Cluster Analysis-K-means

ZiHao Zhang
1024040915
1410352480@qq.com

***Abstract*—This paper explores the application of clustering analysis in modern data analysis, with a particular focus on the performance comparison between the k-means and k-means++ algorithms. As an unsupervised learning method, clustering analysis plays an important role in fields such as data mining, pattern recognition.The k-means algorithm achieves clustering by minimizing the distance between data points and cluster centers, but it is sensitive to the selection of initial centroids. The k-means++ algorithm improves the stability and accuracy of clustering results by enhancing the initial centroid selection process. The paper begins with an introduction to related research efforts in clustering analysis, including density-based clustering methods such as DBSCAN and HDBSCAN, and their comparison with the k-means algorithm. It then formally states the problem: conducting clustering analysis on the xclara.csv dataset using k-means and k-means++ algorithms. The dataset consists of 3000 data points, each with two features, V1 and V2. In the algorithms section, the paper de-scribes in detail the working principles of k-means and k-means++, as well as their differences in the selection of initial centroids. Then it introduces three metrics to evaluate the effectiveness of clustering: the Silhouette Score, the Davies-Bouldin Index, and the Calinski-Harabasz Index. Through experimentation, the paper demonstrates the variation of these metrics with different numbers of clusters and concludes that the best clustering effect is achieved when the number of clusters is set to 3. Furthermore, the paper compares the number of iterations between k-means and k-means++, demonstrating the efficiency advantage of k-means++. Finally, the paper summarizes the excellent performance of k-means++ in handling the dataset and emphasizes the importance of choosing the appropriate clustering method based on the characteristics of the data and the requirements of the clustering task.**

***Index Terms*—Cluster Analysis, Kmeans, and Data Mining.**

I. Introduction

In the era of big data, the ability to extract meaningful insights from vast amounts of information has become increasingly crucial. Clustering analysis, a fundamental technique in unsupervised learning, plays a pivotal role in this endeavor. By grouping similar data points together, clustering helps to uncover hidden patterns and structures within datasets, making it an indispensable tool in various domains such as data mining, pattern recognition, image analysis, and bioinformatics. The primary objective of clustering analysis is to partition a dataset into distinct clusters, ensuring that data points within the same cluster are highly similar, while those in different clusters are dissimilar.

Among the plethora of clustering algorithms, k-means [1] stands out as one of the most widely used and well-studied methods. k-means operates by iteratively assigning data points to the nearest cluster centroid and updating the centroids based on the mean of the assigned points. This process continues until the centroids stabilize, resulting in a partitioning of the data into k clusters. Despite its simplicity and efficiency, the k-means algorithm is not without its limitations. One of the most significant drawbacks is its sensitivity to the initial selection of centroids. Since the algorithm starts with randomly chosen centroids, it often converges to local optima, leading to suboptimal clustering results. This issue is particularly pronounced in datasets with complex structures or when the number of clusters is not well-defined.

However, the k-means algorithm is sensitive to the selection of initial centroids, often leading to the algorithm getting trapped in local optima. To address this issue, the k-means++ [2] algorithm has been improved upon the k-means by adopting a smarter initialization method, which enhances the stability and accurancy of the clustering results.

This paper performs clustering on the xclara.csv dataset available at [3] using the k-means++ algorithm. By running multiple experiments and calculating relevant evaluation metrics, such as the Silhouette Score, Davies-Bouldin Index, and Calinski-Harabasz Index, the appropriate number of clusters is determined. Additionally, by comparing the number of iterations with the K-means algorithm, the advantages of the K-means++ algorithm are illustrated.

The remainder of this paper is organized as follows: Section 2 provides an overview of related work in the field of clustering analysis, discussing various algorithms and their respective strengths and weaknesses. Section 3 describes the dataset used in our experiments and its distribution. Section 4 delves into the details of the k-means and k-means++ algorithms, explaining their underlying principles and differences. Section 5 presents the evaluation metrics used to assess the clustering results and discusses the findings of our experiments. Finally, Section 6 concludes the paper by summarizing the key insights and suggesting directions for future research.

Through this study, we aim to demonstrate the practical benefits of the k-means++ algorithm in clustering analysis, particularly in scenarios where the initial centroid selection is critical to achieving high-quality results. By leveraging the improved initialization strategy of K-means++, researchers and practitioners can enhance the accuracy and efficiency of their clustering tasks, ultimately leading to more robust and reliable data analysis outcomes.

