Protein Consumption in European Countries

Rutwik Guntoorkar

2023-04-03

Introduction:

The accompanied dataset gives estimates of the average protein consumption (in grams per person per day) from different food sources for the inhabitants of 25 European countries. We have to answer the questions below:

A. Use principal components analysis to investigate the relationships between the countries on the basis of these variables

B. Carry out cluster analysis to study relation between countries on their diet

C. Identify the important factors underlying the observed variables and examine the relationships between the countries with respect to these factors

Loading the Dataset

```
library(readr)
library(MVA)

## Loading required package: HSAUR2

## Loading required package: tools

library(HSAUR2)
library(SciViews)
library(scatterplot3d)
library(car)

## Loading required package: carData

library(lattice)
library(GGally)

## Loading required package: ggplot2

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(ggplot2)
library(ggridges)
library(ggvis)
```

```
##
## Attaching package: 'ggvis'

## The following object is masked from 'package:ggplot2':
##
##     resolution

library(ggthemes)
library(cowplot)

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggthemes':
##
##     theme_map

library(gapminder)
library(gganimate)

## No renderer backend detected. gganimate will default to writing frames to
separate files
## Consider installing:
## - the `gifski` package for gif output
## - the `av` package for video output
## and restarting the R session

##
## Attaching package: 'gganimate'

## The following object is masked from 'package:ggvis':
##
##     view_static

library(dplyr)

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:car':
##
##     recode

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

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

library(tidyverse)
```

```
## ── Attaching packages
## ───────────────────────────────────────
## tidyverse 1.3.2 ──

## ✓ tibble  3.1.8     ✓ stringr 1.5.0
## ✓ tidyr   1.3.0     ✓ forcats 1.0.0
## ✓ purrr   1.0.1
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter()     masks stats::filter()
## ✖ dplyr::lag()        masks stats::lag()
## ✖ dplyr::recode()     masks car::recode()
## ✖ ggvis::resolution() masks ggplot2::resolution()
## ✖ purrr::some()       masks car::some()
```

```r
library(grid)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(RColorBrewer)
library(Hotelling)
```

```
## Loading required package: corpcor
##
## Attaching package: 'Hotelling'
##
## The following object is masked from 'package:dplyr':
##
##     summarise
```

```r
library(stats)
library(biotools)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## ---
## biotools version 4.2
```

```r
library(factoextra)
```

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

library(FactoMineR)
library(ggfortify)
library(psych)

##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
##
## The following object is masked from 'package:car':
##
##     logit

library(corrplot)

## corrplot 0.92 loaded

library(devtools)

## Loading required package: usethis

library(cluster)
library(magrittr)

##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract

library(NbClust)
library(MASS)
library(gvlma)
library(leaps)
library(relaimpo)

## Loading required package: boot
##
## Attaching package: 'boot'
##
## The following object is masked from 'package:psych':
##
```

```
##     logit
##
## The following object is masked from 'package:lattice':
##
##     melanoma
##
## The following object is masked from 'package:car':
##
##     logit
##
## Loading required package: survey
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## The following object is masked from 'package:ggvis':
##
##     band
##
## Loading required package: survival
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:boot':
##
##     aml
##
##
## Attaching package: 'survey'
##
## The following object is masked from 'package:graphics':
##
##     dotchart
##
## Loading required package: mitools
## This is the global version of package relaimpo.
##
## If you are a non-US user, a version with the interesting additional metric
pmvd is available
##
## from Ulrike Groempings web site at prof.beuth-hochschule.de/groemping.
```

```
ptn <- read.csv("~/Desktop/Projects/Protein Consumption in European
Countries/Protein_Consumption.csv", row.names=1)

attach(ptn)
```

1. About the Dataset

About

- The dataset contains estimates of average protein consumption (in grams per person per day) from various food sources for the inhabitants of 25 European countries. Each row represents a country, while the columns correspond to different food sources contributing to protein intake. The variables may include, but are not limited to, meat, dairy, legumes, grains, and seafood.

- The dataset aims to capture the dietary habits of European populations and explore the relationships between countries based on their protein consumption patterns. This information is crucial for understanding dietary trends, identifying potential health implications, and exploring cultural and economic factors influencing food choices across different nations.

Data Source

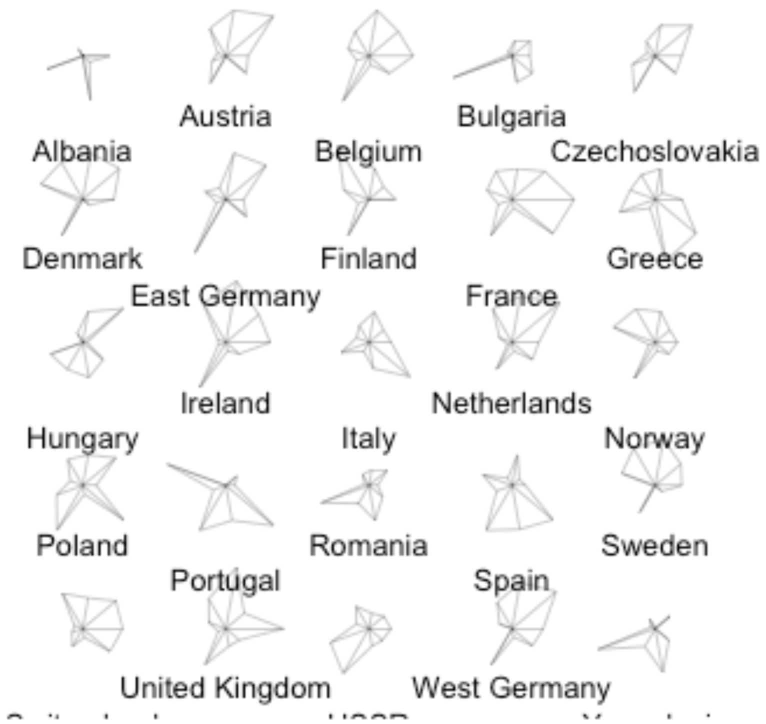
- Link: https://github.com/rutwiksg/Protein-Consumption-in-European-Countries/blob/main/Protein_Consumption.csv

Data Dictionary

- Country: The name of the European country.
- Rest of the columns: Average grams of protein consumed per person per day from the respective item mentioned in the column.

Analysing the Data

```
str(ptn)

## 'data.frame':    25 obs. of  10 variables:
##  $ Red.Meat                : int  10 9 14 8 10 11 8 10 18 10 ...
##  $ White.Meat              : int  1 14 9 6 11 11 12 5 10 3 ...
##  $ Egg                     : int  1 4 4 2 3 4 4 3 3 3 ...
##  $ Milk                    : int  9 20 18 8 13 25 11 34 20 18 ...
##  $ Fish                    : int  0 2 5 1 2 10 5 6 6 6 ...
##  $ Cereals                 : int  42 28 27 57 34 22 25 26 28 42 ...
##  $ Starchy.Foods           : int  1 4 6 1 5 5 7 5 5 2 ...
##  $ Pulses.Nuts.and.Oilseeds: int  6 1 2 4 1 1 1 1 2 8 ...
##  $ Fruits.and.Vegetables   : int  2 4 4 4 4 2 4 1 7 7 ...
##  $ Total                   : int  72 86 89 91 83 91 77 91 99 99 ...

