Multivariate Analysis
Homework #10

Aidan Dykstal
April 15th, 2020

Preliminaries

First, we load the cereal data:

```
# Load the Cereal Data
Cereal <- read.csv('cereal.csv')

# Remove Cereal Names
CerealNoNames <- Cereal[-1]

# Remove Cereal Manufacturers
CerealClear <- CerealNoNames[-1]
```

Now, we are ready to proceed. Note that we remove cereal names and brand labels because agglomerative clustering is unsupervised.

Next, we will write a function that counts how many items are in each cluster for an arbitrary clustering.

```
# Create the Cluster Membership Counter
countMembersPerCluster <- function(clustering) {
  maxClustNum <- max(clustering)
  counts <- rep(0, maxClustNum)
  for (i in 1:length(clustering)) {
    counts[clustering[i]] <- counts[clustering[i]] + 1
  }
  return(counts)
}
```

We're ready to go now!

Problem 1

We will use agglomerative clustering with complete linkage to cluster the Cereal data using Euclidean distances. We will cluster into 2, 3, and 4 clusters, respectively.

```
# Compute the Matrix of Euclidean Distances
dCereal <- dist(CerealClear)

# Cluster Using Complete Linkage with Agglomerative Clustering
cerealClust <- hclust(dCereal, method = "complete")

# Specify Two Clusters for Now
c2 <- cutree(cerealClust, 2)
c2
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 1 2 1 1 2 1 2 1 1 1 1 1 1 1 1 2 1 1
## [39] 1 1 2 2 2
```

```
# Now Specify Three Clusters
c3 <- cutree(cerealClust, 3)
c3
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 2 1 2 1 3 1 1 2 3 1 3 2 1 3 1 3 2 1 2 1 1 1 1 2 1 3 1 1
## [39] 1 1 3 3 3
```

```
# Now Specify Four Clusters
c4 <- cutree(cerealClust, 4)
c4
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 2 1 2 1 3 1 1 4 3 1 3 2 1 3 1 3 2 1 2 1 1 1 1 2 1 3 1 1
## [39] 1 1 3 3 3
```

In the outputs above, the numbers denote which group a given observation is clustered to in each case. For instance, 1 is group/cluster 1, 2 is group/cluster 2, and so on. There are 43 observations in total for each output.

Next, we will count the number of items assigned to each cluster for each number of clusters.

```
# Count the Number of Items per Cluster
count2 <- countMembersPerCluster(c2)
count3 <- countMembersPerCluster(c3)
count4 <- countMembersPerCluster(c4)

# Report these Numbers
cat("Two Clusters: \nClust. 1 had",
    count2[1], "\nClust. 2 had",
    count2[2])
```

```
## Two Clusters:
## Clust. 1 had 34
## Clust. 2 had 9
```

```
cat("Three Clusters: \nClust. 1 had",
    count3[1], "\nClust. 2 had",
    count3[2], "\nClust. 3 had",
    count3[3])
```

```
## Three Clusters:
## Clust. 1 had 27
## Clust. 2 had 7
```

```
## Clust. 3 had 9
```

```
cat("Four Clusters: \nClust. 1 had",
    count4[1], "\nClust. 2 had",
    count4[2], "\nClust. 3 had",
    count4[3], "\nClust. 4 had",
    count4[4])
```

```
## Four Clusters:
## Clust. 1 had 27
## Clust. 2 had 6
## Clust. 3 had 9
## Clust. 4 had 1
```

In the outputs above, the number of items assigned to each cluster for each number of total clusters is given.

Problem 2

We will now cluster the brands into 2, 3, and 4 groups using k-means clustering.

```
# Use k-Means Clustering
# Use k = 2 for Now
k2 <- kmeans(CerealClear, 2, nstart = 10)
ssw2 <- sum(k2$withinss)
c2k <- k2$cluster
c2k
```

```
##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 1 2 2 1 2 1 2 2 2 2 2 2 2 2 2 1 2 2
## [39] 2 2 1 1 1
```

```
# Now Use k = 3
k3 <- kmeans(CerealClear, 3, nstart = 10)
ssw3 <- sum(k3$withinss)
c3k <- k3$cluster
c3k
```

```
##  [1] 3 3 3 3 3 3 3 3 3 2 3 2 3 3 3 3 2 1 3 1 2 3 1 3 1 2 3 2 3 3 3 3 2 3 1 3 3
## [39] 3 3 1 1 1
```

```
# Now Use k = 4
k4 <- kmeans(CerealClear, 4, nstart = 10)
ssw4 <- sum(k4$withinss)
c4k <- k4$cluster
c4k
```

```
##  [1] 2 3 2 2 3 3 3 2 2 2 2 3 1 2 2 2 2 1 2 3 4 2 3 2 3 4 1 2 2 2 2 2 3 1 3 4 3 3
## [39] 3 2 4 4 4
```

In the outputs above, the numbers denote which group a given observation is clustered to in each case. For instance, 1 is group/cluster 1, 2 is group/cluster 2, and so on. There are 43 observations in total for each output.

