PHP2650: Statistical Learning and Big Data

Assignment 3 - Clustering

Antonella Basso

March 22, 2022

Problem 1: Group Presentations

View the slides from another group's clustering presentation on Canvas that intrigued you. Write a 2-3 paragraph summary about the method, why it interested you, and what questions you have remaining. Additionally, compare this method to the one you presented on. Is there any data/question you've maybe seen before that you think would be a good fit for this method? You may also use other references, but be sure to cite them.

Solution

Clustering Algorithm:

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Summary: DBSCAN is a density-based clustering algorithm, meaning that it identifies clusters according to "regions which grow with high density."[1] Specifically, DBSCAN attempts to locate arbitrarily-shaped "dense" and "sparse" regions to cluster the data based on density and level of noise, respectively. For this reason, unlike other clustering algorithms, it does not require that every data point be assigned to a cluster (some points may just be classified as outliers or noise).[2] Additionally, this method takes two parameters to guide the clustering process; Epsilon (ϵ), and "Min Points". Namely, Epsilon is the specified radius of a neighborhood around some arbitrary point p, which must contain at least Min Points within it to be considered a "core point", making it part of a cluster's interior. If p fails to meet these criteria, it is classified as either; a "border point", meaning that the number of points within its neighborhood radius is less than Min Points, but the distance between it and a core point is less than ϵ; or a "noise point" (outlier), otherwise.[3] DBSCAN assesses every point in the data in this way to give its classification label (core point, border point, or outlier) and provide the resulting clusters identified (dense regions containing all core and border points). Given the nature of this approach, DBSCAN is not only great at handling very large data (reasonable computational costs), but it is particularly effective in identifying outliers/noise as well as non-spherical data patterns. In spite of the many benefits of this approach however, some of its limitations include needing user-defined parameters (despite not needing a specified number of clusters), and producing inaccurate results when data reflects a variety of different densities. Additionally, while there are methods out there for determining appropriate values for Epsilon and Min Points (such as k-Nearest Neighbors), DBSCAN is very sensitive to parameters and could produce unreliable outcomes.[3]

Comparison to BIRCH: Both BIRCH and DBSCAN are appealing approaches to clustering, primarily in being able to handle large data sets and high dimensionality. Still however, BIRCH's time complexity of $O(n)$ is a step above DBSCAN's time complexity of $O(n \log n)$, putting it at a slight advantage. Moreover, the two methods differ in that BIRCH requires user-specification of the number of clusters to be identified (along with a threshold value and a branching factor), while DBSCAN finds all naturally-occuring data patterns (for given values of Epsilon and Min Points). Although BIRCH and DBSCAN have fundamentally different clustering procedures, both algorithms are known for their superior handling of large noisy data.

Questions Remaining:

- Is there (or could there be) a way to account for varying data densities to produce better results in such cases?
- Is there a simple way of gauging the algorithms performance for different threshold values? How could we know whether one threshold value is better than another?

Problem 2: Clustering Objective Functions

Consider a matrix M where $M_{i,j}$ is the **dissimilarity** between point i and j. One clustering objective function might be to maximize the sum of dissimilarities of points put into different clusters. Let $\mathcal{C} = \{C_1, C_2, ..., C_k\}$ be a k-way partition of the data. Then, the k-way Max Cut clustering objective is:

$$\max_{\mathcal{C}} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \sum_{x \in C_i, y \in C_j} M_{i,j}$$

As an example of a dissimilarity, we could take $M_{i,j} = ||x_i - x_j||^2$ to be the squared Euclidean distance between points. Under this case, compare the k-means and k-way max cut objective functions. How might a researcher choose between these two objective functions? Illustrate your point by generating data for which the two objective functions would find different optimal clusters.

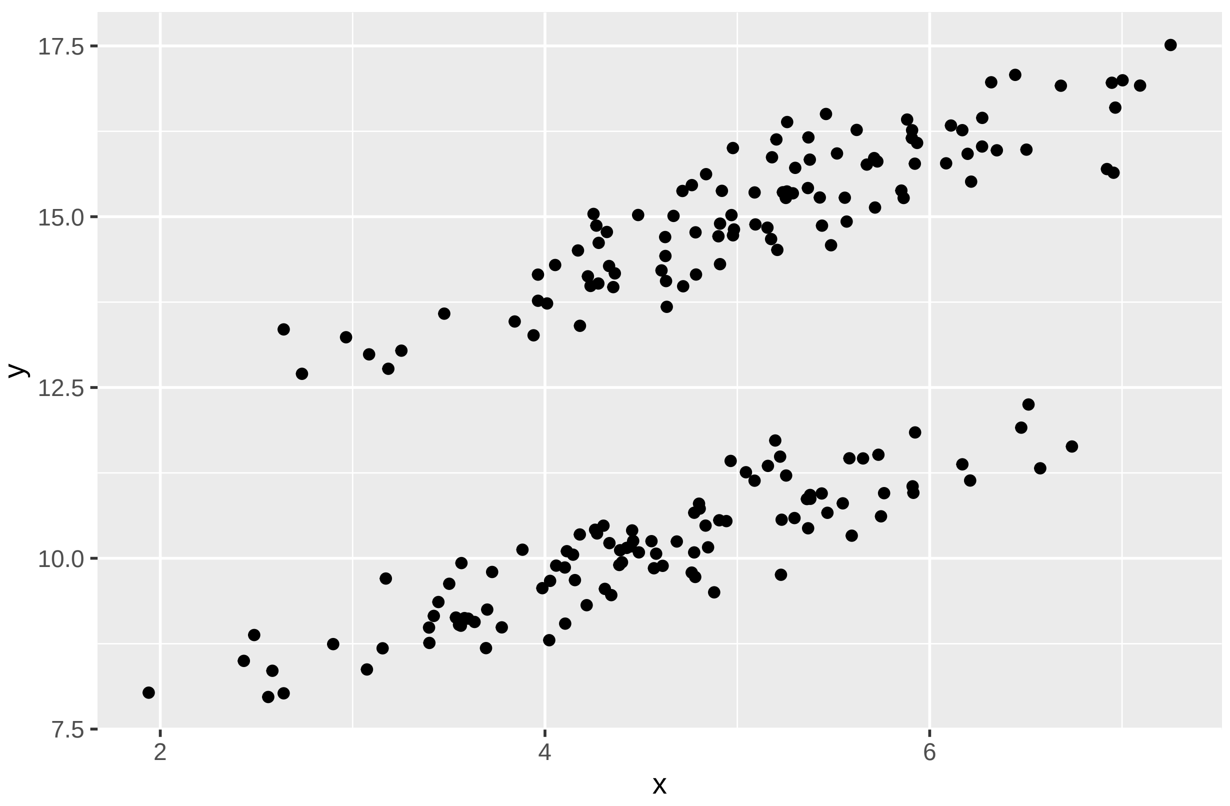
Solution

K-Means Objective: Minimize the sum of dissimilarities of points *within* clusters.

k-way Max Cut Objective: Maximize the sum of dissimilarities of points *between* clusters.

While the k-way Max Cut objective function aims to maximize the sum of dissimilarities (say squared Euclidean distances) of points between clusters, the K-Means objective function aims to minimize the sum of squared Euclidean distances between these points and their respective cluster centroid (mean of points within a cluster). That is, the k-way Max Cut objective function is explicitly based on pairwise dissimilarities between data points, while the K-Means objective function applies them implicitly in that the sum of squared deviations of points from their cluster centroid is equal to the sum of their pairwise squared Euclidean distances divided by the number of points in their given cluster. In either case, both methods have inherently different ways of defining clusters, and hence, may yield distinct optimal clustering outcomes depending on the data. Specifically, while the K-Means algorithm defines its clusters by the means of optimally similar points, the the k-way Max Cut approach would assign clusters to set groups of data for which the sum of their "between" differences is optimally large (at the possible expense of greater "within" cluster variance). This latter approach to clustering would yield better results when data displays noticeable patterns that span a large space, but may be spread in such a way that a K-Means algorithm would sacrifice the difference between these groups to obtain overall "tighter" clusters (groupings of data points that are spacially closer together). For this reason, one (a researcher) may consider using the k-way Max Cut objective function for clustering data that, instead of being naturally grouped in terms of spacial proximity, exhibits significant patterns or trends (non-spherical shapes) that a K-Means algorithm would ignore. To illustrate a case in which both methods would produce different optimal clusters, two sets of correlated data were generated, giving following graph (Figure 1).