prtn <- ptn[,-10]
stars(prtn)
```



- The Total column has been excluded from the analysis.
- This is because it is just the sum of all the rows and causes disruption in the results.
- The stars function helps us identify immediate commonalities between countries.
- East Germany, Ireland, the UK, West Germany, and the Netherlands all countries have similar star diagrams indicating similar characteristics between these countries.

2. The Multivariate Analysis aims to do the following.

Exploring the interrelationships between European countries based on their dietary variables. (Through Principal Component Analysis (PCA))

Investigate the dietary similarities and differences among countries. (Through Cluster Analysis)

Identify the key factors underlying the observed dietary variables and examine the relationships between countries concerning these factors.

3. Principal Component Analysis (PCA)

- Principal Component Analysis (PCA) is a statistical technique for dimensionality reduction and data visualization. It aims to transform a set of possibly correlated variables into a new set of uncorrelated variables called principal components.

These principal components are linear combinations of the original variables and are ordered by the amount of variance they explain in the data.

- PCA finds the directions, or principal components, along which the data varies the most. The first principal component explains the most significant amount of variance in the data, with each subsequent component explaining as much of the remaining variance as possible, subject to the constraint that it is orthogonal (uncorrelated) to the previous components.

Principal Component Analysis (PCA) is like a magic trick that helps you simplify this big table into something easier to understand. It does this by finding patterns in the data.

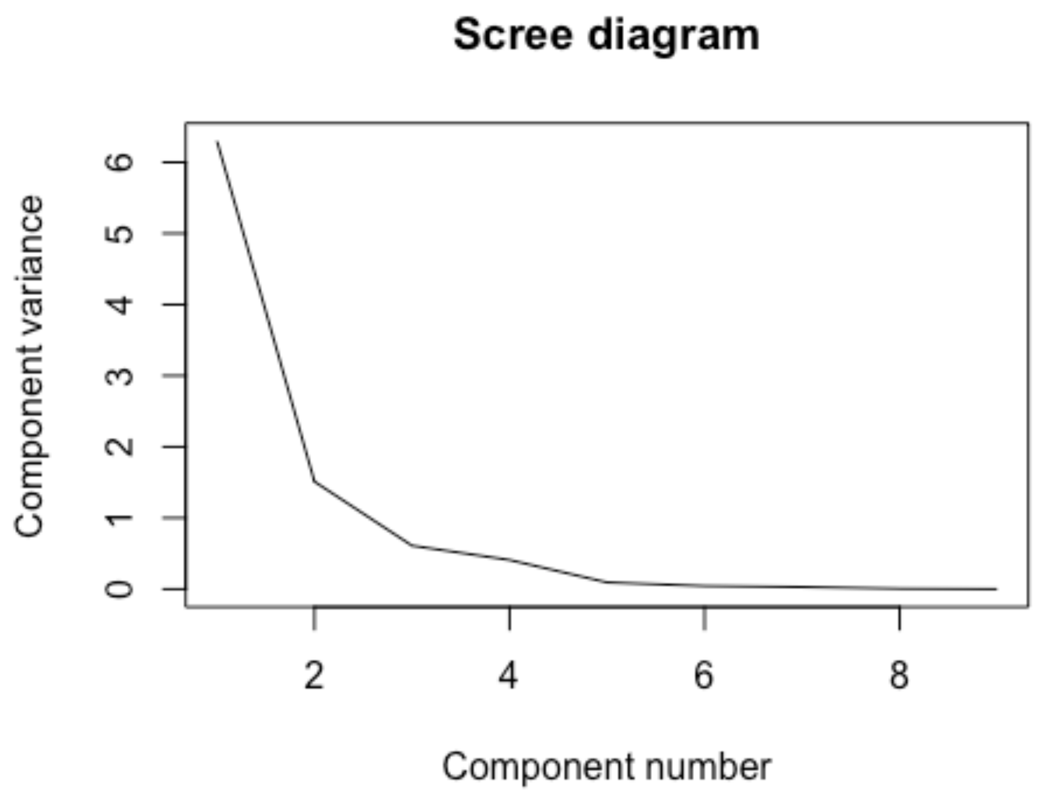
PCA values

```
protein <- cor(prtn)

protein_pca <- prcomp(protein,scale=TRUE)

summary(protein_pca)

## Importance of components:
##                           PC1    PC2    PC3     PC4    PC5     PC6     PC7
## Standard deviation     2.5096 1.2292 0.7812 0.63960 0.3060 0.20803 0.17034
## Proportion of Variance 0.6998 0.1679 0.0678 0.04545 0.0104 0.00481 0.00322
## Cumulative Proportion  0.6998 0.8677 0.9355 0.98091 0.9913 0.99612 0.99935
##                            PC8       PC9
## Standard deviation     0.07659 2.225e-17
## Proportion of Variance 0.00065 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00
```

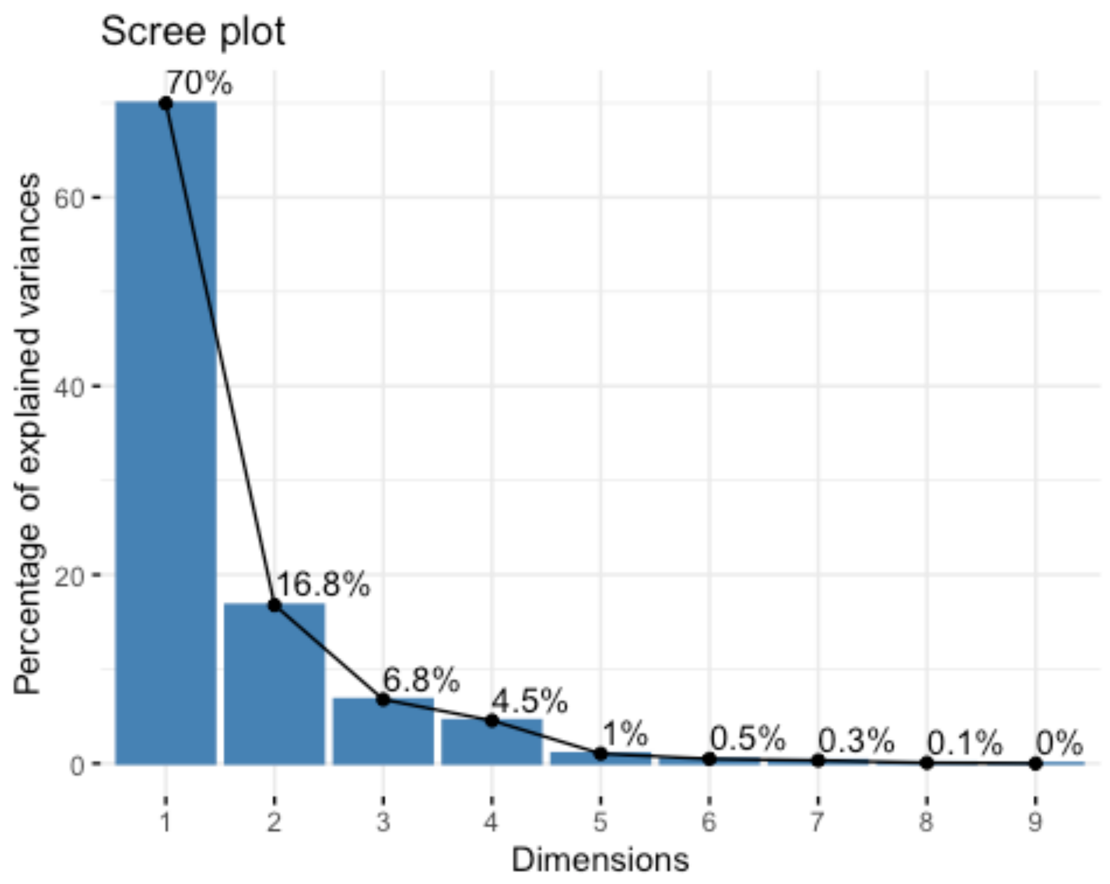
- Further, we can check the schematic diagram to identify how many principal components to consider for our analysis.

Scree Plots

