K-Means Clustering with Outlier Removal

Advised by: Dr. Cogswell

Senior Seminar

Department of Mathematics and Statistics

South Dakota State University

Cole Rausch

Spring 2023

Abstract

KMOR is a modified K-Means algorithm that addresses the major drawback of traditional K-Means clustering in detecting and removing outliers. The study applies KMOR to the IRIS data set and Sanford Breast Cancer Dataset, with randomly generated outliers added only to the IRIS dataset. The paper discusses the theoretical underpinnings of the updated algorithm and its proof of concept, along with an analysis of the results obtained. The study emphasizes the significance of accounting for outlier detection and removal during clustering analysis, and demonstrates that KMOR can deliver improved clustering outcomes compared to traditional K-Means clustering.

K-Means Clustering with Outlier Removal

1 Introduction

Clustering is a powerful technique in data science that enables grouping similar data points based on shared characteristics. This technique is useful in summarizing large data sets and making predictions. K-Means is a commonly used clustering method known for its simplicity and efficiency. However, it has a significant weakness - its susceptibility to outliers. In this paper, a modified version of K-Means called KMOR is proposed as a solution to this problem. While KMOR is not readily available in R, it has been developed and implemented from scratch for the purposes of this study. To demonstrate the efficiency of the approach, outliers will be generated and added to the IRIS data set. The performance of the modified K-Means clustering method, KMOR, will be compared against traditional K-Means. Additionally, KMOR will be applied to a breast cancer research data set provided by Sanford Health to gain insights into the characteristics of breast cancer patients at Sanford Health and highlight the benefits of the outlier detection method.

2 The Data

A data set containing information about breast cancer patients treated at Sanford Health has been received. While the patients remain anonymous, they can be identified using their unique Patient.ID. The data set includes 31 variables, but the focus of this project will be on the numerical variables. These variables are stored as integers and will be used for K-Means clustering to form clusters based on how they vary across different groups of patients. The objective is to gain a better understanding of the characteristics of breast cancer patients at Sanford Health.

Prior to using the KMOR algorithm to analyze the Sanford Breast Cancer Research Data Set, it is necessary to clean the data in order to obtain accurate results. The primary concerns are irrelevant data and errors. The initial step will involve removing any columns that contain irrelevant data for KMOR analysis, such as categorical or text data.

Lastly, the variable "Vital.Status.Desc," which classifies patients as "Dead" or "Alive," will be modified. In order to ensure consistency and accuracy while working with numerical data in K-Means clustering, ”Dead” will be converted to 1 and ”Alive” will be converted to 0. Thoroughly cleaning the data set will instill confidence in the analysis performed with the KMOR algorithm as it will be based on relevant and accurate data.

Table 1: Data Dictionary received from Sanford

Variable Name	**Data Type**	**Description**
Age.at.Diagnosis	int	Age of the patient when they were diagnosed with breast cancer
Vital.Status.Desc	chr	Describes whether the patient was alive or dead at the time of Data Pull (March 2022)
Histo.Behavior.ICD.O.3	int	ICD-O-3 classification of the site, morphology, behavior, and grading of cancers - CODE
Postal.Code.at.Diagnosis	int	Patient’s postal code of residence during diagnosis
Height.at.Admission..ins.	int	Height of patient at admission in inches
Weight.at.Admission..lbs.	int	Weight of patient at admission in pounds
BMI..Body.Mass.Index.	num	Body Mass Index (BMI) of patient at admission
Disease.Free.Survival	int	Number that tells the chances of staying free of a disease or cancer after a particular treatment
Survival	int	?
Cause.of.Death	int	Cause of death - CODE (blank/0 = alive according to records?)

3 K-Means Clustering

K-Means clustering divides a data set into k clusters, where k is a natural number greater than or equal to 2. To start, the data will need to be transformed into an n by m matrix. According to Rao (2022) [4], the K-Means clustering algorithm can be broken down into the following steps:

1. Choose value for k, or number of clusters.

2. Select k number of data points at random to be the center of each cluster.
3. Measure the distance from each data point from the center of each cluster. The cluster closest to the data point will be assigned to the data point.
4. Compute the new center of each cluster by using each data point in the cluster.
5. Measure the quality of each cluster to find areas that need improvement.
6. Repeat steps 3 to 5.

