2. Segmentation

Introduction

For this task, we must use unsupervised learning to split the data into categories.

Common methods used to cluster data are k-means and hierarchical clustering.

We should not use hierarchical clustering in this case because:

1. It does not scale well with large data sets.
2. We are only interested in identifying patterns, not how similar individual observations are to other observations in the same cluster.

Thus, we should go about using k-means clustering

> Study.A <-read.csv(file.choose()) # Choose Study\_A.csv

> Study.B <-read.csv(file.choose()) # Choose Study\_B.csv

> Study.C <-read.csv(file.choose()) # Choose Study\_C.csv

> Study.D <-read.csv(file.choose()) # Choose Study\_D.csv

> Study.E <-read.csv(file.choose()) # Choose Study\_E.csv

We will use Day 0 observations of datasets A-E and group them using k-means.

> Study.A <-subset(Study.A, VisitDay == 0)

> Study.B <-subset(Study.B, VisitDay == 0)

> Study.C <-subset(Study.C, VisitDay == 0)

> Study.D <-subset(Study.D, VisitDay == 0)

> Study.E <-subset(Study.E, VisitDay == 0)

K-means can only be used on numeric values (in this case, the 30 features with values 1-7).

All other features, such as study group, country, ID’s, lead status, etc., can be disregarded when clustering the data (we will keep the study group temporarily for the initial observations).

We can also disregard the PANSS total because it yields no new information if we already have the individual PANSS values.

> Study.A <- Study.A[, -which(names(Study.A) %in% c("Country", "PatientID", "SiteID", "RaterID", "AssessmentID", "TxGroup", "VisitDay", "PANSS\_Total", "LeadStatus"))]

> Study.B <- Study.B[, -which(names(Study.B) %in% c("Country", "PatientID", "SiteID", "RaterID", "AssessmentID", "TxGroup", "VisitDay", "PANSS\_Total", "LeadStatus"))]

> Study.C <- Study.C[, -which(names(Study.C) %in% c("Country", "PatientID", "SiteID", "RaterID", "AssessmentID", "TxGroup", "VisitDay", "PANSS\_Total", "LeadStatus"))]

> Study.D <- Study.D[, -which(names(Study.D) %in% c("Country", "PatientID", "SiteID", "RaterID", "AssessmentID", "TxGroup", "VisitDay", "PANSS\_Total", "LeadStatus"))]

> Study.E <- Study.E[, -which(names(Study.E) %in% c("Country", "PatientID", "SiteID", "RaterID", "AssessmentID", "TxGroup", "VisitDay", "PANSS\_Total", "LeadStatus"))]

Combine into one dataset.

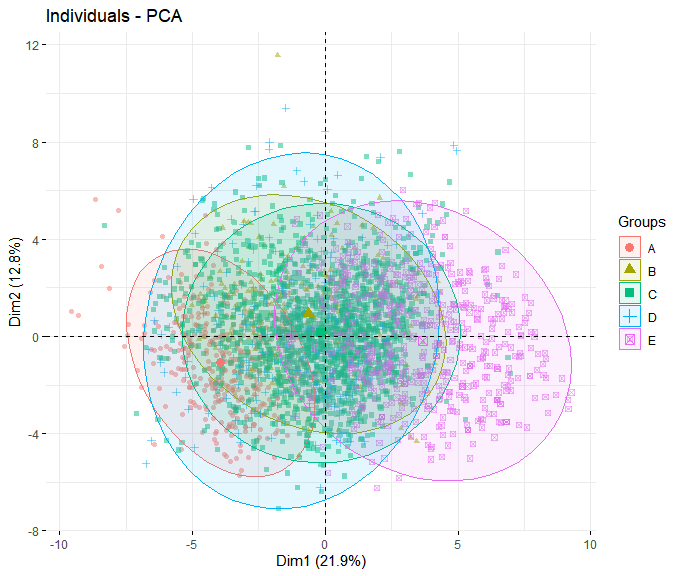
> Combined.Studies <- rbind(Study.A, Study.B, Study.C, Study.D, Study.E)

Initial Observations

Before determining the optimal k value for k-means clustering and implementing it, let’s first view what trends occur in each study group when viewing the data along 2 principal components with the fviz and prcomp functions.