```
(eigen_protein <- protein_pca$sdev^2)

## [1] 6.298011e+00 1.510895e+00 6.102040e-01 4.090876e-01 9.364428e-02
## [6] 4.327677e-02 2.901464e-02 5.866446e-03 4.951441e-34

names(eigen_protein) <- paste("PC",1:9,sep="")

plot(eigen_protein, xlab = "Component number", ylab = "Component variance",
type = "l", main = "Scree diagram")
```



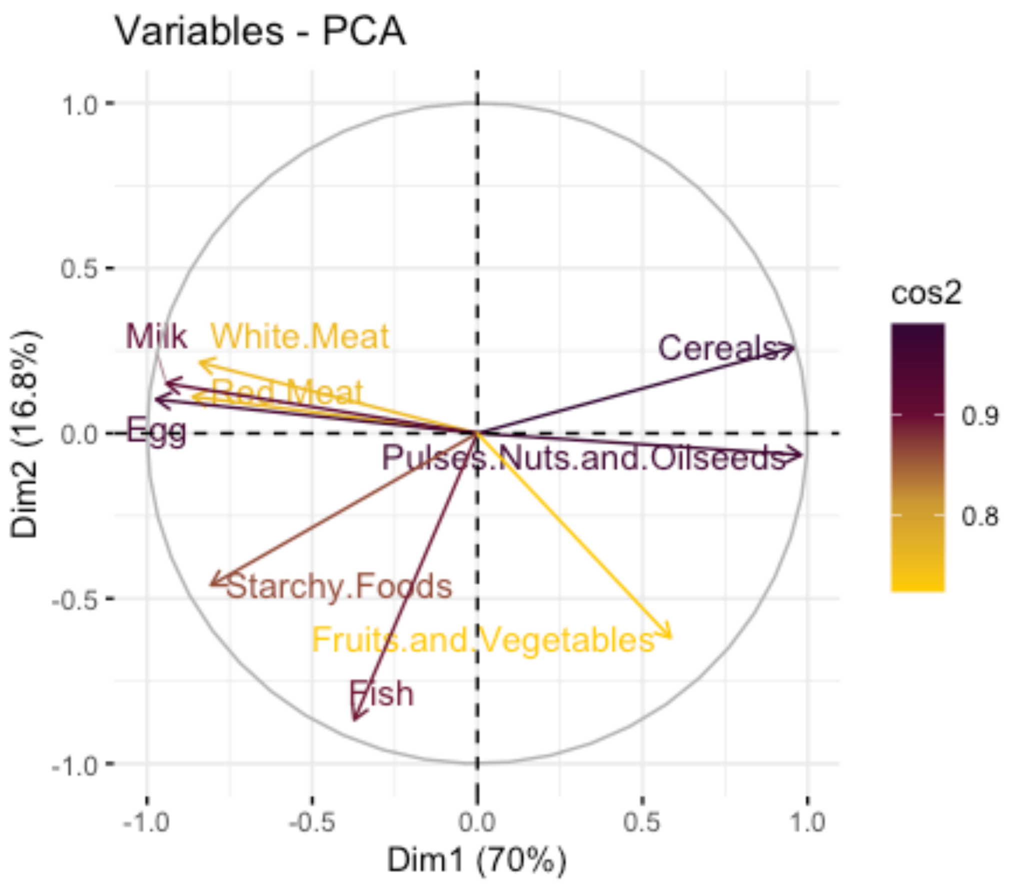
```
fviz_eig(protein_pca, addlabels = TRUE)
```



- The scree plot determines how many Principal Components (PCs) to use for the analysis.
- The significant bend in the plot is used to determine the number of PCs to be used.
- The plot shows us the number of components to be considered is 2. (86.8% of variance)

Biplot

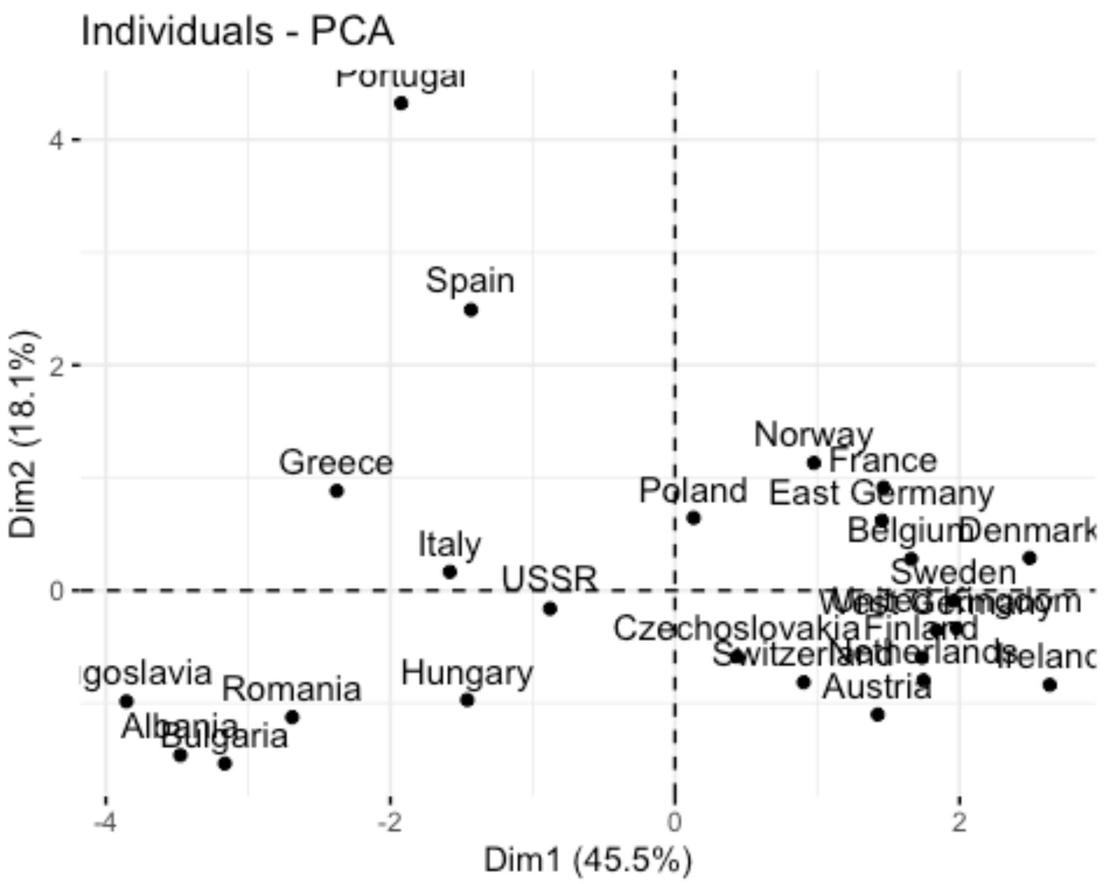
```
fviz_pca_var(protein_pca,col.var = "cos2",
