Journal of Computer and Communications, 2024, 12, 146-159
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

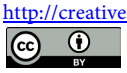
Comprehensive K-Means Clustering

Ethan Xiao

Rye Country Day School, Rye, USA
Email: ethanxiao3@gmail.com

How to cite this paper: Xiao, E. (2024) Comprehensive K-Means Clustering. *Journal of Computer and Communications*, **12**, 146-159.
https://doi.org/10.4236/jcc.2024.123009

Received: January 17, 2024
Accepted: March 23, 2024
Published: March 26, 2024



Abstract

The *k-means* algorithm is a popular data clustering technique due to its speed and simplicity. However, it is susceptible to issues such as sensitivity to the chosen seeds, and inaccurate clusters due to poor initial seeds, particularly in complex datasets or datasets with non-spherical clusters. In this paper, a *Comprehensive K-Means Clustering* algorithm is presented, in which multiple trials of *k-means* are performed on a given dataset. The clustering results from each trial are transformed into a five-dimensional data point, containing the scope values of the x and y coordinates of the clusters along with the number of points within that cluster. A graph is then generated displaying the configuration of these points using *Principal Component Analysis* (*PCA*), from which we can observe and determine the common clustering patterns in the dataset. The robustness and strength of these patterns are then examined by observing the variance of the results of each trial, wherein a different subset of the data keeping a certain percentage of original data points is clustered. By aggregating information from multiple trials, we can distinguish clusters that consistently emerge across different runs from those that are more sensitive or unlikely, hence deriving more reliable conclusions about the underlying structure of complex datasets. Our experiments show that our algorithm is able to find the most common associations between different dimensions of data over multiple trials, often more accurately than other algorithms, as well as measure stability of these clusters, an ability that other *k-means* algorithms lack.

Keywords

K-Means Clustering

1. Introduction

Data clustering is the process of grouping similar data points together based on their intrinsic characteristics or patterns, aiming to reveal the underlying struc-

ture within a dataset. Data clustering has a variety of applications, ranging from biology and genomics, where it aids in the classification of genes with similar functions [1] to marketing and customer segmentation, where it helps businesses tailor their strategies to distinct customer groups [2]. K-means, which is one of the many clustering algorithms, partitions data into a predetermined number of clusters based on centroid proximity. Data is grouped based on the closest centroids to those points, and then the centroids are adjusted to be the average of every point in its group. This process iterates until a stable configuration of clusters and centroids is reached [3]. Lloyd's K-means, a popular k-means algorithm and widely accepted as the "standard" k-means algorithm, initially emerged as a vector quantization technique for Pulse-Code Modulation (PCM) in signal processing. It starts with an initial "seed," or a set of k randomly selected centroids, and iteratively refines it to reach an optimal configuration. With each iteration, Lloyd's k-means assigns each point to its nearest centroid; then, an average is taken from these points, becoming a new centroid. Lloyd's k-means's popularity, especially in machine learning and data mining, is due to its speed and simplicity [4]. However, it has two notable limitations.

Inconsistency. The results of the *k-means* algorithm rely heavily on the algorithm's initial seeding strategy [5]. That is, different starting seeds are likely to yield different clusters. In such cases, a single trial of *k-means* may not encompass the entire spectrum of underlying patterns or relationships present in the data, and multiple trials may result in widely different clusterings, many of them deviating from the most optimal solution. Therefore, it is difficult to find a definite set of clusters with this algorithm.

Inaccuracy. In certain datasets, such as when the inherent data distribution deviates from *kmeans*'s idealized assumption of spherical clusters, the algorithm may create sub-optimal cluster assignments [5]. This limitation is especially significant in the presence of outliers, which can disrupt the algorithm and produce clusters that do not accurately capture the underlying structure of the data.

In this paper, a *Comprehensive K-means* algorithm is presented in order to enhance the robustness and adaptability of the original *k-means* algorithm, particularly for complex datasets with irregularly shaped clusters or outliers. In the following section, related work on improving *k-means* algorithms is discussed. Section 3 provides a detailed explanation of the presented algorithm, followed by experimental results using synthetic and real-world datasets in Section 4. Finally, the paper concludes with a summary and an overview of future work.

2. Related Work

In the standard version of *k-means*, otherwise known as *Lloyd's k-means*, the goal is to identify a set of k centers, in which k is a predetermined value denoting the number of final clusters. Given a dataset X consisting of n data points in the d-dimensional space Rd, the *k-means* algorithm strives to discover a set C containing k centers while minimizing the function $\phi X(C)$, the sum of all distances

between each point *Xn* and the closest centroid to them [3] [4]. *K-means++* is developed by Arthur and Vassilvitskii to improve *Lloyd's K-means* by implementing a more efficient seeding method [6]. That is, once a centroid *C* is chosen randomly among the dataset *X*, the probability P that the point will be selected as a centroid is calculated by squaring the distance between a point *Xn* and the nearest centroid *C*. These probabilities then "weight" their respective points, so the algorithm will favor points that are farther from existing centroids than closer points. This process continues until *k* centroids are selected, which are often further apart than those from *Lloyd's K-means*. In comparison, *K-means++* ensures more accurate results with a guaranteed approximation ratio of O(log k). The computation time is also shorter because the algorithm reaches a stable state more quickly [6].

