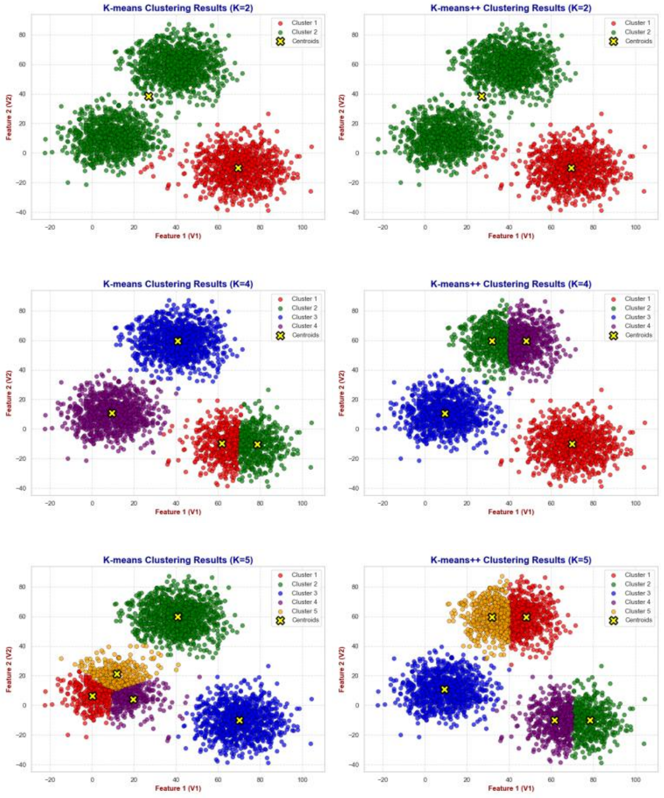

Fig. 6. Comparison of the clustering results of the k-means method and the k-means++ method when k=2, 4, 5.

However, in some cases, K-means also yielded acceptable results, especially when the data had simple distributions and clear differences between clusters. Overall, K-means++ demonstrated its advantage in handling more complex datasets and avoiding the issues caused by random centroid initialization in K-means.

VI. Conclusion

Through this experiment, we compared K-means and K-means++ algorithms' performance at different K values and evaluated the clustering results using inertia and silhouette scores. The results showed that K-means++ effectively addressed the issues of random centroid initialization in K-means, providing better clustering results in most cases. Our visual analysis further confirmed that K=3 was the optimal number of clusters for this dataset.

References

[1] Likas A, Vlassis N, Verbeek J J. The global k-means clustering algorithm[J]. Pattern recognition, 2003, 36(2): 451-461.

[2] Ahmed M, Seraj R, Islam S M S. The k-means algorithm: A comprehensive survey and performance evaluation[J]. Electronics, 2020, 9(8): 1295.

[3] Hamerly G, Elkan C. Learning the k in k-means[J]. Advances in neural information processing systems, 2003, 16.

[4] Syakur M A, Khotimah B K, Rochman E M S, et al. Integration k-means clustering method and elbow method for identification of the best customer profile cluster[C]//IOP conference series: materials science and engineering. IOP Publishing, 2018, 336: 012017.

[5] Shahapure K R, Nicholas C. Cluster quality analysis using silhouette score[C]//2020 IEEE 7th international conference on data science and advanced analytics (DSAA). IEEE, 2020: 747-748.

[6] Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding[R]. Stanford, 2006.

[7] Abdi H, Williams L J. Principal component analysis[J]. Wiley interdisciplinary reviews: computational statistics, 2010, 2(4): 433-459.

[8] Van der Maaten L, Hinton G. Visualizing data using t-SNE[J]. Journal of machine learning research, 2008, 9(11).

[9] Elhamifar E, Vidal R. Sparse subspace clustering: Algorithm, theory, and applications[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(11): 2765-2781.

[10] Schubert E, Sander J, Ester M, et al. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN[J]. ACM Transactions on Database Systems (TODS), 2017, 42(3): 1-21.

[11] Kwak N. Principal component analysis based on L1-norm maximization[J]. IEEE transactions on pattern analysis and machine intelligence, 2008, 30(9): 1672-1680.

[12] Celebi M E. Improving the performance of k-means for color quantization[J]. Image and Vision Computing, 2011, 29(4): 260-271.

[13] Kim K, Ahn H. A recommender system using GA K-means clustering in an online shopping market[J]. Expert systems with applications, 2008, 34(2): 1200-1209.

[14] Nainggolan R, Perangin-angin R, Simarmata E, et al. Improved the performance of the K-means cluster using the sum of squared error (SSE) optimized by using the Elbow method[C]//Journal of Physics: Conference Series. IOP Publishing, 2019, 1361(1): 012015.