II. Related Work

Clustering analysis is a cornerstone of unsupervised learning and has been extensively studied in the field of data mining. The goal of clustering is to group data points into clusters such that points within the same cluster are more similar to each other than to those in other clusters. Over the years, a variety of

clustering algorithms have been developed, each with its own strengths and weaknesses, making them suitable for different types of datasets and applications. In this section, we discuss some of the most prominent clustering algorithms, including k-means, k-means++, DBSCAN [4], HDBSCAN [5], and other notable methods.

A. DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that was proposed by Ester et al. in 1996. Unlike K-means, DBSCAN does not require the number of clusters to be specified in advance. Instead, it identifies clusters based on the density of data points, making it particularly suitable for datasets with noise and clusters of arbitrary shapes. The algorithm works by defining clusters as dense regions of data points separated by regions of lower density. It uses two key parameters: ϵ and minPts, which represent the radius of the neighborhood around a point and the minimum number of points required to form a dense region. DBSCAN is highly effective in identifying clusters of varying shapes and sizes, and it can also detect outliers as noise.

B. HDBSCAN

HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) is an extension of DBSCAN that addresses some of its limitations, particularly when dealing with clusters of varying densities. HDBSCAN builds on the ideas of DBSCAN but introduces a hierarchical approach to clustering. It constructs a hierarchy of clusters by representing the relationships between data points in a tree structure, allowing it to handle clusters with different densities more effectively. HDBSCAN calculates the core distance and relative density of data points to gradually build a hierarchical structure and then extracts clusters based on density thresholds. This approach makes HDBSCAN more robust and flexible, especially in complex datasets where clusters may have varying densities.

C. Other Clustering Methods

In addition to the aforementioned algorithms, there are several other clustering methods that have been developed to address specific challenges in data mining. For example, Mean Shift clustering is a non-parametric algorithm that identifies clusters by finding the modes of the data density function. It is particularly effective for datasets with irregular shapes and varying densities. Another notable method is Affinity Propagation, which identifies exemplars (representative points) within the data and forms clusters around them. This approach is useful in scenarios where the number of clusters is not known in advance.

In summary, the choice of clustering algorithm depends on the specific characteristics of the dataset and the goals of the analysis. By understanding the strengths and limitations of each algorithm, researchers and practitioners can select the most appropriate method for their clustering tasks, leading to more accurate and meaningful results. In this paper, we focus on the k-means++ algorithm and demonstrate its effectiveness in clustering the xclara.csv dataset, while also comparing its performance with the traditional k-means algorithm.

III. Data Distribution

In this paper, we conduct clustering analysis on the xclara.csv dataset using both the K-means and K-means++ algorithms. The dataset consists of 3000 data points, each with two features, labeled as V1 and V2. These features represent certain measured values, and the dataset is structured in a two-dimensional space, making it suitable for visualization and clustering analysis. The distribution of the dataset is shown in Fig.1.

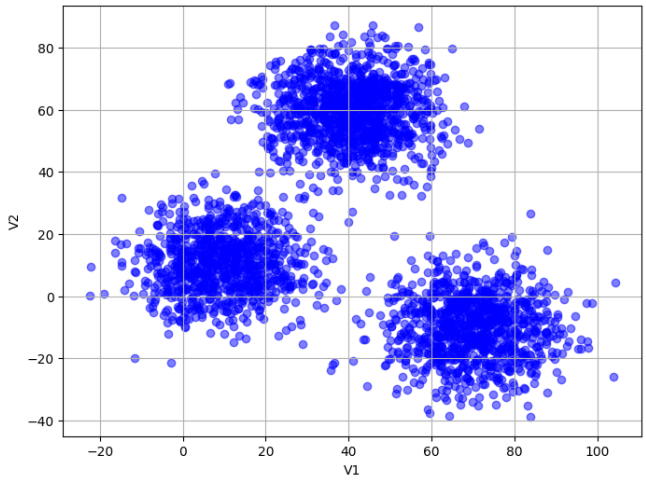


Fig. 1: data distribution

The xclara.csv dataset is a synthetic dataset that is often used to demonstrate clustering algorithms due to its clear and well-defined structure. The dataset contains three distinct clusters, which are visually separable, making it an ideal candidate for evaluating the performance of clustering algorithms. Each cluster in the dataset represents a group of points that are close to each other in the feature space, while the clusters themselves are relatively well-separated.