Deshpande *et al.* developed *Robust K-means++* [7], in which sampling is used to pick candidate centroids such that there is a configuration of *k* of these centroids that will most optimally cluster the data. Even in the worst cases, this seeding method is not susceptible to outliers, as the algorithm discards outliers before finding a starting seed.

Bradley *et al.* then developed *Constrained K-means Clustering* [8], to avoid "empty" clusters, which have too few points to find any notable meaning in datasets, by specifying a minimum number of points per cluster.

To reduce the influence of outliers and improve clustering efficiency on large datasets, *MiniBatch K-means* was presented to cluster a random subset of the dataset for each iteration, until stable clusters can be achieved [9] [10]. For each cluster, an initial data point is randomly selected as the starting point for that cluster. In each iteration, a certain number of points are selected from the dataset, temporarily stored in the set M, a "mini-batch" that represents the set as a whole. Then, standard *k-means* is performed on the mini-batch. This approach greatly reduces computing time on large datasets [9].

Stability Analysis in K-means Clustering [11] is a method developed by Steinley in which multiple clusterings, instead of the most optimal clustering, are used to analyze a dataset's stability. With an object by object co-occurrence matrix, data from locally optimal clusterings are collected and reordered by a steepest ascent quadratic assignment procedure to visualize the dataset's structure. Then, the structure of the data and the most optimal number of clusters can be interpreted from the visualization and data.

Robust Trimmed K-means [12], proposed by Doriabala *et al.*, performs efficiently in both single-membership (in which points can only belong to one cluster) and multi-membership (in which points can belong to multiple clusters) scenarios. Each point has weights pertaining to each cluster, determining the extent to which the point belongs to that cluster. This lack of dichotomy in cluster allocations allows for more precision in clustering, and also reduces the influence of outliers on the accuracy of the clustering of a dataset.

K-means Clustering Based on Observation Point Mechanism [13], proposed

by Zhang *et al.*, aims to achieve results similarly to *k-means* with a smaller subset that mimics the structure of the original dataset, which greatly reduces computation time. Since outliers are removed while taking this representative subset, clustering results are often more accurate and stable.

Some of *k-means*'s weaknesses are due to its sensitivity to outliers and inefficiency with large datasets. Li *et al.* propose K-means Clustering with Bagging and MapReduce [14], which uses an ensemble learning method called bagging [15] and a distributed computing network named MapReduce [16] to make *k-means* less sensitive to outliers and more efficient computationally. Yang *et al.* used a Deep Neural Network (DNN) to perform dimension reduction on the dataset instead of using traditional methods such as PCA [17]. While the *k-means* algorithm itself isn't modified, the usage of Deep Learning in dimensionality reduction produces more accurate clusterings. The DNN, unlike traditional methods, assumes that any dataset is nonlinear, which results in better approximated datasets more efficiently and accurately. Shindler *et al.* developed *Fast and Accurate k-means For Large Datasets* [18], which efficiently deals with clustering large datasets, and uses approximate nearest neighbor search to compute *k-means* in $O(nk)$ time, while also reducing the effect of outliers.

There are many *k-means* algorithms presented that work effectively in specific fields, each with their own set of pros and cons. In this paper, we introduce *Comprehensive K-means*, which is designed to handle a broader range of data types with increased accuracy and reliability. Unlike other algorithms based on *k-means*, it is able to measure the stability of the clusters which it creates by measuring the variance of these clusters over multiple trials, as well as finding the clustering arrangements that appear the most over differently seeded trials.

3. Comprehensive K-Means Clustering

Comprehensive K-means Clustering is a newly designed algorithm that aims to achieve the goals explained in Section 2, as well as improve the algorithm's accuracy and consistency. The implementation flowchart of the *Comprehensive K-means Clustering* algorithm is shown in **Figure 1**, which contains two similar processes with different inputs: random seeds and random subsets. In the random seed process, a random set of centroids is chosen; in the random subset process, a user inputs a certain percentage, $p\%$, of data points to be clustered. Random seeding results in a graph of the trial distributions, called the seed graph, with each point representing the results from one randomly seeded trial on that dataset. By observing the seed graph, we can discover the differences in clustering results over multiple trials and identify the most common solutions. Meanwhile, the result from random subsets is called the subset graph, which represents the trial's results after taking $p\%$ of the data as a subset. From the distribution of these points and the distance which the graph spans, we can under-

stand the dataset's stability. This algorithm is an improvement over current ones as it explores both popular patterns and measures variability from multiple trials of *k-means*, hence it being named *Comprehensive K-means*. This information can be used to study the strength of a dataset's clusters. Furthermore, *Comprehensive K-means* can be run with multiple k-values to find the most consistent k-value for the dataset.

The random seed process is shown on the left side of Figure 1. Given the dataset X, the k-value k, and the number of trials T:

1) Each trial starts with a random set of k centroids, or points selected from X. This set of centroids is known as the seed. T trials of *Lloyd's k-means* are performed with this seed on X, and each trial will result in k clusters.