> library(factoextra)

> fviz(prcomp(Combined.Studies[, -1]), "ind", label = "none", habillage = Combined.Studies$Study, addEllipses = "True", alpha=0.5)



This provides a rough visualization of the clusters of observations based on study. It can be seen that while Studies A-D are relatively close to each other, Study E noticeably has a much higher average first principal component value. Study A also has a lower average first principal component value. What this means will be covered later with further analysis of the principal component values.

K-Means Clustering

Now that we chose k-means as the preferred method, we must know how many clusters to choose. This can be done with the help of the factoextra library. Specifically, we can use the fviz\_nbclust method to gather more information on choosing the optimal value of k for the k-means clustering method. We can also get rid of the study values now.

> Study.A <- Study.A[, -which(names(Study.A) %in% c("Study"))]

> Study.B <- Study.B[, -which(names(Study.B) %in% c("Study"))]

> Study.C <- Study.C[, -which(names(Study.C) %in% c("Study"))]

> Study.D <- Study.D[, -which(names(Study.D) %in% c("Study"))]

> Study.E <- Study.E[, -which(names(Study.E) %in% c("Study"))]

> Combined.Studies <- Combined.Studies[, -which(names(Combined.Studies) %in% c("Study"))]

K-means is sensitive to outliers and noisy data if not standardized, and standardizing the data accounts for the different variances of each feature. Instead of measuring the raw value, the data should be measured by the number of standard deviations from the mean.

> Study.A <- scale(Study.A)

> Study.B <- scale(Study.B)

> Study.C <- scale(Study.C)

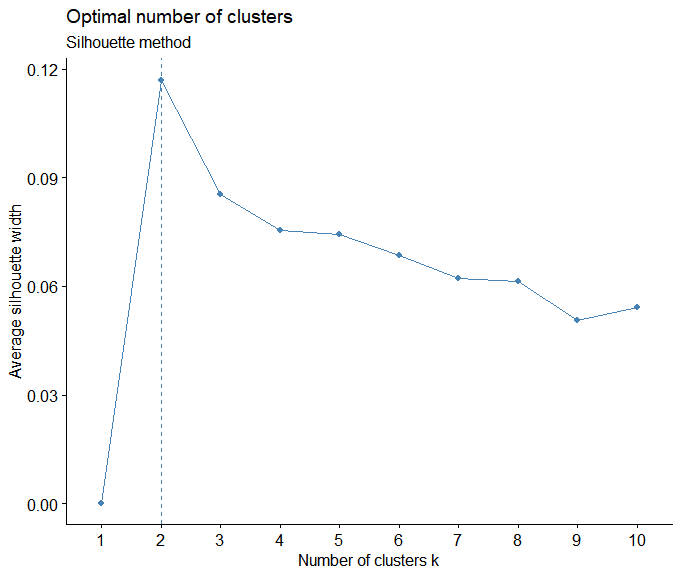
> Study.D <- scale(Study.D)

> Study.E <- scale(Study.E)

> Combined.Studies <- scale(Combined.Studies)

