Learning K-Means Clustering with R: A Step-by-Step Tutorial

Authored by

Mohammed looti

November 6, 2025

Clustering stands as a cornerstone technique within the field of machine learning. Its core purpose is to identify and delineate inherent structures, or natural groupings known as **clusters**, among a collection of data observations. Unlike supervised methods, clustering operates without prior knowledge of labels, focusing purely on the intrinsic relationships between data points.

The fundamental objective of any clustering algorithm is to optimally partition the dataset. This partitioning seeks to maximize the similarity (homogeneity) among observations placed within the same cluster, while simultaneously ensuring maximal dissimilarity (heterogeneity) between observations assigned to different clusters. Effectively, the goal is to create tight, meaningful groups that are clearly distinguishable from one another.

Because this process focuses solely on finding hidden structure and patterns in the data without relying on predefined output labels or target variables, clustering is firmly classified as a form of unsupervised learning. This characteristic makes the technique immensely valuable across diverse industries, particularly for critical tasks such as market segmentation, anomaly detection, and complex pattern recognition.

In the realms of marketing analytics and business intelligence, clustering is an indispensable tool frequently deployed to analyze large volumes of customer data. By analyzing multiple data attributes concurrently, companies can identify distinct, actionable market segments. Attributes commonly leveraged in such analyses include:

Household income and detailed spending habits.
Key demographic factors, such as household size and age profiles.
The occupation of the head of household, providing insight into lifestyle.
Geographic indicators, like proximity or distance from the nearest major urban area.

By segmenting populations based on shared characteristics, businesses can develop highly tailored product offerings and customized advertising campaigns, drastically enhancing outreach effectiveness and return on investment. Among the various clustering methods available, **K-means clustering** remains one of the most widely implemented and computationally efficient algorithms.

The Core Principles of K-Means Clustering

K-means clustering is an iterative partitioning algorithm known for its elegant simplicity and computational speed. The fundamental objective is to partition a dataset containing N observations into K predetermined, non-overlapping subsets or clusters, where K is a parameter specified by the user. The success of the algorithm is measured by its ability to minimize the total variance within each cluster, resulting in groups that are as compact and cohesive as possible.

The method achieves this optimal partitioning through an iterative reassignment process.

Mathematically, the algorithm is designed to minimize the total Within-Cluster Sum of Squares (WSS) across all *K* clusters. Minimizing WSS is equivalent to ensuring that the distance between each data point and the **centroid** of its assigned cluster is as small as possible.

The practical implementation of the K-means clustering algorithm follows a clear, sequential procedure involving three main stages that repeat until stability is achieved:

Selection of *K*: Defining the Number of Clusters. The first critical step requires the user to specify *K*, the desired number of clusters. Since the optimal *K* is rarely known beforehand, various diagnostic techniques--such as the Elbow Method or the Gap Statistic--are typically employed to select a statistically meaningful value before the main clustering routine is executed.

Initialization of Clusters: Setting the Starting Points. Initially, observations are randomly assigned to one of the *K* clusters, or *K* initial cluster centers are randomly placed. Crucially, the outcome of K-means can be sensitive to these starting positions. To mitigate this variability and ensure convergence to a more globally optimal solution, the algorithm is often run multiple times (controlled by the `nstart` parameter), retaining the solution with the lowest WSS.

Iteration and Refinement: The Assignment-Update Cycle. The algorithm performs the following two sub-steps repeatedly until the cluster assignments no longer change significantly (i.e., stability is achieved):

Compute the Centroid: For each temporary cluster, the cluster **centroid** is calculated. The centroid is simply the vector representing the mean feature values of all observations currently assigned to that cluster.
Reassign Observations: Every single observation in the dataset is reassigned to the cluster whose centroid is closest to it. The standard metric used to define "closest" is the squared Euclidean distance.

Applying K-Means Clustering in R: A Step-by-Step Tutorial

To solidify our theoretical understanding, we will now execute a comprehensive, step-by-step example of **K-means clustering** using the robust capabilities of the R programming language. This practical demonstration employs standard R functions and recommended packages, establishing a clear and reproducible analytical workflow.

