Mehra_Kai_Assignment-5

Kai Mehra

2023-03-31

Libraries

```r
# Importing the {tidyverse}, {ggplot2}, {factoextra}, {cluster}, {NbClust},
# and {aricode} libraries.

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggplot2)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
library(cluster)
library(NbClust)
library(aricode)
```

Data

```r
beatles <- read.csv("../Data/Beatles Data/TheBeatlesCleaned.csv")

dim(beatles)
```

```
## [1] 193  11
```

```r
unique(beatles$album)
```

```
##  [1] "Please Please Me"
##  [2] "With The Beatles"
##  [3] "A Hard Day’s Night"
##  [4] "Beatles for Sale"
##  [5] "Help!"
##  [6] "Rubber Soul"
##  [7] "Revolver"
##  [8] "Sgt. Pepper’s Lonely Hearts Club Band"
##  [9] "Magical Mystery Tour"
## [10] "The Beatles (white album)"
## [11] "Yellow Submarine"
## [12] "Abbey Road"
## [13] "Let It Be"
```

```r
length(unique(beatles$album))
```

```
## [1] 13
```

The “beatles” data set contains Spotify metadata on 193 Beatles’s songs based 11 different characteristics. The data set contains songs from 13 albums comprising the Beatles’s core discography. The data set came from Kaggle, and it contains basic demographic information on the song including the song id, release year, name, and album name. The data also contains song characteristic data including the danceability, energy, speechiness, acousticness, liveness, valence, and duration (milliseconds).

Research Question

Can Spotify song characteristics cluster The Beatles’s songs into their albums as they released them? Are there clusters that better group similar songs together?

Hypothesis

The Beatles’s were known for being experimental and trying new things from album to album, and from my experience, I believe The Beatles’s albums are relatively cohesive. Thus, I think that the k-means clustering algorithm will be able to separate some of the albums apart. However, I do not think that the algorithm will be able to identify all 13 albums as there is overlap in style and sound between albums.

Variables of Interest

Dependent Variable:

album: The Beatles released 13 albums, and I will use the k-means clustering algorithm to attempt to classify the songs back into their albums.

Independent Variables:

- length_sec: The length of the song in seconds
- danceability: how suitable a track is for dancing based on a combination of musical elements. 0.0 is least danceable and 1.0 is most danceable.
- energy: a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
- speechiness: Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.
- acousticness: a confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
- liveness: Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
- valence: A measure from 0.0 (very sad) to 1.0 (very happy) describing the musical positiveness conveyed by a track.

These variables provide a comprehensive understanding of the general characteristics of a specific song. Additionally, since they are all continuous, numeric variables, they lend themselves to analysis using linear regression. Intuitively, these variables should have some impact on popularity as songs fitting into certain genres have higher levels of popularity than others. These variables help define genres; for example, a highly danceable, energetic, valent, and loud song would likely be a club/party song which can be incredibly popular.

Data Wrangling

```
# Editing song titles to distinguish different recordings of the same song
beatles$song[id = 109] <- "sgt. Peppers lonely hearts club band - reprise"
beatles$song[id = 152] <- "yellow submarine - yellow submarine"
beatles$song[id = 157] <- "all you need is love - yellow submarine"

# Fixing a typo in the data
beatles$energy[id = 111] <- 0.613
```

```
# ensuring that there are no duplicate songs
length(unique(beatles$song)) == nrow(beatles)
```

```
## [1] TRUE
```

```
beatles <-
  beatles %>%
  mutate(length_sec = duration_ms/1000) %>% # converting length from milsecs to secs
  select(-duration_ms)

beatles <- na.omit(beatles) # ensuring complete data
```

```
# selecting the variables of interest
beatles_interest <-
  beatles %>%
  select(
    year,
    danceability,
    energy,
    speechiness,
    acousticness,
    liveness,
    valence,
    length_sec
  )

str(beatles_interest) # checking the structure of beatles_interest
```

```
## 'data.frame':    193 obs. of  8 variables:
##  $ year        : int  1963 1963 1963 1963 1963 1963 1963 1963 1963 1963 ...
##  $ danceability: num  0.491 0.591 0.608 0.654 0.402 0.605 0.527 0.52 0.635 0.608 ...
##  $ energy      : chr  "0.801" "0.605" "0.565" "0.561" ...
##  $ speechiness : num  0.0361 0.0413 0.0336 0.0304 0.0504 0.0378 0.028 0.0806 0.0291 0.0345 ...
##  $ acousticness: num  0.27 0.707 0.635 0.608 0.607 0.767 0.334 0.386 0.389 0.778 ...
##  $ liveness    : num  0.0665 0.309 0.0601 0.129 0.736 0.0967 0.0702 0.227 0.0828 0.0926 ...
##  $ valence     : num  0.971 0.882 0.835 0.929 0.822 0.597 0.706 0.744 0.77 0.879 ...
##  $ length_sec  : num  174 109 177 145 146 ...
```

```
# converting energy from a character to numeric
beatles_interest$energy <- as.numeric(beatles_interest$energy)
```

```
row.names(beatles_interest) <- NULL
```

```
# storing the actual album values for each song
beatles_actual <- beatles$album
length(unique(beatles_actual)) # number of albums
```

```
## [1] 13
```

```
table(beatles_actual) # true frequency in each album
```

```
## beatles_actual
##                    A Hard Day's Night                            Abbey Road
##                                    13                                    17
##                      Beatles for Sale                                 Help!
##                                    14                                    14
##                             Let It Be                  Magical Mystery Tour
##                                    12                                    11
##                      Please Please Me                              Revolver
##                                    14                                    14
##                           Rubber Soul Sgt. Pepper's Lonely Hearts Club Band
##                                    14                                    13
##               The Beatles (white album)                       With The Beatles
##                                    30                                    14
##                      Yellow Submarine
##                                    13
```

Theoretical Run - 13 clusters

```
# Perform k-means with 13 clusters
theoretical_run <- kmeans(
  x = beatles_interest, # song characteristics
  centers = 13, # number of clusters
  iter.max = 25, # number of maximum iterations
  nstart = 25 # number of random starting values
)
```

```
# Within-cluster sum of squares
theoretical_run$withinss
```

```
##  [1]  167.8787  303.2543  935.7751  429.2801  390.5639  441.6683  882.7801
##  [8]  697.7015  410.9633  602.9617 1135.3135 1200.6902 1462.7289
```

```
# Variance explained
theoretical_run$betweenss / theoretical_run$totss
```

```
## [1] 0.9859843
```

```
# between sum of squares /total sum of squares
```

```
# cluster frequencies compared to actual frequencies
table(theoretical_run$cluster)
```

```
##
##  1  2  3  4  5  6  7  8  9 10 11 12 13
## 21 28 10 21 24 27 23  4 12  2  2 13  6
```

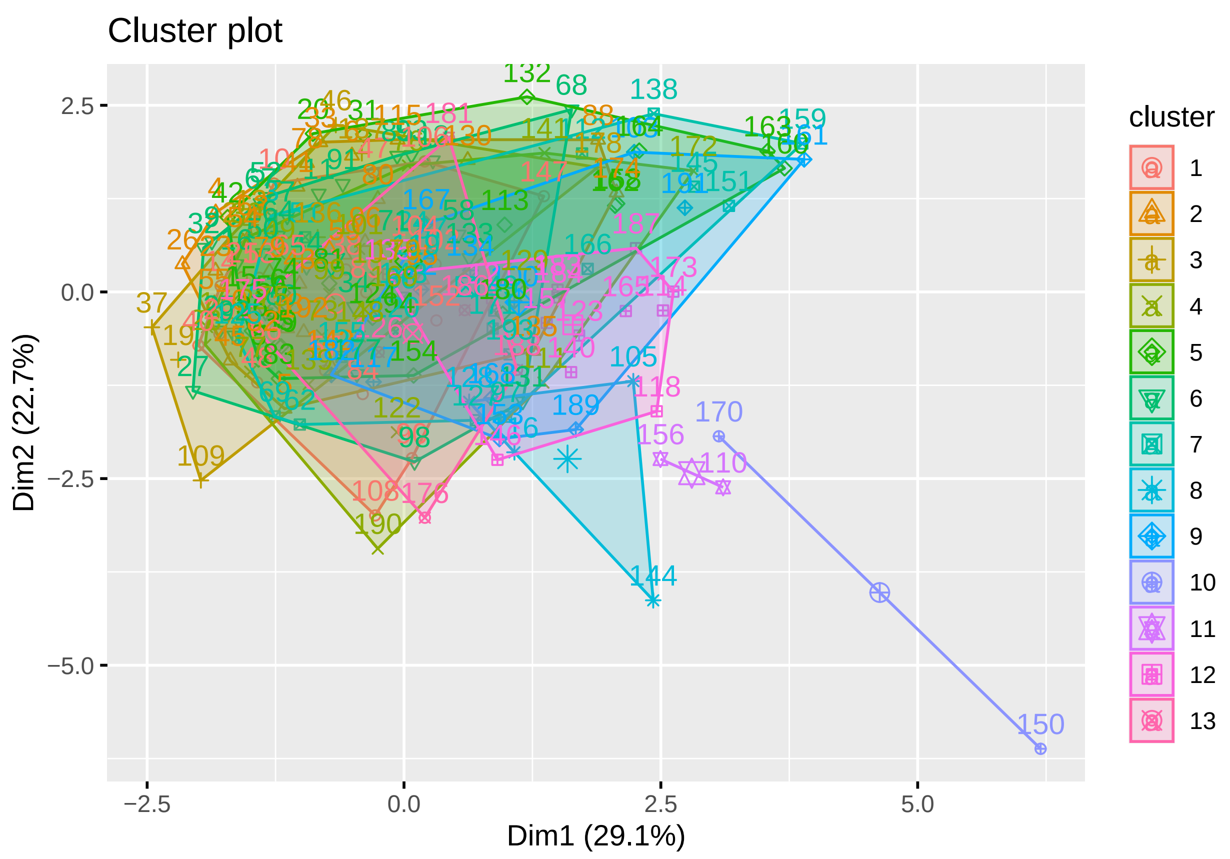
```
table(beatles_actual)
```

```
## beatles_actual
##                   A Hard Day's Night                           Abbey Road
##                                   13                                   17
##                     Beatles for Sale                                Help!
##                                   14                                   14
##                           Let It Be                  Magical Mystery Tour
##                                   12                                   11
##                     Please Please Me                             Revolver
##                                   14                                   14
##                          Rubber Soul Sgt. Pepper's Lonely Hearts Club Band
##                                   14                                   13
##             The Beatles (white album)                      With The Beatles
##                                   30                                   14
##                     Yellow Submarine
##                                   13
```

