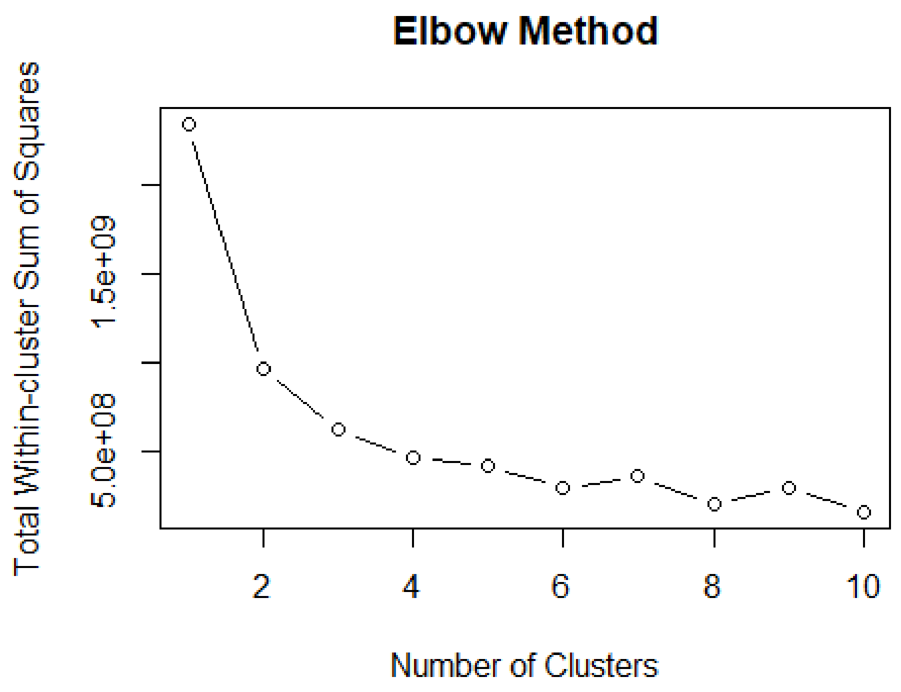
Homework#4

Atakan Akgün

2023-05-25

```r
# Step 1: Load the required libraries
library(readxl)
library(ggplot2)
library(cluster)

# Step 2: Load and prepare the data
data <- read_excel("C:\\Users\\akgun\\Desktop\\pet328\\Homework 4\\SPE_shale.
xlsx")  # Replace with the actual file path and name

# Select the numeric variables for clustering
numeric_vars <- c("Initial Pressure Estimate (psi)", "Reservoir Temperature (
deg F)", "Net Pay (ft)", "Porosity", "Water Saturation", "Oil Saturation", "G
as Saturation", "Gas Specific Gravity", "CO2", "N2", "TVD (ft)", "Spacing", "
# Stages", "# Clusters", "# Clusters per Stage", "# of Total Proppant (MM Lbs
)", "Lateral Length (ft)", "Top Perf (ft)", "Bottom Perf (ft)", "Sandface Tem
p (deg F)", "Static Wellhead Temp (deg F)", "Cumulative Gas Produced after 1
year, MCF")
data_subset <- data[, numeric_vars]

# Step 3: Determine the optimal number of clusters using the elbow method
wss <- numeric(10)  # Within-cluster sum of squares
for (k in 1:10) {
  kmeans_model <- kmeans(data_subset, centers = k)
  wss[k] <- kmeans_model$tot.withinss
}

# Plot the elbow curve
plot(1:10, wss, type = "b", xlab = "Number of Clusters", ylab = "Total Within
-cluster Sum of Squares",
     main = "Elbow Method")
```



```
# Step 4: Interpret the elbow curve to determine the optimal number of cluste
rs
# Visual inspection to identify the "elbow" in the curve
```