             gradient.cols = c("#FFCC00", "#CC9933", "#660033", "#330033"),
             repel = TRUE)
```



- The distance between points in a biplot reflects the generalised distance between them.
- The length of the vector reflects the variance of the variable.
- Correlation of the variables reflected by the angle between them. The smaller the angle, the more significant the correlation.
- For example, it shows that meat and milk are all correlated strongly.

Individual PCA

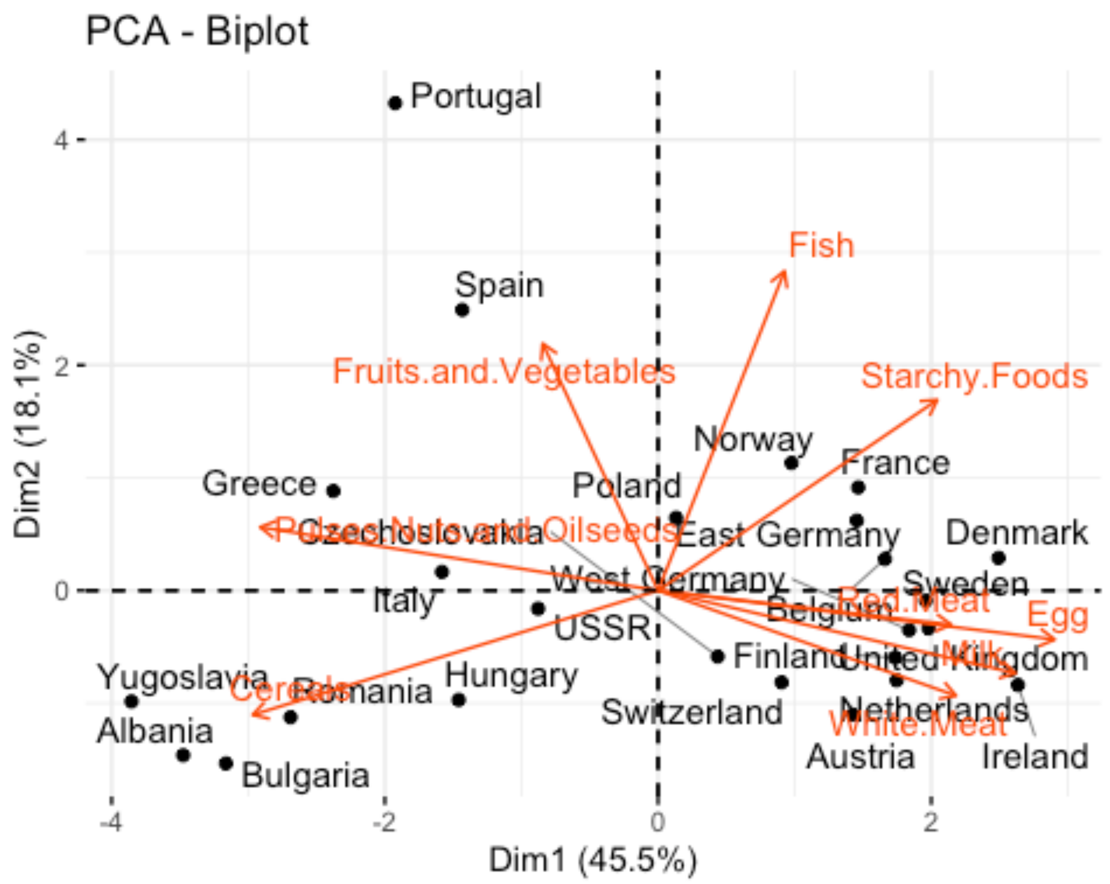
```
res.pca <- PCA(prtn, graph = FALSE)

fviz_pca_ind(res.pca)
```



- The countries have been plotted based on their PCA values in the individual PCA plot.
- The countries are allocated based on their similarities.
- All the South European countries are at the top left of the plot.

PCA - Biplot

