Replication

Applied Stats II

Due: March 31, 2024

A new algorithm for initial cluster centers in k-means algorithm

Clustering is the process of grouping similar data points together. The ultimate goal is to make groups (clusters) where things inside the group are alike, and things between different groups are very different. Clusters are classified into two broad types.The first one is hierarchical that creates a tree-like structure of clusters while the second one is non-hierarchical which makes a simple division of data into a set number of groups.

k-means is a common non-hierarchical clustering method in which the number of clusters are developed typically on the basis of data type and user input. However, the problem arises when the outcomes obtained using kmeans clustering are heavily influenced by initial selection of cluster centers, as the k-means results depend a lot on where the clustering process (the initial cluster centers) has been started. Bad starting points can lead to poor results. therefore, finding good starting points (initial cluster centers) are essential for getting meaningful results. Several efforts have been made in this regard to resolve this initialization of cluster problem as follows:

Method	Approach	Key Idea
Duda and Hart (1973)	Multiple Random Starts	Run k-means multiple times with random starting points, hope one run produces a good clustering result.
Jain and Dubes (1988)	Best of Multiple Runs	Run k-means many times with random starts. Then pick the clustering result with the lowest error and use that as your final clustering.
Bradley and Fayyad (1998)	Subsampling	Break the data into smaller samples, cluster those using k-means. Combine the results of these smaller clusters to seed the clustering of the entire dataset.
Likas et al. (2003)	Global K-means	Incremental approach. Start with one cluster center and gradually add more, carefully choosing the best position for each new center.
Khan and Ahmad (2004)	Cluster center initialization algorithm (CCIA)	Analyze the overall shape of the data to find dense areas. These dense areas are likely to be good starting points for clusters.
Deelers and Auwatanamongkol (2007)	Grid-Based	Divide the data into grid-like cells. Use the center point of each cell as an initial cluster center.

Murat et. al. (2011) has proposed an algorithm to tackle this problem of initializing cluster centers. The algorithm considers all the data variables and the variable with highest coefficient of variation is considered as ”main axis”. Next, it finds a second axis by looking at data variables that exhibits lowest correlation that is as perpendicular to the main axis as possible. The idea is to capture the areas where the data changes the most.

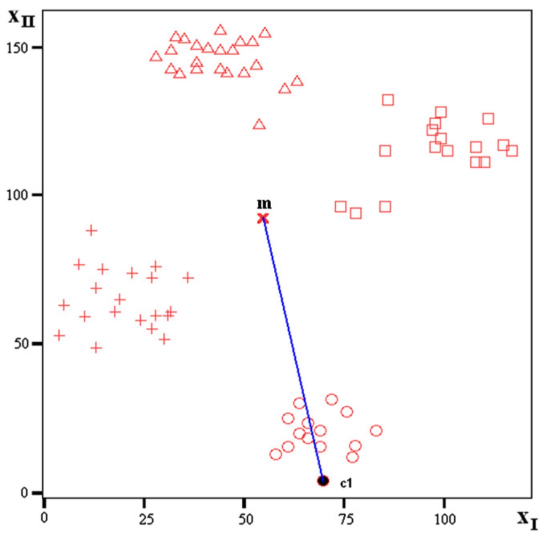


Fig. 1. Selection for first candidates of the initial cluster center.

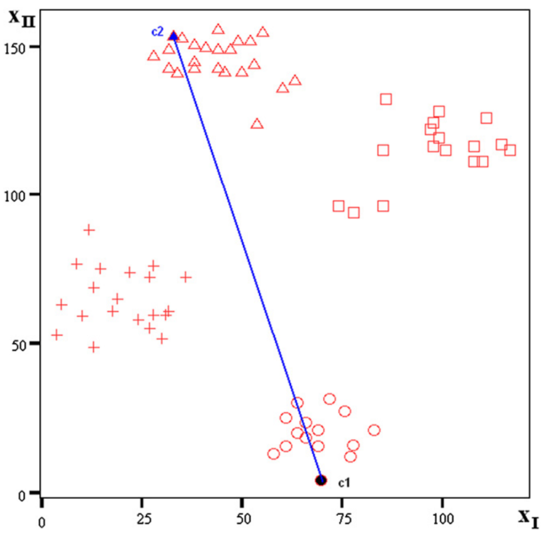


Fig. 2. Selection for second candidates of the initial cluster center.