If the amount of clusters is unknown or not clear from the start, test the quality of clusters using different amounts of k. One common method to determine the optimal number of clusters in a data set is to use a Within-Cluster-Sum-of-Squares (WSS) plot, also known as an elbow plot. To determine the quality of the clustering, the sum of squared distances between each data point and its assigned cluster center is calculated, a technique commonly used in clustering algorithms (Spectral Clustering vs K-Means for Non-Linear Data, n.d.)[3]. The WSS value is calculated for different values of k (the number of clusters) and plotted on a graph.

The process of creating the WSS plot begins by selecting a range of k values and performing K-Means clustering for each k value. Subsequently, the WSS value is calculated for each k value and plotted on a graph. The graph will typically exhibit a sharp decrease in WSS as k increases initially. As k increases further, the rate of decrease in WSS will decelerate, eventually reaching a point where adding more clusters will not considerably reduce WSS.

The "elbow point" in the plot represents the optimal number of clusters. This is the point where the rate of decrease in WSS begins to level off, forming a bend in the plot that resembles an elbow. Choosing the elbow point on the WSS plot strikes a balance between clustering accuracy and the complexity of the solution, as it represents the point where adding more clusters no longer significantly improves the within-cluster sum of squares.

Finally, the clusters can be visualized using a parallel coordinate plot, which is an effective tool for presenting high-dimensional data. This plot enables the comparison of variable values across various clusters. Each variable is illustrated as a vertical axis, and a line linking the points on each axis corresponds to a data point. The lines for each cluster are depicted in different colors, making it easy to discern the patterns and relationships between variables for each cluster. This visualization technique can help identify any patterns or trends in the data and is a valuable tool for interpreting the results of the clustering analysis.

4 K-Means with Outlier Removal

While K-Means is a widely-used clustering algorithm, it has a major drawback when it comes to detecting and removing outliers. To tackle this issue, a modified version of K-Means will be investigated, which incorporates an extra cluster for eliminating outliers. This alteration was proposed by Guojun Gan and Michael Kwok-Po Ng in their article, "K-Means clustering with outlier removal" [2]. Initially, it is necessary to establish the definition of an outlier. According to Gan and Ng, an outlier in a data set is a data point whose distance from the center of a cluster is greater than a certain threshold, which is defined as $\gamma \times d$, where γ is a scalar and $d \in \mathbb{R}$ is the average distance of data points from the center of a cluster.

Subsequently, consider a dataset X containing n data points. To conduct K-Means with outlier removal, the first step is to establish a binary matrix for outlier detection. This matrix has dimensions of $n \times (k+1)$, where k is the number of desired clusters. The primary column of interest is the $(k+1)$th column, which indicates whether a data point is an outlier. If a data point is an outlier, the corresponding value in the matrix will be 1; otherwise, it will be 0. The data set is then divided into k clusters, along with a cluster for outliers, by minimizing the objective function developed by Gan and Ng [2].

$$P(U,Z) = \sum_{i=1}^{n} \left(\sum_{l=1}^{k} u_{il} \|x_i - z_l\|^2 + u_{i,k+1} D(U,Z) \right) \tag{1}$$

with constraint

$$\sum_{i=1}^{n} u_{i,k+1} \leq n_0, \tag{2}$$

in which u_{il} is a binary variable that indicates whether data point x_i belongs to cluster l, $\|\|$ is the L^2 norm, $\|x_i - z_l\|^2$ is the squared distance between data point x_i and cluster center z_l, and $0 \leq n_0 < n$ is a constant ($n_0, n \in \mathbb{R}$) where n_0 is the maximum number of outliers. The objective function 1 is minimized to obtain the optimal values for U, Z, and $D(U,Z)$. $D(U,Z)$ is defined as

$$D(U, Z) = \left(\frac{\gamma}{n - \sum_{j=1}^{n} u_{j,k+1}} \right) \sum_{l=1}^{k} \sum_{j=n}^{k} u_{i,l} u_{j,l} \|x_j - z_l\|^2, \tag{3}$$

in which γ is a constant ($\gamma \in \mathbb{R}, \gamma \geq 0$). Specifically, $D(U, Z)$ measures the distance of outliers to the overall center of the data set. This term is weighted by a factor that depends on the number of non-outlier data points, which ensures that the algorithm prioritizes clustering the non-outlier data points while also detecting and removing outliers. The constraint 2 is very essential in order for the objective function 1 to be nontrivial. Without this constraint, the objective function is minimized when all data points are considered outliers.