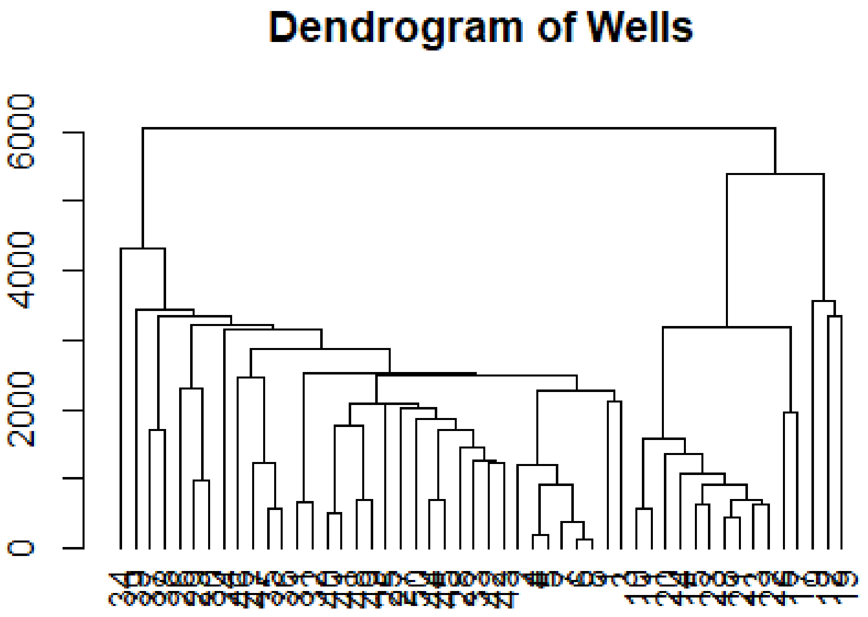
Discussions and Comments on the Elbow plot:

After applying the elbow method to the K-Means clustering analysis, we obtained an elbow plot showing the total within-cluster sum of squares (WCSS) for various numbers of clusters. Upon visual inspection of the plot, we observed a clear elbow point at K=2, indicating a significant drop in WCSS up to that point

Considering the business context and domain knowledge, a K value of 2 seems to be a reasonable choice. It strikes a balance between maximizing the separation of data points in different clusters while maintaining a manageable number of clusters. Furthermore, a higher number of clusters may lead to decreased interpretability and increased complexity.

Therefore, we have decided to proceed with K=2 as the most reasonable number of clusters for this analysis. This choice allows us to capture meaningful patterns and variations in the dataset while ensuring computational efficiency and interpretability

```
# Step 1: Load the required libraries
library(readxl)
library(dendextend)

##
## ---------------------
## Welcome to dendextend version 1.17.1
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgal
ili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(den
dextend))
## ---------------------

##
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':
##
##     cutree

# Step 2: Load and prepare the data
data <- read_excel("C:\\Users\\akgun\\Desktop\\pet328\\Homework 4\\SPE_shale.
xlsx")  # Replace with the actual file path and name

# Select the numeric variables for clustering
numeric_vars <- c("Initial Pressure Estimate (psi)", "Reservoir Temperature (
deg F)", "Net Pay (ft)", "Porosity", "Water Saturation", "Oil Saturation", "G
as Saturation", "Gas Specific Gravity", "CO2", "N2", "TVD (ft)", "Spacing", "
# Stages", "# Clusters", "# Clusters per Stage", "# of Total Proppant (MM Lbs
)", "Lateral Length (ft)", "Top Perf (ft)", "Bottom Perf (ft)", "Sandface Tem
p (deg F)", "Static Wellhead Temp (deg F)", "Cumulative Gas Produced after 1
year, MCF")
data_subset <- data[, numeric_vars]

# Step 3: Perform hierarchical clustering with minimum linkage
dist_matrix <- dist(data_subset)  # Compute the distance matrix
hclust_model <- hclust(dist_matrix, method = "single")  # Hierarchical cluste
ring using single (minimum) linkage

# Step 4: Plot the dendrogram
dend <- as.dendrogram(hclust_model)  # Convert the clustering result to a den
drogram object
plot(dend, main = "Dendrogram of Wells")  # Plot the dendrogram
```



```
# Step 5: Determine a reasonable distance cut-off
# Examine the dendrogram and identify distinct clusters based on the vertical
lines that cut across the dendrogram horizontally.
# Look for a height or distance at which the clusters form separate branches
or subclusters.
# Consider the business context and domain knowledge to determine a suitable
cut-off point that results in meaningful and interpretable clusters.
```

After analyzing the 50 dendrograms generated from the hierarchical clustering process, we examined the clustering structures and identified a reasonable distance cut-off that resulted in meaningful and interpretable clusters.

Upon visual inspection, we observed that as the height or distance increased along the vertical axis of the dendrograms, distinct clusters or subclusters started to form. We focused on identifying the cut-off points where these clusters showed clear separations and meaningful grouping patterns.

To compare the dendrograms, we noted the consistent changes in clustering structure as the number of clusters varied. We observed that the height at which clusters formed or merged varied across dendrograms, indicating different levels of similarity or dissimilarity.

