Exercise_1 solution

Tristan Koning

Sept. 16th, 2024

```
#install.packages("ggdendro")
library(ggplot2)
library(ggdendro)
library(cluster)
library(ggrepel)
par(mfrow = c(1, 1))
```

Problem 1

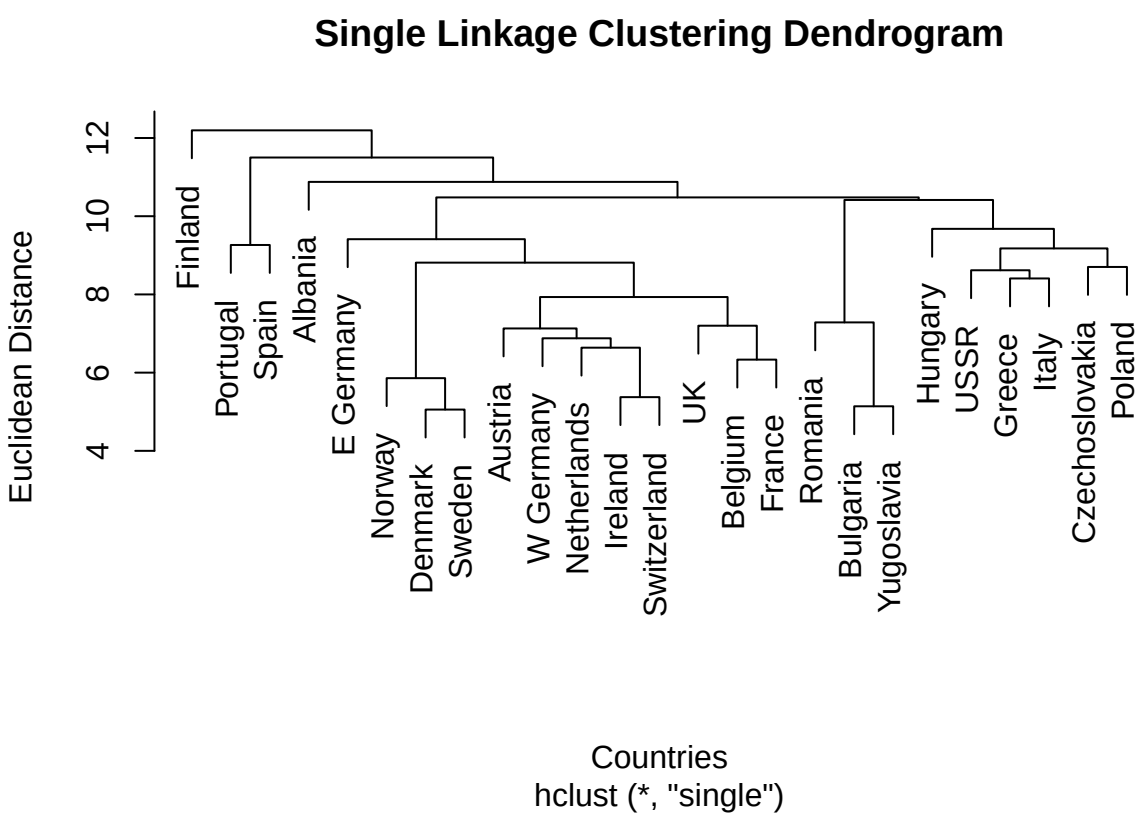
(a)

```
# Setup data for clustering
protein <- read.csv("data/protein.txt", sep = "\t", header = TRUE)
row.names(protein) <- protein$Country
#protein <- protein
#protein <- scale(protein)
```

```
# Single Linkage Clustering
# Setup
single_linkage <- hclust(dist(protein), method = "single")
```

```
## Warning in dist(protein): NAs introduced by coercion
```