Next, we will count the number of items assigned to each cluster for each number of clusters.

```
# Count the Number of Items per Cluster
count2 <- countMembersPerCluster(c2k)
count3 <- countMembersPerCluster(c3k)
count4 <- countMembersPerCluster(c4k)

# Report these Numbers
cat("Two Clusters: \nClust. 1 had",
    count2[1], "\nClust. 2 had",
    count2[2])
```

```
## Two Clusters:
## Clust. 1 had 9
## Clust. 2 had 34
```

```
cat("Three Clusters: \nClust. 1 had",
    count3[1], "\nClust. 2 had",
    count3[2], "\nClust. 3 had",
    count3[3])
```

```
## Three Clusters:
## Clust. 1 had 8
## Clust. 2 had 7
```

```
## Clust. 3 had 28
```

```
cat("Four Clusters: \nClust. 1 had",
    count4[1], "\nClust. 2 had",
    count4[2], "\nClust. 3 had",
    count4[3], "\nClust. 4 had",
    count4[4])
```

```
## Four Clusters:
## Clust. 1 had 4
## Clust. 2 had 20
## Clust. 3 had 13
## Clust. 4 had 6
```

In the outputs above, the number of items assigned to each cluster for each number of total clusters is given.

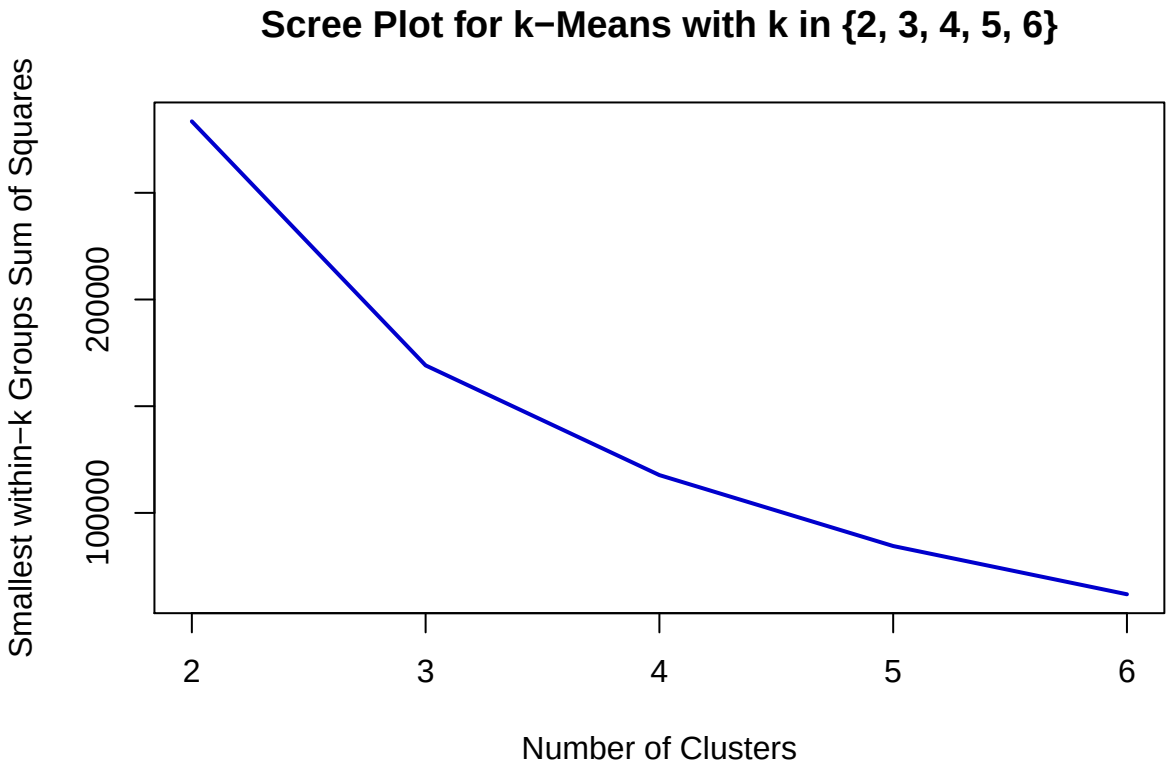
Problem 3

We will build a scree plot for the k-means clusters using two through six clusters. First, we fit k-means using $k = 2$ through $k = 6$.

```
# Use k-Means Clustering with k = 2, ..., 6
# Set Up within Sums of Squares
ssw <- rep(0, 5)
for (k in 2:6) {
  kObj <- kmeans(CerealClear, k, nstart = 10)
  ssw[k - 1] <- sum(kObj$withinss)
}
```

Now, we make the scree plot:

```
# Make the Scree Plot
plot(c(2:6), ssw, type = 'l',
     col = 'blue3', lwd = 2,
     xlab = 'Number of Clusters',
     ylab = 'Smallest within-k Groups Sum of Squares',
     main = 'Scree Plot for k-Means with k in {2, 3, 4, 5, 6}')
```



In this scree plot for the k-means clusters using two through six clusters, the plot does not make a very clear suggestion as to the number of clusters to use. This is because there is no point at which the decrease in slope for our plot is particularly dramatic. If forced to choose, we should likely use three or four clusters since the slope decreases most noticeably when k is near 3 and 4. This means that – as we add more than 3 or 4 clusters – we decrease the SSW, but not so noticeably as to provide significantly more meaningful results at the cost of adding an extra cluster.