```
fviz_pca_biplot(res.pca, repel = TRUE,
                col.var = "#FC4E07", # Variables color
                )
```



- The combination of the variables and individual PCAs gives us the PCA biplot.
- We can see that Albania, Romania, Yugoslavia, and Bulgaria (Balkan countries) tend to have cereals as their primary protein intake.
- Southern European countries (Portugal, Spain, Italy, and Greece) have Fruits, Vegetables, pulses, and oil seeds as their primary source of protein intake.
- Eastern European countries (USSR, Poland, Hungary, East Germany, Czechoslovakia) have a combination of milk, meat, fruits, vegetables, cereal, and oil seeds, all contributing equally towards their protein intake.
- All the rest of the Western and Northern European countries have milk and meat as their primary source of protein in their diet.

Thus, the PCA helped us club countries and tell us their similarities. In our further analysis, we can see if this still stands and find the underlying reason behind this observation.

Cluster Analysis

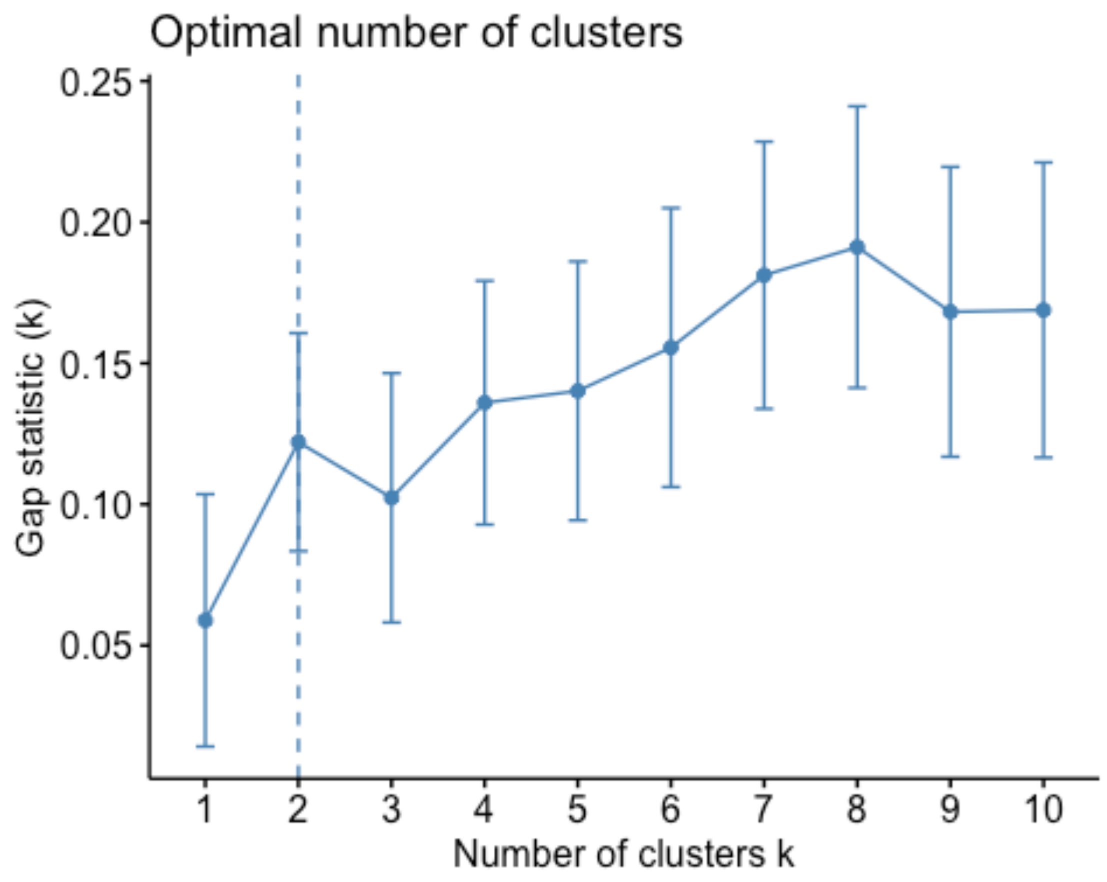
- Cluster analysis categorizes data points into groups based on similarities, revealing patterns and structures within datasets. It aids in exploring relationships, uncovering trends, and facilitating segmentation for targeted strategies in various domains such as business, healthcare, and social sciences. By identifying distinct clusters, analysts can make informed decisions and tailor interventions for more effective outcomes, making cluster analysis an essential tool for data-driven insights and decision-making.

Clustering organizes data points into groups, or "clusters," based on their similarities. It helps us see patterns and similarities within the data, making it easier to understand and analyze.

- We first must define the ideal number of clusters to divide the data into. We can check using a couple of tests.

Optimal Clusters

```
matstd_protein <- scale(prtn)

fviz_nbclust(matstd_protein, kmeans, method = "gap_stat")
```



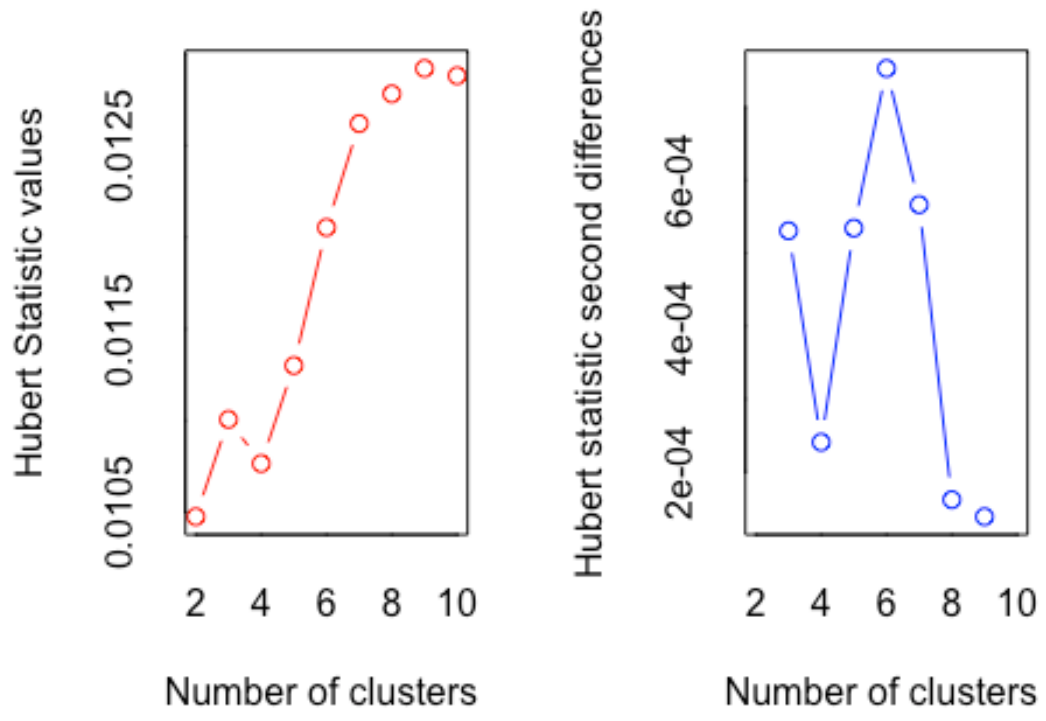
Code for the function to run