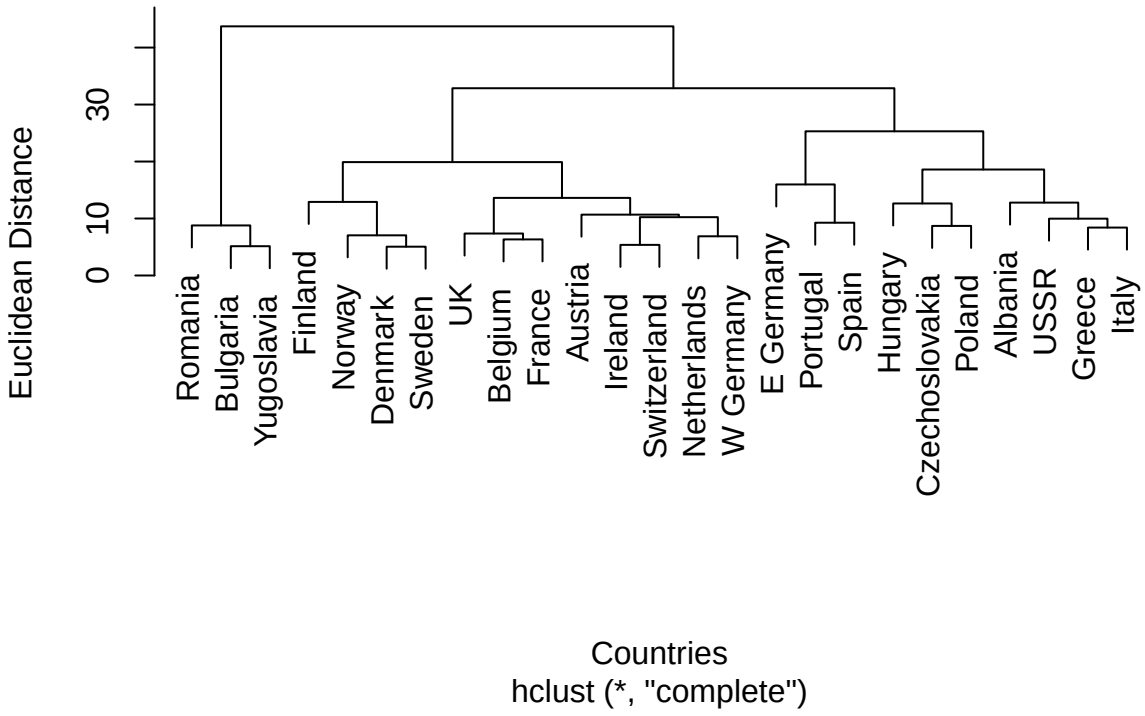
```
# Convert to dendrogram object
plot(single_linkage, main = "Single Linkage Clustering Dendrogram", xlab = "Countries", ylab = "Euclide
```




```
# Complete Linkage Clustering
# Setup
complete_linkage <- hclust(dist(protein), method = "complete")
```

```
## Warning in dist(protein): NAs introduced by coercion
```