Figure 1: Data



Evidently, not all points that are closer together belong to the same natural groups (pattern) we see. Specifically, Figures 2 and 3 below show how each clustering approach would group this data. Although the overall sum of distances between clusters in Figure 2 would be small compared to Figure 3, the K-Means objective function ensures that the within-cluster distance is as small as possible. However, in a case like this, we would want the data to be grouped as it is in Figure 3. Thus, since the k-way Max Cut method performs better with pattern recognition, it may be preferable for gene sequencing, or image; text; and speech recognition, to name a few examples.

Figure 2: Clustering with K−Means Objective Function

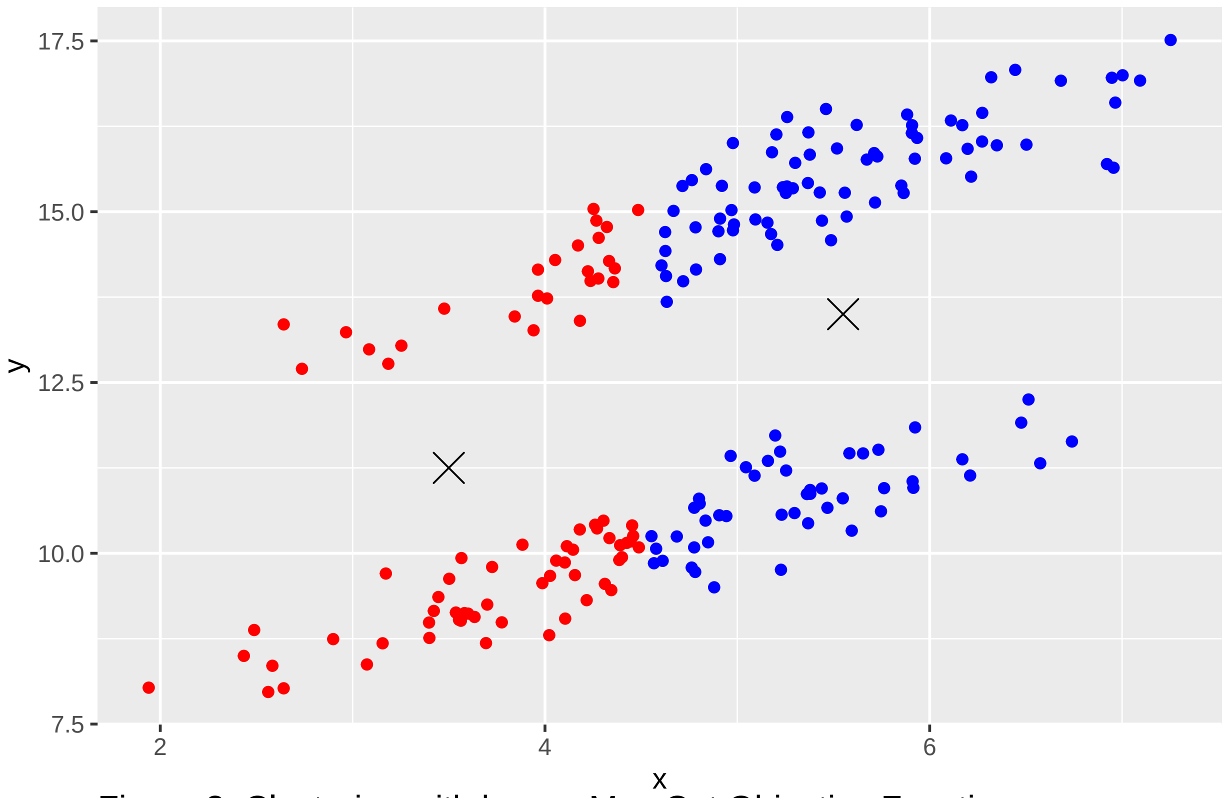
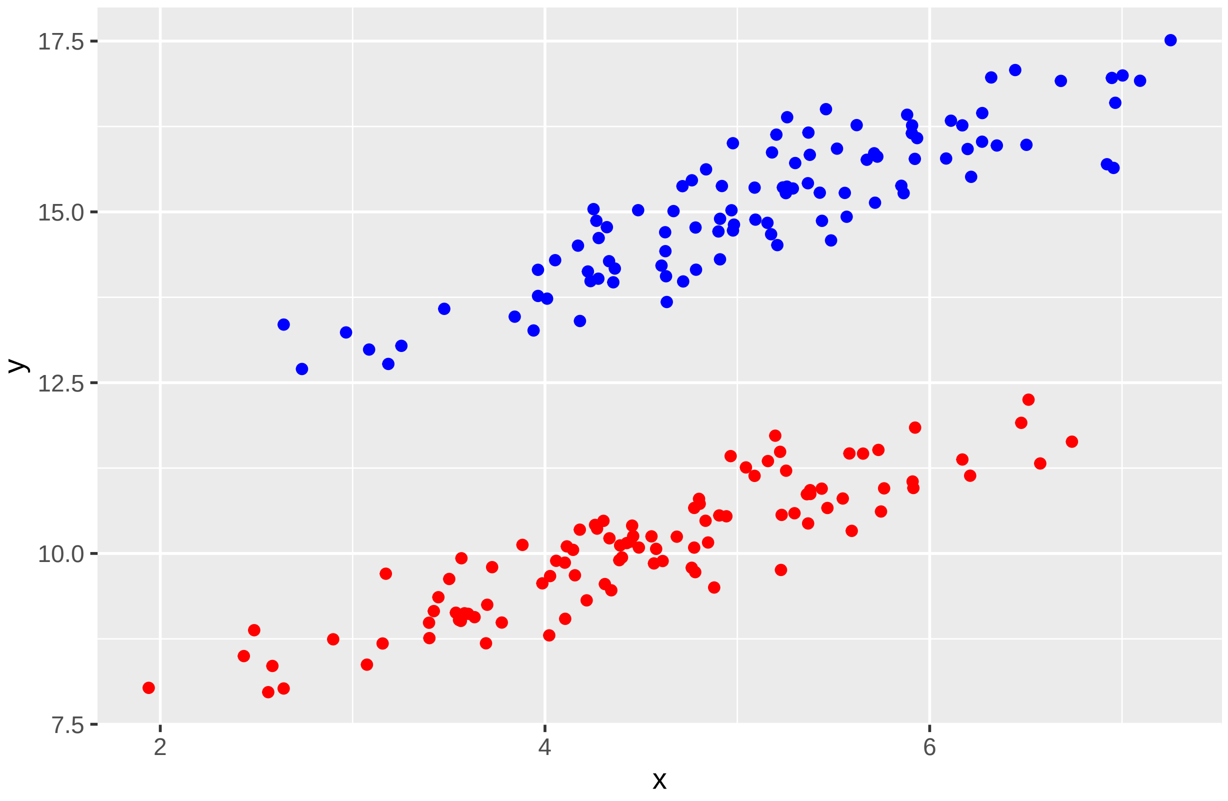


Figure 3: Clustering with k−way Max Cut Objective Function



Problem 3: Application to Pain Data

Read the paper "Hierarchical clustering by patient-reported pain distribution alone identifies distinct chronic pain subgroups differing by pain intensity, quality, and clinical outcomes". The paper has publicly available data (see S1 Dataset in the supplementary information). Using what you read, the questions the authors aimed to address, and your own exploratory analysis, perform an alternative clustering analysis. You should justify why you are interested in this alternative method and interpret your results in comparison to the original methods.