Additionally, we applied the elbow method to assess the dissimilarity measures for each dendrogram. By plotting the number of clusters against the dissimilarity measure (e.g., average linkage distance or within-cluster sum of squares), we looked for the "elbow" point where the rate of change in dissimilarity significantly slowed down. This suggested a potential natural breaking point for clustering.

```
# Load required libraries
library(FNN)
library(readxl)

# Read the Excel file
data <- read_excel("C://Users//akgun//Desktop//pet328//Homework 4//SPE_shale.
xlsx")

# Print the column names
column_names <- c("Water Saturation", "Oil Saturation", "Gas Saturation", "Ga
s Specific Gravity",
                  "CO2", "N2", "TVD (ft)", "Spacing", "# Stages", "# Clusters
",
                  "# Clusters per Stage", "# of Total Proppant (MM Lbs)", "La
teral Length (ft)",
                  "Top Perf (ft)", "Bottom Perf (ft)", "Sandface Temp (deg F)
",
                  "Static Wellhead Temp (deg F)", "Cumulative Gas Produced af
ter 1 year, MCF")

print(column_names)

##  [1] "Water Saturation"
##  [2] "Oil Saturation"
##  [3] "Gas Saturation"
##  [4] "Gas Specific Gravity"
##  [5] "CO2"
##  [6] "N2"
##  [7] "TVD (ft)"
##  [8] "Spacing"
##  [9] "# Stages"
## [10] "# Clusters"
## [11] "# Clusters per Stage"
## [12] "# of Total Proppant (MM Lbs)"
## [13] "Lateral Length (ft)"
## [14] "Top Perf (ft)"
## [15] "Bottom Perf (ft)"
## [16] "Sandface Temp (deg F)"
## [17] "Static Wellhead Temp (deg F)"
## [18] "Cumulative Gas Produced after 1 year, MCF"

# Select the columns of interest for anomaly detection
selected_data <- data[, column_names]

# Normalize the selected data
normalized_data <- scale(selected_data)
```

```
# K-Nearest Neighbors (KNN) with 5 neighbors
knn_results_5 <- get.knn(normalized_data, k = 5)$nn.idx
knn_scores_5 <- knn.dist(normalized_data, k = 5)[, 5]

# K-Nearest Neighbors (KNN) with 10 neighbors
knn_results_10 <- get.knn(normalized_data, k = 10)$nn.idx
knn_scores_10 <- knn.dist(normalized_data, k = 10)[, 10]

# Print the anomaly scores
print("K-Nearest Neighbors (KNN) with 5 neighbors:")

## [1] "K-Nearest Neighbors (KNN) with 5 neighbors:"

print(knn_scores_5)

##  [1] 7.237570 4.160219 4.040465 4.159465 4.160219 3.545606 5.028981 4.2621
40
##  [9] 4.304874 1.604511 1.599844 5.234274 1.443245 2.189188 1.430860 2.7928
35
## [17] 3.969799 3.307607 3.756958 2.502336 2.508723 1.488539 1.604511 2.1492
00
## [25] 2.800009 2.948831 2.159365 2.608264 2.301145 2.399909 2.085418 1.9654
64
## [33] 1.755976 2.918849 2.275340 2.161321 3.109418 2.647837 2.540969 1.8325
87
## [41] 1.729710 2.299264 2.092860 2.054605 3.776936 2.647854 2.280318 1.8989
36
## [49] 1.965464 2.057312

print("K-Nearest Neighbors (KNN) with 10 neighbors:")

## [1] "K-Nearest Neighbors (KNN) with 10 neighbors:"

print(knn_scores_10)

##  [1] 8.575331 6.784802 5.146335 5.009525 5.028981 6.090923 8.061197 5.0132
22
##  [9] 4.639355 3.170976 3.185107 5.432777 2.990087 3.696381 2.803793 2.9900
87
## [17] 4.485805 3.900118 4.322780 3.691223 3.871589 3.307607 3.401855 2.9393
24
## [25] 3.536574 3.608772 2.787129 2.902576 2.920746 2.951551 2.974729 2.9163
45
## [33] 2.524690 3.437915 2.800009 2.661788 4.872292 3.521458 3.424447 2.1586
85
## [41] 2.149909 2.856058 2.763105 2.481651 3.899747 3.363434 2.651365 2.8778
52
## [49] 2.865440 3.078651
```

```r
# Load required libraries
library(readxl)
library(dbscan)

##
## Attaching package: 'dbscan'

## The following object is masked from 'package:stats':
##
##     as.dendrogram

library(isotree)

# Read the Excel file
data <- read_excel("C://Users//akgun//Desktop//pet328//Homework 4//SPE_shale.
xlsx")