To better understand the dataset, we first visualize its distribution. As shown in Fig.1, the data points are plotted in a 2D space, with V1 on the x-axis and V2 on the y-axis. From the plot, we can observe that the dataset contains three prominent clusters, each with a relatively dense core and a clear separation from the other clusters. This visual inspection suggests that the dataset is well-suited for clustering analysis, as the clusters are distinct and can be easily identified.

IV. Algorithms

This paper primarily employs the k-means++ algorithm for clustering analysis, with a comparison to the k-means algorithm.

A. Kmeans

The k-means algorithm is an iterative clustering method that partitions a dataset into K clusters. The specific principle is illustrated in Fig. 2. Its execution process typically involves the following steps. First, initialization is performed by randomly selecting K data points as the initial cluster centers. The choice of these initial centers can significantly impact the final clustering results, so heuristic methods like the k-means++ algorithm are often used to improve initialization. Next, in the assignment step, each data point is assigned to the nearest cluster center based on the Euclidean distance, which is calculated as:

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

where x_i is a data point, μ_j is a cluster center, x_{im} is the m-th feature of x_i, and μ_{jm} is the m-th feature of μ_j. After assigning all data points, the update step recalculates the cluster centers as the mean of all data points in each cluster:

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

where C_j is the set of data points in the j-th cluster, and $|C_j|$ is the number of data points in C_j. Finally, the algorithm iterates the assignment and update steps until the cluster centers converge (i.e., they no longer change significantly) or a predefined maximum number of iterations is reached.

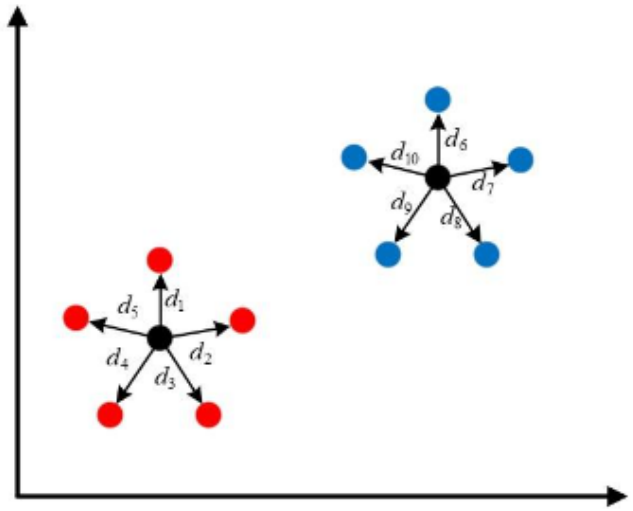


Fig. 2: kmeans

While the k-means algorithm is widely used for clustering due to its simplicity and efficiency, it has several limitations. The algorithm's performance heavily depends on the initial selection of cluster centers. Poor initialization can lead to suboptimal clustering results. The algorithm may converge to a local optimum rather than the global optimum, especially if the initial centroids are poorly chosen.

B. Kmeans++

The k-means++ algorithm was proposed to address the issues of k-means, particularly its sensitivity to the initial selection of cluster centers and its tendency to converge to local optima. k-means++ improves the initialization process, significantly enhancing the stability and clustering performance of the k-means algorithm.

Instead of selecting initial cluster centers completely at random, k-means++ uses a probabilistic approach. Specifically, it first randomly selects one data point as the initial cluster center. Then, it selects subsequent centers based on the distance of each data point to the already chosen centers, with a higher probability of selecting points that are farther away.

By selecting initial centers that are spread out, k-means++ ensures a more uniform coverage of the dataset, reducing the likelihood of the algorithm converging to local optima while also reducing sensitivity to initial conditions.

V. Evaluation

A. Evaluation Metrics

To assess the results, we employ three evaluation metrics. These are the Silhouette Coefficient, the Davies-Bouldin Index, and the Calinski-Harabasz Index. The Silhouette Coefficient is used to measure the similarity of each sample to its own cluster compared to its similarity to the nearest cluster.A value closer to 1 indicates better clustering performance. The formula is as follows:

The Silhouette Coefficient measures how similar a data point is to its own cluster compared to other clusters. It ranges from -1 to 1, where a value closer to 1 indicates better clustering performance. The formula for the Silhouette Coefficient is as follows:

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

where:

- $a(i)$ is the average distance from sample i to other points within the same cluster,
- $b(i)$ is the average distance from sample i to all points in the nearest cluster that is not its own.