Solution

The data used in the study relevant to this analysis consists of 21,658 unique patient observations of self-reported pain status (with "1" indicating the presence of pain, and "0" the absence thereof) in 74 distinct body regions. This data, hence, takes the form of a 21,658 $\times$ 75 data frame, with the first column giving patient (ID) numbers and the remaining 74 giving the (binary) pain statuses for each body region. To gain a general sense of how frequently pain is reported within the data (both in terms of patients and body region), Figures 4 and 5 below were created to show the distributions of pain absence across patients (observations) and body regions (variables), respectively. Noticeably, the majority of individuals in the study reported "no pain" for most of the 74 body regions, with fewer and fewer numbers reporting much greater magnitudes of pain, giving the shape seen in Figure 4. That being the case, we can make sense of the graph in Figure 5, which shows that the greatest portion of patients make up the majority of body regions for which practically no pain was reported. More than providing means of visualizing the distribution pain presence (or lack thereof), these graphs give us an idea of how much more significant a positive pain status (1 for "pain") is than a negative one in the context of the study.

Figure 4: Distribution of Pain Absence Across Patients

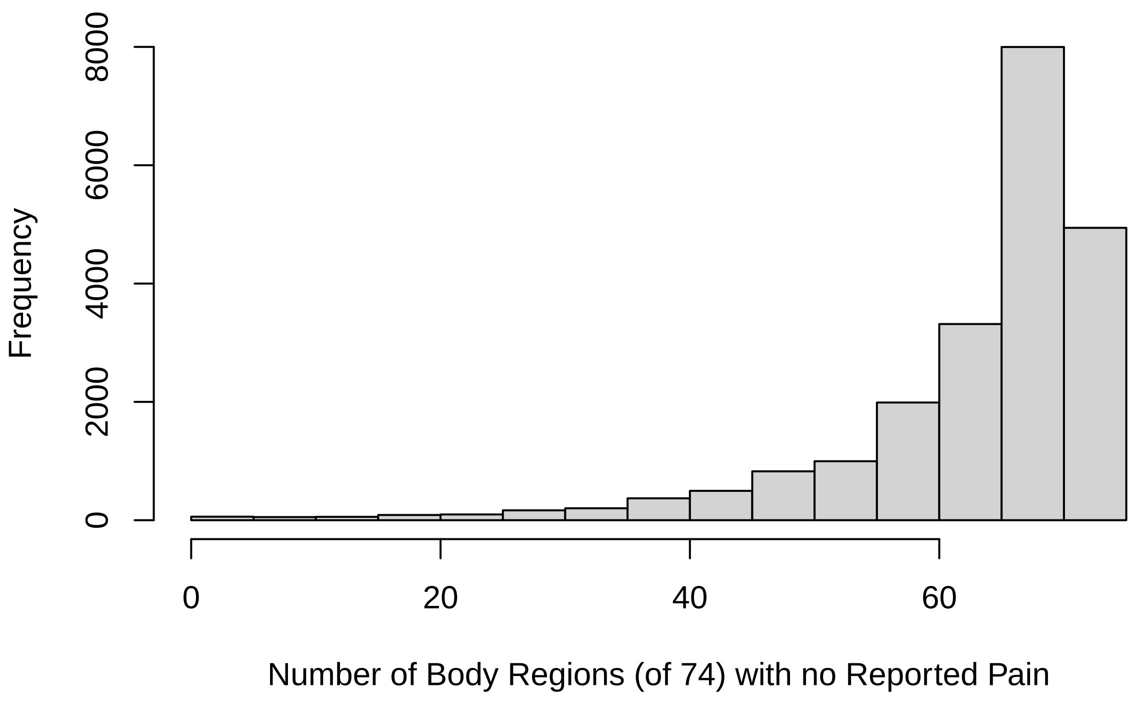
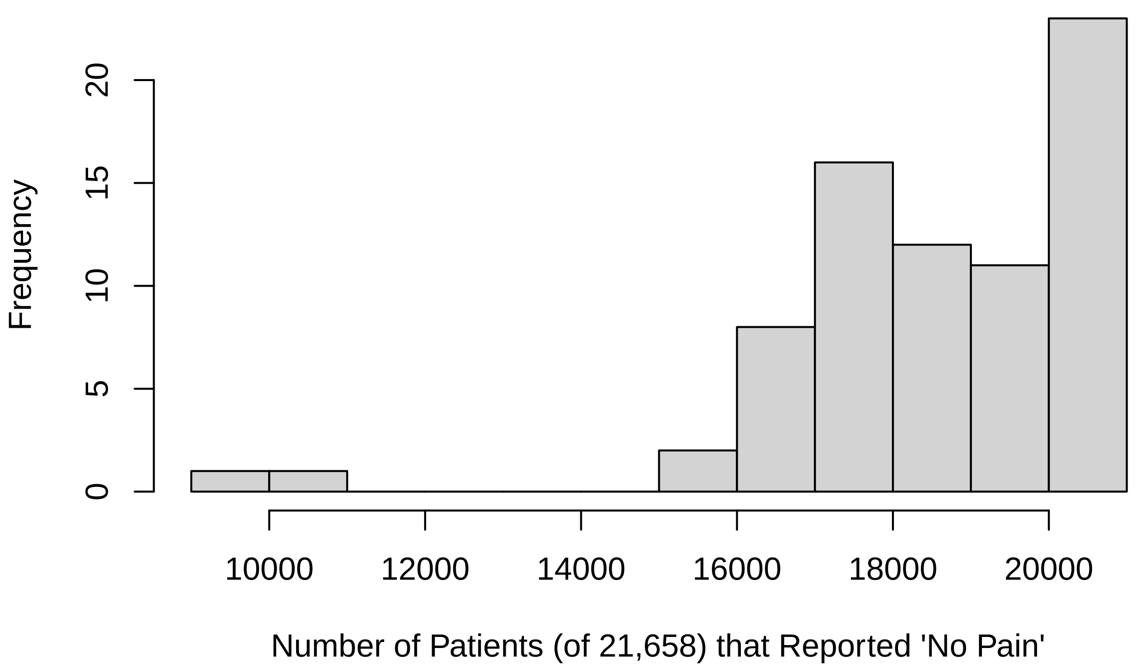
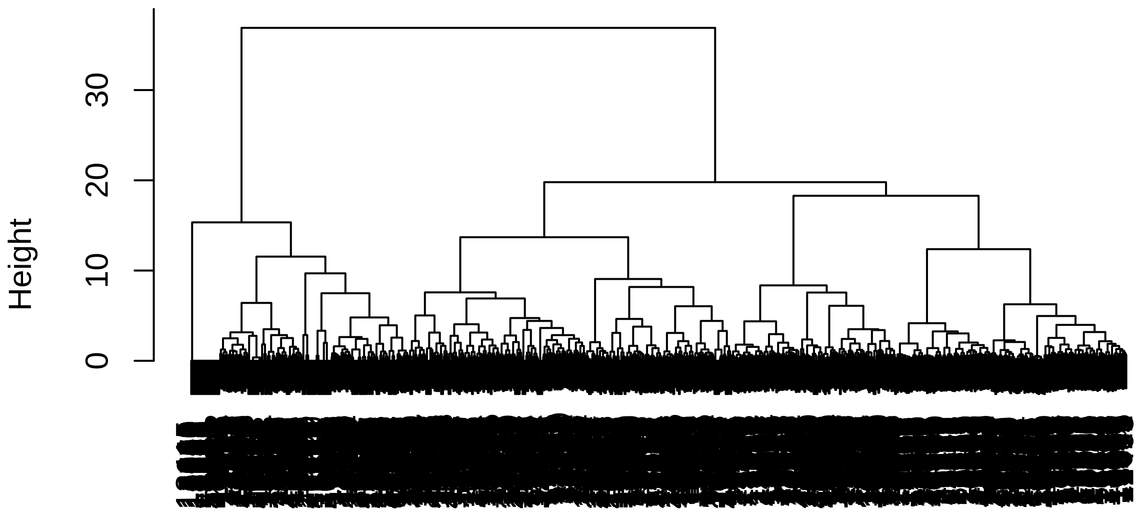


Figure 5: Distribution of Pain Absence Across Body Regions



To cluster the data, for the purpose of "evaluating the clinical relevance of patterns of pain distribution", the researchers in this study used an agglomerative hierarchical clustering algorithm, also known as a "Bottom-Up" approach or "AGNES", given its ascending clustering nature. Namely, this type of algorithm (agglomerative) starts off by assigning a cluster to each observation, and then successively fuses them together based on a distance/"dissimilarity" metric through a specified "linkage" method, producing an ascending hierarchy of clusers, which is visually represented as a "tree" (called a dendrogram). Given that the data is binary (and asymmetric due to the presence/absence nature of values), the study uses a dissimilarity metric based on the Jaccard distance and a Ward's linkage method as a basis for conducting agglomerative clustering. Moreover, the researchers in the study chose to cluster the data based on observations. Thus, their 9 chosen clusters are each grouped sets of patients for which the algorithm identified significant similarities in their corresponding reported pain status patterns (with respect to the 74 body regions in question). Given the sheer number of observations in the data as well as the computational cost (cubic, $O(n^3)$) of this clustering approach, a random sample of $n = 1,000$ observations was used to illustrate the study's clustering process (shown in Figure 6 below).