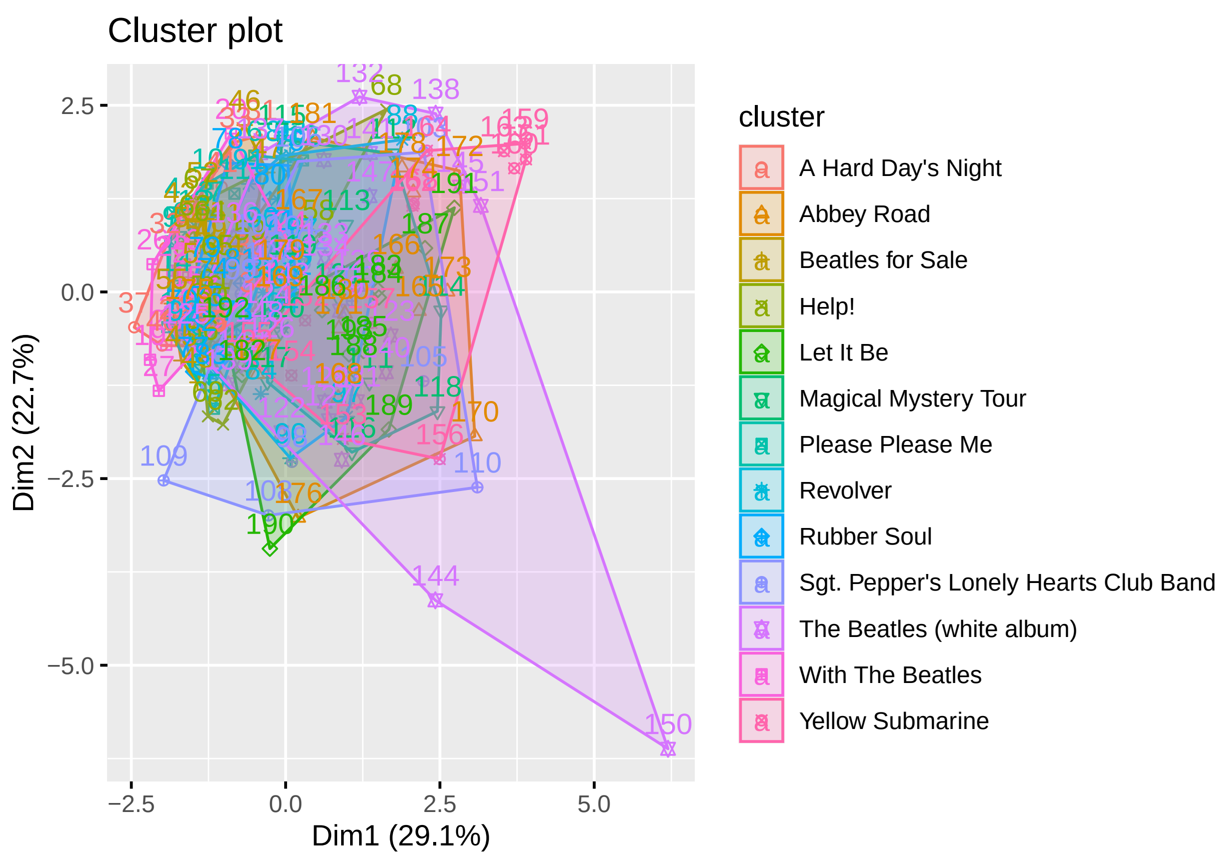
```
# centers of the 13 clusters
round(theoretical_run$centers, 4)
```

```
##        year danceability energy speechiness acousticness liveness valence
## 1  1965.286       0.5495 0.5969      0.0472       0.4143   0.2906  0.7617
## 2  1965.250       0.5757 0.5399      0.0395       0.4402   0.1634  0.7467
## 3  1965.800       0.5739 0.5496      0.0368       0.4339   0.2715  0.7891
## 4  1966.333       0.5490 0.5894      0.0451       0.2519   0.1758  0.7037
## 5  1966.083       0.5283 0.4889      0.0466       0.4178   0.2188  0.5687
## 6  1964.815       0.5210 0.5319      0.0406       0.4220   0.2339  0.7164
## 7  1967.000       0.4991 0.5065      0.0401       0.3778   0.2082  0.5275
## 8  1967.500       0.3430 0.6300      0.0516       0.1107   0.5145  0.4978
## 9  1968.333       0.4448 0.4827      0.0359       0.3565   0.2044  0.5667
## 10 1968.500       0.2890 0.5650      0.1871       0.3962   0.4603  0.2605
## 11 1968.000       0.3770 0.6190      0.0520       0.1450   0.5110  0.1870
## 12 1968.461       0.4069 0.4734      0.0458       0.2866   0.1703  0.4784
## 13 1969.167       0.6168 0.5862      0.1163       0.3776   0.3367  0.5562
##    length_sec
## 1    157.2985
## 2    147.7976
## 3    101.4667
## 4    166.7474
## 5    136.6073
## 6    123.4053
## 7    183.8795
## 8    283.8302
## 9    211.2033
## 10   484.6665
## 11   361.2130
## 12   239.6502
## 13    51.4400
```

```
# Visualize 13 clusters
fviz_cluster(
  object = theoretical_run,
  data = beatles_interest
)
```



```r
# plotting actual album clusters
ground_truth <- theoretical_run
ground_truth$cluster <- beatles_actual
fviz_cluster(
  object = ground_truth,
  data = beatles_interest,
  show.clust.cent = FALSE
)
```



Removing Outliers

Songs 110, 144, 150, 156, and 170 (A Day in the Life, I Want You, Helter Skelter, Revolution 9, It's All Too Much,) are outlying songs as visualized in the cluster plot. I will remove them to see if that effects how the clustering is completed.

```
# removing outlier songs
beatles_interest_no_outliers <- beatles_interest[-c(110, 144, 150, 156, 170),]
beatles_actual_no_outliers <- beatles_actual[-c(110, 144, 150, 156, 170)]
```

```
# Perform k-means with 13 clusters
theoretical_run <- kmeans(
  x = beatles_interest_no_outliers, # song characteristics
  centers = 13, # number of clusters
  iter.max = 25, # number of maximum iterations
  nstart = 25 # number of random starting values
)
```

```
# Within-cluster sum of squares
theoretical_run$withinss
```

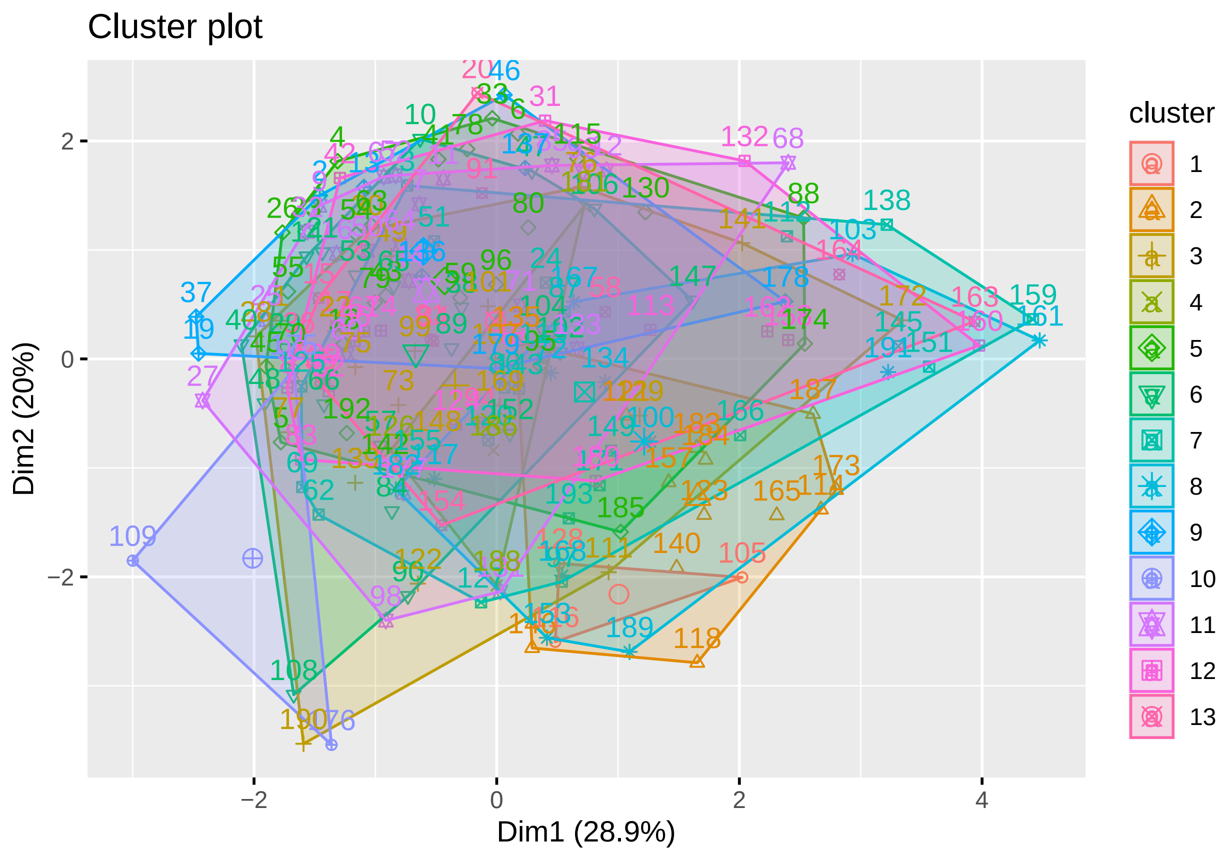
```
##  [1]  434.12407 1200.69023  429.28011  455.33157  271.21351  167.87872
##  [7]  882.78011  410.96331  375.90386   81.74334  300.70998  141.13998
## [13]  131.46743
```

```r
# Variance explained
theoretical_run$betweenss / theoretical_run$totss
```

```
## [1] 0.9845689
```

```r
# between sum of squares /total sum of squares
```