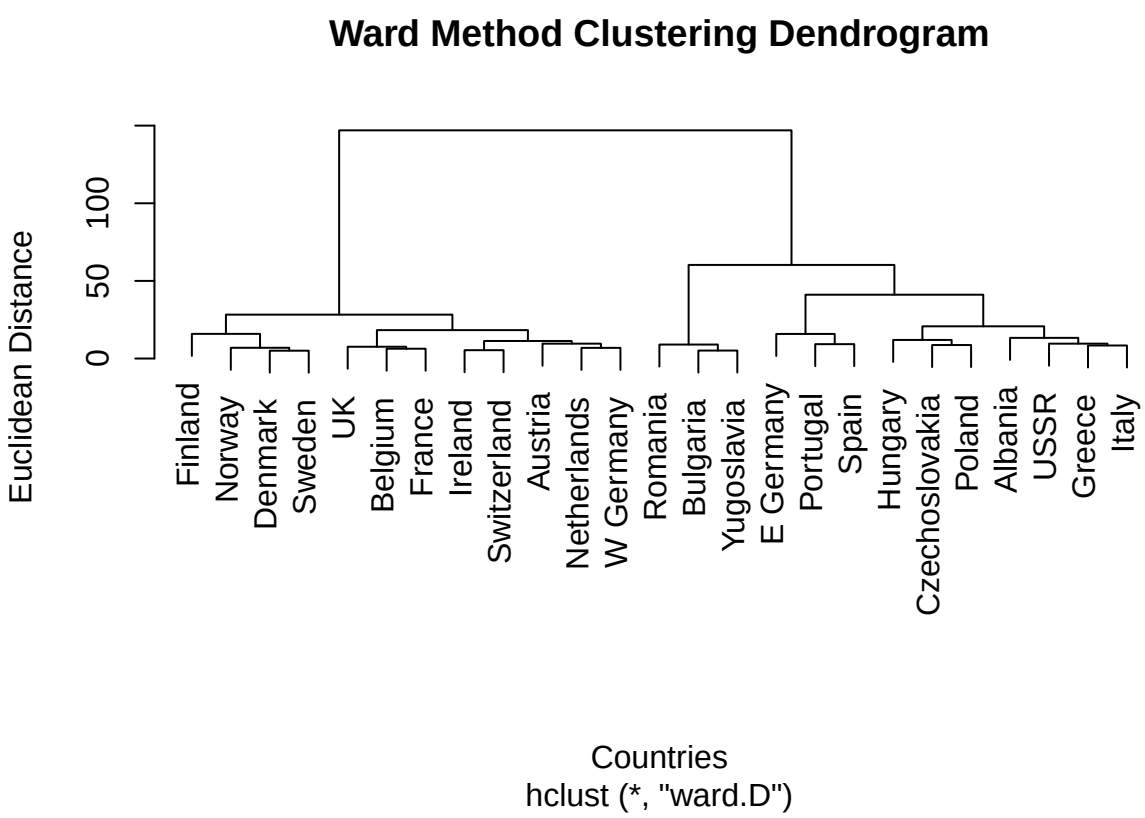
```
plot(complete_linkage, main = "Complete Linkage Clustering Dendrogram", xlab = "Countries", ylab = "Eucl
```




```
# Ward Method Clustering
# Setup
ward_linkage <- hclust(dist(protein), method = "ward.D")
```

```
## Warning in dist(protein): NAs introduced by coercion
```

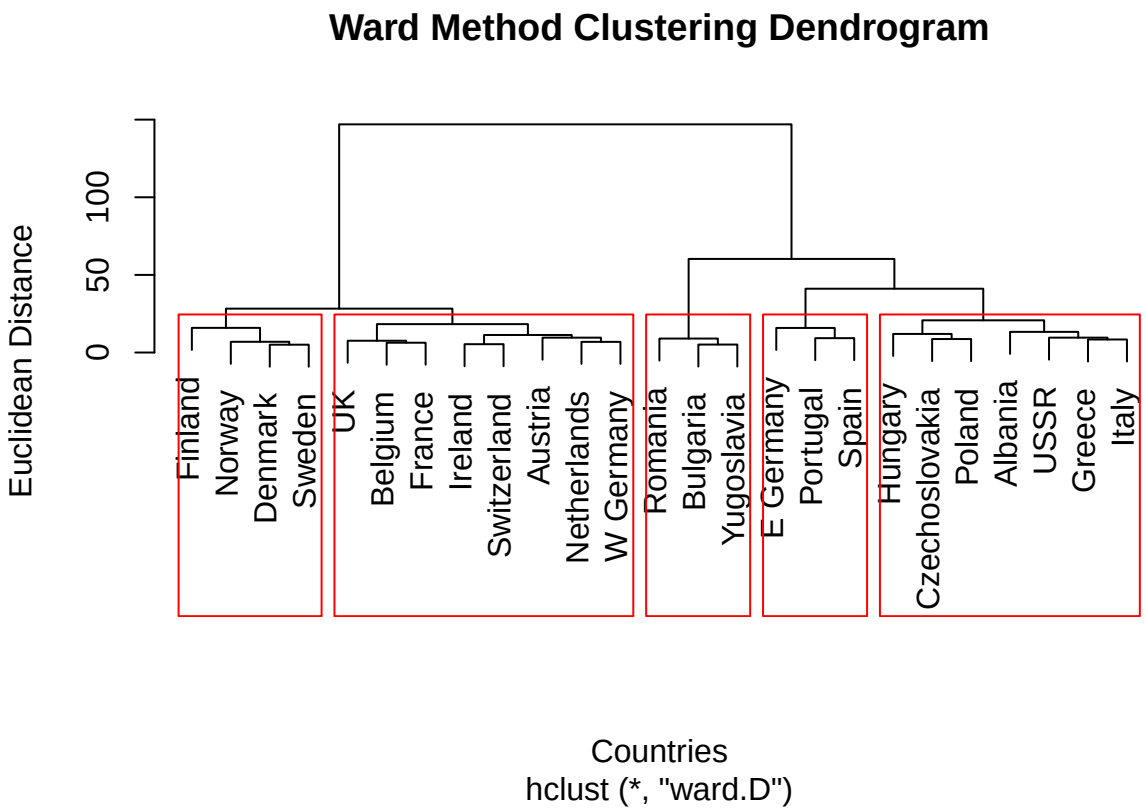
```
plot(ward_linkage, main = "Ward Method Clustering Dendrogram", xlab = "Countries", ylab = "Euclidean Dis
```