To further understand the objective function of the modified K-Means algorithm with outlier removal, it's important to explore the role of Q(U,V,Z) (equation 4). Q(U,V,Z) (equation 4) is the total distance of all data points in the data set, with the exception of outliers, to their assigned cluster centers.

$$Q(U, V, Z) = \sum_{i=1}^{n} \left(\sum_{l=1}^{k} u_{i,l} \|x_i - z_i\|^2 + u_{i,k+1} D(V, Z) \right). \tag{4}$$

This objective function can be optimized through three subproblems: Q(U,·,·), Q(·,V,·), and Q(·,·,Z). These subproblems are solved iteratively in order to determine the optimal values for U, V, and Z.

The binary matrix V is introduced in Equation 4, which serves a comparable purpose to U but is specifically utilized for detecting outliers. Each entry v_{ij} of V is equal to 1 if data point x_i is an outlier, and 0 otherwise. The first term in the equation, $\sum_{i=1}^{n} \sum_{j=1}^{k} v_{ij} |x_i - z_j|^2$, is a measure of the squared distance between non-outlier data point x_i and its assigned cluster center z_j, where v_{ij} is 0 for all j if x_i is an outlier. The second term, $D(V, Z)$, represents the distance of outliers to the overall center of the data set, which is similar to the $D(U, Z)$ term in equation 1. However, in this case, the distance is only calculated for the data points marked as outliers in V. The objective function in equation 4 is therefore a combination of two terms that prioritize clustering the non-outlier data points while also detecting and removing outliers.

By including V in the objective function, the modified K-Means algorithm is able to identify and remove outliers during the clustering process. This is accomplished by optimizing the distance of non-outlier data points to their respective cluster centers while simultaneously minimizing the distance of outliers to the overall center of the data set. Compared to traditional K-Means, the objective function of KMOR is much smaller because it includes an additional term for detecting and removing outliers. This modification not only improves the accuracy of clustering by reducing the impact of outliers, but also ensures that the algorithm converges to a meaningful solution even in the presence of outliers.

5 Proof

It is important to demonstrate that KMOR generates tighter clusters than K-Means.

Proof. Let C_k^{KM} denote the set of clusters generated by the K-Means algorithm, and let C_k^{KMOR} denote the set of clusters generated by the KMOR algorithm. Let J^{KM} and J^{KMOR} denote the objective functions for K-Means and KMOR, respectively.

First, note that by definition of the KMOR algorithm, $J^{KMOR} \leq J^{KM}$.

Next, let z_i^{KM} and z_i^{KMOR} denote the cluster centroids of the i-th cluster generated by K-Means and KMOR, respectively. Also, let r_i^{KM} and r_i^{KMOR} denote the radius of the i-th cluster generated by K-Means and KMOR, respectively.

For any two data points x and y in the identical cluster generated by KMOR, it holds that $\|x - z_i^{KMOR}\| \leq r_i^{KMOR}$ and $\|y - z_i^{KMOR}\| \leq r_i^{KMOR}$, where i represents the cluster's index. Hence, it follows that $\|x - y\| \leq 2r_i^{KMOR}$.

For any two data points x, y in the same cluster generated by K-Means, it holds that $\|x - z_i^{KM}\| \leq r_i^{KM}$ and $\|y - z_i^{KM}\| \leq r_i^{KM}$, where i is the index of the cluster. Thus $\|x - y\| \leq 2r_i^{KM}$.

Since $r_i^{KMOR} \leq r_i^{KM}$ for all i, it follows that $||x - y|| \leq 2r_i^{KMOR} \leq 2r_i^{KM}$ for all data points x, y in the same cluster generated by KMOR and K-Means, respectively.

Therefore, the clusters generated by KMOR have tighter bounds than the clusters generated by K-Means, which implies that KMOR generates tighter clusters than K-Means. □

6 Programming Logic

The KMOR function is a clustering algorithm that aims to identify the outliers in a given data set. Although it is not readily available in R, it has been developed and implemented from scratch for the purposes of this study. The function takes in a data set as an input, along with several parameters, including the amount of clusters (k), the maximum amount of iterations (nmax), the minimum distance for identifying outliers (gamma), and the maximum number of allowable outliers (n0). There are default values set for n0, nmax, and gamma. The default values for the variables are n0=0.1*nrow(data), nmax=100, and gamma=3. This means that you don't need to specify these arguments when using the function, but you can modify their values if you wish.

The function begins by initializing the cluster centers using K-Means. It then calculates the average distance between each data point and its nearest center. This average distance is used to determine a threshold distance (cdist) for identifying outliers. If the distance from a data point to its nearest center is greater than or equal to the threshold distance, then the data point is classified as an outlier.