Figure 6: Original Approach − Cluster Dendrogram (by Observations



Bottom−Up (AGNES)
Jaccard Dissimilarity Metric, Ward Linkage

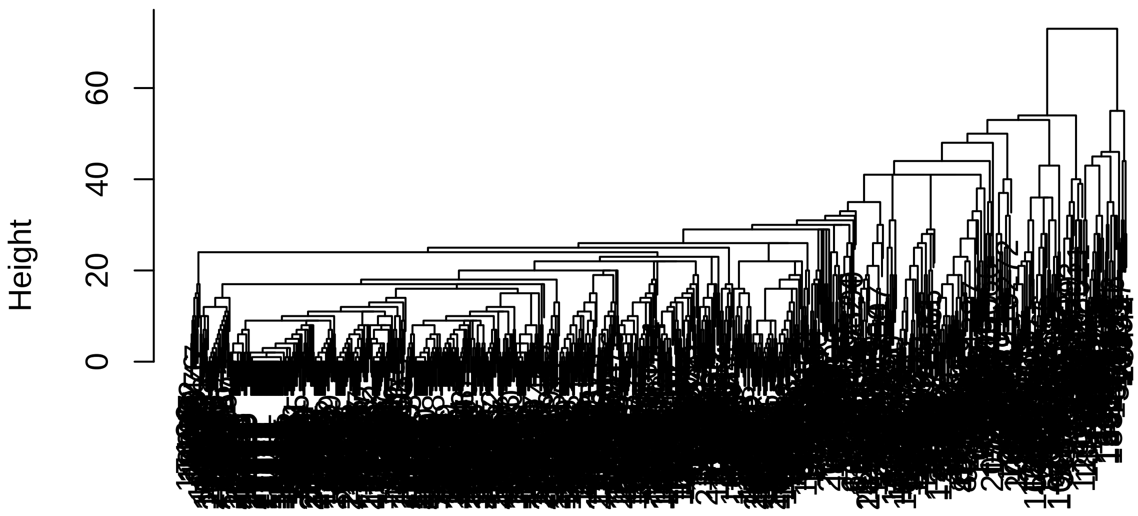
In addition to being computationally expensive, the Bottom-Up approach is better suited for discovering small clusters and may hence lose sight of the greater overarching patterns of pain distribution. For this reason, we propose the following two clustering alternatives which may yield more desirable outcomes, while preserving the underlying hierarchical framework used in the study.

1. **Alternative**:

   Clustering by patients/observations using a divisive hierarchical clustering ("Top-Down" or "DIANA") approach, based on a Manhattan distance dissimilarity metric.

   Given that a divisive hierarchical clustering approach works better for identifying larger clusters, we might consider using this Top-Down method to cluster patient responses in such a way that provides a more adequate grouping of the data with respect to the study objective. Specifically, a divisive hierarchical clustering algorithm functions in the reverse way that an agglomerative clustering algorithm does, starting out with all observations in a single cluster and iteratively breaking them up such that the more "similar" values are grouped together to create a decending tree-based representation of clusters. While this method allows for a user-specified distance/dissimilarity metric, it differs from the agglomerative approach in that it does not need a linkage for fusing clusters together. Instead, it simply divides clusters by the "maximum average (specified) dissimilarity". Given the high dimensionality of the data, we specify a dissimilarity metric based on the Manhattan distance, which (for binary data) is equivalent to the Hamming distance used for measuring distance/dissimilarity between equal-length vectors/strings of binary data. Employing the corresponding algorithm on the same random sample (of size $n = 1,000$) from the data used above (Figure 6), we obtain the dendogram shown in Figure 7 below. Evidently, the clusters produced convey a completely different hierarchical structure than those form the original approach. Looking at the general cluster taxonomy, we notice that there are some similarities between the dendogram and the histogram produced in the exploratory data analysis (EDA) above (Figure 4). Namely, the clustering shows that the largest difference exists between only a small number of patients and the remaining individuals in the study, which may be the same individuals for which there was little to no pain reported across body regions. This indicates that a divisive approach to hierarchical clustering could be better for capturing the distibution pain absense accross patients. That is, by taking this massive difference into consideration first (and then grouping individuals accordingly), rather than doing so later on in the clustering process.

Figure 7: Alternative 1 − Cluster Dendrogram (by Observations)



Top−Down (DIANA)
Manhattan Dissimilarity Metric, 'Maximum Average' Linkage

2. **Alternative**:

   Clustering by body regions/variables using an agglomerative hierarchical clustering ("Bottom-Up" or "AGNES") approach, with Ward's linkage and a dissimilarity metric based on either a Hamming, Cosine, or Jaccard distance.

   To reduce computational cost, but maintain a hierarchical clustering scheme, we may also consider clustering by variables or body regions. Since, the aim is to find relevant patterns of pain distribution, using the original agglomerative or "Bottom-Up" approach on the variables (of which there are only 74) may yield equally optimal groupings of body regions for which patients display similar patterns of pain. To see which dissimilarity metric and linkage method would yield the best clustering structure, we run the algorithm on each combination of the following dissimilarities and linkages to obtain the corresponding agglomerative coefficient (for which larger values close to 1 indicate a strong clustering structure).

   Dissimilarity

   - Hamming Distance: Dissimilarity between equal-length binary vectors/strings.
   - Cosine Distance: Dissimilarity between any-length vectors in a multi-dimensional space (result of converting cosine similarity (cosine of the angle between two vectors) to dissimilarity through `as.dist(1-(cosine_similarity))`).

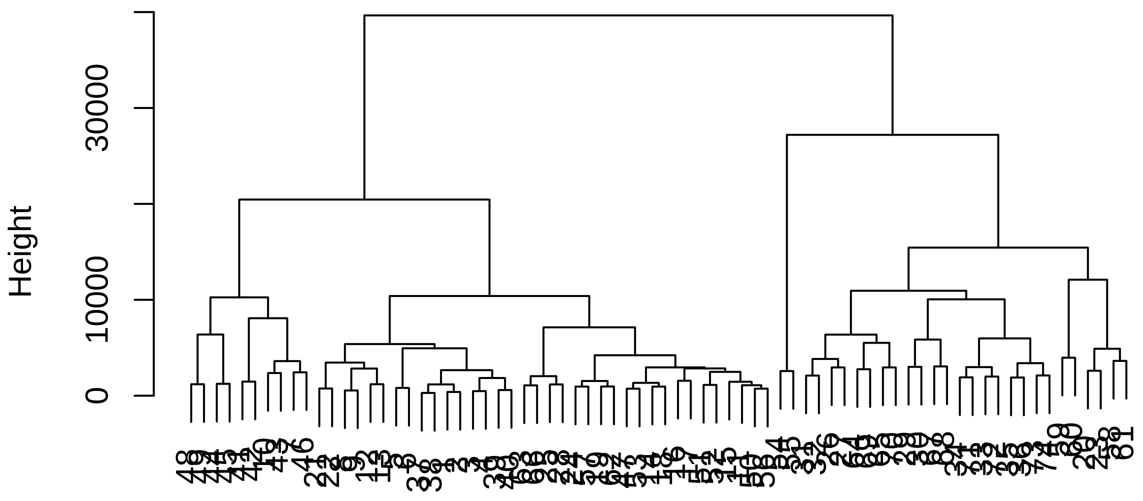
- Jaccard Distance: Dissimilarity between two sets (and asymmetric binary data).