After setting up both the axes, the algorithm attempts to calculate the mean of all data points along these two axes. The data point that's furthest away from this mean, becomes first cluster center. Similarly, the algorithm calculates distances from the previously found centers to every data point iteratively for each additional cluster center, keeping track of the total distance each data point has from all previous centers and the data point with the largest total distance becomes the next candidate for a cluster center. This process repeats until the desired number of center candidates generated. By carefully choosing initial cluster centers based on variation and avoiding clustering together, this approach aims to improve the quality and meaningfulness of the final clusters produced by k-means.

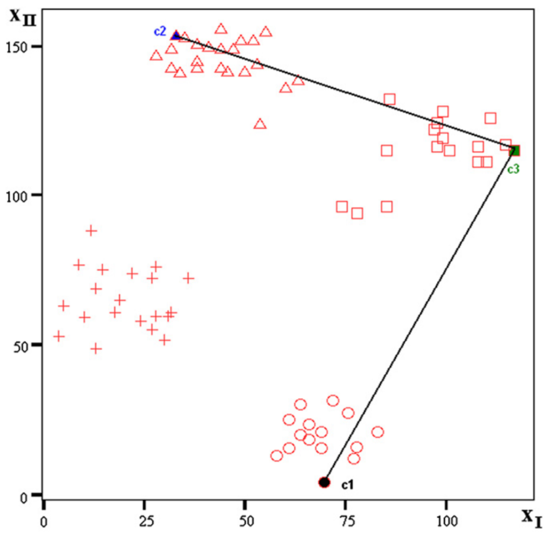


Fig. 3. Selection for third candidates of the initial cluster center.

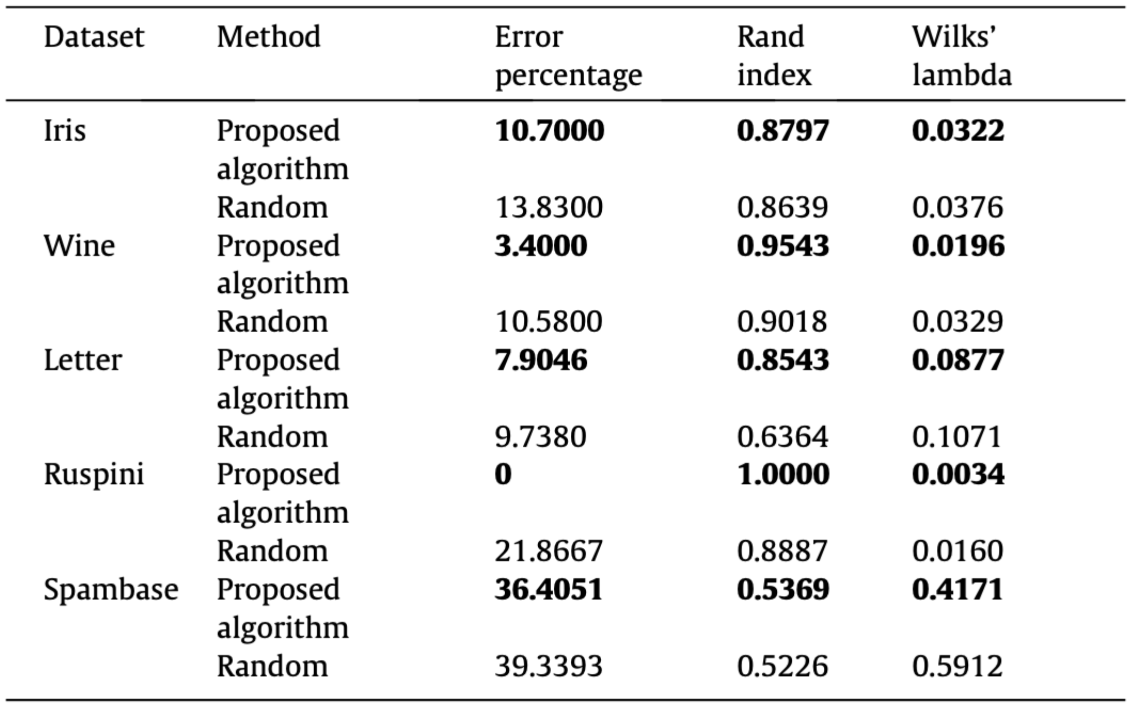
To assess the quality of these clusters, Error percentage, Rand index and Wilks' lambda test statistics are used. Error percentage tells the percentage of data points that got put into the wrong cluster compared to their "true" category through counting the misclassified points and divide by the total number of data points..A low error percentage means our clustering did a good job.

