Assignment 5

Rob DiVincenzo 11-30-2023 BA 64060

This is the submission for Assignment 5.

```
#Read data
DF=read.csv("./Cereals.csv") # Read the Cereals csv file
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(cluster)
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
#Load libraries
#library(ISLR)
#library(tidyverse)  # data manipulation
#library(factoextra) # clustering algorithms & visualization
#library(flexclust)

set.seed(123)
```

```
# Data Preprocessing
#Select only the 9 numerical variables
DF <- select(DF, calories, protein, fat, sodium, fiber, carbo, sugars, potass, vitamins, shelf, weight,

cleaned_data <- na.omit(DF) #omit any rows with na column values

normalized_data <- scale(cleaned_data) #normalize data
```

```r
# 1) Ward is the best result

# Compute with agnes and with different linkage methods
hc_single <- agnes(normalized_data, method="single")
hc_complete <- agnes(normalized_data, method="complete")
hc_average <- agnes(normalized_data, method="average")
hc_ward <- agnes(normalized_data, method="ward")

print(hc_single$ac)
```

```
## [1] 0.6067859
```

```r
print(hc_complete$ac)
```

```
## [1] 0.8353712
```

```r
print(hc_average$ac)
```

```
## [1] 0.7766075
```

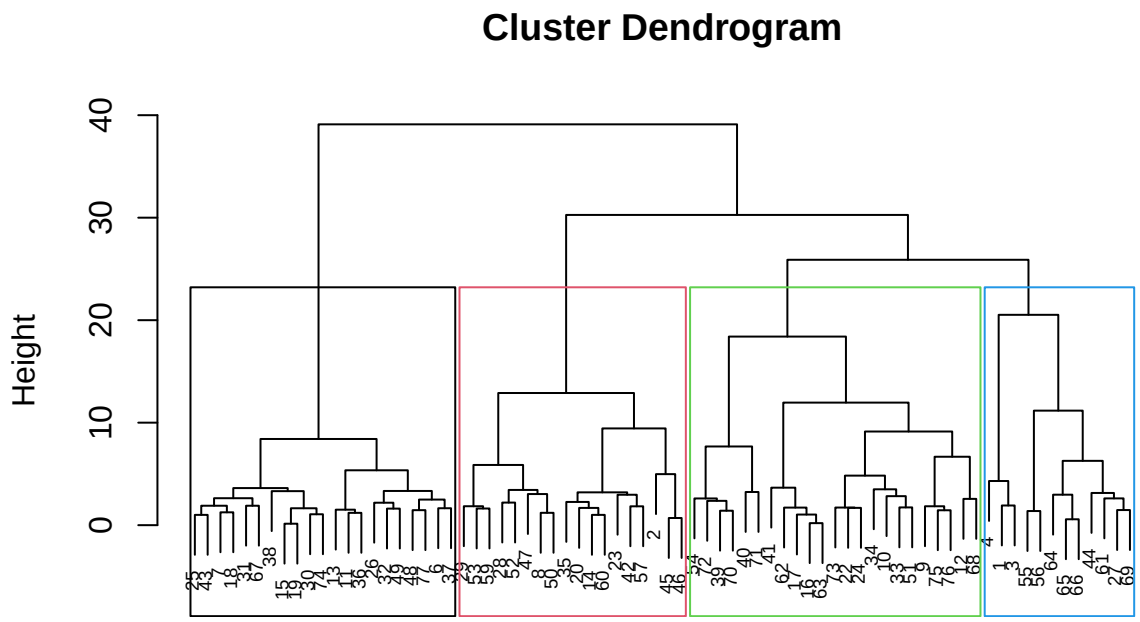
```r
print(hc_ward$ac) # best result
```

```
## [1] 0.9046042
```

```r
# 2) I would choose 4 clusters for our dendrogram since clusters in k=3 seemed too large and clusters i
d <- dist(normalized_data, method = "euclidean")

hc_ward2 <- hclust(d, method = "ward.D")

#plot the denrogram
plot(hc_ward2, cex = 0.6)
rect.hclust(hc_ward2, k=4, border = 1:4)
```




```r
#3) # It appears that most observations remained in their original clusters, indicating a good level of

#3a)

#Combine the cluster you obtained in the first question to the data, and then make a partition A and B

# Extract cluster assignments
ward_clusters <- cutree(hc_ward, k = 4)
# Combine cluster assignments with normalized data using data.frame without converting to factors
data_with_clusters <- data.frame(normalized_data, Ward_Cluster = ward_clusters)

# Make partition A (70%) and partition B (30%)
index_partition_A <- createDataPartition(data_with_clusters$Ward_Cluster, p = 0.7, list = FALSE)
partition_A <- data_with_clusters[index_partition_A, ]
partition_B <- data_with_clusters[-index_partition_A, ]

#3b)
#Compute the mean/average for each numerical variable in each cluster in partition A (so you obtain the

# Assuming your numerical variables are columns 1 to n in 'partition_A'
centroid_A <- partition_A %>%
  group_by(Ward_Cluster) %>%
  summarize(across(everything(), mean))

#3c)
```

```r
# Exclude Ward Cluster number
numeric_columns_B <- partition_B[, -14]

# Initialize a vector to store new cluster numbers in partition B
new_clusters_B <- integer(nrow(partition_B))