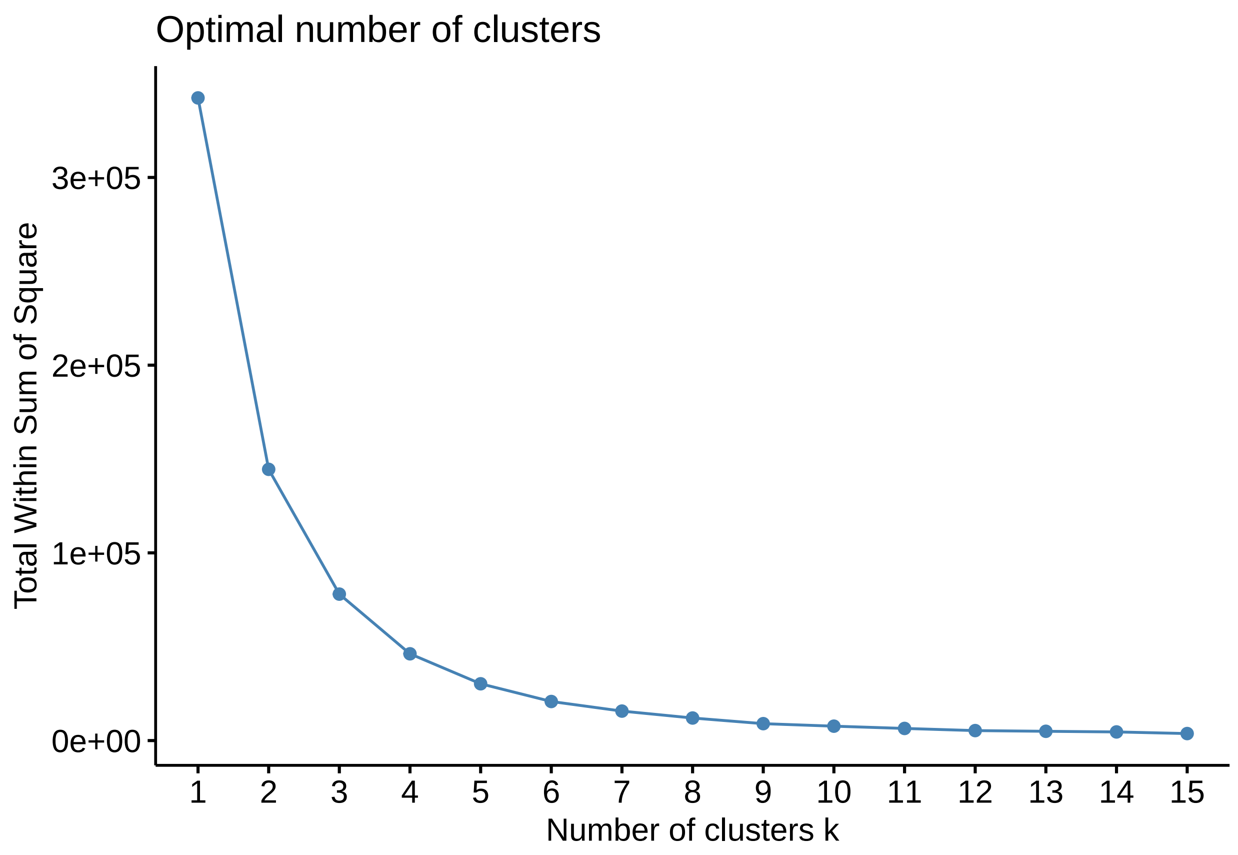
```r
# Visualize 13 clusters
fviz_cluster(
  object = theoretical_run,
  data = beatles_interest_no_outliers
)
```



Unsupervised Methods

Elbow Method

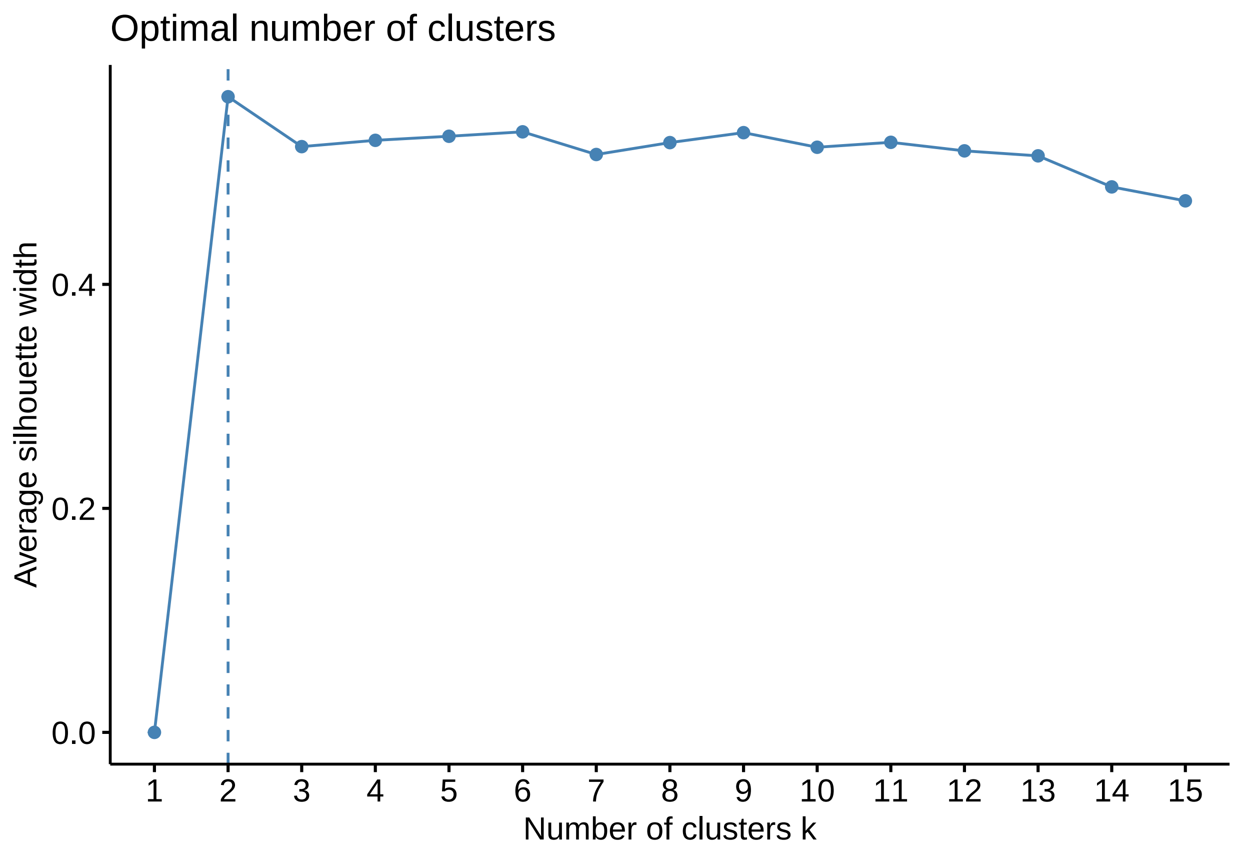
```r
fviz_nbclust(
  x = beatles_interest_no_outliers,
  FUNcluster = kmeans, # cluster function
  method = "wss", # within-cluster sum of squares
  k.max = 15,  # maximum number of clusters
  iter.max = 25, # same as our k-means setup
  nstart = 25 # same as our k-means setup
)
```



The elbow method indicates that around 2 to 4 clusters would be optimal in this analysis. However, the intepretation of the graph is subjective, so I will employ the silhouette and gap statistic methods to form a more concrete opinion.

Silhouette Method

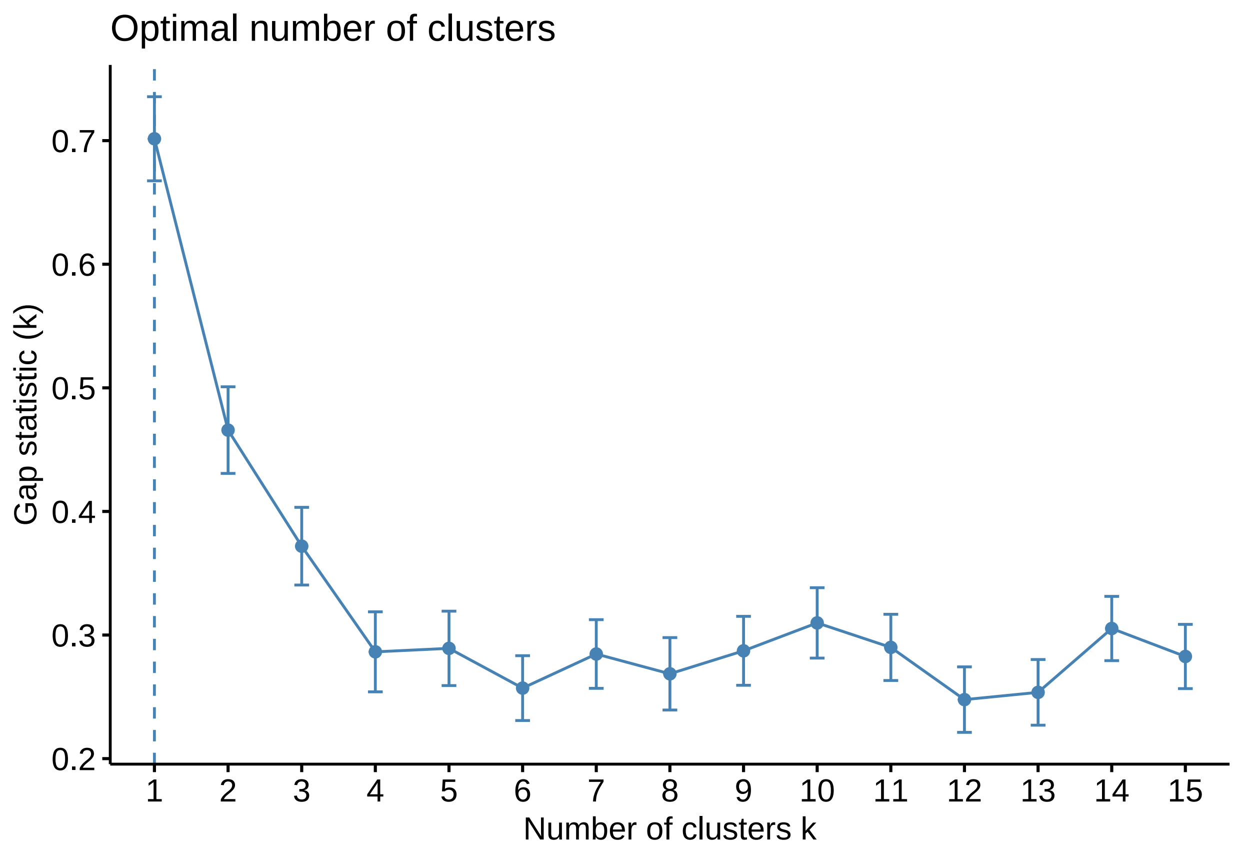
```
fviz_nbclust(
  x = beatles_interest_no_outliers,
  FUNcluster = kmeans, # cluster function
  method = "silhouette", # silhouette
  k.max = 15,  # maximum number of clusters
  iter.max = 25, # same as our k-means setup
  nstart = 25 # same as our k-means setup
)
```



The silhouette method indicates that 2 clusters are optimal.