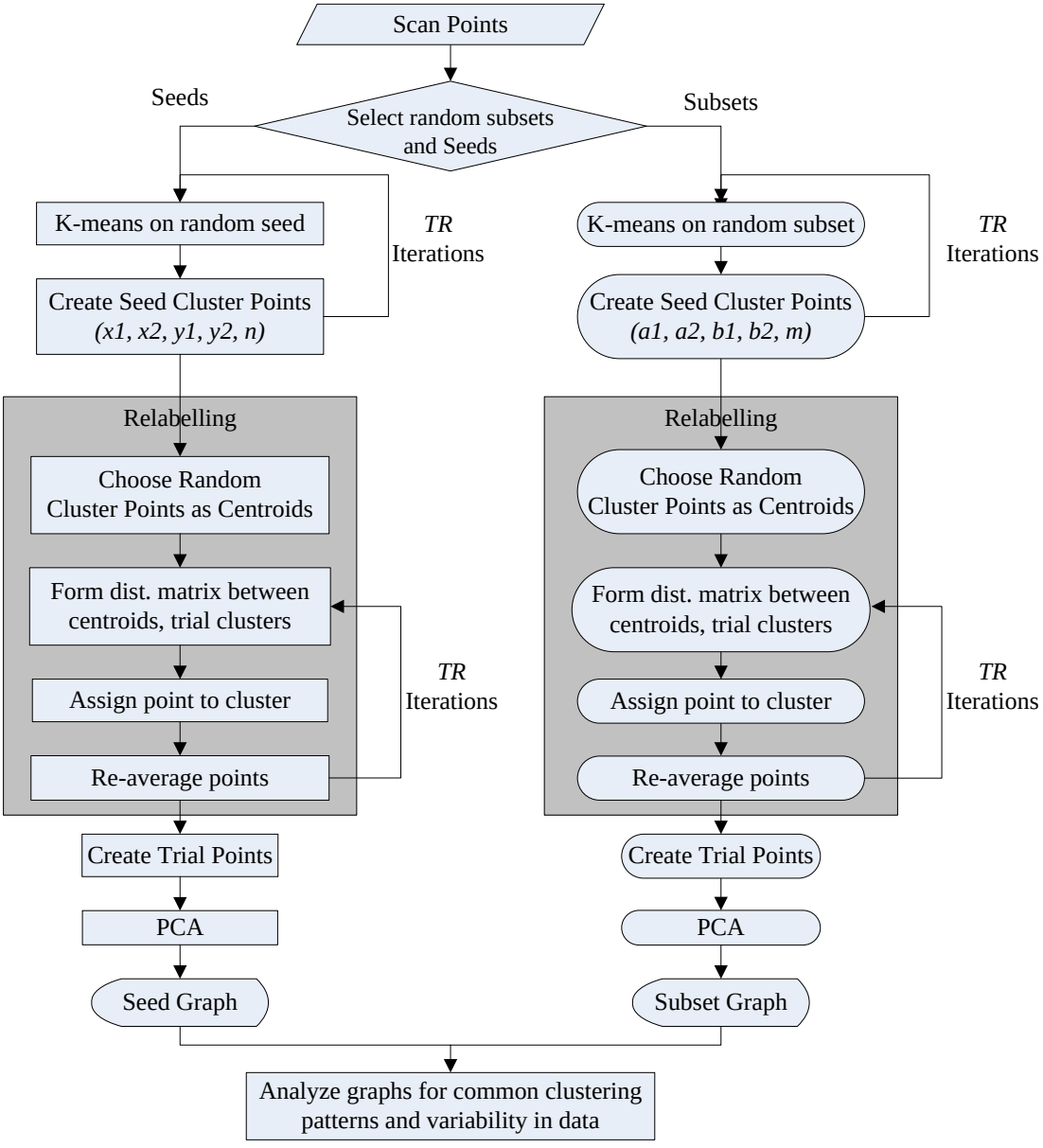


Figure 1. Flowchart of comprehensive K-means.

2) Each cluster generated is described as a 5 dimensional dataset (x_1, x_2, y_1, y_2, n): the left-bound of the domain x_1, the right-bound of the domain x_2, the lower bound of the range y_1, the upper bound of the range y_2, and the number of points in the cluster n.

An example of the dataset is shown in Figure 2. For instance, Cluster 1 is stored in the format (x_{11}, x_{21}, y_{11}, y_{21}, 18). This method takes into consideration the cluster's location, span, and membership while also being computationally efficient and accounting for the potential influence of outliers. The result of this step is a set of "cluster-points," containing all clusters across all trials stored as points.

3) In the relabeling step, a set of k centroids is chosen from the set of cluster-points.

For every k cluster-points, a matrix is formed, holding all cosine similarities between all k centroids and these cluster-points. The purpose of the distance matrix is to be used to optimally assign each cluster-point to a centroid. Cosine similarities are used, as when the data is normalized around the origin, cosine similarities will guarantee accurate labeling compared to Euclidean distance.

4) The maximum value in the matrix is chosen, and the cluster assigned to that value is assigned to the centroid it shares the value with.

5) Steps 4 and 5 are reiterated for each trial until every cluster has been assigned.