The Davies-Bouldin Index (DBI) is a clustering evaluation metric that measures the quality of clustering by assessing both the separation and compactness of clusters. A lower DBI value indicates better clustering. The formula for the Davies-Bouldin Index is:

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left(\frac{s(C_i) + s(C_j)}{d(C_i, C_j)} \right),$$

where:

- k is the number of clusters,
- $s(C_i)$ is the compactness of cluster C_i, measured by the average distance between points within the cluster,
- $d(C_i, C_j)$ is the separation between clusters C_i and C_j, typically measured by the distance between their centroids,

- $\max_{j \neq i}$ indicates that for each cluster C_i, the maximum value is chosen from the comparisons with all other clusters C_j.

The Calinski-Harabasz Index (also known as the Variance Ratio Criterion) evaluates clustering quality based on the ratio of between-cluster variance to within-cluster variance. A higher Calinski-Harabasz score indicates better clustering. The formula is:

$$CH = \frac{\text{Tr}(B_k)}{\text{Tr}(W_k)} \times \frac{N-k}{k-1},$$

where:

- $\text{Tr}(B_k)$ is the trace of the between-cluster scatter matrix, representing the variance between cluster centroids,
- $\text{Tr}(W_k)$ is the trace of the within-cluster scatter matrix, representing the variance within clusters,
- N is the total number of data points,
- k is the number of clusters.

Due to the similar values obtained by these two algorithms on the dataset, we have counted the number of iterations to demonstrate that k-means++ can achieve better results with fewer iterations compared to k-means. This is because k-means relies on randomly selecting initial centroids, which can lead to a poorer initial assignment and thus requires more iterations to con-verge. k-means++ improves this by selecting new centroids that are farther away from the existing ones, resulting in a more uniform and rational initial distribution of centroids. This initialization method reduces the likelihood of the algorithm getting stuck in local optima, hence k-means++typically converges to a better solution in fewer iterations.

B. Parameter Settings

To determine the optimal number of clusters, we use the K-means++ implementation from the `sklearn` library. We test different values of k and calculate the evaluation metrics for each. The results are shown in Fig.3.

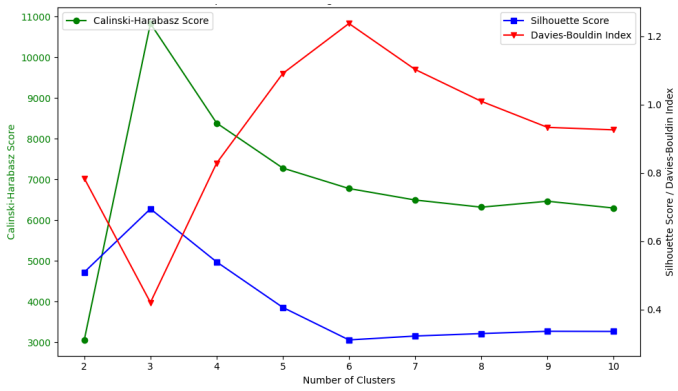


Fig. 3: Selecting the Number of Clusters

The Silhouette Score measures how well each data point fits within its assigned cluster compared to other clusters. A higher Silhouette Score indicates better-defined clusters. In Fig.3, the Silhouette Score (represented by square markers) reaches its maximum value of 0.695 when $k = 3$. As k increases beyond 3, the score decreases significantly, indicating that adding more clusters leads to overfitting or less meaningful groupings. This trend suggests that $k = 3$ is the optimal number of clusters for the dataset, as it maximizes the separation between clusters while maintaining high within-cluster cohesion.

The Davies-Bouldin Index evaluates clustering quality by measuring the ratio of within-cluster compactness to between-cluster separation. A lower DBI value indicates better clustering. In Fig.3, the DBI (represented by inverted triangle markers) achieves its minimum value of 0.421 at $k = 3$. As k increases, the DBI rises, but the increase is relatively small.