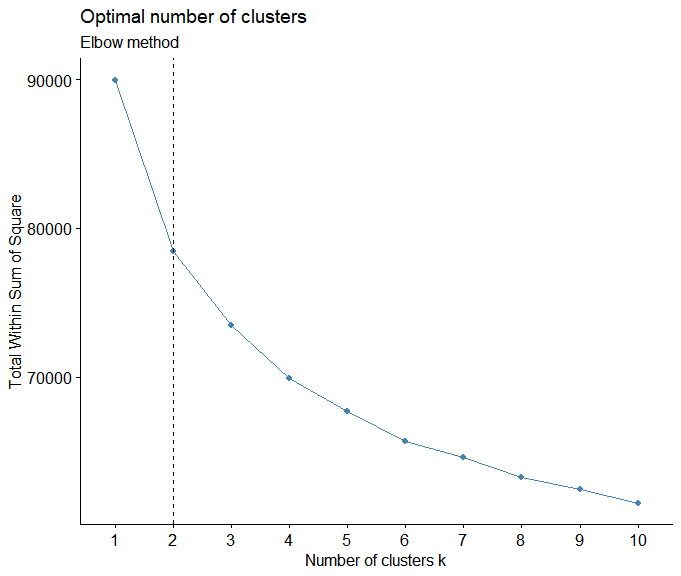
Average Silhouette Method – measures how well each observation is contained within its cluster. A higher average silhouette width generally means a good clustering. In this case, the appropriate number of clusters to choose is 2.

> fviz\_nbclust(Combined.Studies, kmeans, method = "silhouette") + labs(subtitle = "Silhouette method")



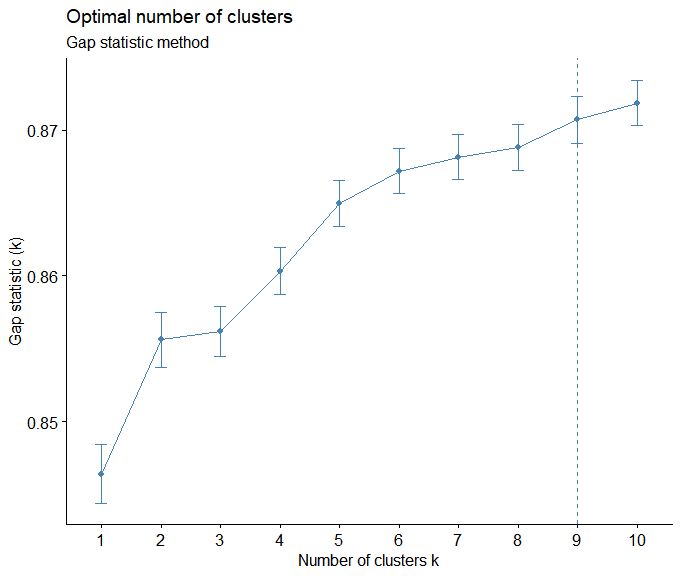
Elbow Method – defines clusters by minimizing the within sum of squares error (WSS) as a function of the number of clusters. WSS measures how close together a cluster is, and generally a smaller WSS is better. As for selecting an appropriate number of clusters, locations of a bend (in this case at 2 clusters) is considered an appropriate number.

> fviz\_nbclust(Combined.Studies, kmeans, method = "wss") + geom\_vline(xintercept = 2, linetype = 2) + labs(subtitle = "Elbow method")



Gap Statistic Method – measures the within intra-cluster variation as a function of the number of clusters chosen. For this method, the ideal number of clusters is the smallest value of k such that its gap statistic is within one standard deviation of the value that maximizes the gap statistic. The ideal number here is 9 since it is the greatest value smaller than 10 that is within 1 standard deviation. However, we can see that at 2 clusters, the gap value is comparable to the gap value at 3 clusters since the difference is somewhat negligible.

> fviz\_nbclust(Combined.Studies, kmeans, nstart = 25, method = "gap\_stat", nboot = 100) + labs(subtitle = "Gap statistic method")



NbClust() – The final way we can determine the optimal number of clusters to use is by using this function from the NbClust library. It provides 30 indices for determine the appropriate number of clusters, and determines the number of clusters using a majority vote. More information about the 30 indices can be found in the NbClust library documentation.

