Vowel

Georgios Papadopoulos Vasileios Papadopoulos

Load Dataset

```
vowel <- read.arff("vowel.arff")
head(vowel)
```

```
##   Train or Test Speaker Number  Sex Feature 0 Feature 1 Feature 2 Feature 3
## 1         Train         Andrew Male    -3.639     0.418    -0.670     1.779
## 2         Train         Andrew Male    -3.327     0.496    -0.694     1.365
## 3         Train         Andrew Male    -2.120     0.894    -1.576     0.147
## 4         Train         Andrew Male    -2.287     1.809    -1.498     1.012
## 5         Train         Andrew Male    -2.598     1.938    -0.846     1.062
## 6         Train         Andrew Male    -2.852     1.914    -0.755     0.825
##   Feature 4 Feature 5 Feature 6 Feature 7 Feature 8 Feature 9 Class
## 1    -0.168     1.627    -0.388     0.529    -0.874    -0.814   hid
## 2    -0.265     1.933    -0.363     0.510    -0.621    -0.488   hId
## 3    -0.707     1.559    -0.579     0.676    -0.809    -0.049   hEd
## 4    -1.053     1.060    -0.567     0.235    -0.091    -0.795   hAd
## 5    -1.633     0.764     0.394    -0.150     0.277    -0.396   hYd
## 6    -1.588     0.855     0.217    -0.246     0.238    -0.365   had
```

Drop columns

```
#vowel_drop <- subset(vowel, select = -c("Train or Test", "Speaker Number", "Sex", "Class"))
vowel_drop <- select(vowel, -1, -2, -3, -14)
head(vowel_drop)
```

```
##   Feature 0 Feature 1 Feature 2 Feature 3 Feature 4 Feature 5 Feature 6
## 1    -3.639     0.418    -0.670     1.779    -0.168     1.627    -0.388
## 2    -3.327     0.496    -0.694     1.365    -0.265     1.933    -0.363
## 3    -2.120     0.894    -1.576     0.147    -0.707     1.559    -0.579
## 4    -2.287     1.809    -1.498     1.012    -1.053     1.060    -0.567
## 5    -2.598     1.938    -0.846     1.062    -1.633     0.764     0.394
## 6    -2.852     1.914    -0.755     0.825    -1.588     0.855     0.217
##   Feature 7 Feature 8 Feature 9
## 1     0.529    -0.874    -0.814
## 2     0.510    -0.621    -0.488
## 3     0.676    -0.809    -0.049
## 4     0.235    -0.091    -0.795
## 5    -0.150     0.277    -0.396
## 6    -0.246     0.238    -0.365
```

K Means

K-means algorithm works as presented below:

1. Choose groups in the feature plan randomly
2. Minimize the distance between the cluster center and the different observations (centroid). It results in groups with observations
3. Shift the initial centroid to the mean of the coordinates within a group.
4. Minimize the distance according to the new centroids. New boundaries are created. Thus, observations will move from one group to another
5. Repeat until no observation changes groups

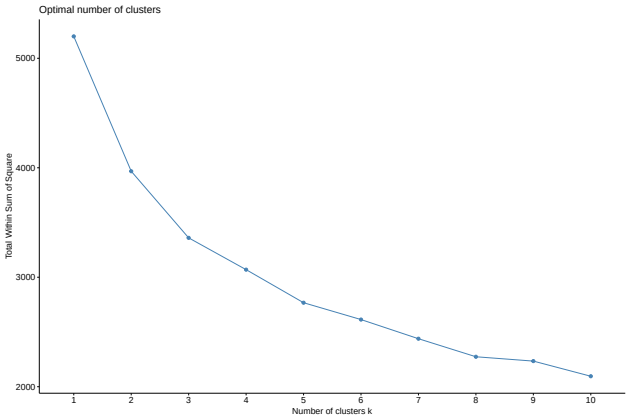
Since M-means is based on distances of among data points, there few different measures it take into account. For example, *Euclidean* distance is the most common method, though we could also use *Manhattan* and *Minlowski* distances as well. Below, present the mathematical formulation of *Euclidean* distance.

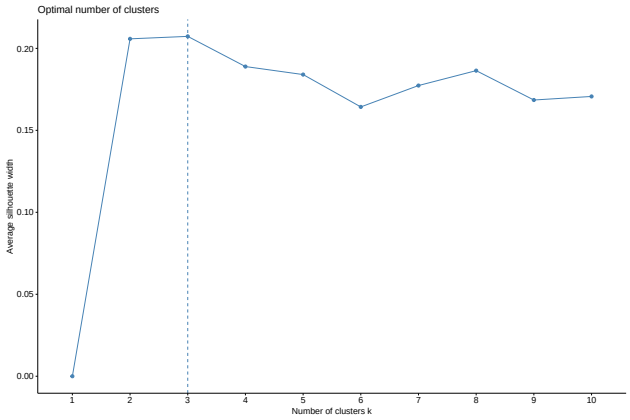
$$distance(x, y) = \sum_{i}^{n} (x_i - y_i)^2 \tag{1}$$

Optimal K

One technique to choose the best k is called the elbow method. This method uses within-group homogeneity or within-group heterogeneity to evaluate the variability. Another approach is called *Silhouette*. We will measure within groups sum of squares(variance) and the quality to clusters to determine the optimal value of k. For this, we will install the package *factoextra*.

```
set.seed(123)
fviz_nbclust(vowel_drop, kmeans, method = "wss")
fviz_nbclust(vowel_drop, kmeans, method = "silhouette")
```





We see the best value of k is 3.