Similarly, Rand index shows the agreement between the created clusters and the "real" clusters the data should belong to. It looks at pairs of data points and checks, were the pair put in the same cluster by both our algorithm and the real data? were they put in different clusters by both? did our algorithm put them together, but they really belong in different clusters? and did our algorithm separate them, but they really belong together? A Rand index close to 1 means our clustering was very similar to the true structure of the data.

Finally, Wilks' lambda demonstrates the difference between the clusters in a statistical sense. It involves comparing how spread out data points are within each cluster vs. how spread out the clusters are overall. Small Wilks' lambda means the clusters are distinct; data points within a cluster are close together while the clusters themselves are far apart.

The researchers performed the three statistical tests on five publicly available datasets that are the iris dataset, the letter image recognition data, the spambase dataset, the wine recoginition data and the Ruspini data to compare their proposed algoritm with traditional k-means random initialization method. The results of their comparison are as follows:

Comparison results between proposed algorithm and random initial centers according to error percentage, Rand index and Wilks' lambda test statistic.



Dataset	Method	Error percentage	Rand index	Wilks' lambda
Iris	Proposed algorithm	**10.7000**	**0.8797**	**0.0322**
	Random	13.8300	0.8639	0.0376
Wine	Proposed algorithm	**3.4000**	**0.9543**	**0.0196**
	Random	10.5800	0.9018	0.0329
Letter	Proposed algorithm	**7.9046**	**0.8543**	**0.0877**
	Random	9.7380	0.6364	0.1071
Ruspini	Proposed algorithm	**0**	**1.0000**	**0.0034**
	Random	21.8667	0.8887	0.0160
Spambase	Proposed algorithm	**36.4051**	**0.5369**	**0.4171**
	Random	39.3393	0.5226	0.5912

The results in terms of consistency among clusters improved as a result of using the proposed algorithm as compare to random initialization.