> library(NbClust)

> NbClust(data = Combined.Studies, distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans")

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Among all indices:

\* 11 proposed 2 as the best number of clusters

\* 4 proposed 3 as the best number of clusters

\* 5 proposed 4 as the best number of clusters

\* 1 proposed 5 as the best number of clusters

\* 1 proposed 6 as the best number of clusters

\* 1 proposed 10 as the best number of clusters

\*\*\*\*\* Conclusion \*\*\*\*\*

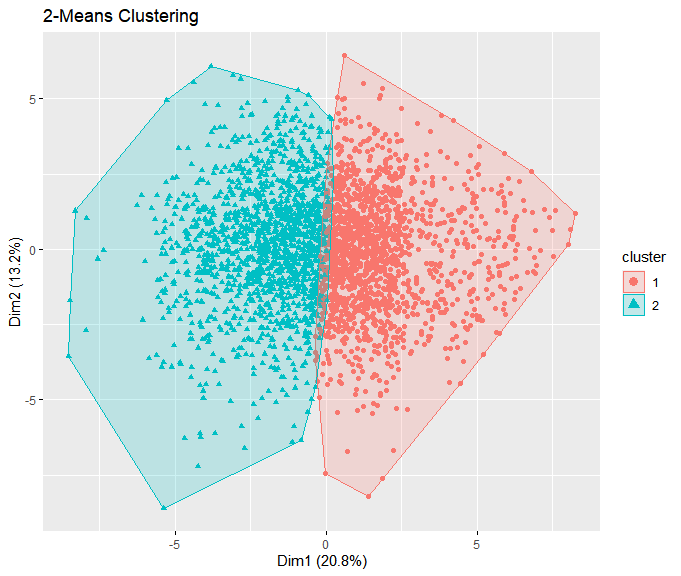
\* According to the majority rule, the best number of clusters is 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Now that we choose k = 2, we can partition the dataset using 2-means clustering.

> kmeans.clust <- kmeans(Combined.Studies, 2, nstart = 100)

> fviz\_cluster(kmeans.clust, Combined.Studies, geom = c("point")) + labs(title = "2-Means Clustering")



There are only two dimensions shown here from a reduction of 30 dimensions because the k-means clustering algorithm shows the data in a system of two principal components. Clusters 1 and 2 can be separated fairly well along where the first principal component switches from positive to negative (where Dim1 0).

Analysis

From the k-means clustering with k = 2, we can see that the data forms two distinct groups when visualized using two principal components. However, we do not yet know what these two principal components mean in terms of the original features of the data. This may be useful to know since then we can draw conclusions on the reasoning behind how these two groups are formed.

For example, because we know that the two clusters are separated along the first principal component (where Dim1 0), if we knew which features contributed most to a change in the first principal component, we could figure out how exactly these two groups of patients are split.

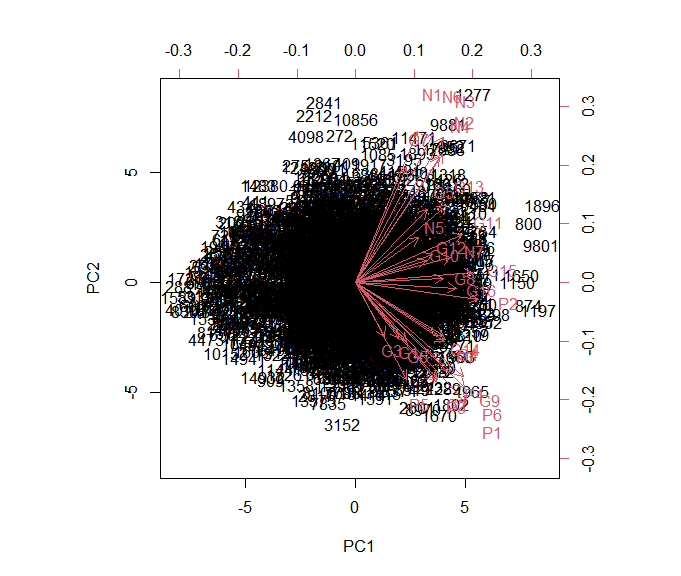
To dig further on this matter, a principal component analysis using the built-in R function prcomp() can be used again to determine the contributions each feature makes to each principal component.