The Calinski-Harabasz Score evaluates clustering quality based on the ratio of between-cluster variance to within-cluster variance. A higher score indicates better-defined clusters. In Fig.3, the Calinski-Harabasz Score (represented by dot markers) peaks at 10826.601 when $k = 3$. As k increases beyond 3, the score decreases, indicating that additional clusters do not improve the overall clustering quality. This trend further supports the conclusion that $k = 3$ is the optimal number of clusters, as it maximizes the between-cluster variance while minimizing the within-cluster variance.

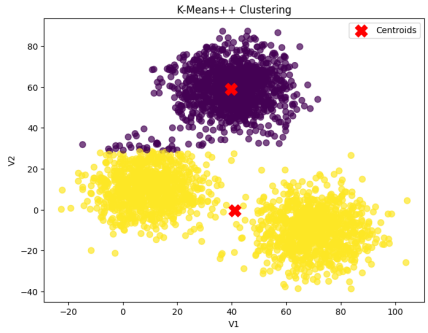
These results demonstrate that $k = 3$ provides the best balance between cluster separation and cohesion, making it the most suitable choice for this dataset. This is quite consistent with the data distribution.

In addition to the evaluation metrics, we further validated our findings by visualizing the clustering results for $k = 2, 3,$ and 4, as shown in Fig.4. These visualizations provide an intuitive understanding of how the K-means++ algorithm partitions the dataset for different values of k.

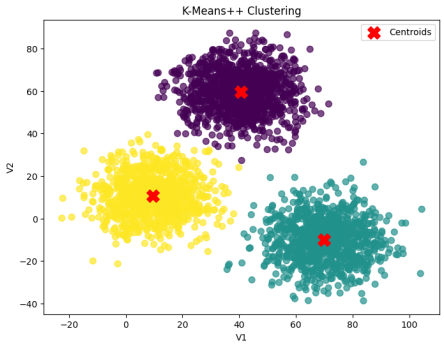
1) Analysis of Visualizations: The visualizations in Fig.4 align closely with the evaluation metrics presented in Fig.3:

- For $k = 2$ (Figure 4a), the dataset is divided into two clusters. However, this partitioning fails to capture the natural structure of the data, as one of the clusters combines two distinct groups. This is reflected in the lower Silhouette Score and higher Davies-Bouldin Index for $k = 2$.
- For $k = 3$ (Figure 4b), the clustering results are optimal. The three clusters correspond well to the natural groupings in the dataset, with clear separation and compactness. This matches the peak Silhouette Score, minimum Davies-Bouldin Index, and maximum Calinski-Harabasz Score observed for $k = 3$ in Fig.3.
- For $k = 4$ (Figure 4c), the algorithm introduces an additional cluster, splitting one of the natural groups into two smaller clusters. This over-segmentation is reflected in the decline of the evaluation metrics for $k = 4$, as the additional cluster does not improve the overall clustering quality.

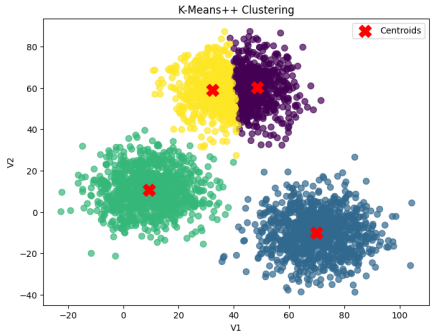
2) Conclusion: The visualizations in Figure 4 confirm that $k = 3$ provides the best clustering results, both quantitatively (based on evaluation metrics) and qualitatively (based on visual inspection). This consistency between the metrics and visualizations further validates our conclusion that $k = 3$ is the optimal number of clusters for this dataset.



(a) Clustering results for $k = 2$.



(b) Clustering results for $k = 3$.



(c) Clustering results for $k = 4$.

Fig. 4: Visualization of clustering results for $k = 2, 3$, and 4. The results for $k = 3$ show the best clustering performance, with clear separation and compactness.

C. KMeans++ and KMeans

To demonstrate the superiority of the k-means++ algorithm over the standard k-means algorithm, we compare the number of iterations and the iteration time required for both algorithms to converge. Using the `KMeans` function from the `sklearn` library, we set the `init` parameter to `random` for the k-means algorithm and `k-means++` for the k-means++ algorithm. The results are shown in Table I.

Algorithm	Iterations	Iteration Time (seconds)
KMeans	4	0.1395
KMeans++	3	0.0548

TABLE I: Comparison of the number of iterations and iteration time between KMeans and KMeans++.