Step 1: Installing and Loading Essential R Packages

For both the efficient execution of the clustering routine and the effective visualization of our results, we rely on two specialized R packages. We will load the **factoextra** package, which simplifies visualization and optimal cluster determination, and the **cluster** package, which provides

key utility functions necessary for various partitioning methods.

```
library(factoextra)
library(cluster)
```

Step 2: Data Acquisition and Preprocessing (Scaling)

Our analysis utilizes the well-known intrinsic R dataset, **USArrests**. This dataset documents crime statistics for all 50 U.S. states recorded in 1973. The variables include rates of *Murder*, *Assault*, and *Rape* (per 100,000 residents), alongside *UrbanPop* (the percentage of the population residing in urban areas).

Crucial data preprocessing is mandatory before deploying K-means. Since the algorithm relies fundamentally on distance measurements (like Euclidean distance), features with vastly different scales can unfairly dominate the resulting clusters. For instance, the Assault rate typically has a much larger numerical range than the Murder rate. Therefore, we must standardize (or scale) the variables to ensure each feature contributes equally to the distance calculation. The following R code loads the data, handles potential missing values, and standardizes the features.

```
# Load the built-in USArrests dataset
df <- USArrests

# Remove rows with missing values (if any)
df <- na.omit(df)

# Scale each variable to have a mean of 0 and standard deviation of 1
df <- scale(df)

# View the structure of the scaled dataset
head(df)
```

```
Murder Assault UrbanPop Rape
Alabama 1.24256408 0.7828393 -0.5209066 -0.003416473
Alaska 0.50786248 1.1068225 -1.2117642 2.484202941
Arizona 0.07163341 1.4788032 0.9989801 1.042878388
Arkansas 0.23234938 0.2308680 -1.0735927 -0.184916602
California 0.27826823 1.2628144 1.7589234 2.067820292
Colorado 0.02571456 0.3988593 0.8608085 1.864967207
```

Step 3: Determining the Optimal Number of Clusters (K)

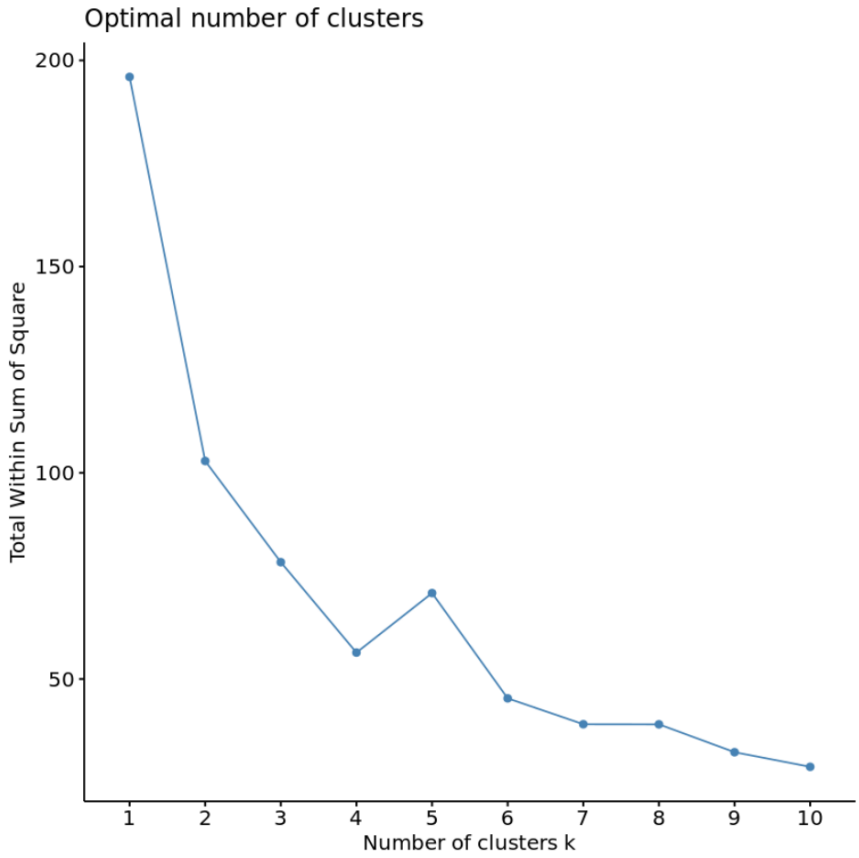
Before running the final K-means model using R's robust **kmeans()** function, we must first address the crucial hyperparameter: the number of clusters, *K* (specified via the `centers` argument). Since arbitrarily selecting *K* can lead to misleading results, we rely on established analytical methods to provide objective evidence for the optimal count.