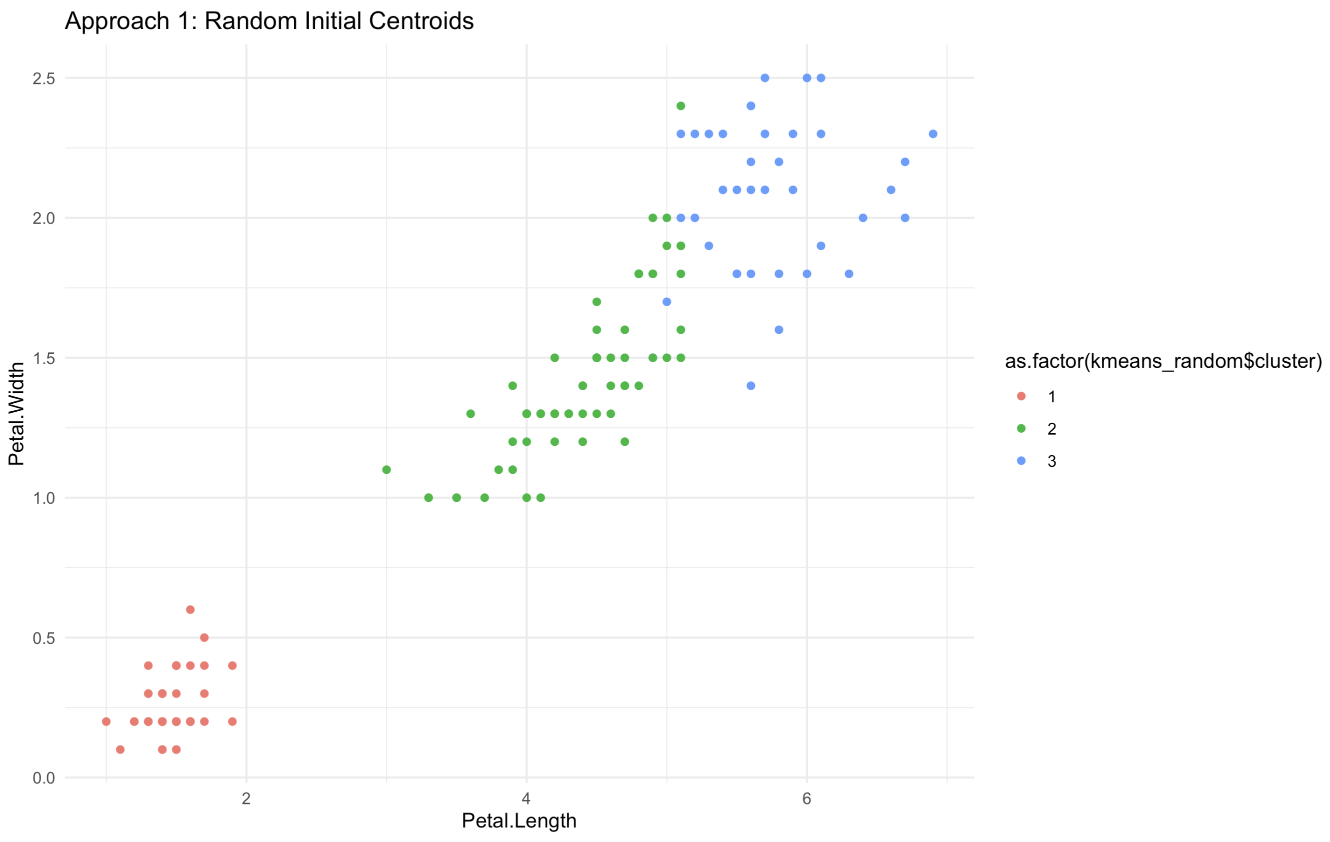
However, their claim cannot be verified when we run the R code to support the findings:

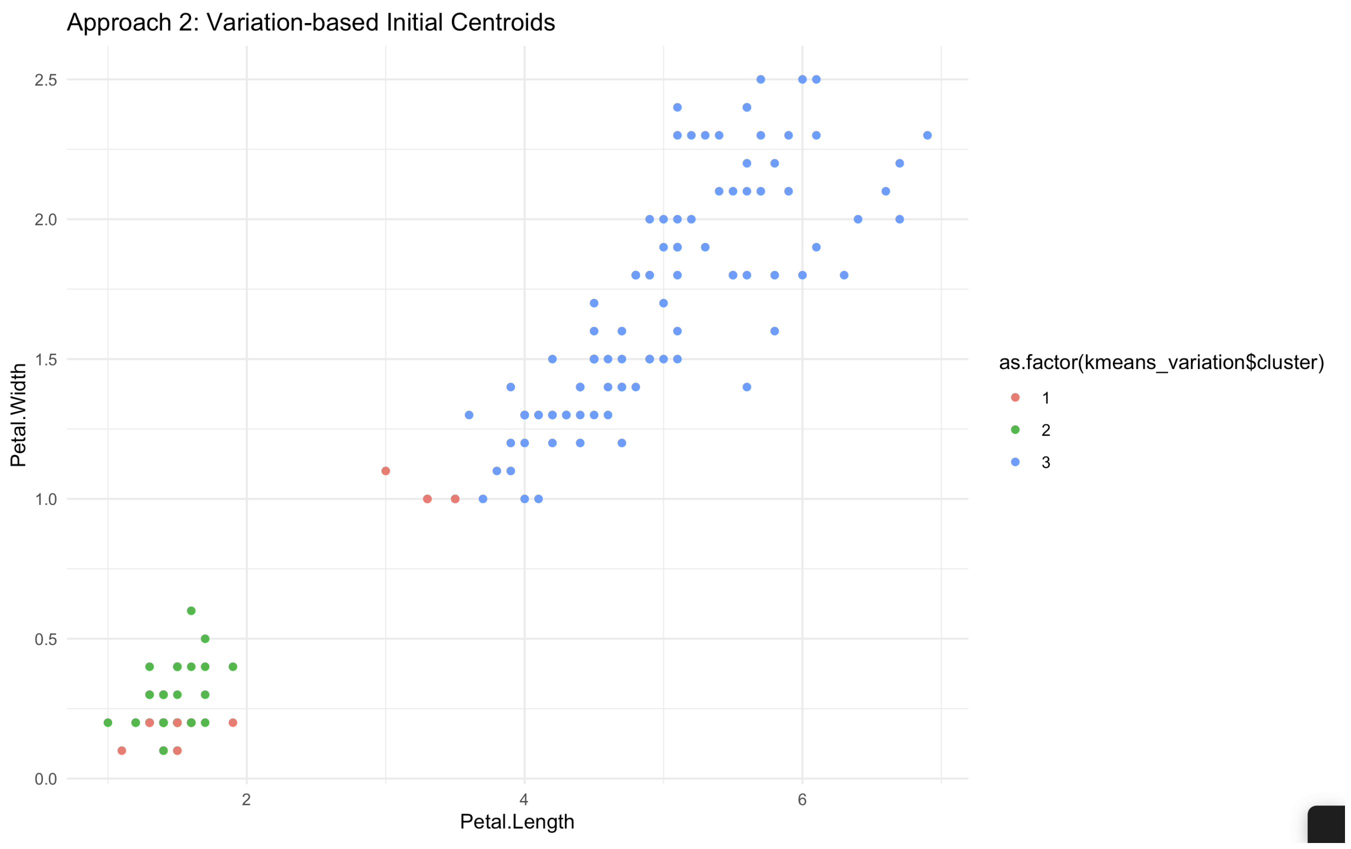
The R script for k-means clustering comparing the random initialization and proposed variation based method is:

```
# Load required libraries
library(cluster)
library(MASS)
library(ggplot2)

# Function to calculate error percentage
calculate_error_percentage <- function(clusters, labels) {
  error <- sum(clusters != labels)
  error_percentage <- (error / length(labels)) * 100
  return(error_percentage)
}

# Function to calculate the Rand index
calculate_rand_index <- function(clusters, labels) {
  n <- length(clusters)
  a <- sum(outer(clusters, clusters, "==") & outer(labels, labels, "==")) #
    Number of pairs with the same cluster and label
  b <- sum(outer(clusters, clusters, "!=") & outer(labels, labels, "!=")) #
    Number of pairs with different cluster and label
  rand_index <- (a + b) / choose(n, 2)  # Rand index calculation
  return(rand_index)
}

# Function to calculate Wilks' Lambda
calculate_wilks_lambda <- function(data, clusters) {
  lda_data <- cbind(data, clusters)  # Combine data with clusters
  lda_model <- lda(lda_data[, -ncol(lda_data)], clusters)
  wilks_lambda <- lda_model$svd^2 / sum(lda_model$svd^2)
  return(wilks_lambda)
}

# Load Iris dataset
data(iris)

# Set the seed for reproducibility
set.seed(123)

# Approach 1: Random initial centroids with iteration for optimal clustering
kmeans_random <- kmeans(iris[-5], centers = 3, nstart = 20)

# Calculate error percentage and Rand index for approach 1
error_percentage_random <- calculate_error_percentage(kmeans_random$cluster,
    as.numeric(iris$Species))
rand_index_random <- calculate_rand_index(kmeans_random$cluster, as.numeric(
```

```r
    iris$Species))

# Fit LDA model for approach 1 with random centroids
wilks_lambda_random <- calculate_wilks_lambda(iris[-5], kmeans_random$cluster)

# Approach 2: Initial centroids based on maximum variation and minimum correlation
variances <- apply(iris[-5], 2, var)  # Calculate variances for each feature
max_var_index <- which.max(variances)  # Index for maximum variation
min_corr_indices <- which(cor(iris[-5]) == min(cor(iris[-5])))  # Indices for minimum correlation

# Ensure the indices are distinct
if (max_var_index %in% min_corr_indices) {
  min_corr_indices <- min_corr_indices[min_corr_indices != max_var_index]
}

# Initialize centroids based on selected indices
centroids <- iris[c(max_var_index, min_corr_indices), -5]