6) Then, each centroid is updated as the average of the set of all clusters that have been as-signed to it, similarly to *k-means*.

7) Steps 4-7 are repeated until the centroids converge.

8) Each cluster in a trial will be given labels from 1 through k so that across trials, the most similar clusters all have the same label, and when the trial points are being constructed, clusters 1 through k will be ordered in the form 1, 2, … k, and the ensuring trial-point will be a point with $k*5$ dimensions in the form (x_{11}, x_{21}, y_{11}, y_{21}, n_1, x_{12}, x_{22}, …, x_{1k}, x_{2k}, y_{1k}, y_{2k}, n_k).

9) These points are then projected onto the two-dimensional plane using *Principal Component Analysis (PCA)*, creating a dataset with all T trials in point form. The two-dimensional form allows these points to be visualized on a graph.

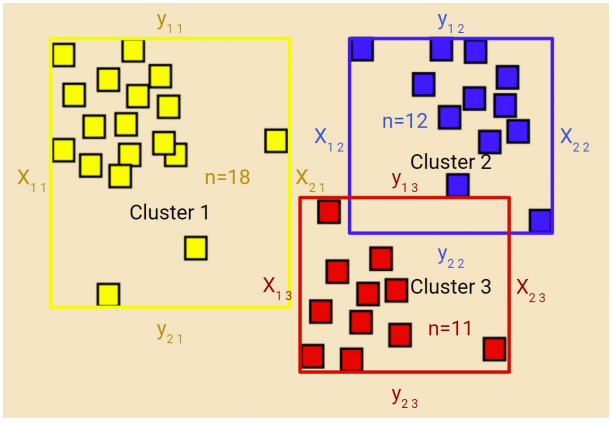


Figure 2. The "bounding box" method of converting clusters into points.

PCA is a dimensionality reduction method that can reduce the number of dimensions in a dataset to a value less than or equal to the former number of dimensions. The graph at the end of this process displays each trial as a 2-dimensional point, showing the similarities and differences of each trial of *k-means* performed on the data. This information collected over multiple trials is graphed in two dimensions so that the similarities and differences among all trial points can easily be visualized.

The same process is carried out with random subsets, shown at the right side of the flow chart. The constants k, T, and X are all defined as before, and the percentage of points kept in each trial p is determined by the user. In each trial, a subset of p% of the original dataset is randomly selected as the input, and the algorithm generates one constant seed that is used to start all trials. The following steps are the same as the random seed process, and the generated subset graph reveals the extent at which the removal of a certain number of points might impact the clustering of the dataset.

The time complexity for the *Comprehensive K-means* algorithm can be expressed as $O(TNIK)$, where:

T: number of trials of the *k-means* algorithm performed

N: number of data points in the dataset

I: *number* of iterations

K: chosen k-value

The most computationally intensive step in *k-means* is assigning points to centroids, which iterates through each point of the dataset K times; hence, the time complexity of *k-means* is NK. This occurs for I iterations for T trials, hence the time complexity of $TNIK$. The algorithm comprises four main components: Scanning each data point in the dataset; the time complexity is $O(N)$. Executing T trials of the *k-means* algorithm on the dataset; the time complexity is $O(TNIK)$. Performing the relabeling step; the time complexity is $O(TIK^2)$. Applying *Principal Component Analysis (PCA)* to T final points; the time complexity is $O(25K^2N + 125K^3)$.

The computational burden associated with the scanning step is negligible compared to the time required for executing the *k-means* trials, considering that $TIK > 1$. Consequently, as the number of data points N surpasses the threshold of $125K^3/(TIK - 25K^2)$, which is highly likely in the case of large datasets, the time complexity $O(TRNTK)$ becomes more computationally demanding than the *PCA* time complexity $O(25K^2N + 125K^3)$.

4. Case Study

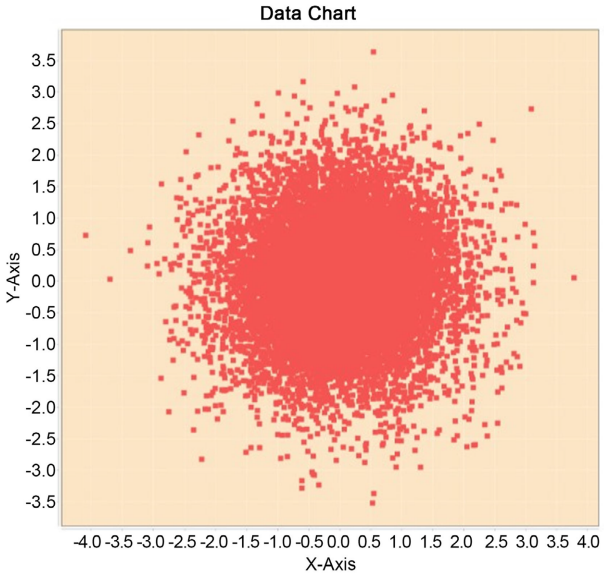
A Gaussian is an arrangement of points in which points are arranged around the Gaussian's center based on the Gaussian probability distribution. That is, 66% of the points will be generated within 1 std from the Gaussian's center, 95% of the points will be within 2 std of the Gaussian, and henceforth. By generating synthetic Gaussians with tens of thousands of points, we simulated large datasets with defined clusters. An example Gaussian is shown in **Figure 3(a)**. **Figure**