Gap Statistic Method

```
set.seed(1234) # set seed

kmeans_gap <- clusGap(
  x = beatles_interest_no_outliers,
  FUNcluster = kmeans,
  iter.max = 25, # same as our k-means setup
  nstart = 25, # same as our k-means setup
  K.max = 15, # maximum number of clusters
  B = 100 # takes some time...
)

# Plot gap statistic
fviz_gap_stat(kmeans_gap)
```



The gap statistic method indicates that 1 cluster is optimal which makes some intuitive sense as all of these songs were made by The Beatles in a seven year period.

Combining all of the results from the three unsupervised methods indicates that 2 clusters are the optimal clustering for The Beatles's songs.

Unsupervised Run - 2 Clusters

```
set.seed(1234) # set seed

# Perform k-means with 2 clusters
unsupervised_run <- kmeans(
  x = beatles_interest_no_outliers,
  centers = 2,
  iter.max = 25,
  nstart = 25
)
```

```
# Within-cluster sum of squares
unsupervised_run$withinss
```

```
## [1] 52375.16 92123.42
```

```r
# Variance explained
unsupervised_run$betweenss / unsupervised_run$totss
```

```
## [1] 0.5779522
```

```r
# between sum of squares / # total sum of squares

# Check out cluster frequencies
table(unsupervised_run$cluster)
```

```
##
##   1   2
##  54 134
```

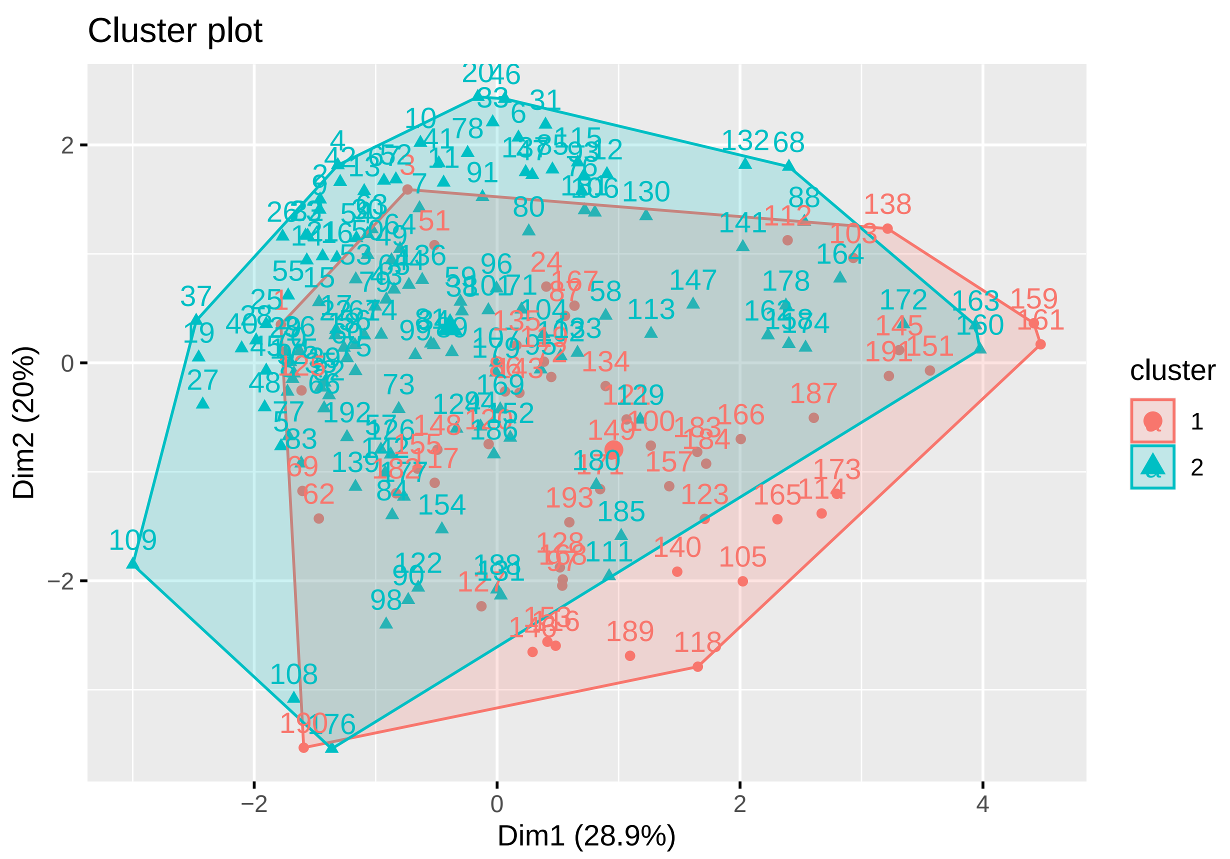
```r
# Actual frequencies
table(beatles_actual)
```

```
## beatles_actual
##                      A Hard Day's Night                          Abbey Road
##                                      13                                  17
##                        Beatles for Sale                               Help!
##                                      14                                  14
##                               Let It Be                Magical Mystery Tour
##                                      12                                  11
##                        Please Please Me                            Revolver
##                                      14                                  14
##                             Rubber Soul Sgt. Pepper's Lonely Hearts Club Band
##                                      14                                  13
##               The Beatles (white album)                    With The Beatles
##                                      30                                  14
##                        Yellow Submarine
##                                      13
```

```r
# centroids of unsupervised run
unsupervised_run$centers
```

```
##       year danceability   energy speechiness acousticness  liveness   valence
## 1 1967.667    0.4626111 0.508487   0.0412500    0.3230285 0.2186981 0.5465556
## 2 1965.664    0.5500448 0.544497   0.0459709    0.4032437 0.2206381 0.6959172
##   length_sec
## 1   208.6487
## 2   136.9757
```

```r
# plot the clusters for the unsupervised_run
fviz_cluster(
  object = unsupervised_run,
  data = beatles_interest_no_outliers
)
```



```
original_albums <-
  c("A Hard Day's Night",
  "Abbey Road",
  "Beatles for Sale",
  "Help!",
  "Let It Be",
  "Magical Mystery Tour",
  "Please Please Me",
  "Revolver",
  "Rubber Soul",
  "Sgt. Pepper's Lonely Hearts Club Band",
  "The Beatles (white album)",
  "With The Beatles",
  "Yellow Submarine")

# convert clusters into a comparable formal
names(original_albums) <- original_albums
album_classes <- original_albums[beatles_actual_no_outliers]

numeric_classes <- as.numeric(factor(album_classes))

theoretical_classes <- theoretical_run$cluster
```

```
# Compare using Adjusted Rand Index
ARI(
  numeric_classes,
  theoretical_classes
)
```

```
## [1] 0.02128763
```

```
# Compare using Adjusted Mutual Information
AMI(
  numeric_classes,
  theoretical_classes
)
```

```
## [1] 0.065915
```

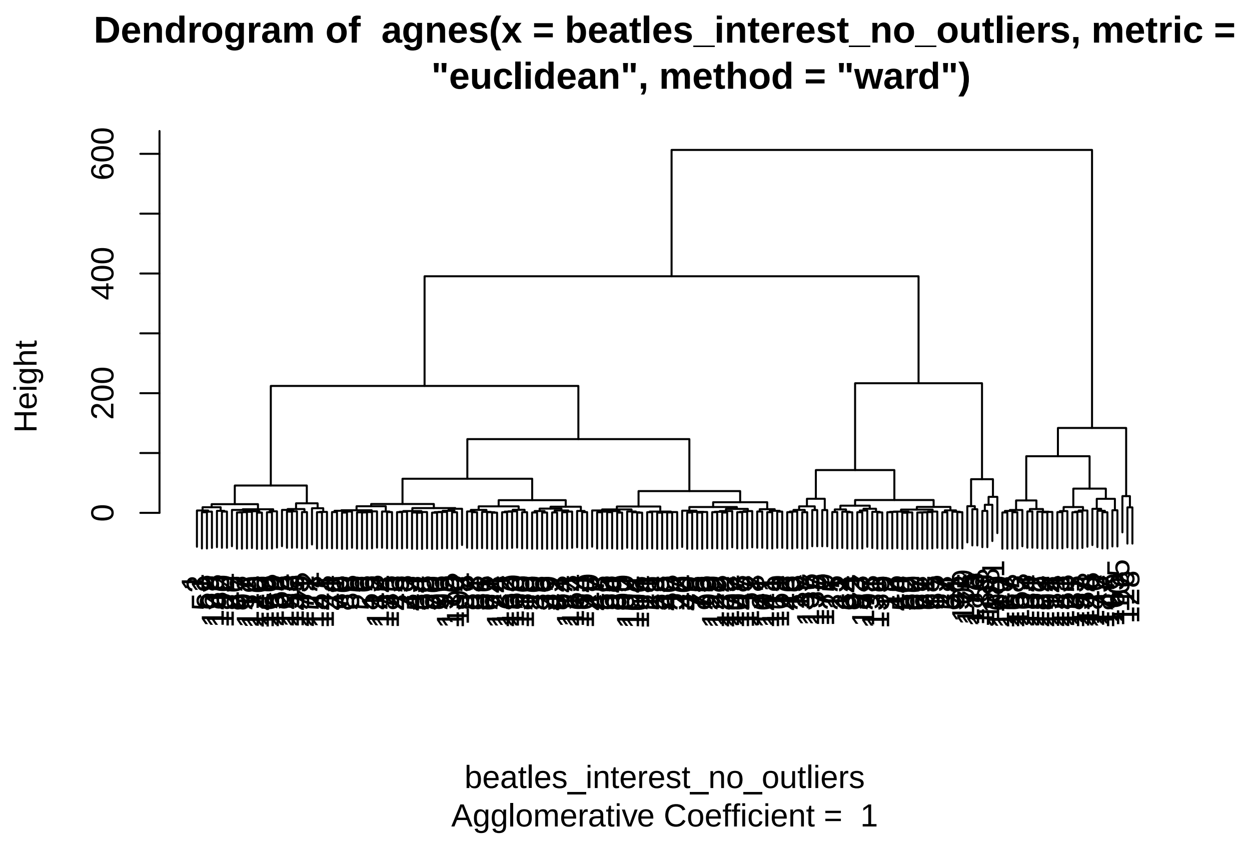
Hierarchical Clustering

Theoretical Methods

```
methods <- c(
  "complete", "average", "single",
  "complete", "ward"
)

# Get agglomerative coefficient results
sapply(
  X = methods,
  function(method){

    # Apply agglomerative methods
    agnes(
      x = beatles_interest_no_outliers, # data
      metric = "euclidean", # distance
      method = method # linking
    )$ac

  }
)
```

```
##  complete   average    single  complete      ward
## 0.9920831 0.9807607 0.9082052 0.9920831 0.9964185
```