We will leverage two highly recognized graphical techniques available in the **factoextra** and **cluster** packages: the Elbow Method (based on WSS) and the more statistically rigorous Gap Statistic.

Method 1: The Elbow Method (Total Within Sum of Squares)

The Elbow Method evaluates clustering performance by plotting the Total Within Sum of Squares (WSS) against the number of clusters (K). WSS inherently decreases as K increases, since adding more clusters necessarily reduces the variance within each group. The ideal K value is identified at the "elbow" point--the inflection point where the reduction in WSS begins to sharply diminish, suggesting that any further increase in K provides only marginal improvement in cluster compactness.

fviz_nbclust(df, kmeans, method = "wss")



Upon reviewing the plot, the rate of decrease in WSS clearly exhibits a sharp bend, or "elbow," at K = 4. This graphical evidence strongly suggests that using **four clusters** provides the most efficient partitioning for the standardized USArrests data.

Method 2: The Gap Statistic Validation

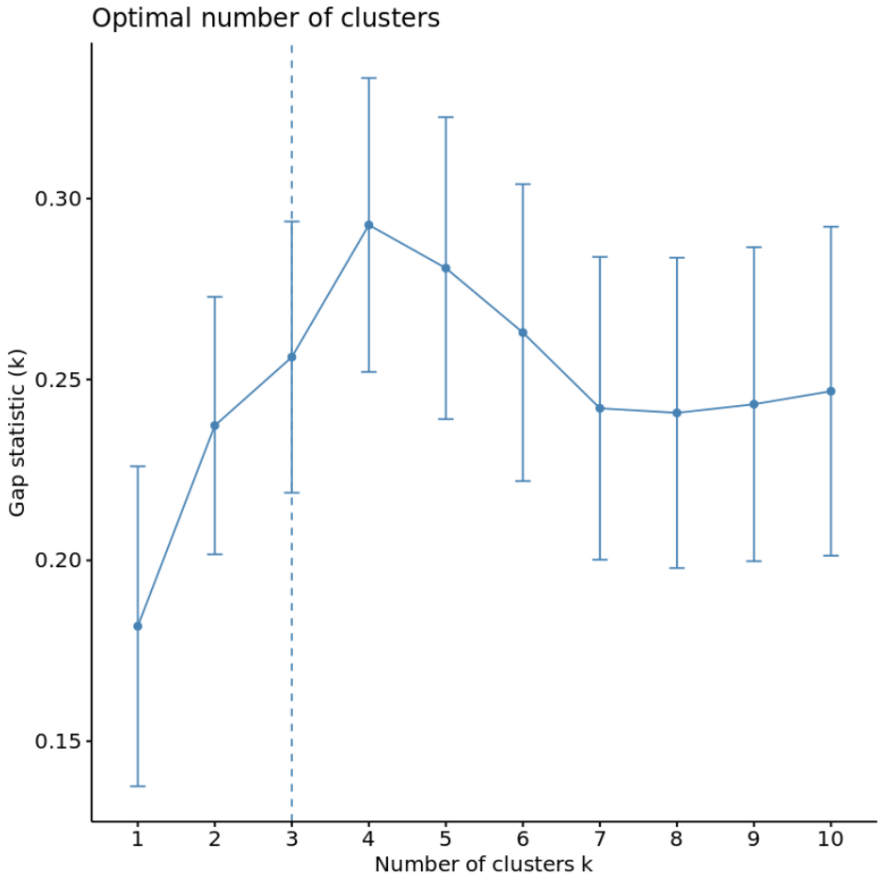
The gap statistic offers a more statistically robust approach to determining optimal K. It works by quantifying the improvement gained by clustering the actual data compared to clustering a reference null distribution--a dataset randomly generated without inherent clustering structure. The optimal K is the value that maximizes this gap statistic, indicating the most significant deviation from randomness.