3(b) represents the results of the algorithm run on Figure 3(a), with the graph on the left representing seeding results with a *k*-value of 2 over 100 trials, and the graph on the right representing subset results with a *k*-value of 2 over 100 trials with 50% of points kept each trial run. To validate the algorithm's efficacy, we started with one Gaussian with no definite clusters to test if our algorithm could find that the Gaussian has no definite clustering based on the subset chart's spread. Then, we tested two Gaussian datasets with the same number of points and standard deviation, which had centers set at a certain number of standard deviations apart. The results from *Comprehensive K-means* stayed consistent, even as the two Gaussians' centers reached within 0.125 standard deviations of each other, but the number of different clustering results drastically increased under 0.25 standard deviations, as seen on Table 1. The spread is calculated by approximating the distance between the two farthest points in the graph. Besides, we also observed that as the number *n* of Gaussians increased, so did the minimum standard deviation for our results to begin destabilizing.

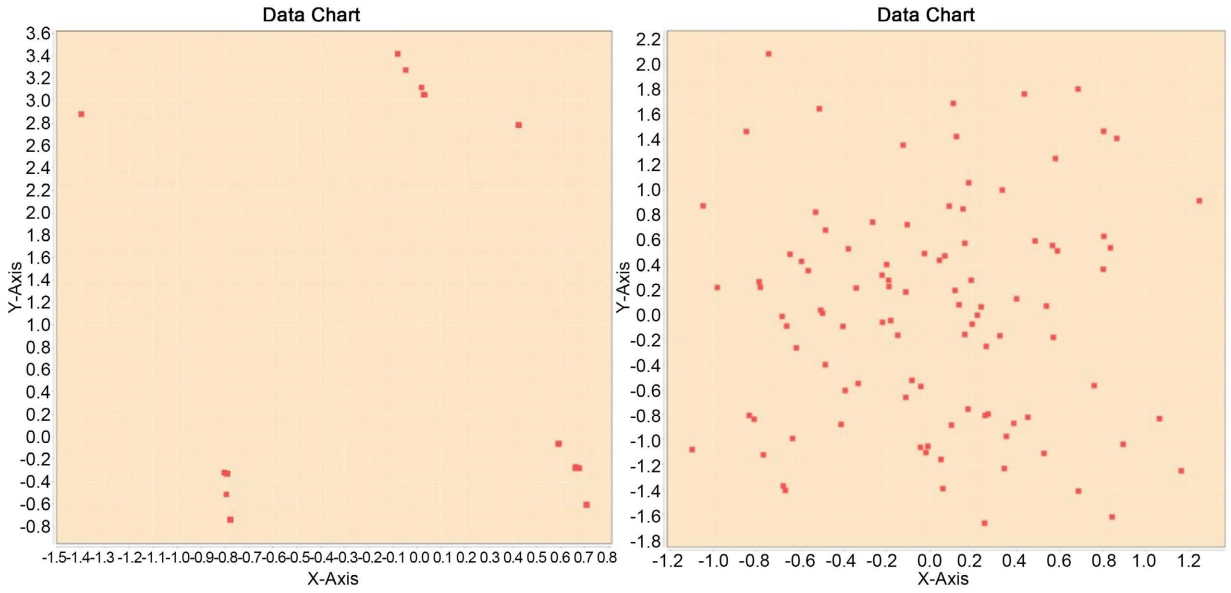
Gaussians with differing numbers of points were used to simulate skewed data; we also inserted artificial outliers to visualize the effect such as outliers would have on the stability of the clusterings of otherwise stable datasets, as shown in Figure 4(a).

Table 1. Results of two-Gaussian data with differing standard deviations.

StDev	Center 1	Center 2	Points	Observed Patterns	Subset Spread
1	−0.5, 0	0.5, 0	5	1	1.0
0.5	−0.25, 0	0.25, 0	4	1	1.8
0.25	−0.125, 0	0.125, 0	3	1	1.9
0.125	0.0625, 0	0.0625, 0	14	1	2.1 (3.1 w/outlier)
0	0, 0	NA	14	4	3.9



(a)



(b)

Figure 3. Single dataset results. (a) A Gaussian centered at (0, 0) with standard deviation 1; (b) The results of performing Comprehensive K means Clustering on the Gaussian shown in **Figure 3(a)**.

Through the tests, we noticed that the presented algorithm generally showed consistent clustering patterns for these data. The left graph in **Figure 4(b)** is the seed graph; the right graph of **Figure 4(b)** is the subset graph with 50% of points kept in each run. Both graphs have had a *k*-value of 2 performed on the data for 100 trials. The seed graph is stable due to the main body of the dataset; yet, the subset graph's spread is higher than expected due to the two outliers, meaning that the presence of the outliers heavily affected the stability of the dataset, resulting in vastly different configurations of clusters when subsets of the data were taken. Furthermore, we observed that as the data became more difficult to cluster, the common clustering patterns began diversifying. In the data with synthetic outliers, our subset results were much more spread apart compared to relatively stable data.