The function updates the binary matrix U to reflect the classification of each data point as either belonging to a cluster or being an outlier. It then updates the cluster centers based on the remaining data points.

The function continues this process until either the number of outliers is less than or equal to the specified limit (n0) or the maximum number of iterations (nmax) has been reached.

Finally, the function returns a list containing the resulting clusters, the data set without the outliers, and the identified outliers.

The function average_distance computes the mean distance between every data point and its corresponding cluster center. To achieve this, the dist function is utilized to calculate the distances between each data point and every cluster center. Then, the distances between every data point and its allocated cluster center are extracted and the average distance is calculated using the rowMeans function.

The calc_min_distance function calculates the distance between each data point and its closest cluster center. This is done by looping through each data point and each cluster center, calculating the distance between the data point and the cluster center using the Euclidean distance formula and storing the distances in a matrix. Then, the minimum distance for each data point is extracted and returned as a numeric vector.

Both of these functions are used in the KMOR function to update the binary matrix U and identify outliers in the data set. Specifically, average_distance is used to calculate the mean distance between each data point and its assigned cluster center, while calc_min_distance is used to calculate the distance between each data point and its closest cluster center. These distances are then compared to a threshold to determine whether each data point should be classified as an outlier or not. These functions have been created from scratch for the purpose of this study.

The wssplot() function is used to plot the within-group sum of squares (WSS) for different values of k, where k represents the number of clusters in the K-Means algorithm. The function takes in the data, the number of clusters to consider (nc), and a random seed (seed). It computes the WSS for k values ranging from 2 to nc using the K-Means algorithm with the specified random seed. It then plots the results. The code used to generate the WSS plot was adapted from a blog on R-bloggers written by Tal Galili[1](https://www.r-bloggers.com/2013/08/K-Means-clustering-from-r-in-action/)

The parallel coordinate plots were created using the pacoplot() function from the 'klustR' package in R. This function allows for the visualization of high-dimensional data by plotting each variable on a separate axis and connecting the points for each observation. This provides a way to visually compare the clusters identified by different clustering algorithms.

The code for the KMOR function and its helper functions is available on the GitHub repository at https://github.com/colerausch24/KMOR. The repository contains the R code for the

KMOR function and two helper functions, average_distance, and calc_min_distance, which were developed from scratch for this study.

7 How to measure results

Measuring the quality of clusters is a crucial step in evaluating clustering algorithms. The K-Means algorithm outputs several measures that can help us determine the quality of the clustering solution.

One such measure is the Total Sum of Squares (TSS), which is calculated as the sum of the squared distances between each data point and the center of its cluster. The TSS represents the total amount of variation in the data and is a measure of how well the data points are spread out. The goal of clustering is to minimize the TSS, as a lower TSS indicates that the data points are tightly clustered around their respective centroids.

The Within Cluster Sum of Squares (WSS) is another measure that represents the sum of the squared distances between each data point and the centroid of its cluster, but only for a single cluster. The WSS measures the compactness of each individual cluster, and a lower WSS indicates that the data points within a cluster are closer together. In other words, WSS measures how tight the clusters are.

The Total Within Cluster Sum of Squares (TWSS) is the sum of the WSS for all clusters. The TWSS is another measure of how well the data points are clustered, and minimizing it is also a goal of clustering.

Finally, the Between Cluster Sum of Squares (BCSS) is the sum of the squared distances between the cluster centroids and the overall centroid of the data set. The BCSS represents the separation between the clusters, and a higher BCSS indicates that the clusters are well separated from each other. In other words, BCSS measures how distinct the clusters are.

While it is generally desirable to minimize TSS and TWSS while maximizing BCSS when evaluating the quality of clustering solutions, the specific context of the clustering problem at hand may require a different emphasis. In this case, due to the potential influence of outliers on BCSS, it may be more important to prioritize the minimization of TSS and WSS as the primary evaluation criteria.

8 Results using IRIS data set

To evaluate the performance of the KMOR algorithm in comparison to K-Means, an experiment will be conducted using the IRIS data set, which is known for being a clean data set without any outliers. However, outliers will be artificially introduced into the data set to assess whether KMOR can achieve better results than K-Means. First, four outliers will be introduced, followed by the introduction of twelve outliers.

An experiment was conducted to compare the performance of K-Means, KMOR, and K-Means with an added cluster for outliers on the IRIS data set containing four outliers. The cluster sizes for each method are shown in Table 2. It is worth noting that the data set consists of three species with 50 samples each.