We calculate this statistic using the **clusGap()** function from the `cluster` package, specifying that 25 random starts (`nstart = 25`) and 50 bootstrapped reference datasets (`B = 50`) should be used for reliable estimation. The results are then visualized using **fviz_gap_stat()**.

 # Calculate gap statistic based on number of clusters
gap_stat <- clusGap(df,
FUN = kmeans,

nstart = 25,
K.max = 10,
B = 50)

# Plot number of clusters vs. gap statistic
fviz_gap_stat(gap_stat)



Consistent with the Elbow Method, this plot definitively shows that the maximum value for the gap statistic is achieved precisely at **K = 4**, providing strong, independent confirmation for our chosen number of clusters.

Step 4: Executing the K-Means Model and Statistical Interpretation

With the optimal value of *K* = 4 confirmed by two separate methods, we proceed to execute the final K-means clustering model. We set a random seed (`set.seed(1)`) to guarantee that the results are fully reproducible. By setting `nstart = 25`, we ensure the algorithm tries 25 different initial configurations and selects the resulting partition that yields the lowest total WSS, thus minimizing the risk of settling for a local optimum.

 # Ensure the example is reproducible
set.seed(1)

# Perform k-means clustering with k = 4 centers
km <- kmeans(df, centers = 4, nstart = 25)

View the detailed results of the model
km

K-means clustering with 4 clusters of sizes 16, 13, 13, 8

Cluster means:
Murder Assault UrbanPop Rape
1 -0.4894375 -0.3826001 0.5758298 -0.26165379
2 -0.9615407 -1.1066010 -0.9301069 -0.96676331
3 0.6950701 1.0394414 0.7226370 1.27693964
4 1.4118898 0.8743346 -0.8145211 0.01927104

Clustering vector:
Alabama Alaska Arizona Arkansas California Colorado
4 3 3 4 3 3
Connecticut Delaware Florida Georgia Hawaii Idaho
1 1 3 4 1 2
Illinois Indiana Iowa Kansas Kentucky Louisiana
3 1 2 1 2 4
Maine Maryland Massachusetts Michigan Minnesota Mississippi
2 3 1 3 2 4
Missouri Montana Nebraska Nevada New Hampshire New Jersey
3 2 2 3 2 1
New Mexico New York North Carolina North Dakota Ohio Oklahoma
3 2 4 2 1 1
Oregon Pennsylvania Rhode Island South Carolina South Dakota Tennessee
1 1 1 4 2 4
Texas Utah Vermont Virginia Washington West Virginia
3 1 2 1 1 2
Wisconsin Wyoming
2 1

Within cluster sum of squares by cluster:
16.212213 11.952463 19.922437 8.316061
(between_SS / total_SS = 71.2 %)

Available components:

"cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss"
"size" "iter" "ifault"

The resulting model summary provides crucial diagnostic information regarding the cluster sizes and the quality of the separation. The states have been distributed into the four clusters as follows:

Cluster 1 is the largest, containing **16** states.
Cluster 2 and Cluster 3 both contain **13** states each.
Cluster 4 is the smallest, grouping **8** states.

A critical metric reported is the ratio `(between_SS / total_SS = 71.2 %)`. This signifies that **71.2%** of the total variance present in the dataset is successfully explained by the separation *between* the clusters. A high percentage like this indicates that the clustering solution is strong and that the identified groups are highly distinct from one another.