Linkage

- Complete (“Maximum”): Merges clusters based on the smallest distance between clusters’ maximum pairwise dissimilarities.
- Average (“UPGMA”): Merges clusters based on the smallest distance between clusters’ mean pairwise dissimilarities.
- Single (“Minimum”): Merges clusters based on the smallest distance between clusters’ minimum pairwise dissimilarities.
- Ward’s: Merges clusters based on the smallest between-cluster variance between clusters’ minimized within-cluster variance.

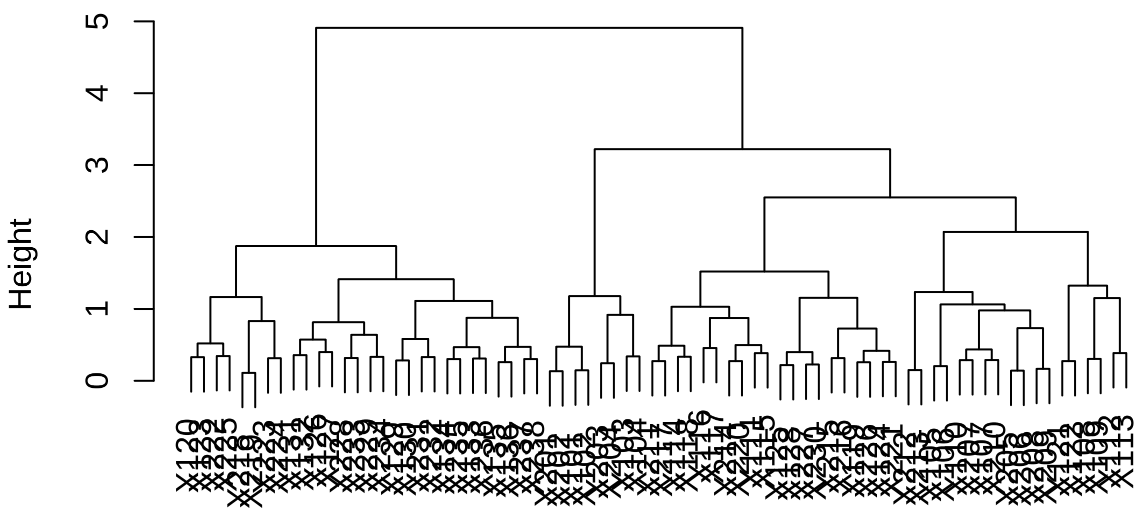
Of all linkage methods, we see that, across dissimilarity measures, Ward’s produced the highest agglomerative coefficient values of approximately 0.9579, 0.9431, and 0.8975, for Hamming, Cosine, and Jaccard distances, respectively. Although the Hamming dissimilarity metric (in tandem with Ward’s linkage method) produced the largest of the three agglomerative coefficients (with Cosine not being very far off), all three indicate strong clustering structures. Moreover, given each metric’s appropriateness for this type of data, the use of either would yield equally valid and meaningful results. Thus, an agglomerative approach to clustering the data by body regions/variables, would be best using Ward’s linkage method and a Hamming, Cosine, or Jaccard dissimilarity metric (with Hamming or Cosine being of preference). The dendograms produced under each of these dissimilarity metrics are given by Figures 8-10 below.

Figure 8: Alternative 2a − Cluster Dendrogram (by Observations)



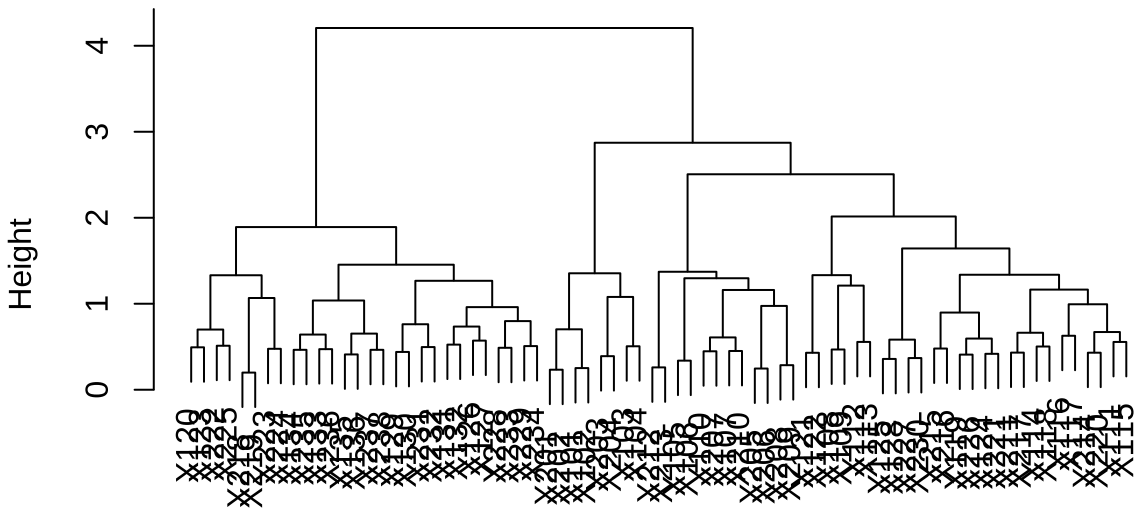
Bottom−Up (AGNES)
Hamming Dissimilarity Metric, Ward Linkage − AC=0.9579

Figure 9: Alternative 2b − Cluster Dendrogram (by Observations)



Bottom−Up (AGNES)
Cosine Dissimilarity Metric, Ward Linkage − AC=0.9431

Figure 10: Alternative 2c − Cluster Dendrogram (by Observations)



Bottom−Up (AGNES)
Jaccard Dissimilarity Metric, Ward Linkage − AC=0.8975

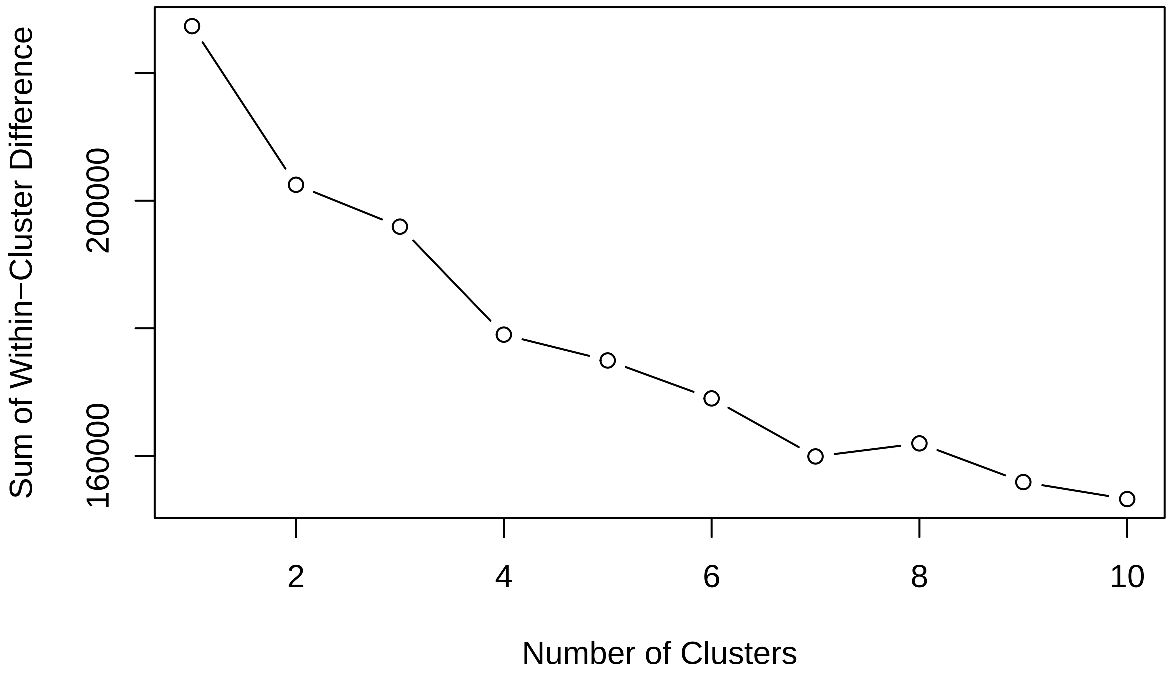
As both of these alternatives produce potentially improved (with trade-offs), but not optimal clustering results, we propose one final clustering scheme. Given the computational expense of hierarchical clustering methods, the size of this data, and the primary research objective, we shall argue that a more promising approach to clustering for the purposes of this study is the following:

3. **Preferred Approach**:

   Clustering by patients/observations using a K-Modes algorithm, with $k = 9$ clusters.