- **Number of Iterations**:
  - The k-means algorithm required 4 iterations to converge. This is because the random initialization of cluster centers often leads to suboptimal starting points, requiring more iterations to reach a stable solution.
  - The k-means++ algorithm required only 3 iterations to converge. This reduction in iterations is due to the smarter initialization method, which selects initial centroids that are more representative of the true cluster structure. As a result, k-means++ converges faster to a better solution.
- **Iteration Time**:
  - The k-means algorithm took 0.1395 seconds to complete. The longer runtime is a direct consequence of the higher number of iterations and the less efficient initialization process.
  - The k-means++ algorithm took only 0.0548 seconds to complete. This significant reduction in runtime is attributed to both the fewer iterations required and the more efficient initialization process, which reduces the computational overhead.

Although the dataset is small and the difference in results is not significant, we believe that the efficiency of the k-means++ algorithm would greatly exceed that of the k-means algorithm, especially when applied to larger datasets.

VI. Discussion and Future Work

While this paper provides valuable insights into the performance comparison between the k-means and k-means++ algorithms, it has certain limitations that should be addressed in future research. Below, we discuss these limitations and propose potential directions for future work.

A. Limitations of the Study

- **Limited Dataset Scope**: The experiments were conducted on the xclara.csv dataset. While this dataset is suitable for demonstrating the basic principles of clustering, it may not fully capture the challenges posed by real-world datasets, such as high-dimensional data, noisy data, or datasets with overlapping clusters.
- **Evaluation Metrics**: The study relied on internal evaluation metrics to assess clustering quality. While these metrics are useful, they may not fully reflect the true performance of the algorithms, especially in the absence of ground truth labels.

B. Future Work

To address the limitations of this study and further advance the field of clustering analysis, we propose the following directions for future research:

- **Experiments on Diverse Datasets**: Future work should evaluate the performance of k-means and k-means++ on a wider range of datasets, including high-dimensional data, noisy data, and datasets with complex cluster structures. This would provide a more comprehensive understanding of the algorithms' strengths and weaknesses.

- **Incorporation of External Evaluation Metrics**: Future studies should incorporate external evaluation metrics to complement the internal metrics used in this study. This would provide a more robust assessment of clustering quality, especially in scenarios where ground truth labels are available.

While this study demonstrates the advantages of k-means++ over the standard k-means algorithm, there are several limitations that warrant further investigation. By addressing these limitations and exploring the proposed future directions, researchers can advance the field of clustering analysis and develop more robust and efficient algorithms for a wide range of applications.

VII. Conclusion

In this paper, we introduce the use of clustering analysis in the field of data mining, where we discuss several common algorithms, including k-means, k-means++, and DBSCAN. We perform clustering analysis on the dataset xclara.csv using both k-means and k-means++ algorithms. We evaluate the quality of clustering using three assessment metrics: the Silhouette Score, the Davies-Bouldin Index, and the Calinski-Harabasz Index. By analyzing the number of iterations and the time of iteration, we demonstrate that the k-means++ algorithm outperforms the k-means algorithm under certain conditions. We have implemented the clustering analysis and concluded that the k-means++ method per-forms excellently with this dataset, showing significant advantages in many practical applications.Nevertheless, the selection of the appropriate clustering method still needs to be made based on the characteristics of the data and the requirements of the clustering task.

References

[1] J. MacQueen, "Some methods for classification and analysis of multivariate observations," 1967. [Online]. Available: https://api.semanticscholar.org/CorpusID:6278891

[2] S. V. David Arthur, "k-means++: The advantages of careful seeding," 2007. [Online]. Available: https://theory.stanford.edu/%7Esergei/papers/kMeansPP-soda.pdf

[3] mubaris, "sclara.csv," https://github.com/mubaris/friendly-fortnight/blob/master/xclara.csv.

[4] B. Fang-Ming, W. Wei-Kui, and C. Long, "Dbscan: Density-based spatial clustering of applications with noise," *Journal of Nanjing University(Natural Sciences)*, vol. 48, no. 4, pp. 491–498, 2012.

[5] L. Mcinnes, J. Healy, and S. Astels, "hdbscan: Hierarchical density based clustering," *The Journal of Open Source Software*, vol. 2, no. 11, 2017.