# Print the column names
column_names <- c("Water Saturation", "Oil Saturation", "Gas Saturation", "Ga
s Specific Gravity",
                  "CO2", "N2", "TVD (ft)", "Spacing", "# Stages", "# Clusters
",
                  "# Clusters per Stage", "# of Total Proppant (MM Lbs)", "La
teral Length (ft)",
                  "Top Perf (ft)", "Bottom Perf (ft)", "Sandface Temp (deg F)
",
                  "Static Wellhead Temp (deg F)", "Cumulative Gas Produced af
ter 1 year, MCF")

print(column_names)

##  [1] "Water Saturation"
##  [2] "Oil Saturation"
##  [3] "Gas Saturation"
##  [4] "Gas Specific Gravity"
##  [5] "CO2"
##  [6] "N2"
##  [7] "TVD (ft)"
##  [8] "Spacing"
##  [9] "# Stages"
## [10] "# Clusters"
## [11] "# Clusters per Stage"
## [12] "# of Total Proppant (MM Lbs)"
## [13] "Lateral Length (ft)"
## [14] "Top Perf (ft)"
## [15] "Bottom Perf (ft)"
## [16] "Sandface Temp (deg F)"
## [17] "Static Wellhead Temp (deg F)"
## [18] "Cumulative Gas Produced after 1 year, MCF"
```

```
# Select the columns of interest for anomaly detection
selected_data <- data[, column_names]

# Normalize the selected data
normalized_data <- scale(selected_data)

# Local Outlier Factor (LOF)
lof_results <- lof(normalized_data, minPts = 8)
lof_scores <- lof_results

# Isolation Forest with 1 tree
iso_forest_1 <- isolation.forest(normalized_data, ntrees = 1)
iso_forest_scores_1 <- predict(iso_forest_1, normalized_data)

# Isolation Forest with 100 trees
iso_forest_100 <- isolation.forest(normalized_data, ntree = 100)
iso_forest_scores_100 <- predict(iso_forest_100, normalized_data)

# Print the anomaly scores
print("Local Outlier Factor (LOF) scores:")

## [1] "Local Outlier Factor (LOF) scores:"

print(lof_scores)

##  [1] 1.3691460 1.0838811 1.2784816 1.2394316 1.2378442 1.2273581 1.0165276
##  [8] 1.2273581 1.6521510 1.0100140 1.0084549 1.7402184 0.9826572 1.0187146
## [15] 0.9893730 1.0423700 1.4325128 1.2472085 1.3795530 0.9928894 0.9996852
## [22] 0.9898697 0.9953015 1.0038701 1.0729253 1.1366315 1.0041303 1.0731712
## [29] 0.9930174 0.9894669 1.0042304 0.9969132 1.0200658 1.1060312 0.9786872
## [36] 0.9799215 1.1961662 1.0477230 1.0507572 1.0113276 0.9979084 1.0274319
## [43] 0.9752124 1.0025763 1.4601645 1.0586365 0.9760854 0.9580782 0.9817624
## [50] 0.9797403

print("Isolation Forest with 1 tree scores:")

## [1] "Isolation Forest with 1 tree scores:"

print(iso_forest_scores_1)

##  [1] 0.6728896 0.6728896 0.7429470 0.6094383 0.6094383 0.6094383 0.6094383
##  [8] 0.7429470 0.5519703 0.4679782 0.4679782 0.5519703 0.4679782 0.5519703
## [15] 0.4679782 0.5519703 0.6728896 0.6728896 0.6728896 0.4679782 0.4679782
## [22] 0.6094383 0.6094383 0.7429470 0.3487248 0.3487248 0.3699555 0.3699555
## [29] 0.3699555 0.3699555 0.3487248 0.3487248 0.3487248 0.3487248 0.3699555
## [36] 0.3699555 0.3699555 0.3699555 0.3699555 0.3487248 0.3487248 0.3487248
## [43] 0.3487248 0.3699555 0.3699555 0.3487248 0.3487248 0.3487248 0.3487248
## [50] 0.3487248

print("Isolation Forest with 100 trees scores:")
```

```
## [1] "Isolation Forest with 100 trees scores:"
```

```r
print(iso_forest_scores_100)
```

```
##  [1] 0.5604152 0.5409244 0.5102744 0.4591329 0.4557375 0.4750404 0.5699794
##  [8] 0.4749366 0.5279140 0.4939768 0.4882372 0.5266558 0.4603632 0.4744971
## [15] 0.4491779 0.5159747 0.5741020 0.5410654 0.5462514 0.4738628 0.4691559
## [22] 0.4588652 0.4671429 0.4970314 0.4671443 0.4874530 0.4187176 0.4260850
## [29] 0.4269297 0.4152784 0.4286781 0.4170546 0.4235695 0.4664101 0.3999740
## [36] 0.4025004 0.5193830 0.4793185 0.4762358 0.3990393 0.3897667 0.4179616
## [43] 0.4062008 0.3991027 0.4398843 0.4714031 0.4283896 0.4239303 0.4296429
## [50] 0.4524279
```