```
fviz_nbclust <- function (x, FUNcluster = NULL, method = c("silhouette", 
"wss", 
                                                         "gap_stat"), diss 
= NULL, k.max = 10, nboot = 100, verbose = interactive(), 
                          barfill = "steelblue", barcolor = "steelblue", 
linecolor = "steelblue", 
                          print.summary = TRUE, ...)
```

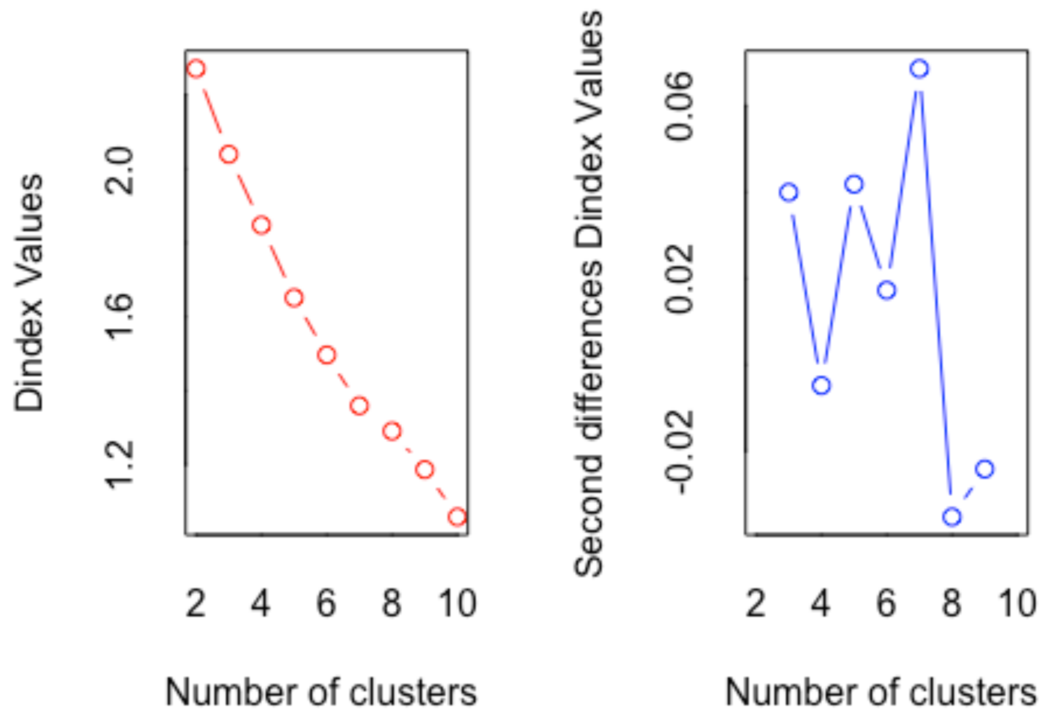
```
{
  set.seed(123)
  if (k.max < 2)
    stop("k.max must bet > = 2")
  method = match.arg(method)
  if (!inherits(x, c("data.frame", "matrix")) & !("Best.nc" %in%
                                                  names(x)))
    stop("x should be an object of class matrix/data.frame or ",
         "an object created by the function NbClust() [NbClust package].")
  if (inherits(x, "list") & "Best.nc" %in% names(x)) {
    best_nc <- x$Best.nc
    if (any(class(best_nc) == "numeric") )
      print(best_nc)
    else if (any(class(best_nc) == "matrix") )
      .viz_NbClust(x, print.summary, barfill, barcolor)
  }
  else if (is.null(FUNcluster))
    stop("The argument FUNcluster is required. ", "Possible values are
kmeans, pam, hcut, clara, ...")
  else if (!is.function(FUNcluster)) {
    stop("The argument FUNcluster should be a function. ",
         "Check if you're not overriding the specified function name
somewhere.")
  }
  else if (method %in% c("silhouette", "wss")) {
    if (is.data.frame(x))
      x <- as.matrix(x)
    if (is.null(diss))
      diss <- stats::dist(x)
    v <- rep(0, k.max)
    if (method == "silhouette") {
      for (i in 2:k.max) {
        clust <- FUNcluster(x, i, ...)
        v[i] <- .get_ave_sil_width(diss, clust$cluster)
      }
    }
    else if (method == "wss") {
      for (i in 1:k.max) {
        clust <- FUNcluster(x, i, ...)
        v[i] <- .get_withinSS(diss, clust$cluster)
      }
    }
    df <- data.frame(clusters = as.factor(1:k.max), y = v,
                     stringsAsFactors = TRUE)
    ylab <- "Total Within Sum of Square"
    if (method == "silhouette")
      ylab <- "Average silhouette width"
    p <- ggpubr::ggline(df, x = "clusters", y = "y", group = 1,
                        color = linecolor, ylab = ylab, xlab = "Number of
clusters k",
```

```r
                         main = "Optimal number of clusters")
    if (method == "silhouette")
      p <- p + geom_vline(xintercept = which.max(v), linetype = 2,
                          color = linecolor)
    return(p)
  }
  else if (method == "gap_stat") {
    extra_args <- list(...)
    gap_stat <- cluster::clusGap(x, FUNcluster, K.max = k.max,
                                 B = nboot, verbose = verbose, ...)
    if (!is.null(extra_args$maxSE))
      maxSE <- extra_args$maxSE
    else maxSE <- list(method = "firstSEmax", SE.factor = 1)
    p <- fviz_gap_stat(gap_stat, linecolor = linecolor,
                       maxSE = maxSE)
    return(p)
  }
}

.viz_NbClust <- function (x, print.summary = TRUE, barfill = "steelblue",
                          barcolor = "steelblue")
{
  best_nc <- x$Best.nc
  if (any(class(best_nc) == "numeric") )
    print(best_nc)
  else if (any(class(best_nc) == "matrix") ) {
    best_nc <- as.data.frame(t(best_nc), stringsAsFactors = TRUE)
    best_nc$Number_clusters <- as.factor(best_nc$Number_clusters)
    if (print.summary) {
      ss <- summary(best_nc$Number_clusters)
      cat("Among all indices: \n===================\n")
      for (i in 1:length(ss)) {
        cat("*", ss[i], "proposed ", names(ss)[i],
            "as the best number of clusters\n")
      }
      cat("\nConclusion\n=========================\n")
      cat("* According to the majority rule, the best number of clusters is
",
          names(which.max(ss)), ".\n\n")
    }
    df <- data.frame(Number_clusters = names(ss), freq = ss,
                     stringsAsFactors = TRUE)
    p <- ggpubr::ggbarplot(df, x = "Number_clusters",
                           y = "freq", fill = barfill, color = barcolor) +
      labs(x = "Number of clusters k", y = "Frequency among all indices",
           title = paste0("Optimal number of clusters - k = ",
                          names(which.max(ss))))
    return(p)
  }
}
```

```
res.nbclust <- prtn %>% scale() %>% NbClust(distance = "euclidean", min.nc =
2, max.nc = 10, method = "complete", index ="all")