# Compute Euclidean distance between observations in partition B and centroids in A using a for loop
for (i in 1:nrow(partition_B)) {
  observation <- as.matrix(numeric_columns_B[i, , drop = FALSE])
  distances <- apply(centroid_A[, -1], 1, function(centroid) sqrt(sum((observation - centroid)^2)))
  new_clusters_B[i] <- which.min(distances)
}

# Create a new data frame with the original columns and the new cluster assignments
partition_B_new <- data.frame(partition_B, New_Cluster = new_clusters_B)

#3d)
#Compare the difference between the old and new cluster numbers in partition B to check its stability.

# Check stability by comparing old and new cluster assignments
stability_comparison <- table(partition_B_new$Ward_Cluster, partition_B_new$New_Cluster)

# Add labels and show a clearer stability comparison
stability_labels <- c("Cluster 2 (old)", "Cluster 3 (old)", "Cluster 4 (old)")
rownames(stability_comparison) <- colnames(stability_comparison) <- stability_labels

print(stability_comparison)
```

```
##
##                   Cluster 2 (old) Cluster 3 (old) Cluster 4 (old)
##   Cluster 2 (old)               5               1               0
##   Cluster 3 (old)               0               6               0
##   Cluster 4 (old)               0               0               9
```

```r
# It appears that most observations remained in their original clusters, indicating a good level of sta
```

```r
#4) Cluster 1 is the healthiest cluster. We absolutely need to use normalized data here because we are
```

```r
# Define the criteria for "Healthy Cereals" (healthy_factors and unhealthy factors)
healthy_factors <- c("protein", "fiber", "potass", "vitamins") # high is good
unhealthy_factors <- c("calories", "fat", "carbo", "sugars") # high is bad

# Obtain the cluster centroids
centroids <- data_with_clusters %>%
  group_by(Ward_Cluster) %>%
  summarize(across(healthy_factors, mean), across(unhealthy_factors, mean))
```

```
## Warning: There were 2 warnings in `summarize()`.
## The first warning was:
## i In argument: `across(healthy_factors, mean)`.
## Caused by warning:
```

```
## ! Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(healthy_factors)
##
##   # Now:
##   data %>% select(all_of(healthy_factors))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## i Run ‘dplyr::last_dplyr_warnings()‘ to see the 1 remaining warning.
```

```
# Calculate the "health score" score for each cluster by summing up the healthy columns and subtracting
cluster_scores <- rowSums(centroids[, -1][, 1:4] - centroids[, -1][, 5:8])

# Identify the healthiest Ward Cluster, the one with the highest health score.
healthiest_cluster <- centroids$Ward_Cluster[which.max(cluster_scores)]

print(healthiest_cluster)
```

```
## [1] 1
```

```
#Cluster 1 is the healthiest cluster.
```