- We can observe that Ward method clustering shows the clearest structure in the dendrogram. It groups countries with small distances together early, and then forms larger clusters incrementally.
- The other two methods have a more erratic structure, with single linkage clustering showing the most erratic structure, where some countries dont belong to a cluster until the very end.

(b)

```
plot(ward_linkage, main = "Ward Method Clustering Dendrogram", xlab = "Countries", ylab = "Euclidean Dis
rect.hclust(ward_linkage, k = 5, border = "red")
```



Looking at the dendrogram of the Ward method, we can observe that the countries are getting grouped together by their geographic location. We can split the dendrogram into 5 clusters, which are shown in the plot by the red borders. The clusters are as follows: Scandinavia, Eastern Europe, Western Europe, The Balkans, and the Mediterranean.

(c)

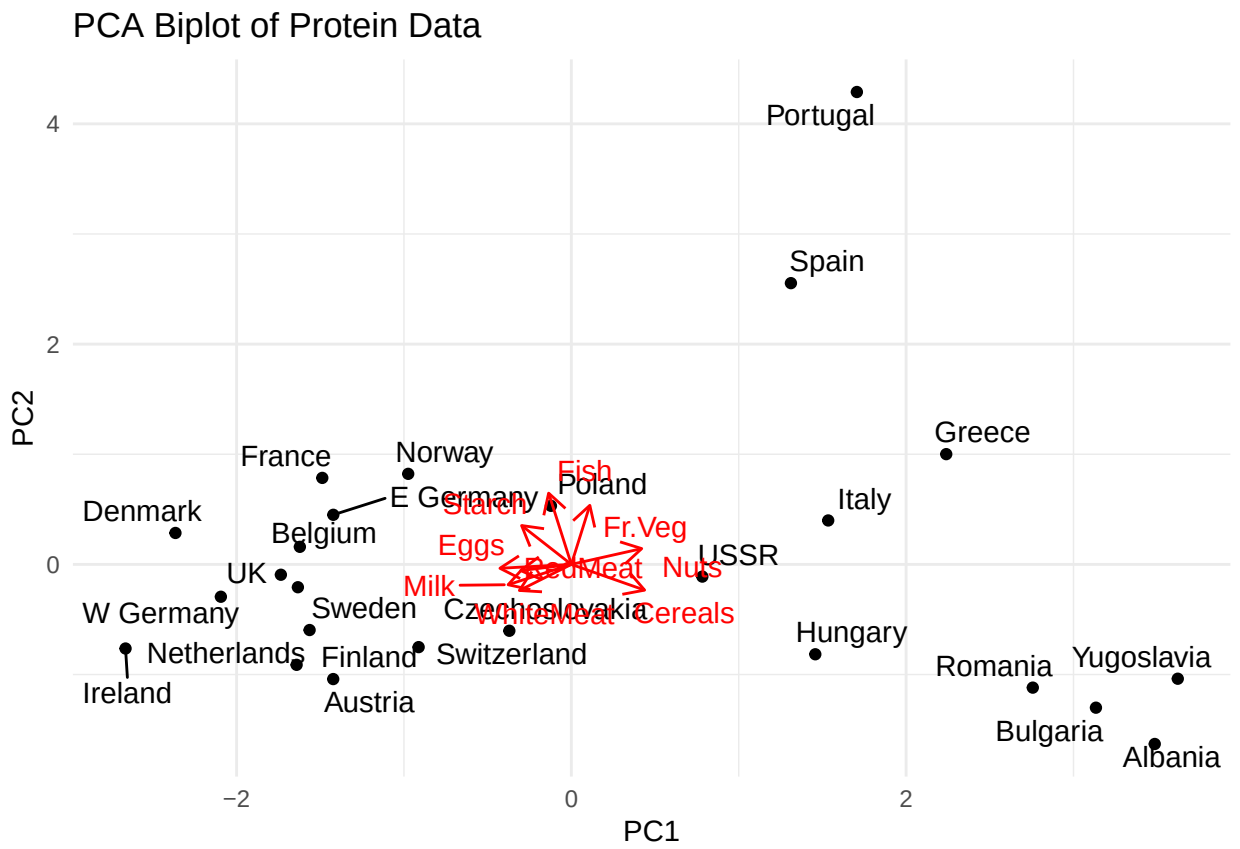
```
# Perform PCA on data
protein_pca <- protein
row.names(protein_pca) <- protein_pca$Country
protein_pca <- protein_pca[, -1]
protein_pca <- scale(protein_pca)
pca <- prcomp(protein_pca, scale = TRUE)

# Get loadings of PC
loadings <- pca$rotation
loadings_data <- data.frame(Variable = rownames(loadings), PC1 = loadings[, 1], PC2 = loadings[, 2])

# Perform Dimensionality reduction
scores <- pca$x
pca_data <- data.frame(Country = protein$Country, PC1 = scores[, 1], PC2 = scores[, 2])

ggplot() +
  geom_point(data = pca_data, aes(x = PC1, y = PC2)) +
```

```
  geom_text_repel(data = pca_data, aes(x = PC1, y = PC2, label = Country)) +
  geom_segment(data = loadings_data, aes(x = 0, y = 0, xend = PC1, yend = PC2),
               arrow = arrow(length = unit(0.3, "cm")), color = "red") +
  geom_text_repel(data = loadings_data, aes(x = PC1, y = PC2, label = Variable), color = "red") +
  labs(title = "PCA Biplot of Protein Data", x = "PC1", y = "PC2") +
  theme_minimal()
```