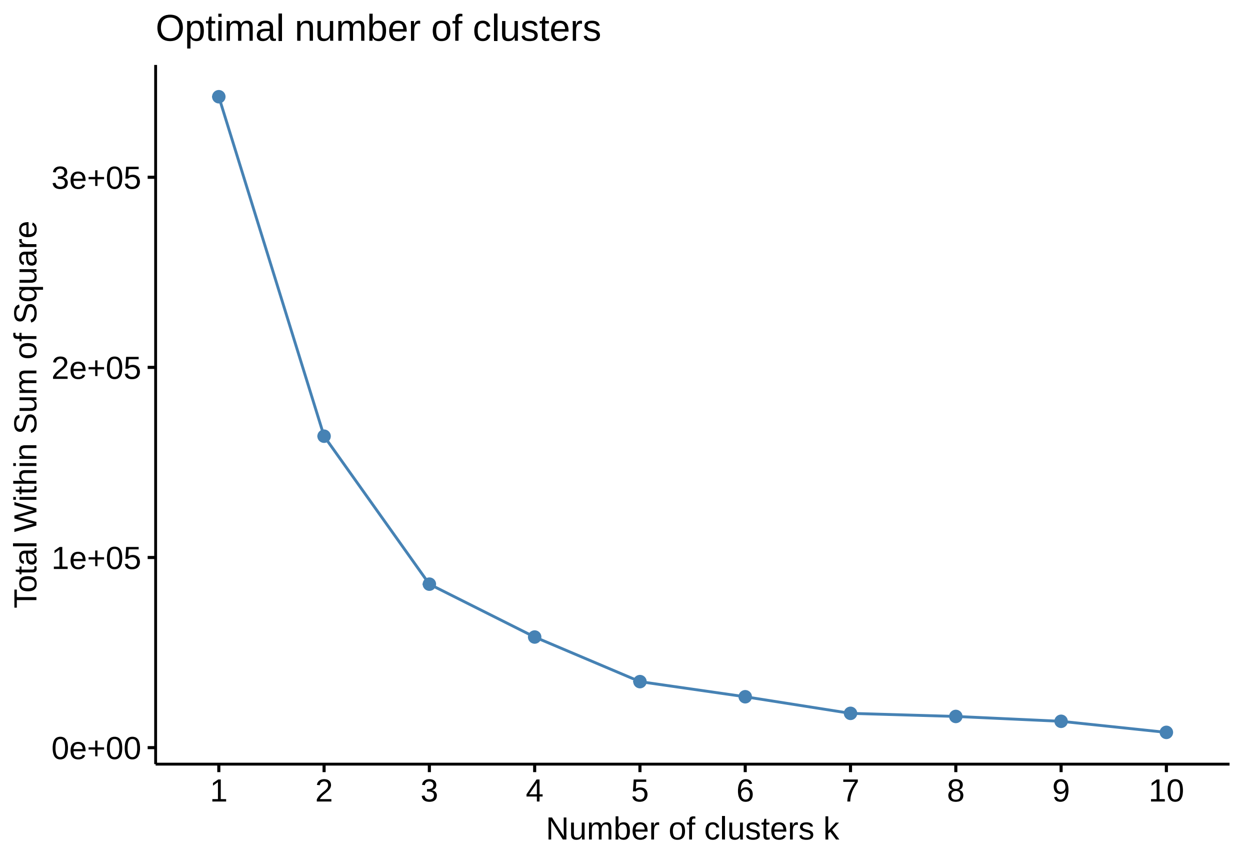
```
# using ward linking method
beatles_hclust_ward <- agnes(
  x = beatles_interest_no_outliers, # data
  metric = "euclidean", # distance
  method = "ward" # linking
)

# plot ward dendogram
plot(beatles_hclust_ward, which.plots = 2)
```



Unsupervised Methods

Elbow Method

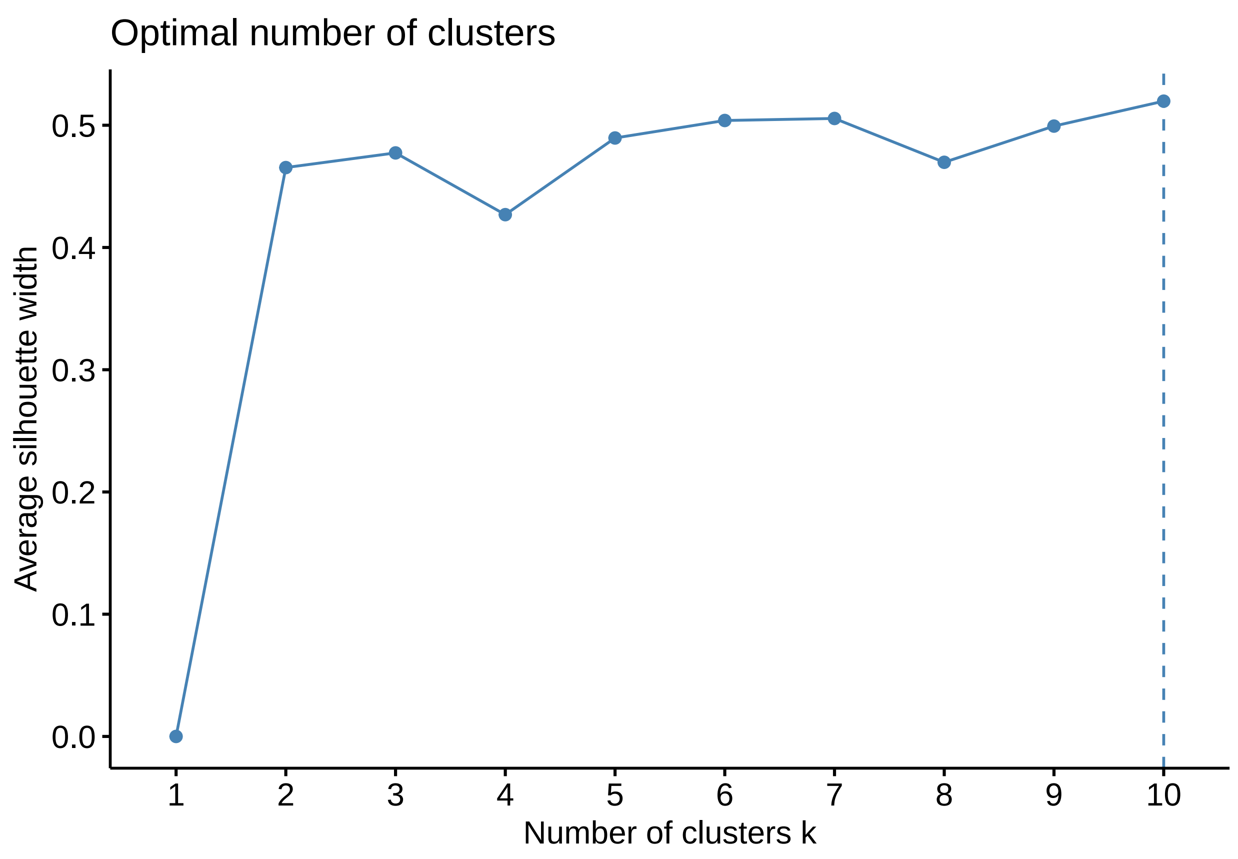
```r
fviz_nbclust(
  x = beatles_interest_no_outliers,
  FUNcluster = hcut, # cluster function
  hc_method = "ward.D", # use Ward's
  method = "wss" # within-cluster sum of squares
)
```



The elbow method seems to indicate that around 2 to 4 clusters is optimal.

Silhouette Method

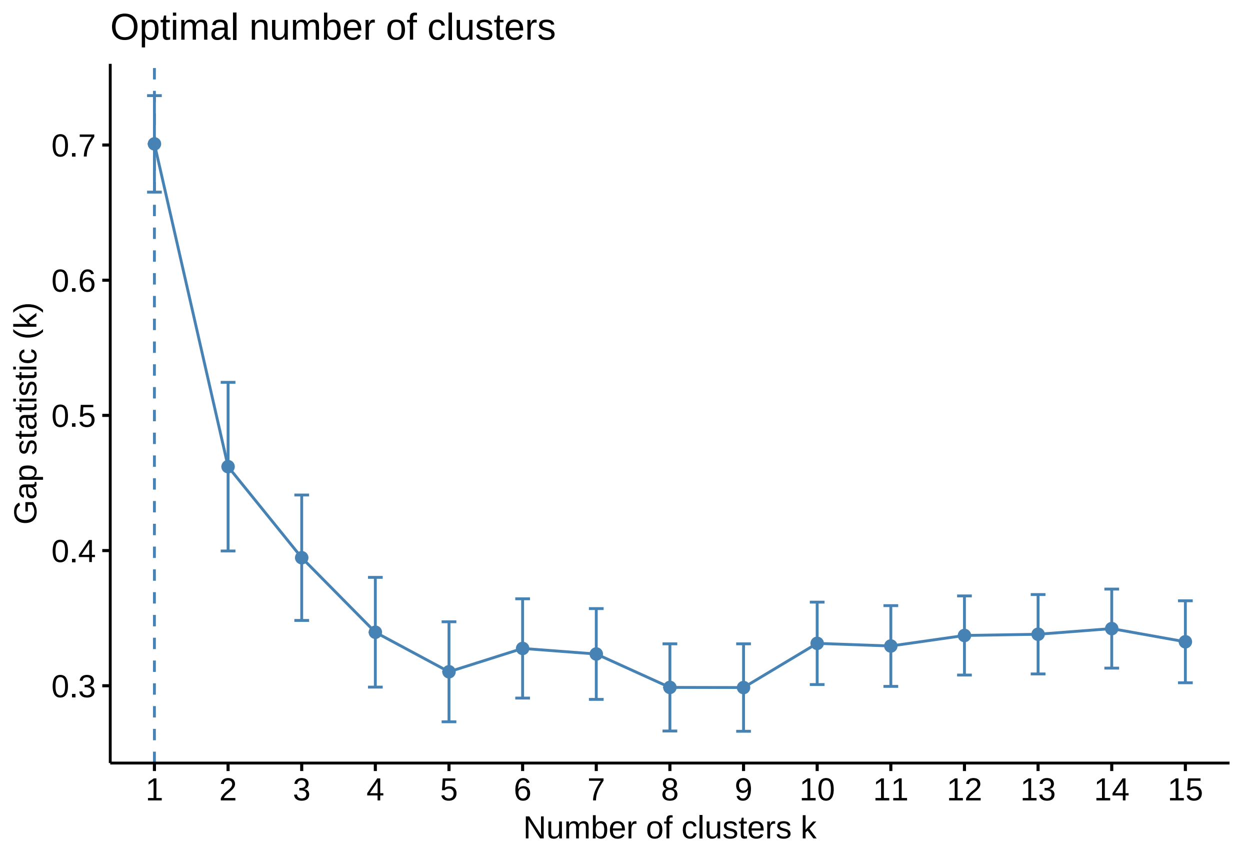
```
fviz_nbclust(
  x = beatles_interest_no_outliers,
  FUNcluster = hcut, # cluster function
  hc_method = "ward.D", # use Ward's
  method = "silhouette" # silhouette method
)
```



The silhouette method indicates that 10 clusters are optimal.