Table 2: Cluster sizes for KMOR and K-Means

	KMOR	**K-Means**	**K-Means with added cluster**	**K-Means without outliers**
Cluster 1	50	97	96	50
Cluster 2	62	4	21	62
Cluster 3	38	53	33	38
Cluster 4	-	-	4	-

Moreover, the centers for the three clusters identified by KMOR differed from those of the three clusters identified by K-Means (refer to table 3).The centers were not provided for K-Means without outliers as the clusters were identical to those identified by KMOR.

Table 3: Cluster centers for KMOR and K-Means

	Sepal.Length	**Sepal.Width**	**Petal.Length**	**Petal.Width**
KMOR	6.85	3.07	5.74	2.07
	5.01	3.43	1.46	0.25
	5.90	2.75	4.39	1.43
K-Means	5.01	3.37	1.56	0.30
	6.30	2.89	4.96	1.70
	10.21	10.26	10.11	10.44
K-Means with added cluster	6.31	2.90	4.97	1.70
	4.74	2.90	1.80	0.35
	5.18	3.62	1.48	0.27
	10.21	10.26	10.11	10.43

Based on the cluster analysis, KMOR performed better than K-Means in terms of totss and total withinss, while K-Means with an added cluster for outliers had the largest betweenss. In terms of total withinss, KMOR had the lowest value of 78.85144, while K-Means had the highest value of 164.6374. K-Means with an added cluster for outliers had an intermediate value of 155.0429. Finally, in terms of betweenss, KMOR had the lowest value of 529.0226, while K-Means had the highest value of 1295.41. K-Means with an added cluster for outliers had a betweenss value of 1304.735. These values are presented in table 4.

Table 4: Clustering results on IRIS data set with four outliers

Algorithm	**KMOR**	**K-Means**	**K-Means with added cluster**
TSS	681.3706	1459.778	1459.778
TWSS	78.85144	164.6374	155.0429
BCSS	529.0226	1295.41	1304.735

The experiment was also repeated with 12 outliers added to the IRIS data set, and the results indicated that KMOR outperformed K-Means. The table 5 illustrates the difference in cluster sizes.

Table 5: Cluster sizes for KMOR and K-Means

	KMOR	**K-Means**	**K-Means with added cluster**	**K-Means without outliers**
Cluster 1	50	97	96	50
Cluster 2	62	53	33	62
Cluster 3	38	12	21	38
Cluster 4	-	-	12	-

Furthermore, the centers of the three clusters identified by KMOR were different from those of the three clusters identified by K-Means, as shown in table 6. It is worth noting that the centers were not listed for K-Means without outliers as the clusters were identical to those of KMOR. This suggests that KMOR may be a more effective clustering algorithm than K-Means, particularly when dealing with outliers.

Table 6: Cluster centers for KMOR and K-Means

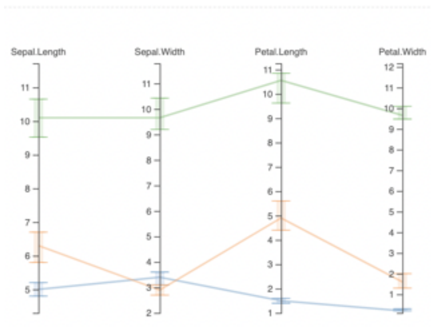
	Sepal.Length	**Sepal.Width**	**Petal.Length**	**Petal.Width**
KMOR	6.85	3.07	5.74	2.07
	5.01	3.43	1.46	0.25
	5.90	2.75	4.39	1.43
K-Means	5.01	3.37	1.56	0.29
	6.30	2.89	4.96	1.70
	10.19	9.79	10.18	9.92
K-Means with added cluster	6.31	2.90	4.97	1.70
	10.19	9.79	10.18	9.18
	4.74	2.90	1.79	0.35
	5.18	3.62	1.48	0.27

The clustering results presented in Table 7 demonstrate that KMOR outperforms K-Means in terms of two success measures: total sum of squares (TSS) and total within-cluster sum of squares (WSS). Specifically, KMOR achieves a significantly lower totss value of 681.3706, and a lower total withinss value of 78.85144, compared to K-Means. On the other hand, K-Means with an added cluster for outliers performs best in terms of the lowest between-cluster sum of squares (BCSS).