## Warning in pf(beale, pp, df2): NaNs produced

## Warning in pf(beale, pp, df2): NaNs produced
```



```
## *** : The Hubert index is a graphical method of determining the number of
clusters.
##                 In the plot of Hubert index, we seek a significant knee
that corresponds to a
##                 significant increase of the value of the measure i.e the
significant peak in Hubert
##                 index second differences plot.
##
```

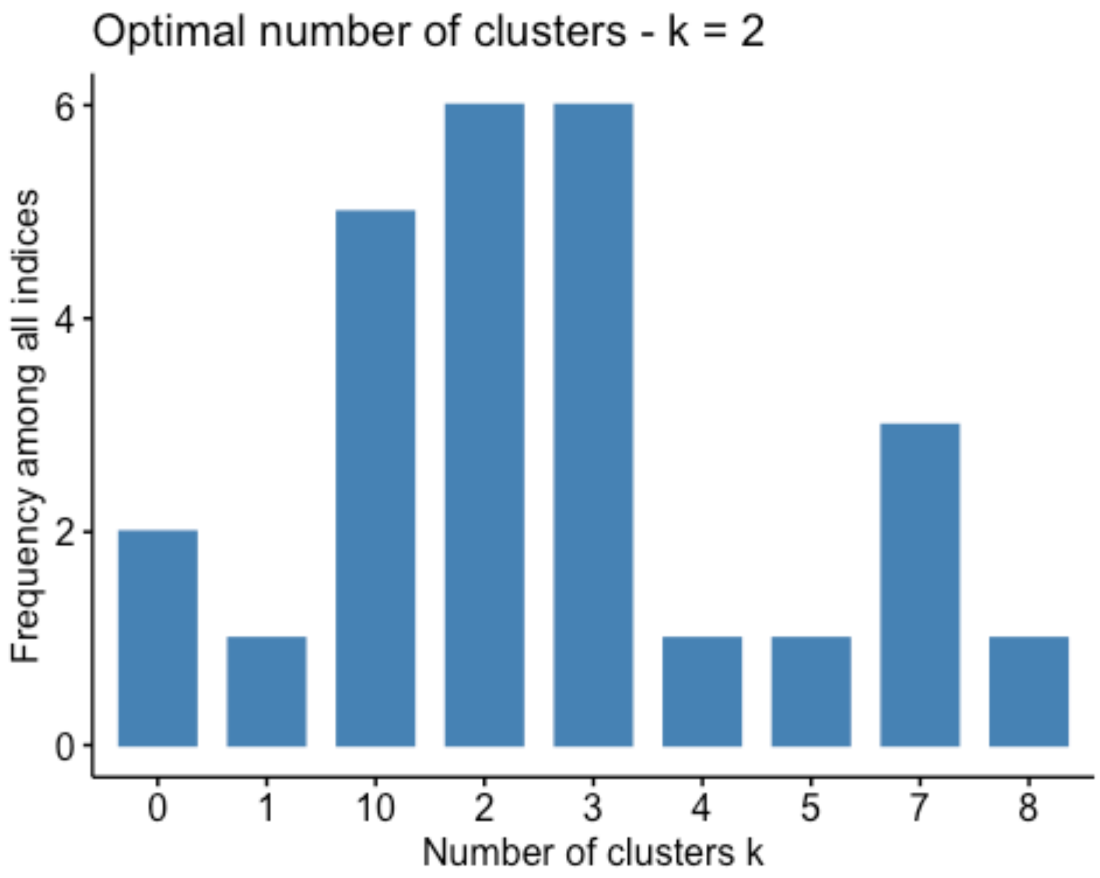


```
## *** : The D index is a graphical method of determining the number of
clusters.
##                 In the plot of D index, we seek a significant knee (the
significant peak in Dindex
##                 second differences plot) that corresponds to a significant
increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 6 proposed 2 as the best number of clusters
## * 6 proposed 3 as the best number of clusters
## * 1 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 3 proposed 7 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 5 proposed 10 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
```

```
##
## *******************************************************************
```