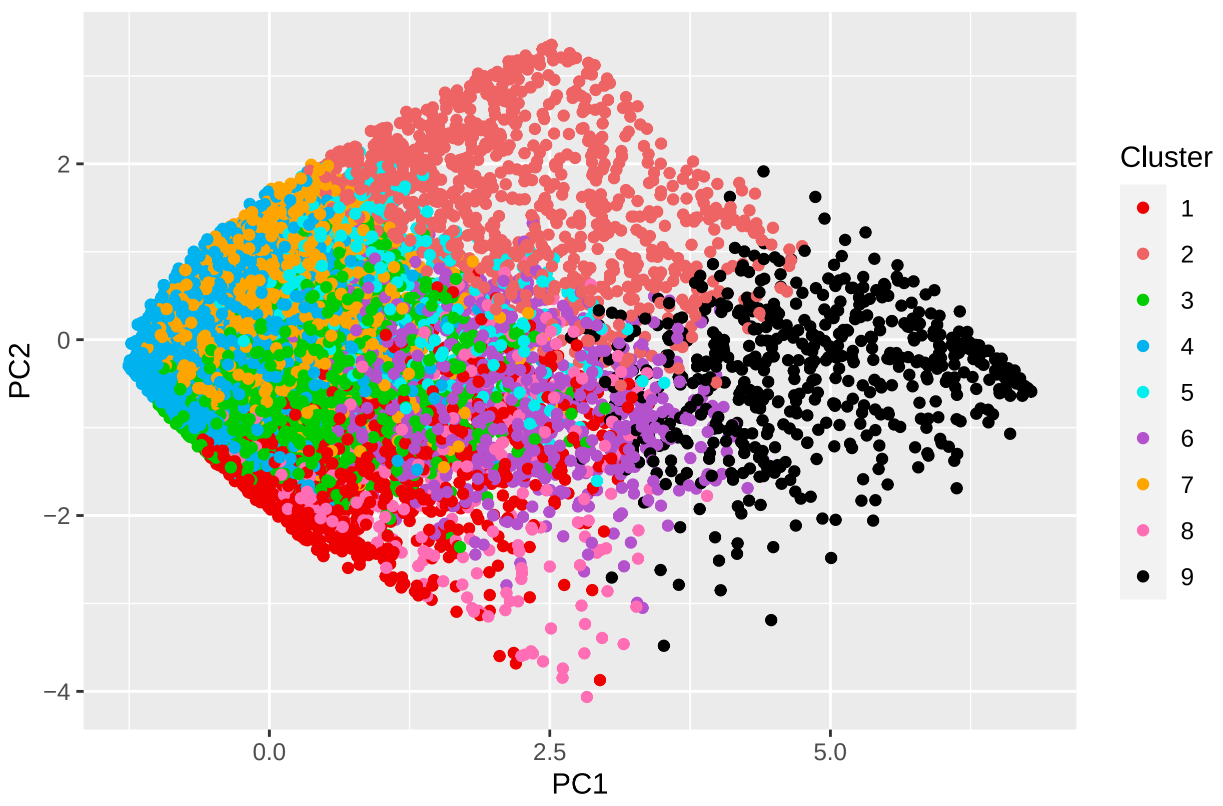
   The K-Modes clustering algorithm is the proposed K-Means alternative for clustering binary and/or hierarchical data. In short, this approach (as with K-Means) aims to minimize the distance between data points in a cluster and their corresponding mode, using simple "matches" as a dissimilarity metric, rather than the traditional K-Means Euclidean distance metric. As we would with K-Means, we produce an elbow plot (Figure 11 below) to identify an optimal number of clusters. The sums of within-cluster differences for the corresponding number of clusters indicates that $k = 5$ clusters may be preferable, but given the lack of clarity in this graph, we choose to run the algorithm with $k = 9$ clusters to stay somewhat consistent with the original approach (also providing more effective means for comparing results).

Figure 11: K−Modes Elbow Plot



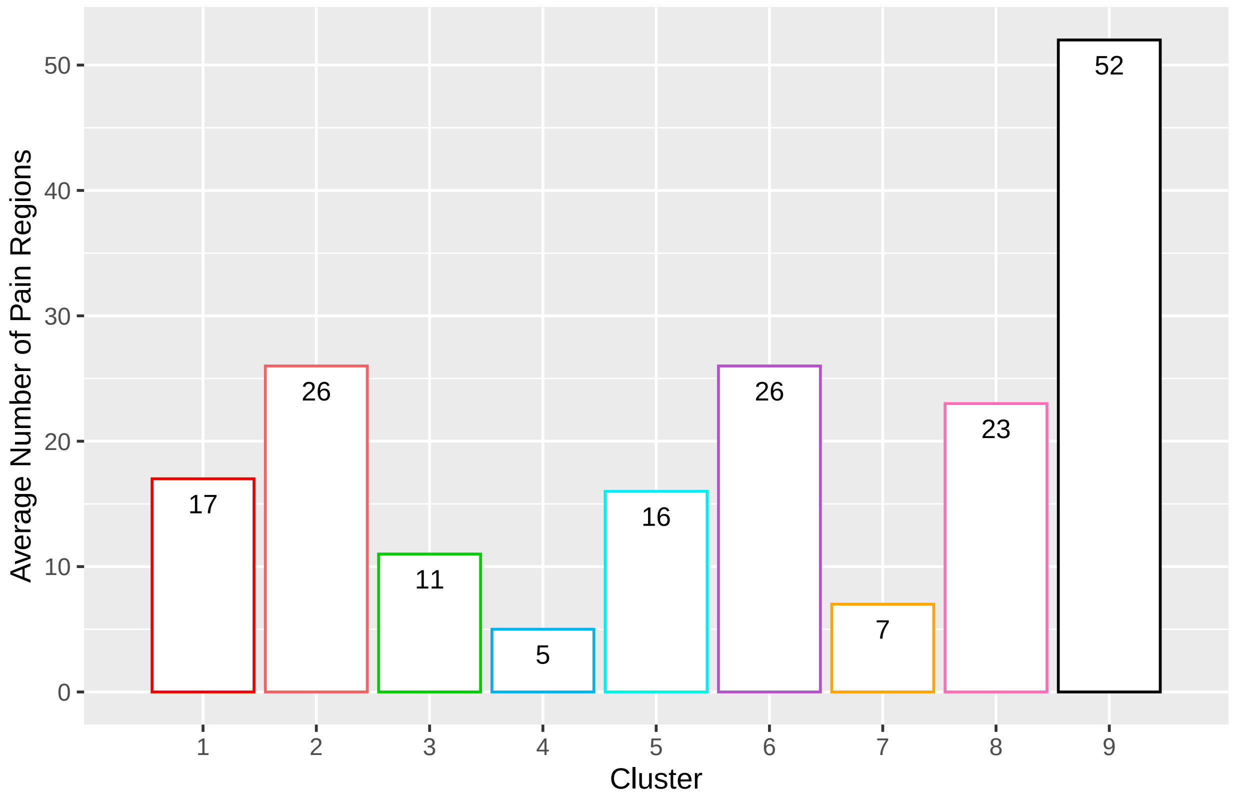
After running the algorithm, the output was appended as a vector to the data and PCA was conducted to reduce dimensionality and provide the following two-dimensional visualization of the K-Modes clustered data (Figure 12 below).