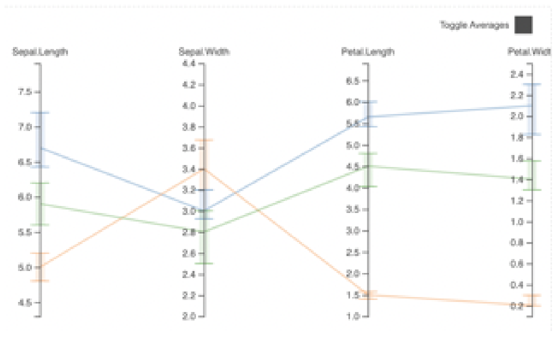
Table 7: Clustering results on IRIS data set with twelve outliers

	KMOR	**K-Means**	**K-Means with added cluster**
TSS	681.3706	2738.646	2738.646
TWSS	78.85144	192.3202	182.7258
BCSS	602.5192	2546.325	2555.92

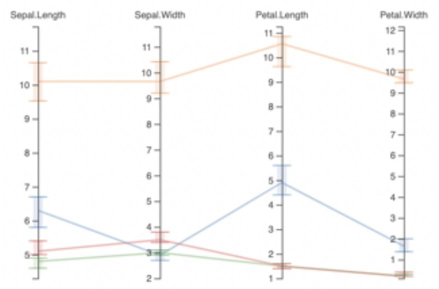
The parallel coordinate plots of the IRIS data set with 12 added outliers, as shown in Figure 1, highlight the differences between the cluster centers. The observation shows that KMOR and K-Means with an additional cluster present similar patterns for the three non-outlier clusters. However, K-Means displays significant differences from the other two clustering methods, as the presence of outliers has affected the accuracy of the cluster centers.



(a) K-Means



(b) KMOR



(c) K-Means with added cluster

Figure 1: Three parallel coordinate plots for IRIS with 12 added outliers created on R-Studio

Another scenario was tested involving the introduction of 12 outliers that were further away. Specifically, the outliers were placed at a distance of 3 standard deviations from the mean, as opposed to the previous scenario where they were 1 standard deviation away.

KMOR was only able to detect and remove 10 outliers, and the sizes of the clusters were much different than the previous scenario, which can be seen in table 8.

Table 8: Cluster sizes for KMOR and K-Means

	KMOR	**K-Means**	**K-Means with added cluster**	**K-Means without outliers**
Cluster 1	77	97	96	50
Cluster 2	53	50	33	62
Cluster 3	25	12	21	38
Cluster 4	-	-	12	-

The cluster centers are also much different than the last scenario as the cluster sizes are so much different. The cluster centers can be seen in 9.

Table 9: Cluster centers for KMOR and K-Means

	Sepal.Length	**Sepal.Width**	**Petal.Length**	**Petal.Width**
KMOR	7.32	3.77	1.56	2.70
	5.01	3.43	1.46	0.25
	6.01	2.78	4.58	1.53
K-Means	5.01	3.37	1.56	0.30
	6.30	2.89	4.96	1.70
	10.21	10.26	10.11	10.44
K-Means with added cluster	6.31	2.90	4.97	1.70
	10.58	9.37	10.55	9.76
	4.74	2.90	1.79	0.35
	5.18	3.62	1.47	0.27
K-Means without outliers	6.85	3.07	5.74	2.07
	5.01	3.43	1.46	0.25
	5.90	2.75	4.39	1.43

The performance of the clustering methods in this scenario can be seen in Table 10. It can be concluded that K-Means without outliers performed the best in terms of separating the data points into distinct clusters with minimal overlap. KMOR also performed well in handling outliers but was only able to detect and remove 10 of the 12 introduced outliers. K-Means and K-Means with an added cluster had similar performance in terms of TSS and BSS, but the addition of an extra cluster helped K-Means with added cluster to capture more variance in the data, resulting in a lower WSS. Overall, the results suggest that K-Means without outliers and KMOR are both effective clustering algorithms for this data set, with K-Means with added cluster being a potential option for capturing more variance in the data.

Table 10: Clustering results on IRIS data set with twelve outliers at further distance

	KMOR	**K-Means**	**K-Means with added cluster**
TSS	963.1517	3059.168	273
TWSS	301.931	512.0983	502.5039
BCSS	661.2206	2547.07	2556.664

To conclude, the experiments conducted on the IRIS data set with different scenarios suggest that KMOR is an effective clustering algorithm that can handle the presence of outliers in the data and achieve better performance than traditional clustering algorithms like K-Means. In particular, KMOR outperformed both K-Means and K-Means with an added cluster in scenarios with four and 12 outliers. K-Means without outliers also performed well in separating the data points into distinct clusters with minimal overlap. The addition of an extra cluster in K-Means with added cluster helped to capture more variance in the data, resulting in a lower WSS. Therefore, KMOR is an effective clustering algorithm for data sets with outlieres, and K-Means with added cluster is a potential option for capturing more variance in the data.