Visualizing the Final Cluster Solution

To visually inspect the quality and spatial distribution of the groups, we use the **fviz_cluster()** function. Since the USArrests data is four-dimensional, this function automatically utilizes Principal Component Analysis (PCA) to project the data onto the first two principal components (PC1 and PC2). This allows us to view the multidimensional cluster separation effectively in a clear 2D scatter plot.

 # Plot results of final k-means model, projected onto the first two principal components
fviz_cluster(km, data = df)

Interpreting Cluster Characteristics using Original Data

While the clustering was performed on scaled data, interpreting the results requires examining the cluster means based on the original, unscaled variable values. This provides meaningful context for stakeholders. We use the powerful **aggregate()** function in R to compute the mean of the original USArrests variables for every state assigned to each of the four clusters.

 # Find means of each cluster using the original USArrests data
aggregate(USArrests, by=list(cluster=km$cluster), mean)

cluster Murder Assault UrbanPop Rape

1 3.60000 78.53846 52.07692 12.17692
2 10.81538 257.38462 76.00000 33.19231
3 5.65625 138.87500 73.87500 18.78125
4 13.93750 243.62500 53.75000 21.41250

Based on these mean values, we can clearly profile the distinct crime behavior patterns captured

by our K-means clustering solution:

Cluster 1 (16 states): Characterized by **low overall crime rates** across all metrics, coupled with moderate urbanization levels (Mean Murder: 3.6; Mean Assault: 78.5). These are generally the safest states.
Cluster 2 (13 states): These states exhibit **very high crime rates**, specifically high Assault and Rape figures, and are highly urbanized (Mean UrbanPop: 76.0%).
Cluster 3 (13 states): Identified by **moderate crime rates** and a high degree of urbanization (Mean UrbanPop: 73.9%).
Cluster 4 (8 states): Defined by the **highest average Murder rates** and high Assault rates, yet they display relatively low urbanization (Mean UrbanPop: 53.8%).

Finally, the cluster assignments can be appended back to the original dataset, providing a complete, labeled data frame ready for further geographical or comparative analysis.

 # Add cluster assignment to original data frame
final_data <- cbind(USArrests, cluster = km$cluster)

# View the resulting data frame with cluster labels
head(final_data)

Murder Assault UrbanPop Rape cluster

Alabama 13.2 236 58 21.2 4
Alaska 10.0 263 48 44.5 2
Arizona 8.1 294 80 31.0 2
Arkansas 8.8 190 50 19.5 4
California 9.0 276 91 40.6 2
Colorado 7.9 204 78 38.7 2

Evaluating K-Means: Advantages and Limitations

While K-means clustering is a powerful and widely adopted technique in machine learning, practitioners must be aware of its inherent strengths and weaknesses to determine if it is the optimal choice for a given dataset. These trade-offs guide the selection of appropriate clustering methods.

Key Advantages of K-Means:

Efficiency and Scalability: K-means is exceptionally fast and computationally **efficient**, allowing for rapid deployment and quick processing even when dealing with extremely large datasets (high

N).

Simplicity: Its underlying logic is straightforward, making the algorithm easy to implement, interpret, and explain, which is highly beneficial for exploratory data analysis.

Potential Drawbacks and Constraints:

Requirement for K: It fundamentally requires the user to **pre-specify the number of clusters (K)**, which can sometimes be subjective or arbitrary.

Sensitivity to Outliers: Since the cluster center is defined by the **centroid** (mean), the algorithm is highly **sensitive to outliers**. Extreme values can pull the centroid significantly, distorting the shape and position of the resulting cluster.

Geometric Assumptions: K-means implicitly assumes that clusters are convex (spherical) and roughly equal in size and density, making it ineffective for identifying clusters with complex, non-linear shapes.

For datasets where these assumptions are violated--for instance, data containing many outliers or complex shapes--alternative methods are often preferred. These alternatives include K-medoids clustering (which uses medoids instead of means, making it less sensitive to outliers) or hierarchical clustering (which eliminates the need to pre-specify K).

This tutorial successfully demonstrated the entire workflow for applying **K-means clustering** in R, from data preparation and optimal K determination to final interpretation and visualization. The complete R code used to generate this analysis is available for reference here.