Gap Statistic

```
# Set seed
set.seed(1234)

# Custom hierarchical clustering function
custom_hclust <- function(x, k, ...){
  list(
    cluster = cutree(
      # Base R version of `agnes`
      # Much faster
      hclust(
        dist(x), method = "ward.D",
        ...
      ),
      k = k
    )
  )
}

## Perform bootstrap
hclust_gap <- clusGap(
  x = beatles_interest_no_outliers,
  FUNcluster = custom_hclust,
  K.max = 15, # same as k-means setup
  B = 100 # takes some time...
)

# Plot gap statistic
fviz_gap_stat(hclust_gap)
```

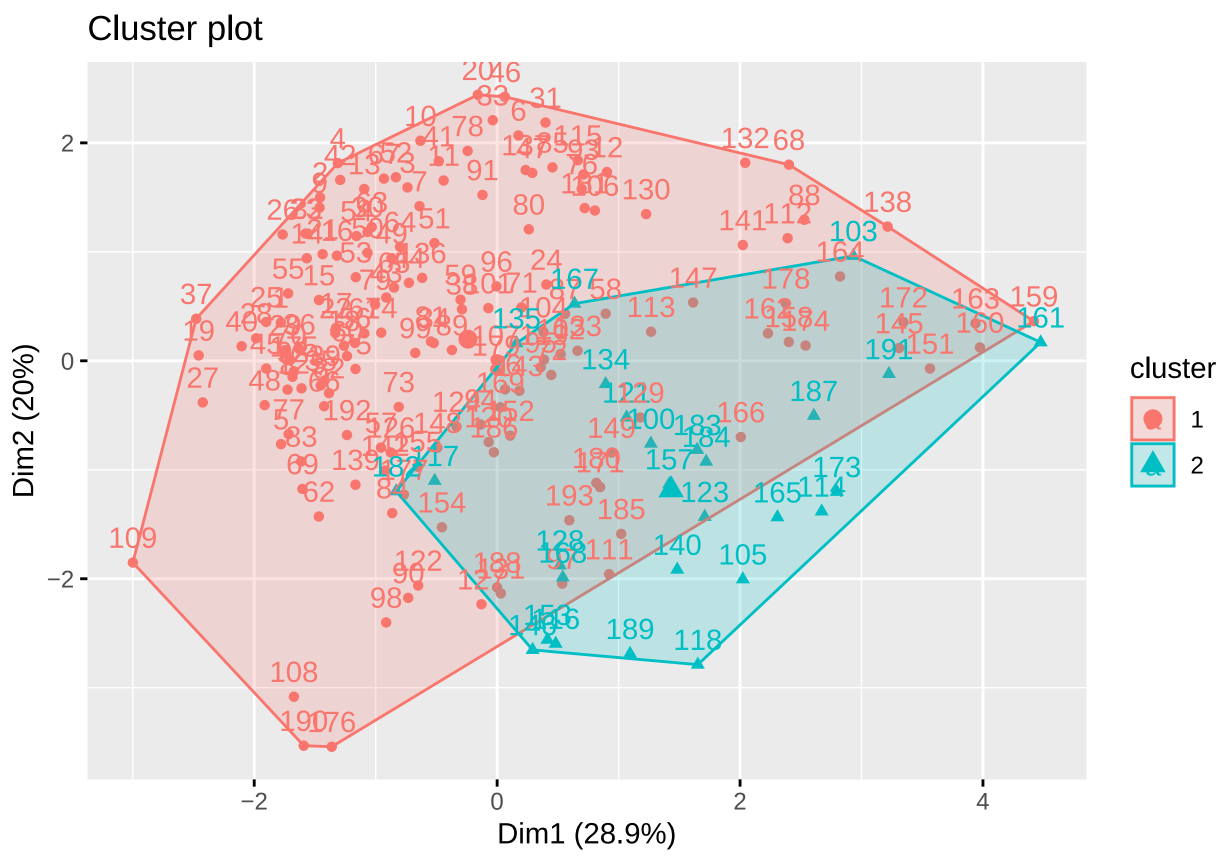


The gap statistic indicates that one cluster is optimal.

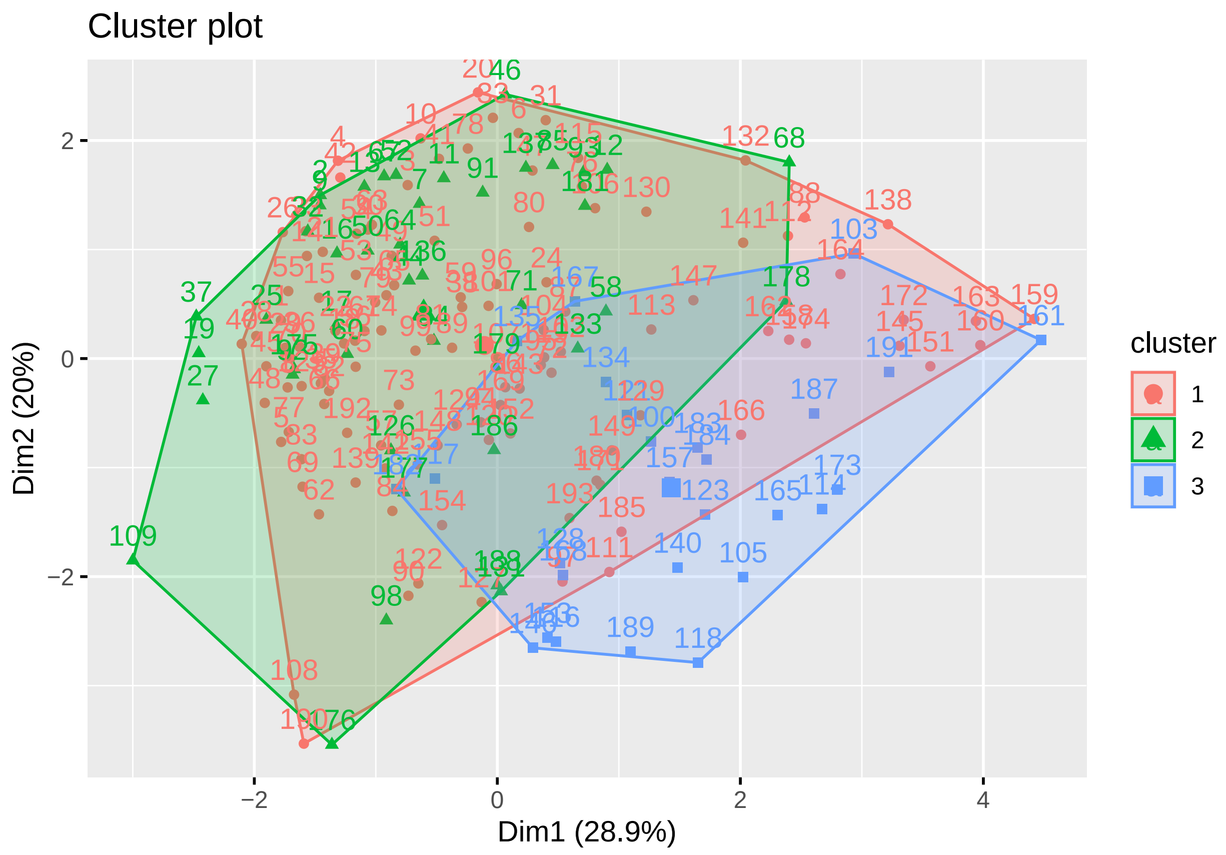
Hierarchical Cluster Plots

```
# corresponding cuts of the dendogram
cut_two <- cutree(beatles_hclust_ward, k = 2)
cut_three <- cutree(beatles_hclust_ward, k = 3)
cut_ten <- cutree(beatles_hclust_ward, k = 10)
```