Real-World Data Results

We evaluated our algorithm using datasets from Kaggle, a data science community website that shares a variety of datasets, and several example datasets from CellRank, a computational framework used in single-cell RNA sequencing data analysis [19]. The algorithm is tested with a wine dataset [20], a California housing dataset [21], and a zebrafish dataset from CellRank, k, as shown in **Figure 5**.

PCA has been used on each of these datasets to reduce them into two dimensions.

The wine dataset is a 13-dimensional dataset, containing information on variables like wine quality, acidity, etc. about 178 different variants of the "Vinho Verde" wine [20]. With *Comprehensive K-means*, we can discover associations between variables like wine acidity, sulfur dioxide content, pH, etc. and wine

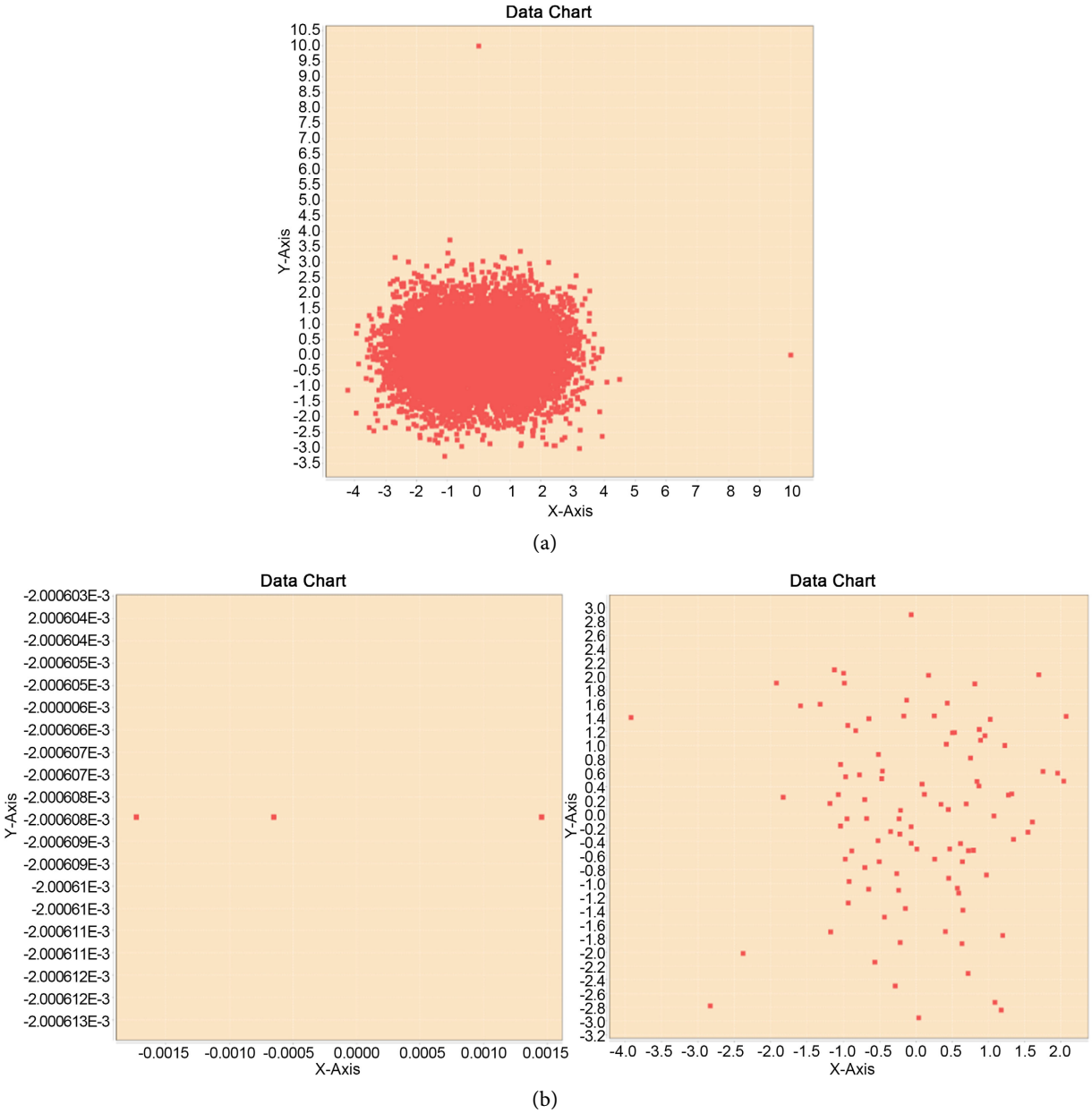


Figure 4. Outlier Gaussian results. (a) Two Gaussians that are 2 standard deviations apart with 2 artificial outliers; (b) Results of using *Comprehensive K-means* on data shown in **Figure 4(a)**.

quality. To convert the data into two dimensions, we converted all of the other variables into one dimension, and then created two-dimensional points with those as the x-coordinates and their associated normalized quality variables as the y-coordinates. As shown on **Table 2**, we tested a variety of *k*-values on the dataset's two-dimensional form and found that the *k*-value of 2 yielded the most consistent results, but a *k*-value of 3 had the lowest sensitivity. The clustering result using our optimal *k*-value shows the associativity between quality and the other variables. For instance, by analyzing the most common clusterings, we found that wine with higher quality often had higher alcohol, sulfates, and citric acid levels, while also having lower volatile acidity, density and sulfur dioxide.