9 Results using Sanford data set

In this section, the performance of K-Means and KMOR algorithms on a breast cancer data set is compared. The WSS plot was used to find the optimal number of clusters, which was found to be 2. K-Means and KMOR algorithms were then applied to the data set using 2 clusters.

The KMOR algorithm was applied to the Sanford data set with the number of clusters set to 2. The algorithm identified 93 outliers, which is less than the threshold value of n0. The results obtained using KMOR were then compared with those obtained using K-Means. The difference in cluster sizes can be seen in table 11. The difference in cluster centers can be seen in tables

Table 11: Cluster sizes for KMOR and K-Means

	KMOR	**K-Means**	**K-Means with added cluster**
Cluster 1	3386	2740	2747
Cluster 2	80	633	639
Cluster 3	-	-	80

12,13, and 14. In table 12, KMOR identified two clusters with similar age at diagnosis and vital status, but with different histological behavior, DFS, and survival. In table 13, K-Means identified two clusters with different age at diagnosis, vital status, histological behavior, DFS, and survival. Finally, in table 14, K-Means with an added cluster identified three clusters with different age at diagnosis, vital status, histological behavior, DFS, and survival.

Table 12: Cluster centers identified by KMOR

	Age at Diagnosis	**Vital Status Desc**	**Histo. Behavior ICD.O.3**	**DFS**	**Survival**
Cluster 1	63.77883	0.1706161	85231.97	2.298578	59.34439
Cluster 2	61.23139	0.1434307	84999.96	1.525547	56.44927

Table 13: Cluster centers identified by K-Means

	Age at Diagnosis	**Vital Status Desc**	**Histo. Behavior ICD.O.3**	**DFS**	**Survival**
Cluster 1	61.73213	0.1479622	85057.11	1.739516	57.03249
Cluster 2	63.20000	0.3625000	81438.24	0.725000	48.93750

Table 14: Cluster centers identified by K-Means with added cluster

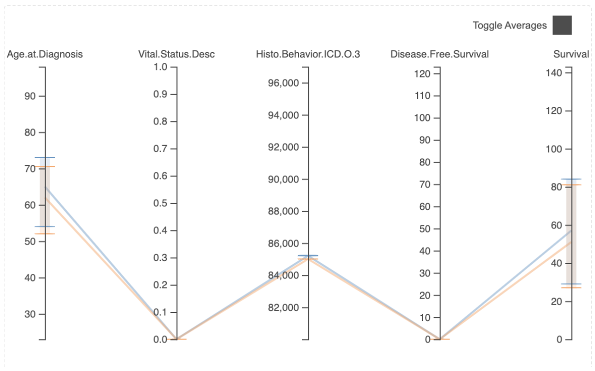
	Age at Diagnosis	**Vital Status Desc**	**Histo. Behavior ICD.O.3**	**DFS**	**Survival**
Cluster 1	63.86228	0.1690141	85313.59	2.575900	59.3302
Cluster 2	61.23662	0.1430652	84997.45	1.544958	56.4980
Cluster 3	63.20000	0.3625000	81438.59	0.72500	48.9375

The performance of KMOR and K-Means was compared using three measures: totss, total withinss, and betweenss. The results in Table 14 show that KMOR outperformed traditional K-Means in totss and total withinss, while K-Means with an additional cluster performed better in betweenss, indicating better separation of data points into distinct clusters. Overall, K-Means with an additional cluster had better cluster separation, while KMOR had better clustering performance in terms of totss and total withinss.