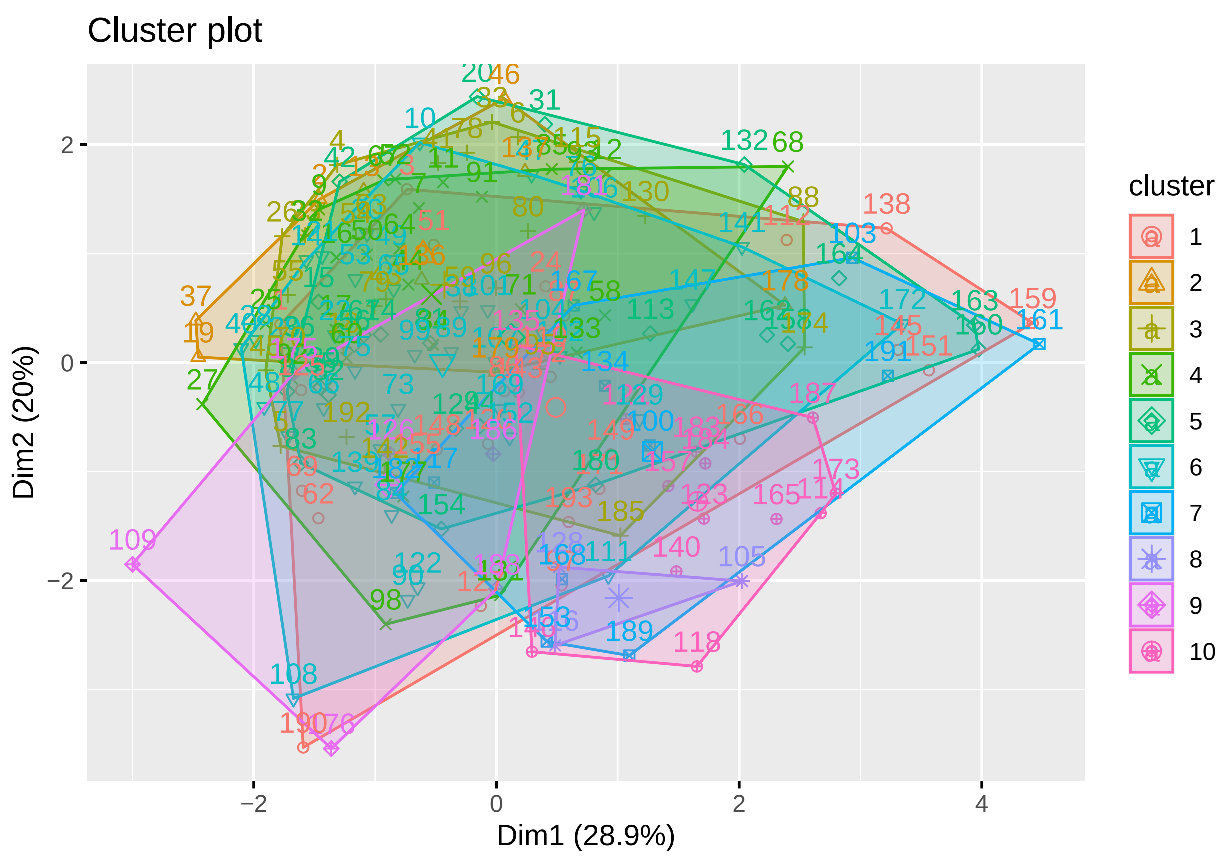
```
# Two Clusters
fviz_cluster(
  list(
    data = beatles_interest_no_outliers,
    cluster = cut_two
  )
)
```



```
# Three Clusters
fviz_cluster(
  list(
    data = beatles_interest_no_outliers,
    cluster = cut_three
  )
)
```



```
# Ten Clusters
fviz_cluster(
  list(
    data = beatles_interest_no_outliers,
    cluster = cut_ten
  )
)
```



Final Analysis

```
# putting hierarchical clusters into data
final_analysis <-
  beatles_interest_no_outliers %>%
  mutate(cut_two, cut_three, cut_ten)
```

```
# summarizing based on two clusters
final_analysis %>%
  group_by(cut_two) %>%
  summarise("Mean Year" = mean(year),
            "Mean Danceability" = mean(danceability),
            "Mean Energy" = mean(energy),
            "Mean Speechiness" = mean(speechiness),
            "Mean Acousticness" = mean(acousticness),
            "Mean Liveness" = mean(liveness),
            "Mean Valence" = mean(valence),
            "Mean Length" = mean(length_sec))
```

```
## # A tibble: 2 x 9
##   cut_two `Mean Year` Mean Dan~1 Mean ~2 Mean ~3 Mean ~4 Mean ~5 Mean ~6 Mean ~7
##     <int>       <dbl>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1       1       1966.      0.542   0.542  0.0452   0.392   0.221   0.674    145.
## 2       2       1968.      0.420   0.485  0.0412   0.308   0.217   0.528    234.
## # ... with abbreviated variable names 1: ‘Mean Danceability‘, 2: ‘Mean Energy‘,
## #   3: ‘Mean Speechiness‘, 4: ‘Mean Acousticness‘, 5: ‘Mean Liveness‘,
## #   6: ‘Mean Valence‘, 7: ‘Mean Length‘
```

```
# summarizing based on three clusters
final_analysis %>%
  group_by(cut_three) %>%
  summarise("Mean Year" = mean(year),
            "Mean Danceability" = mean(danceability),
            "Mean Energy" = mean(energy),
            "Mean Speechiness" = mean(speechiness),
            "Mean Acousticness" = mean(acousticness),
            "Mean Liveness" = mean(liveness),
            "Mean Valence" = mean(valence),
            "Mean Length" = mean(length_sec))
```

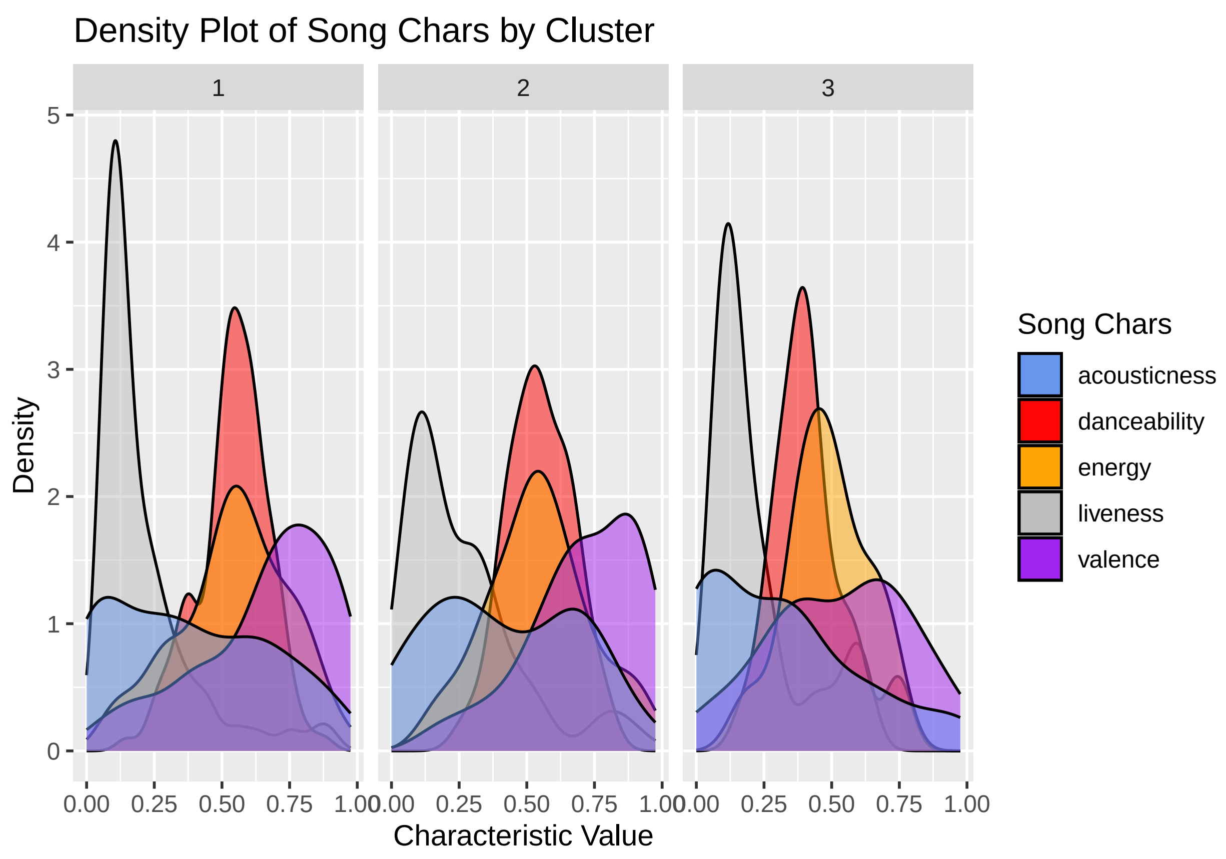
```
## # A tibble: 3 x 9
##   cut_three ‘Mean Year‘ Mean D~1 Mean ~2 Mean ~3 Mean ~4 Mean ~5 Mean ~6 Mean ~7
##       <int>       <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1         1       1966.    0.541   0.542  0.0433   0.383   0.207   0.660    158.
## 2         2       1966.    0.547   0.544  0.0503   0.419   0.257   0.711    108.
## 3         3       1968.    0.420   0.485  0.0412   0.308   0.217   0.528    234.
## # ... with abbreviated variable names 1: ‘Mean Danceability‘, 2: ‘Mean Energy‘,
## #   3: ‘Mean Speechiness‘, 4: ‘Mean Acousticness‘, 5: ‘Mean Liveness‘,
## #   6: ‘Mean Valence‘, 7: ‘Mean Length‘
```

```
# summarizing based on ten clusters
final_analysis %>%
  group_by(cut_ten) %>%
  summarise("Mean Year" = mean(year),
            "Mean Danceability" = mean(danceability),
            "Mean Energy" = mean(energy),
            "Mean Speechiness" = mean(speechiness),
            "Mean Acousticness" = mean(acousticness),
            "Mean Liveness" = mean(liveness),
            "Mean Valence" = mean(valence),
            "Mean Length" = mean(length_sec))
```