> results <- prcomp(Combined.Studies)

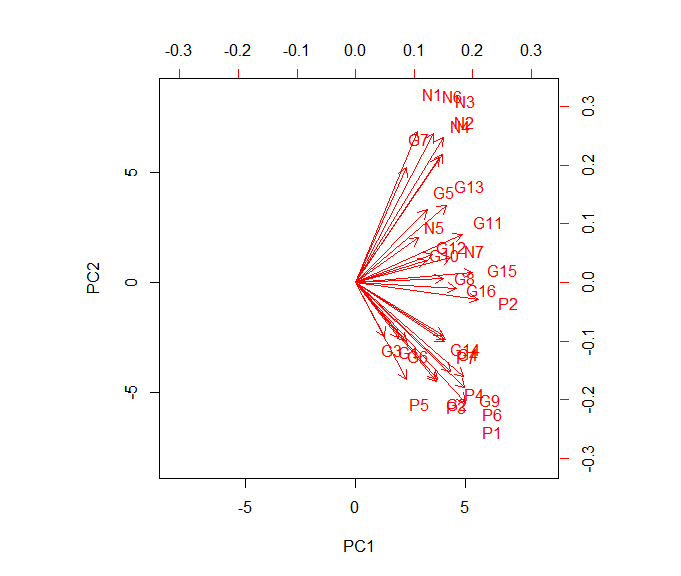
> results$rotation <- -1 \* results$rotation #Eigenvectors in R by default point in the negative direction. This reverses the signs.

> results$x <- -1 \* results$x #This reverses the signs of the principal component scores for each observation as well.

> biplot(results, scale = 0) #This is a little cluttered, we can visualize the graph without the observation points.



> biplot(results, col = c("white", "red"), scale = 0)



From this, we can see the proportion that each feature contributes to the two principal components. Since we are mostly interested in the first principal value since the clusters are split near when its value 0, we can see that the features that contribute most to PC1 (vectors furthest right on the graph) are P2, G15, P1, P6, G9, G11, P4, G4, P7, etc. This shows that the clusters are mainly divided on how patients scored on the positive and general symptoms. Indeed, a quick look at the trends in each cluster show that the left cluster (cluster 2) has mostly patients who scored higher on these symptoms while the right cluster (cluster 1) has mostly patients who scored lower on these symptoms.

> kmeans.clust$centers

P1 P2 P3 P4 P5 P6

1 -0.4724259 -0.5346738 -0.3816866 -0.4048736 -0.2373839 -0.4758130

2 0.5083891 0.5753756 0.4107423 0.4356944 0.2554547 0.5120341

P7 N1 N2 N3 N4 N5

1 -0.3946165 -0.2128975 -0.3249632 -0.2962271 -0.3168242 -0.2546761

2 0.4246565 0.2291042 0.3497008 0.3187773 0.3409423 0.2740633

N6 N7 G1 G2 G3

1 -0.2698701 -0.3386144 -0.1385484 -0.3177588 -0.09534452

2 0.2904139 0.3643912 0.1490953 0.3419481 0.10260258

G4 G5 G6 G7 G8 G9

1 -0.3395045 -0.2556141 -0.1872483 -0.1533706 -0.3787357 -0.4751995

2 0.3653491 0.2750726 0.2015025 0.1650459 0.4075667 0.5113738

G10 G11 G12 G13 G14 G15

1 -0.2861119 -0.3900442 -0.3216501 -0.3415269 -0.3713398 -0.4730922

2 0.3078921 0.4197361 0.3461356 0.3675255 0.3996078 0.5091062

G16

1 -0.4152765

2 0.4468892

As for the initial observations from earlier, we can now explain them using this new information. From the principal component analysis, we can see that points with a higher first principal component value tend to score poorly on the positive and general symptoms. Since much of Study E lies in cluster 1, we can conclude that the study differs from the other studies due to patients scoring lower in these positive and general symptoms on average. The opposite can be said for Study A, where the patients score higher on these symptoms on average.