Looking at the plot of the Dimensionality-Reduction and the dendogram, we can observe that they build similar clusters, which are based on the geographic regions of europe, Although the PCA plot has more of a tendency to show Western vs. Eastern Europe in a socio-economic context, while the dendogram shows more of a geographic clustering.

Problem 2

```
set.seed(1)

# k-means function on 1-dimensional data vector x
my.kmean <- function(x, k) {
  # Init data frame to store cluster assignments
  cluster_assignments <- data.frame(x = x, cluster = rep(NA, length(x)))

  # Initialize k clusters with random centroids
  centroids <- runif(k, min(x), max(x))
```

```r
  for (i in 1:1000) {
    # Assign each data point to the nearest cluster
    cluster_assignments$cluster <- sapply(x, function(x) which.min(abs(x - centroids)))

    # Update centroids
    new_centroids <- rep(NA, k)

    for (cluster in 1:k) {
      points_in_cluster <- x[cluster_assignments$cluster == cluster]
      new_centroids[cluster] <- mean(points_in_cluster)
    }

    # Check for convergence
    if (all(centroids == new_centroids)) {
      break
    } else {
      centroids <- new_centroids
    }
  }

  print(paste("Centroids: ", centroids))
  print(paste("Converged after ", i, " iterations"))
  return(cluster_assignments)
}

my.kmean(c(1,2,1,3,2,6,5,7,6,12), 3)
```

```
## [1] "Centroids:  1.8" "Centroids:  6"   "Centroids:  12"
## [1] "Converged after  3  iterations"
```

```
##     x cluster
## 1   1       1
## 2   2       1
## 3   1       1
## 4   3       1
## 5   2       1
## 6   6       2
## 7   5       2
## 8   7       2
## 9   6       2
## 10 12       3
```