Figure 12: Preferred Approach − K−Modes Clustering (by Observations)



Although it is not entirely clear from the paper which variable names correspond to which body regions (making it hard to compare all results), the following plot (Figure 13), which shows the average magnitude of pain by cluster membership (that is, the average number of body regions for which patients in a given cluster reported pain), gives a similar assessment of the clustered observations as does Figure 4 (with regards to pain interference) in the study report. Specifically, results show that the original hierarchical approach clustered patients such that t-scores produced for average pain interference at each cluster bare some resemblance to the graph in Figure 13 given below. Here we see that there is a decent range of average number of pain regions across clusters, with one cluster in particular having a decently higher average than the rest (as in the study results). Despite not being able to make any more significant comparisons or interpretations of the clustering results from the K-Modes approach, the similarity between these specific graphs hint at the possibility of a comparable clustering of individuals in the study by both methods (and hence, a comparable identification of similarities/dissimilarities between observations in the data). Ultimately, this sheds light on the promise and appeal of the K-Modes algorithm as a method for clustering categorical and/or hierarchical data without the computational cost and complexity of the discussed hierarchical clustering approaches.

Figure 13: Average Magnitude of Pain by Cluster Membership



Code

```r
set.seed(47)

# Libraries
library(MASS)
library(ggplot2)

# Simulating Correlated Data
sig <- rbind(c(1,0.90), c(0.90, 1))
data1 <- mvrnorm(n=100, mu=c(4.5, 10), Sigma=sig)
data2 <- mvrnorm(n=100, mu=c(5, 15), Sigma=sig)

data <- ggplot() +
  geom_point(aes(data1[,1], data1[,2]), color="black") +
  geom_point(aes(data2[,1], data2[,2]), color="black") +
  labs(title="Figure 1: Data", x="x", y="y")

data
```

```r
set.seed(47)

# K-Means Approach
km <- ggplot() +
  geom_point(aes(data1[,1], data1[,2], color=cut(data1[,1], c(-Inf, 4.5, Inf)))) +
  geom_point(aes(data2[,1], data2[,2], color=cut(data2[,1], c(-Inf, 4.5, Inf)))) +
  scale_color_manual(values=c("red", "blue")) +
  geom_point(aes(x=3.5, y=11.25), shape=4, size=5) +
  geom_point(aes(x=5.55, y=13.5), shape=4, size=5) +
  labs(title="Figure 2: Clustering with K-Means Objective Function", x="x", y="y") +
  theme(legend.position="none")

# k-way Max Cut Approach
mc <- ggplot() +
  geom_point(aes(data1[,1], data1[,2]), color="red") +
  geom_point(aes(data2[,1], data2[,2]), color="blue") +
  labs(title="Figure 3: Clustering with k-way Max Cut Objective Function", x="x", y="y")

km
mc
```

```r
# Libraries
library(dplyr)
library(cluster) # AGNES & DIANA Hierarchical Clustering, 'daisy'
library(cultevo) # Hamming Distance Metric
library(lsa) # Cosine Distance Metric
library(vegan) # Jaccard Distance Metric
library(klaR) # K-Modes Clustering
```

```r
# Importing Data
journal_df <- read.csv("/Users/antonellabasso/Desktop/PHP2650/DATA/journal.pone/flatfile-Table 1.csv")

# Data Frame with Columns of Interest (first 75)
pain <- journal_df[-11749,1:75] #also removing extra row from file conversion
#dim(pain)
```

```r
#head(pain)
```

```r
# EDA

# Distribution of "Zeros" (No Pain) for Patients/Observations
zeros_per_patient <- c()
for (i in 1:nrow(pain)){
  zeros <- sum(pain[i,]==0)
  zeros_per_patient <- c(zeros_per_patient, zeros)
}
hist(zeros_per_patient,
     main="Figure 4: Distribution of Pain Absence Across Patients",
     xlab="Number of Body Regions (of 74) with no Reported Pain")
#NOTE: Distribution for "ones" looks the same but flipped
```

```r
# Distribution of "Zeros" (No Pain) for Body Regions/Variables
zeros_per_region <- c()
for (i in 2:ncol(pain)){
  zeros <- sum(na.omit(pain[,i]==0))
  zeros_per_region <- c(zeros_per_region, zeros)
}
hist(zeros_per_region,
     main="Figure 5: Distribution of Pain Absence Across Body Regions",
     xlab="Number of Patients (of 21,658) that Reported 'No Pain'")
```

```r
set.seed(47)

# CLUSTERING DATA: Data in Matrix Form
pain_mat <- as.matrix(pain[,2:75]) #can use transpose for clustering by body region

# CLUSTERING DATA: Matrix Sample to Cluster by Patients/Observations (due to large data)
pain_mat_sample <- pain_mat[sample(nrow(pain_mat),
                                   size=1000,
                                   replace=FALSE),]
```

```r
set.seed(47)

# ORIGINAL APPROACH: Top-Down (AGNES) - Jaccard Distance Dissimilarity Metric, Ward Linkage
jaccard_study <- vegdist(pain_mat_sample, method="jaccard") # Jaccard Distance
hc_ward_j_study <- hclust(jaccard_study, method="ward.D") # Ward Linkage

plot(hc_ward_j_study,
     main="Figure 6: Original Approach - Cluster Dendrogram (by Observations)",
     xlab="Bottom-Up (AGNES)",
     sub="Jaccard Dissimilarity Metric, Ward Linkage")

# Note: We have to sample due to large data
```

```r
# ALTERNATIVE 1: Top-Down (DIANA), Manhattan Dissimilarity Metric, Clustering by Patients/Observations

#top_down_e <- diana(pain_mat_sample) # Euclidean Distance
# pltree(top_down_e)

top_down_m <- diana(pain_mat_sample, metric="manhattan") # Manhattan Distance
pltree(top_down_m,
       main="Figure 7: Alternative 1 - Cluster Dendrogram (by Observations)",
       xlab="Top-Down (DIANA)",
       sub="Manhattan Dissimilarity Metric, 'Maximum Average' Linkage")

# Note: We have to sample due to large data
# Note: Manhattan is better for binary data (same as Hamming)
# Note: Plot resembles the EDA graph made for "0" frequencies for patients
```

```
# ALTERNATIVE 2: Bottom-Up (AGNES), Clustering by Body Region

# Dissimilarity Matrices
hamming <- hammingdists(t(pain_mat)) # Hamming Distance
cosine_similarity <- cosine(pain_mat) # Cosine Distance (calculates by vectors/cols)
cosine <- as.dist(1-cosine_similarity) # converting similarity matrix to dissimilarity matrix
jaccard <- vegdist(t(pain_mat), method="jaccard") # Jaccard Distance

#euclidean <- daisy(t(pain_mat), metric="euclidean") # Euclidean Distance
#manhattan <- daisy(t(pain_mat), metric="manhattan") # Manhattan (same as Hamming)
#gower <- daisy(t(pain_mat), metric="gower") # Gower Distance (same as Hamming)
#binary <- dist(t(pain_mat), method="binary") # Asymmetric Binary Distance (same as Jaccard)


# A: Different HC for Hamming Dissimilarity Metric
hc_complete_h <- hclust(hamming, method="complete") # Complete Linkage
hc_average_h <- hclust(hamming, method="average") # Average/UPGMA Linkage
hc_single_h <- hclust(hamming, method="single") # Single Linkage
hc_ward_h <- hclust(hamming, method="ward.D") # Ward Linkage

# B: Different HC for Cosine Dissimilarity Metric
hc_complete_c <- hclust(cosine, method="complete") # Complete Linkage
hc_average_c <- hclust(cosine, method="average") # Average/UPGMA Linkage
hc_single_c <- hclust(cosine, method="single") # Single Linkage