```
fviz_nbclust(res.nbclust, ggtheme = theme_minimal())
```

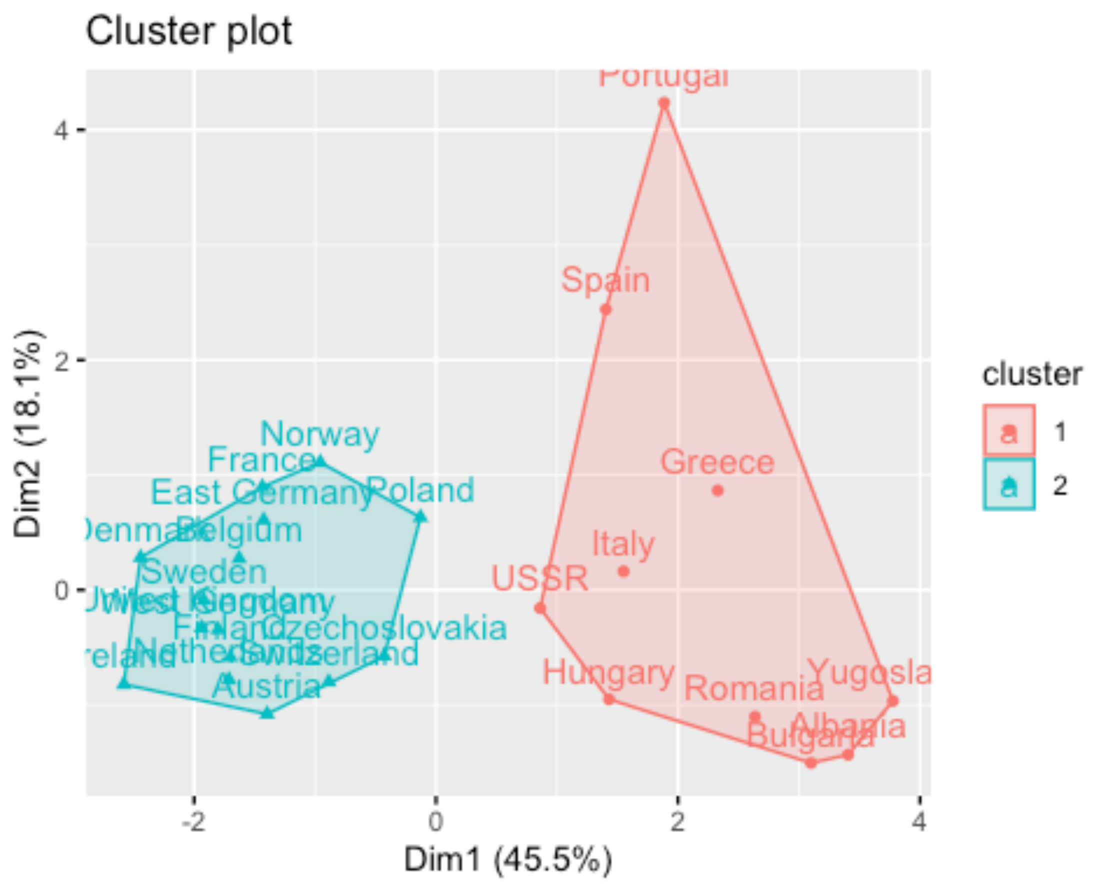
```
## Among all indices:
## ===================
## * 2 proposed  0 as the best number of clusters
## * 1 proposed  1 as the best number of clusters
## * 6 proposed  2 as the best number of clusters
## * 6 proposed  3 as the best number of clusters
## * 1 proposed  4 as the best number of clusters
## * 1 proposed  5 as the best number of clusters
## * 3 proposed  7 as the best number of clusters
## * 1 proposed  8 as the best number of clusters
## * 5 proposed  10 as the best number of clusters
##
## Conclusion
## =========================
## * According to the majority rule, the best number of clusters is  2 .
```



- Both the tests suggest that the optimal number of clusters should be 2.

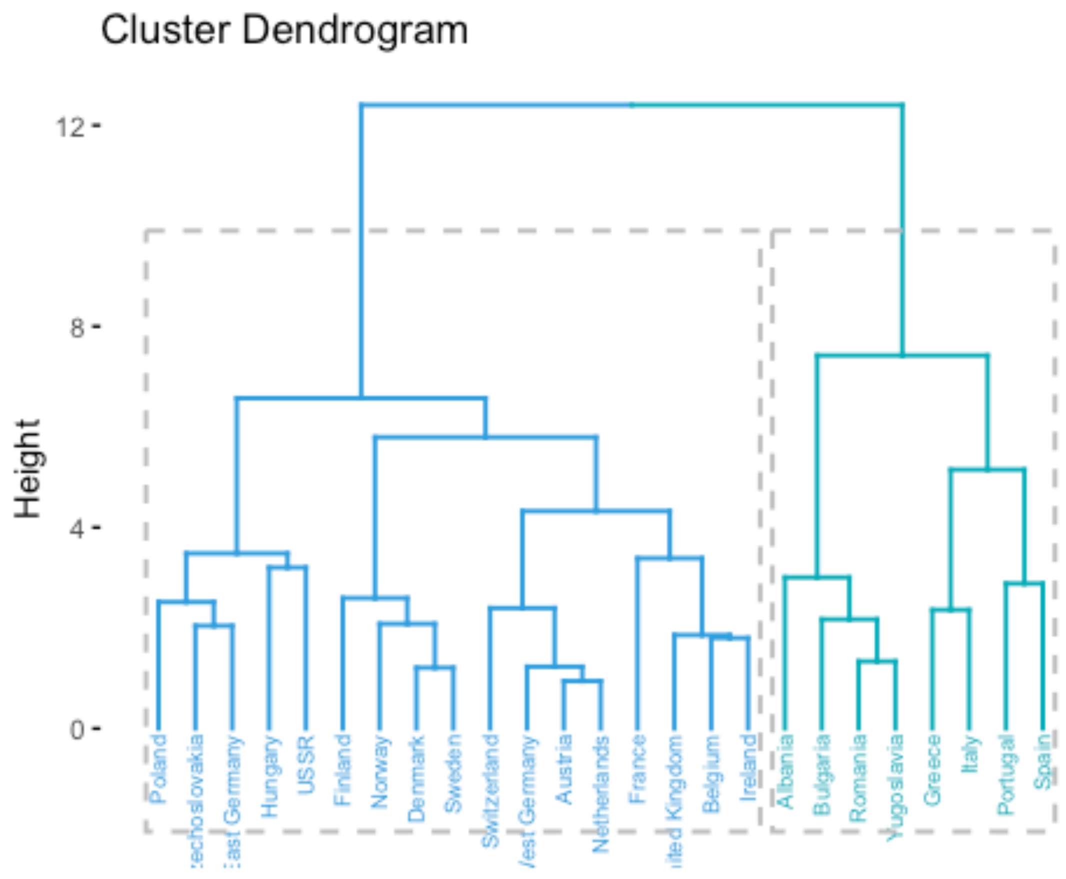
Visuals using 2 clsuters

```
pam.res <- pam(matstd_protein, 2)
# Visualize
fviz_cluster(pam.res)
```



Hierarchial Clustering

```
res.hc <- matstd_protein %>% scale() %>% dist(method = "euclidean") %>%
  hclust(method = "ward.D2")

fviz_dend(res.hc, k = 2, # Cut in four groups
          cex = 0.5, # label size
          k_colors = c("#2E9FDF", "#00AFBB", "#E7B800", "#FC4E07"),
          color_labels_by_k = TRUE, # color labels by groups
          rect = TRUE # Add rectangle around groups
          )

## Warning in get_col(col, k): Length of color vector was longer than the
number of
## clusters - first k elements are used

## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use
"none" instead as
## of ggplot2 3.3.4.
## ℹ The deprecated feature was likely used in the factoextra package.
##   Please report the issue at
<]8;;https://github.com/kassambara/factoextra/issueshttps://github.com/kassam
bara/factoextra/issues]8;;>.
```



- Clustering also aligns with our findings of the PCA.
- However, as we have 2 clusters, Balkan and Southern European countries are clustered together for dairy and meat protein intake, which is less.
- On the other hand, Northern, Eastern, and Western European countries are clustered, for they have dairy and meat intake as part of their protein intake.

In the final section, we can review additional support material for our findings and conclude.

5. Observations and Conclusion

- For both PCA and Clustering, we found that the given countries have been segregated into four segments – Balkan, Southern, Eastern European, and Northern & Western European.

- A deeper analysis is done to identify the root cause for this classification. The findings are as follows:

Balkan Countries:

- This group consists of Albania, Romania, Yugoslavia, and Bulgaria. These countries are geographically close and share similarities in their traditional diets. Balkan cuisine often includes a variety of cereals and bread based on wheat. Additionally, cereal growth is also higher in these countries due to its proximity to the Adriatic and Black Seas. Therefore, the protein diet in these countries will likely include a significant amount of cereal.

Southern European Countries:

- Portugal, Spain, Italy, and Greece fall into this group. Mediterranean cuisine characterises these countries' diets, rich in seafood, olive oil, fruits, vegetables, and grains. Therefore, the protein diet in these countries is likely to include a balance of fruits, vegetables, oil seeds, and pulses.

Eastern European Countries:

- The USSR, Hungary, Poland, East Germany, and Czechoslovakia are part of this group. These countries historically had diets influenced by Soviet food policies, which emphasized meat, potatoes, grains, and dairy. Meat, especially pork, was a staple in these diets, along with potatoes and bread. Additionally, traditional dishes often include soups and stews with meat and vegetables. Therefore, the protein diet in these countries likely consists of a significant amount of pork, beef, poultry, and dairy products (a mixture of all).

Western and Northern European Countries:

- This group includes Austria, Belgium, Denmark, inland, France, Ireland, Netherlands, Norway, Sweden, Switzerland, the United Kingdom, and West Germany. These countries have diverse diets, but common elements include a reliance on seafood (especially in coastal regions), dairy products, and a variety of meats such as beef, pork, and poultry. However, compared to the other groups, the emphasis on meat and dairy products is higher.

These groupings are based on geographical proximity, historical influences, and culinary traditions, all of which play a role in shaping the protein diets of these countries.