**K-means done with built in dataset all.mammals.milk.1956**

K-means clustering in R

1. Install package cluster.datasets

From the package make use of all.mammals.milk.1956(built in)

It contains

* A data frame with 25 observations on the following 6 variables.
* Name a character vector for the animal name
* Water a numeric vector for the percentage of water
* Protein a numeric vector for the percentage of protein
* Fat a numeric vector for the percentage of fat
* Lactose a numeric vector for the percentage of lactose
* Ash a numeric vector for the percentage of ash.

1. Install tidyverse package in R

**What is tidyverse package?**

* Tidyverse is a collection of essential R packages for data science. The packages under the tidyverse umbrella help in performing and interacting with the data. There are a whole host of things you can do with your data, such as subsetting, transforming, visualizing, etc.
* The "tidyverse" collects some of the most versatile R packages: ggplot2, dplyr, tidyr, readr, purrr, and tibble. The packages work in harmony to clean, process, model, and visualize data.

1. Install gridExtra package in R

**What is gridExtra package ?**

* Provides a number of user-level functions to work with ``grid''graphics, notably to arrange multiple grid-based plots on a page, and draw tables

4.Plot the data

Plot all mammals vs each of water, protein, lactose, fat, ash etc

4.set.seed(N)

* The *seed* number you choose is the starting point used in the generation of a sequence of random numbers

**5.kmeans function in R**

The format of the K-means function in R is kmeans(x, centers) where x is a numeric dataset (matrix or data frame) and centers is the number of clusters to extract. The function returns the cluster memberships, centroids, sums of squares (within, between, total), and cluster sizes.

The kmeans() function has an nstart option that attempts multiple initial configurations and reports on the best one. For example, adding nstart=25 will generate 25 initial configurations.

The output of kmeans is a list with several bits of information. The most important being:

* cluster: A vector of integers (from 1:k) indicating the cluster to which each point is allocated.
* centers: A matrix of cluster centers.
* totss: The total sum of squares.
* withinss: Vector of within-cluster sum of squares, one component per cluster.
* tot.withinss: Total within-cluster sum of squares, i.e. sum(withinss).
* betweenss: The between-cluster sum of squares, i.e. $totss-tot.withinss$.
* size: The number of points in each cluster.

In the R code , observe

The **kmeans()** function outputs the results of the clustering. We can see the centroid vectors (cluster means), the group in which each observation was allocated (clustering vector) and a percentage (89.9%) that represents the compactness of the clustering, that is, how similar are the members within the same group. If all the observations within a group were in the same exact point in the n-dimensional space, then we would achieve 100% of compactness.

**Clustering Validation**

We may use the silhouette coefficient (silhouette width) to evaluate the goodness of our clustering.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

The silhouette coefficient is calculated as follows:

1. For each observation **i**, it calculates the average dissimilarity between **i** and all the other points within the same cluster which **i** belongs. Let’s call this average dissimilarity **“Di”**.
2. Now we do the same dissimilarity calculation between **i** and all the other clusters and get the lowest value among them. That is, we find the dissimilarity between **i** and the cluster that is closest to **i** right after its own cluster. Let’s call that value **“Ci”**
3. The silhouette (**Si**) width is the difference between Ci and Di (Ci — Di) divided by the greatest of those two values (max(Di, Ci)).  
   **Si = (Ci — Di) / max(Di, Ci)**

So, the interpretation of the silhouette width is the following:

* Si > 0 means that the observation is well clustered. The closest it is to 1, the best it is clustered.
* Si < 0 means that the observation was placed in the wrong cluster.
* Si = 0 means that the observation is between two clusters.

The silhouette plot below gives us evidence that our clustering using four groups is good because there’s no negative silhouette width and most of the values are bigger than 0.5.