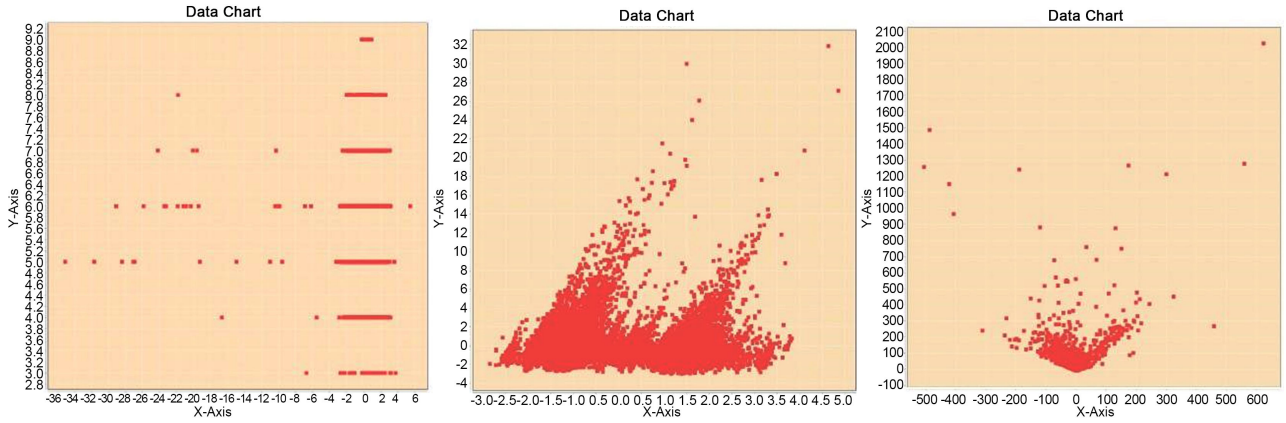


Figure 5. Real-world datasets. 2-dimensional representations of the real-world datasets. The wine data is on the left, the housing data is in the middle, and the zebrafish data is on the right.

Table 2. Wine dataset results.

K-value	Num. Points	Observed Patterns	Subset Spread
2	4	2	5.9
3	29	4	2.6
4	47	3	3.4
5	9	Not Discernible	10.5

The housing dataset has 10 dimensions and originally contains information about 20,640 houses from the 1990 California census [21]. As a popular machine learning dataset, this housing dataset is an effacious test case for our algorithm, allowing us to explore clusters between variables like location, number of bedrooms, price, etc. In its two dimensional form, we found that *k*-values of 2 and 3 worked very well, yet 3 created slightly more stable clusters compared to 2, as shown in **Table 3**. Some information can be found from the dataset by analyzing the clustering result; for instance, by using our optimal *k*-value to cluster the data, we found that as houses gradually converged on a specific latitude and decreased in longitude, the median house value, median income, number of households, population, and number of rooms would all increase, suggesting that areas in the mid-west portion of California are more densely populated and have larger houses than other areas in California.

The zebrafish dataset is an example dataset from CellRank, originally containing information about single-cell RNA sequencing of zebrafish embryos. This dataset is a real-world, biological dataset with 2434 dimensions and 27,934 points. Specific information about the dataset, including associativity among cells, is available on CellRank. By using the presented algorithm on the dataset, we found that while a k-value of 2 seemed to work best in terms of finding only one clustering pattern, a k-value of 4 yielded the most stable clusterings, as its graph had the smallest spread, shown in **Table 4**.

Table 3. Housing dataset results.

K-value	Num. Points	Observed Patterns	Subset Spread
2	2	1	3.9
3	3	1	3.5
4	12	2	4.0
5	60	Not Discernible	5.0

Table 4. Zebrafish dataset results.

K-value	Num. Points	Observed Patterns	Subset Spread
2	1	1	1040
3	3	1	1000 (1600 w/outlier)
4	12	2	800 (2000 w/outlier)
5	45	4	1290 (2000 w/outlier)

5. Conclusions and Future Work

Comprehensive K-means Clustering is an insightful and consistent algorithm that is capable of detecting instability in clusters formed by *k-means* and common patterns of clustering. Its method of gathering data over multiple trials makes it more favorable over other *k-means* variations such as *k-means++* and *Mini-batch k-means* in that aspect. By evaluating the algorithm using several synthetic and real datasets, we have evidence showing that the algorithm can gather much more information compared to the original *k-means*.