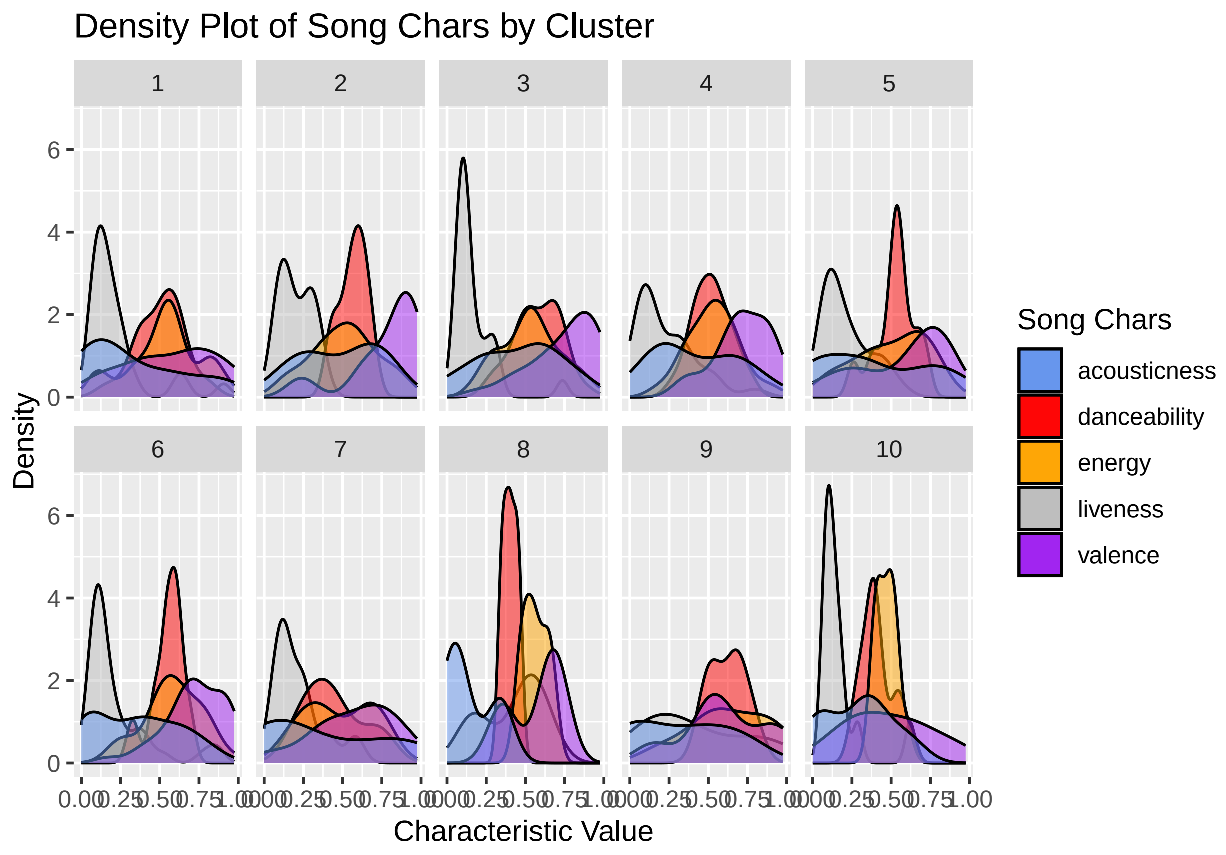
```
## # A tibble: 10 x 9
##    cut_ten ‘Mean Year‘ Mean Da~1 Mean ~2 Mean ~3 Mean ~4 Mean ~5 Mean ~6 Mean ~7
##      <int>       <dbl>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1       1       1967.     0.505   0.532  0.0413   0.338   0.220   0.565   183.
##  2       2       1966.     0.565   0.512  0.0359   0.481   0.203   0.779   104.
##  3       3       1965.     0.579   0.532  0.0397   0.456   0.166   0.745   148.
##  4       4       1965.     0.521   0.532  0.0406   0.422   0.234   0.716   123.
##  5       5       1966.     0.527   0.500  0.0461   0.401   0.214   0.578   137.
##  6       6       1966.     0.549   0.583  0.0455   0.351   0.223   0.721   161.
##  7       7       1969.     0.441   0.478  0.0364   0.377   0.218   0.575   212.
##  8       8       1967.     0.402   0.563  0.0389   0.147   0.416   0.57    289.
##  9       9       1969.     0.622   0.629  0.106    0.325   0.416   0.602    55.4
## 10      10       1968.     0.407   0.473  0.0458   0.287   0.170   0.478   240.
## # ... with abbreviated variable names 1: ‘Mean Danceability‘, 2: ‘Mean Energy‘,
## #   3: ‘Mean Speechiness‘, 4: ‘Mean Acousticness‘, 5: ‘Mean Liveness‘,
## #   6: ‘Mean Valence‘, 7: ‘Mean Length‘
```

Plots of Final Analysis

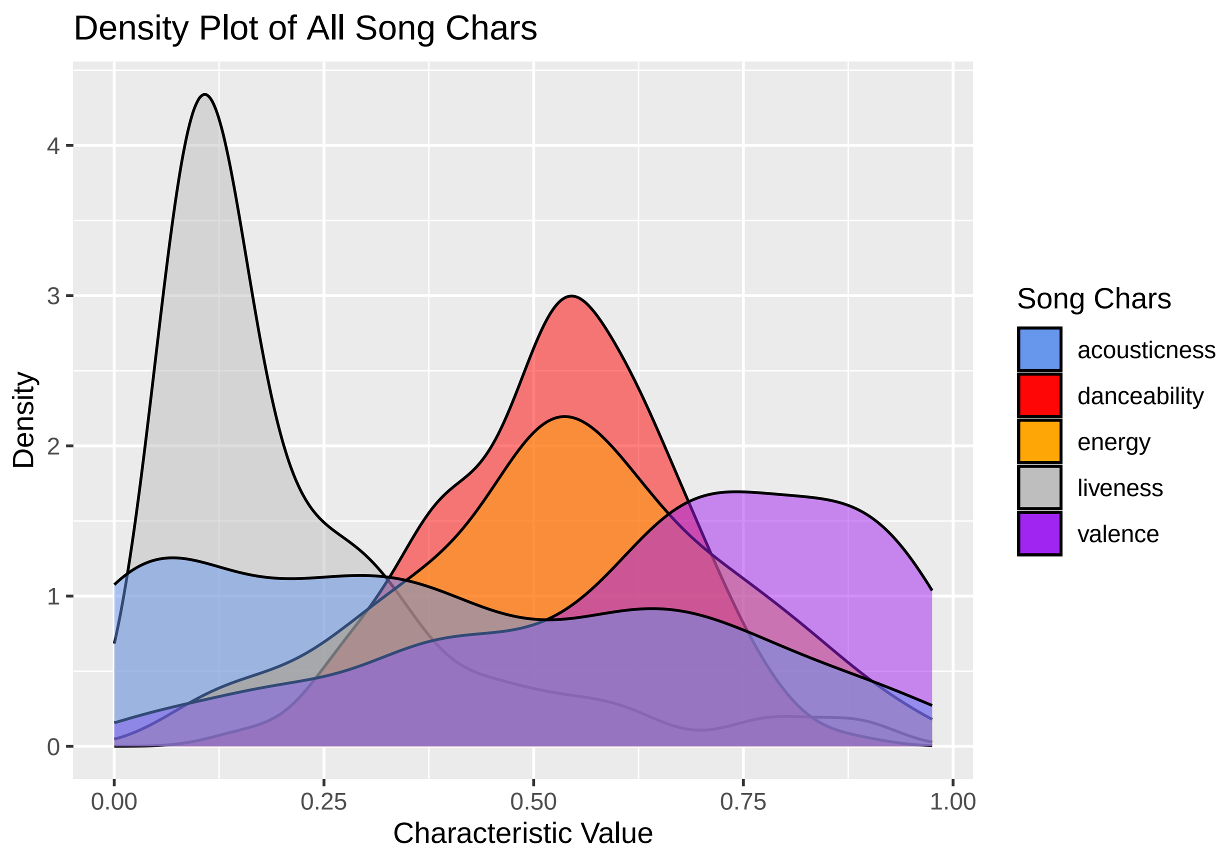
```
cols <- c("liveness"="grey","danceability"="red","energy"="orange", "valence"="purple", "acousticness" =

final_analysis %>%
  ggplot() +
  geom_density( # density plot
    aes(
    x = liveness,
    fill = "liveness"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = danceability,
    fill = "danceability"),
    alpha = 0.5
    ) +
  geom_density(
    aes(x = energy,
    fill = "energy"),
    alpha = 0.5
    ) +
  geom_density(
    aes(
    x = valence,
    fill = "valence"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = acousticness,
    fill = "acousticness"),
    alpha = 0.5
    ) +
  facet_grid(~cut_three) +
  scale_fill_manual(name = "Song Chars", values = cols) + # add legend
  labs(x = "Characteristic Value",
       y = "Density",
       title = "Density Plot of Song Chars by Cluster") # better labels
```



```
# plot of char density for ten clusters
final_analysis %>%
  ggplot() +
  geom_density( # density plot
    aes(
    x = liveness,
    fill = "liveness"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = danceability,
    fill = "danceability"),
    alpha = 0.5
    ) +
  geom_density(
    aes(x = energy,
    fill = "energy"),
    alpha = 0.5
    ) +
  geom_density(
    aes(
    x = valence,
    fill = "valence"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = acousticness,
    fill = "acousticness"),
    alpha = 0.5
    ) +
  facet_wrap(~cut_ten,
             ncol = 5) +
  scale_fill_manual(name = "Song Chars", values = cols) + # legend
  labs(x = "Characteristic Value",
       y = "Density",
       title = "Density Plot of Song Chars by Cluster") # better labels
```



```
# Plot of char density for all songs
final_analysis %>%
  ggplot() +
  geom_density(
    aes(
    x = liveness,
    fill = "liveness"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = danceability,
    fill = "danceability"),
    alpha = 0.5
    ) +
  geom_density(
    aes(x = energy,
    fill = "energy"),
    alpha = 0.5
    ) +
  geom_density(
    aes(
    x = valence,
    fill = "valence"),
    alpha = 0.5
  ) +
  geom_density(
    aes(
    x = acousticness,
    fill = "acousticness"),
    alpha = 0.5
    ) +
  scale_fill_manual(name = "Song Chars", values = cols) +
  labs(x = "Characteristic Value",
       y = "Density",
       title = "Density Plot of All Song Chars")
```



Discussion

The Beatles were revolutionary artists during the 1960s who churned out 13 of the best albums of all time. Many of these albums explored different genres from rock and roll to pop to incorporating international (specifically South Asian) influences. Using k-means clustering, I found that there was little ability for the algorithm to separate songs into their respective albums based on Spotify metadata. While the variance explained was high, this was likely due to there being 13 clusters. The clusters had vastly different frequencies than the actual albums, and the centroids and visualization indicated high overlap in the data. Removing outliers from the data improved the clustering somewhat, but the original data, when clustered by actual album is not very divisible into groups. While some claim that the Beatles created drastically different albums, the data shows that the albums mainly overlapped with similar characteristics between them. This is backed up by the AMI and ARI computed between the ground truth (the actual albums) and the 13 k-means clusters. The AMI and ARI are very close to 0 indicating that the clusters do not line up very well showing how Spotify metadata can not classify the Beatles albums. Using unsupervised methods showed that two clusters would be optimal for this data, based on elbow, silhouette, and the gap statistic. The first cluster was older Beatles songs that were more danceable, energetic, speechiness, acousticness, liveness, and valence but shorter songs.

Hierarchical clustering again reinforced the fact that the Spotify metadata is not good at classifying Beatles songs into their albums. Unsupervised methods indicated that 2, 3 or 10 would be best. Hierarchical clustering does not generate vastly different results than k-means clustering. After completing the clustering, I studied how the song characteristics were distributed for each cluster. For three clusters, the first cluster was more low liveness and medium danceable songs while the second cluster was more balanced. The third cluster had the most medium energy songs. Thus, hierarchical clustering can identify some patterns in Beatles songs, but not to the extent that I initially predicted. The 10 clusters provided other interesting results, especially when compared to the characteristic distribution of all of the songs.