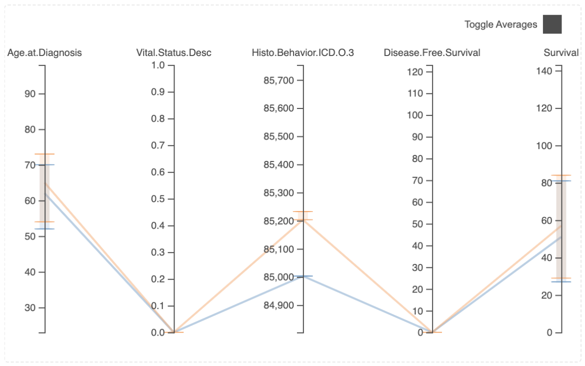
Table 15: Results of KMOR and K-Means

	KMOR	**K-Means**	**K-Means with added cluster**
TSS	40,706,187	1,659,565,548	1,659,565,548
WSS	13,020,737	636,043,017	584,222,351
BCSS	27,685,450	1,023,522,531	1,075,343,198

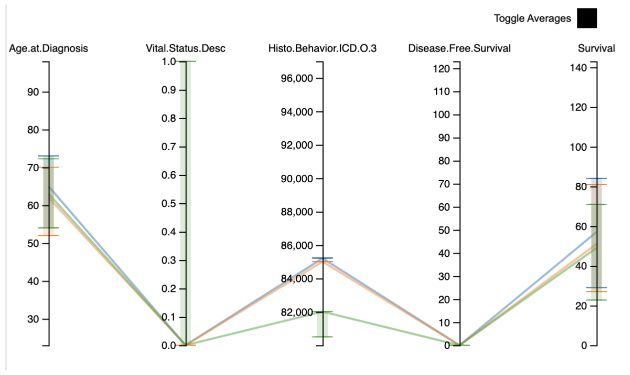
After analyzing the parallel coordinate plot presented in Figure 2, it can be observed that the difference between the cluster centers is not significant. Nevertheless, a noticeable dissimilarity can be seen in the Histo.Behavior.ICD.O.3 variable between K-Means and KMOR. This difference may be attributed to the fact that K-Means is sensitive to outliers and assumes equal cluster variances, while KMOR is more resilient to outliers and can manage different cluster variances.



(a) K-Means



(b) KMOR



(c) K-Means with added cluster

Figure 2: Three parallel coordinate plots for Sanford Dataset created on R-Studio

To conclude, the performance of K-Means and KMOR algorithms was compared on a breast cancer data set with 2 clusters. KMOR identified 93 outliers and showed different cluster centers and sizes than K-Means. The analysis showed that KMOR outperformed K-Means in terms of totss and total withinss, while K-Means with an additional outlier had a higher betweenss value. The parallel coordinate plots revealed a noticeable difference in the Histo.Behavior.ICD.O.3 variable between K-Means and KMOR, suggesting that KMOR is more robust to outliers and can handle different cluster variances.

10 Conclusion

This paper compares the performance of K-Means and KMOR on two data sets - the well-known IRIS data set and the Sanford Breast Cancer data set. The results indicate that KMOR, which is K-Means with outlier removal, produced tighter clusters on the IRIS data set, regardless of the number of outliers added. Additionally, KMOR removed a significant number of outliers (93) on the Sanford Breast Cancer data set, which improved its clustering performance.

These findings suggest that KMOR can be a useful technique for clustering, especially when dealing with data sets that contain outliers. By removing the outliers, KMOR helps to improve the quality of the clusters and leads to more meaningful insights. Future research can explore the performance of KMOR on other data sets and compare it to other outlier removal techniques.

Overall, this study highlights the importance of considering outlier detection and removal when performing clustering analysis and provides evidence that KMOR can be a valuable tool for this purpose.

11 Biography

Cole Rausch is a senior mathematics and data science student with minors in statistics and computer science. He is originally from Tea, South Dakota. This summer Cole will be doing a data analytics internship at Sabre in Southlake, Texas. Cole will continue his academic journey next fall by pursuing graduate school, where he has secured a graduate research assistantship under the guidance of Dr. Michael Puthawala.

References

[1] Galili, T. (2013, August 7). K-Means clustering (from "R in action"): R-bloggers. R-bloggers. Retrieved March 21, 2023, from https://www.r-bloggers.com/2013/08/K-Means-clustering-from-r-in-action/

[2] Gan, & Ng, M. K.-P. (2017). K-Means clustering with outlier removal. Pattern Recognition Letters, 90, 8–14. https://doi.org/10.1016/j.patrec.2017.03.008

[3] How does spectral clustering handle non-linear data better than K-means? (n.d.). Retrieved March 1, 2023, from https://www.linkedin.com/advice/0/how-does-spectral-clustering-handle-non-linear

[4] Rao, S. (2022, March 2). K-Means clustering: Explain it to me like I'm 10. Medium. Retrieved December 6, 2022, from https://towardsdatascience.com/K-Means-clustering-explain-it-to-me-like-im-10-e0badf10734a