In *Comprehensive K-means*, *k-means* is iterated through multiple trials, and then *k* points from the clusters in each trial are created in order to find common patterns and instability. Yet, if we use other algorithms such as *k-means++* instead of *k-means* to collect data, the results and the computation time of *Comprehensive K-means* could change due to the increased efficiency of each trial. *K-means++*'s seeding algorithm is "too accurate," which causes it to ignore "erroneous" clusters generated by *k-means*. These clusters, as proof of a dataset's instability, are necessary to analyze a dataset's consistency. Thus, in future work, we plan to further enhance *Comprehensive K-means* using other variations of *k-means* in order to find a balance between efficiency and variability.

We also only enabled the algorithm to work with two-dimensional data; the cluster-points and trial-points generally were restricted to 5 variables per cluster, due to the use of bounding boxes to describe them. Future work could be done to expand the algorithm to process higher-dimensional data and also to find more precise ways to turn clusters into points.

Acknowledgements

I'd like to thank Dr. Andrew Blumberg for his support and guidance in writing this paper. Without his mentorship, this project would not be possible.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Eisen, M.B., *et al.* (1998) Cluster Analysis and Display of Genome-Wide Expression Patterns. *Proceedings of the National Academy of Sciences*, **95**, 14863-14868. https://doi.org/10.1073/pnas.95.25.14863

[2] Provost, F. and Fawcett, T. (2013) Data Science and Its Relationship to Big Data and Data-Driven Decision Making. Mary Ann Liebert, Inc., Larchmont. https://doi.org/10.1089/big.2013.1508

[3] Lloyd, S. (1982) Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, **28**, 129-137. https://doi.org/10.1109/TIT.1982.1056489

[4] Ikotun, A.M., *et al.* (2023) K-Means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data. *Information Sciences*, **622**, 178-210. https://doi.org/10.1016/j.ins.2022.11.139

[5] Fränti, P. and Sieranoja, S. (2019) How Much Can k-Means Be Improved by Using Better Initialization and Repeats? *Pattern Recognition*, **93**, 95-112. https://doi.org/10.1016/j.patcog.2019.04.014

[6] Arthur, D. and Vassilvitskii, S. (2007) K-Means++: The Advantages of Careful Seeding. *Proceedings of the* 18*th Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, 7-9 January 2007, 1027-1035.

[7] Deshpande, A., *et al.* (2020) Robust *k*-Means++. *Proceedings of the* 36*th Conference on Uncertainty in Artificial Intelligence* (*UAI*), Toronto, 3-6 August 2020, 799-808.

[8] Bennett, K.P., *et al.* (2018) Constrained K-Means Clustering. Microsoft Research. https://www.microsoft.com/en-us/research/publication/constrained-k-means-clustering/

[9] Feizollah, A., Anuar, N.B., Salleh, R. and Amalina, F. (2014) Comparative Study of k-Means and Mini Batch k-Means Clustering Algorithms in Android Malware Detection Using Network Traffic Analysis. 2014 *International Symposium on Biometrics and Security Technologies* (*ISBAST*), Kuala Lumpur, 26-27 August 2014, 193-197. https://doi.org/10.1109/ISBAST.2014.7013120

[10] Bejar, J. (2013) K-Means vs Mini Batch K-Means: A Comparison. UPCommons.

[11] Steinley, D. (2008) Stability Analysis in K-Means Clustering. *British Journal of Mathematical and Statistical Psychology*, **61**, 255-273. https://doi.org/10.1348/000711007X184849

[12] Dorabiala, O., *et al.* (2021) Robust Trimmed k-Means.

[13] Zhang, X.L., *et al.* (2020) A Robust k-Means Clustering Algorithm Based on Observation Point Mechanism. *Complexity*, **2020**, Article ID: 3650926. https://www.hindawi.com/journals/complexity/2020/3650926/ https://doi.org/10.1155/2020/3650926

[14] Li, H.-G., *et al.* (2011) K-Means Clustering with Bagging and MapReduce. 2011 44*th Hawaii International Conference on System Sciences*, Kauai, 4-7 January 2011, 1-8. https://doi.org/10.1109/HICSS.2011.265

[15] Rocca, J. (2021) Ensemble Methods: Bagging, Boosting and Stacking. Medium. https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking

-c9214a10a205

[16] Dean, J. and Ghemawat, S. (2008) MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, **51**, 107-113. https://doi.org/10.1145/1327452.1327492

[17] Yang, B., *et al.* (2016) Towards K-Means-Friendly Spaces: Simultaneous Deep Learning and Clustering.

[18] Shindler, M., *et al.* (2011) Fast and Accurate k-Means for Large Datasets. In: Shawe-Taylor, J., *et al.*, Eds., *Advances in Neural Information Processing Systems*, Curran Associates, New York. http://proceedings.neurips.cc/paper_files/paper/2011/file/52c670999cdef4b09eb656850da777c4-Paper.pdf

[19] Baccin, C., *et al.* (2019) CellRank: A Scalable and Unbiased Algorithm for Mapping and Characterizing the Early Lineage Bifurcation in Single-Cell RNA-seq Data.

[20] Pramar, R. (2018) Wine Quality. Kaggle. https:///www.kaggle.com/datasets/rajyellow46/wine-quality?rvi=1

[21] Nugent, C. (2017) California Housing Prices. Kaggle. https://www.kaggle.com/datasets/camnugent/california-housing-prices?resource=download