hc_ward_c <- hclust(cosine, method="ward.D") # Ward Linkage

# C: Different HC for Jaccard Dissimilarity Metric
hc_complete_j <- hclust(jaccard, method="complete") # Complete Linkage
hc_average_j <- hclust(jaccard, method="average") # Average/UPGMA Linkage
hc_single_j <- hclust(jaccard, method="single") # Single Linkage
hc_ward_j <- hclust(jaccard, method="ward.D") # Ward Linkage


# A: Agglomerative Coefficients (Hamming Dissimilarity Metric) <- overall best (highest)
#coef(hc_complete_h)
#coef(hc_average_h)
#coef(hc_single_h)
#coef(hc_ward_h) # best (highest): 0.9579

# B: Agglomerative Coefficients (Cosine Dissimilarity Metric)
#coef(hc_complete_c)
#coef(hc_average_c)
#coef(hc_single_c)
#coef(hc_ward_c) # best (highest): 0.9431

# C: Agglomerative Coefficients (Jaccard Dissimilarity Metric)
```

```
#coef(hc_complete_j)
#coef(hc_average_j)
#coef(hc_single_j)
#coef(hc_ward_j) # best (highest): 0.8975


# A Plots
# plot(hc_complete_h, xlab="Bottom-Up (AGNES)",
#      sub="Hamming Dissimilarity Metric, Complete Linkage")
# plot(hc_average_h, xlab="Bottom-Up (AGNES)",
#      sub="Hamming Dissimilarity Metric, Average (UPGMA) Linkage")
# plot(hc_single_h, xlab="Bottom-Up (AGNES)",
#      sub="Hamming Dissimilarity Metric, Single Linkage")
plot(hc_ward_h,
     main="Figure 8: Alternative 2a - Cluster Dendrogram (by Observations)",
     xlab="Bottom-Up (AGNES)",
     sub="Hamming Dissimilarity Metric, Ward Linkage - AC=0.9579")

# B Plots
# plot(hc_complete_c, xlab="Bottom-Up (AGNES)",
#      sub="Cosine Dissimilarity Metric, Complete Linkage")
# plot(hc_average_c, xlab="Bottom-Up (AGNES)",
#      sub="Cosine Dissimilarity Metric, Average (UPGMA) Linkage")
# plot(hc_single_c, xlab="Bottom-Up (AGNES)",
#      sub="Cosine Dissimilarity Metric, Single Linkage")
plot(hc_ward_c,
     main="Figure 9: Alternative 2b - Cluster Dendrogram (by Observations)",
     xlab="Bottom-Up (AGNES)",
     sub="Cosine Dissimilarity Metric, Ward Linkage - AC=0.9431")

# C Plots
# plot(hc_complete_j, xlab="Bottom-Up (AGNES)",
#      sub="Jaccard Dissimilarity Metric, Complete Linkage")
# plot(hc_average_j, xlab="Bottom-Up (AGNES)",
#      sub="Jaccard Dissimilarity Metric, Average (UPGMA) Linkage")
# plot(hc_single_j, xlab="Bottom-Up (AGNES)",
#      sub="Jaccard Dissimilarity Metric, Single Linkage")
plot(hc_ward_j,
     main="Figure 10: Alternative 2c - Cluster Dendrogram (by Observations)",
     xlab="Bottom-Up (AGNES)",
     sub="Jaccard Dissimilarity Metric, Ward Linkage - AC=0.8975")

#Note: Centroid/UPGMC linkage gives agglomerative coefficient errors
#Note: manhattan=gower=hamming, binary=jaccard (for binary data)
```

```
# PREFFERED APPROACH: K-Modes Clustering on Patients/Observations

# K-Modes Elbow Plot (Determining Optimal Number of Clusters)
num_clust <- 1:10
sum_diffs <- c()
for (i in num_clust){
   km <- kmodes(pain_mat, i)
   sum_diff <- sum(km$withindiff)
   sum_diffs <- c(sum_diffs, sum_diff)
}
plot(x=num_clust, y=sum_diffs, type="b",
     main="Figure 11: K-Modes Elbow Plot",
     xlab="Number of Clusters",
     ylab="Sum of Within-Cluster Difference")

#Note: Based on the plot we might have chosen 5 clusters, but go with 9 to stay somewhat consistent wit
```

```
set.seed(47)

# PREFFERED APPROACH: K-Modes Clustering on Patients/Observations
k_modes <- kmodes(pain_mat, 9)

# Appending Corresponding Cluster Column to Data Matrix
cluster <- k_modes$cluster
pain_mat_kmodes <- cbind(pain_mat, cluster)

# PCA (for Visualization)
pca <- prcomp(pain_mat)
pca12 <- pca$x[,1:2]
pain_mat_kmodes_pca <- cbind(pain_mat_kmodes, pca12)
pain_mat_kmodes_pca <- as.data.frame(pain_mat_kmodes_pca)

# Visualizing Clusters in 2-D (PC1 & PC2)
ggplot(pain_mat_kmodes_pca) +
  geom_point(aes(x=PC1, y=PC2, color=as.factor(cluster))) +
  labs(title="Figure 12: Preferred Approach - K-Modes Clustering (by Observations)",
       color="Cluster") +
  scale_color_manual(values=c("red2", "indianred2", "green3",
                              "deepskyblue2", "cyan2", "mediumorchid3",
                              "orange", "hotpink1", "black"))
# ggplot(pain_mat_kmodes_pca) +
#   geom_point(aes(x=PC1, y=PC2), color=cluster)
```

```
# Comparison to Cluster Results from Paper (Figure 4 in study)

# Average Magnitude of Pain (Average Number of Pain Regions in Each Cluster)
# (average number of body regions for which patients reported pain (1) in each cluster)
pain_mat_kmodes_pca$sum_pain <- rowSums(pain_mat_kmodes_pca[,1:74])
pain_mat_kmodes_pca_grouped <- pain_mat_kmodes_pca %>% group_by(cluster)
region_pain_means <- pain_mat_kmodes_pca_grouped %>%
  summarise(mean_pain=round(mean(sum_pain)), mean_pain_prop=mean_pain/74) #%>%
#region_pain_means

ggplot(region_pain_means, aes(x=cluster, y=mean_pain, color=as.factor(cluster))) +
  geom_bar(stat="identity", fill="white") +
  geom_text(aes(label=mean_pain), vjust=1.8, color="black", size=3.5) +
  scale_x_discrete(limits=1:9) +
  labs(title="Figure 13: Average Magnitude of Pain by Cluster Membership",
       x="Cluster",
       y="Average Number of Pain Regions") +
  theme(legend.position="none") +
  scale_color_manual(values=c("red2", "indianred2", "green3",
                              "deepskyblue2", "cyan2", "mediumorchid3",
                              "orange", "hotpink1", "black"))
```

```
# Additional computed table for cluster analysis

# Average/Proportion of Patients in Each Cluster that Reported Pain (Observation=1) for Each Given Body
region_means_by_cluster <- c()
for (i in 1:74){
  means_i <- as.data.frame(as.list(tapply(pain_mat_kmodes_pca[,i],
                             pain_mat_kmodes_pca$cluster, mean)))
  region_means_by_cluster <- rbind(region_means_by_cluster, means_i)
}
region <- colnames(pain_mat)
region_means_by_cluster2 <- cbind(region, region_means_by_cluster) %>%
  rename("Body Region"=region, "Cluster 1"=X1, "Cluster 2"=X2, "Cluster 3"=X3, "Cluster 4"=X4,
         "Cluster 5"=X5, "Cluster 6"=X6, "Cluster 7"=X7, "Cluster 8"=X8, "Cluster 9"=X9)
region_means_by_cluster2
```

Sources:

1. https://www.ijcsmc.com/docs/papers/August2017/V6I8201725.pdf
2. https://hdbscan.readthedocs.io/en/latest/comparing_clustering_algorithms.html
3. https://medium.com/analytics-vidhya/all-you-need-to-know-about-the-dbscan-algorithm-f1a35ed8e712#:~:text=Time%20Complexity%20of%20DBSCAN%20Algorithm,Algorithm%20is%20O%20(n).

- Study Report: "Hierarchical clustering by patient-reported pain distribution alone identifies distinct chronic pain subgroups differing by pain intensity, quality, and clinical outcomes" (https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0254862#sec029)
- https://bradleyboehmke.github.io/HOML/hierarchical.html#ref-kaufman2009finding
- https://towardsdatascience.com/hierarchical-clustering-on-categorical-data-in-r-a27e578f2995
- https://www.machinelearningplus.com/nlp/cosine-similarity/
- https://stats.stackexchange.com/questions/31565/compute-a-cosine-dissimilarity-matrix-in-r
- https://christophscheuch.github.io/post/unsupervised-learning/clustering-binary-data/