# Shuffle the order of the data points
shuffled_indices <- sample(nrow(iris))
shuffled_iris <- iris[shuffled_indices, ]

# Perform k-means clustering with shuffled data and centroids based on maximum variation and minimum correlation
kmeans_variation <- kmeans(shuffled_iris[-5], centers = centroids)

# Calculate error percentage and Rand index for approach 2
error_percentage_variation <- calculate_error_percentage(kmeans_variation$cluster, as.numeric(shuffled_iris$Species))
rand_index_variation <- calculate_rand_index(kmeans_variation$cluster, as.numeric(shuffled_iris$Species))

# Fit LDA model for approach 2 with variation-based centroids
wilks_lambda_variation <- calculate_wilks_lambda(shuffled_iris[-5], kmeans_variation$cluster)

# Visualize the clustering results
# Approach 1: Random Initial Centroids
ggplot(iris, aes(Petal.Length, Petal.Width, color = as.factor(kmeans_random$cluster))) +
  geom_point() +
  labs(title = "Approach 1: Random Initial Centroids") +
  theme_minimal()

# Approach 2: Variation-based Initial Centroids
ggplot(shuffled_iris, aes(Petal.Length, Petal.Width, color = as.factor(kmeans_variation$cluster))) +
  geom_point() +
  labs(title = "Approach 2: Variation-based Initial Centroids") +
  theme_minimal()

# Print the results
cat("Approach 1 (Random Initial Centroids):\n")
cat("Error Percentage:", error_percentage_random, "%\n")
cat("Rand Index:", rand_index_random, "\n")
cat("Wilks' Lambda:", wilks_lambda_random, "\n\n")

cat("Approach 2 (Variation-based Initial Centroids):\n")
cat("Error Percentage:", error_percentage_variation, "%\n")
cat("Rand Index:", rand_index_variation, "\n")
cat("Wilks' Lambda:", wilks_lambda_variation, "\n")
```

The results generated using random initialization metnod as a result of this code suggests an Error percentage of 10.67% , Rand index of 1.77 and Wilks lamda of 0.97 while the proposed algorithm on the basis of variation generated much higher Error percentage of 55.33%, a lower Rand index of 1.45 and approximately the same Wilks' lambda of 0.95. The clustering images shown below further supports our claim that clusters created using the traditional random initialization method shows better cluster formation as compare to the variation based algorithm proposed by the authors





Moreover, other test statistics such as Silhouette score and Dunn index may further be incorporated in the analysis to support the findings.