Brief Intro To Clustering in R (And Also R Markdown)

Khoi Trinh

2023-02-22

Before working with R markdown, we need a few packages

```
# The rmarkdown package
#install.packages('rmarkdown')
```

If you want to generate pdf files, you will need to install LaTeX

If you don't plan on using LaTeX anywhere outside of R markdown, I suggest TinyTex

```
#install.packages('tinytex')
#tinytex::install_tinytex()
```

Note that you can and should run these above commands in the RStudio console

Let's Explore Clustering

```
library(cluster)
library(NbClust)
library(factoextra)
library(dplyr)
library(kmed)
```

First, we need some data

Data description

The data I chose is my own Spotify streaming history for the past year; you can find how to get your own Spotify data here

Then, follow these instructions to obtain the song traits.

Misc data processing:

Read in the data

```
spotify = read.csv("final.csv")
```

The data had almost 60,000 observations, out of those, only the numeric data will be considered.

And out of the numeric columns, we will drop columns 1, 2, 5, 7, 14, and 15 as they are not song traits.

```
numericData = spotify %>% #Add data
  dplyr::select(where(is.numeric)) #finds where is.numeric is true

# drop the mentioned columns
numericData <- subset(numericData, select=-c(1,2,5,7,14,15))
```

Time To Create Some Clusters

One last data processing step

Scale the data, and use that to create our clusters. We need to scale the data as most of the traits are < 1; but tempo are not

So scaling is needed to not skewed the clusters.

```
clusterData <- scale(numericData)
```

K means

For this clustering method, the `kmeans()` function from the stats package is used. Let's start with 2 clusters. Normally, there are ways to determine an optimal number of clusters, but for the sake of simplicity, let's stick to 2, maybe we can change it later.

```
kmean <- kmeans(clusterData,2, nstart=10)
```

Clusters Analysis

First, let's see the size of the clusters

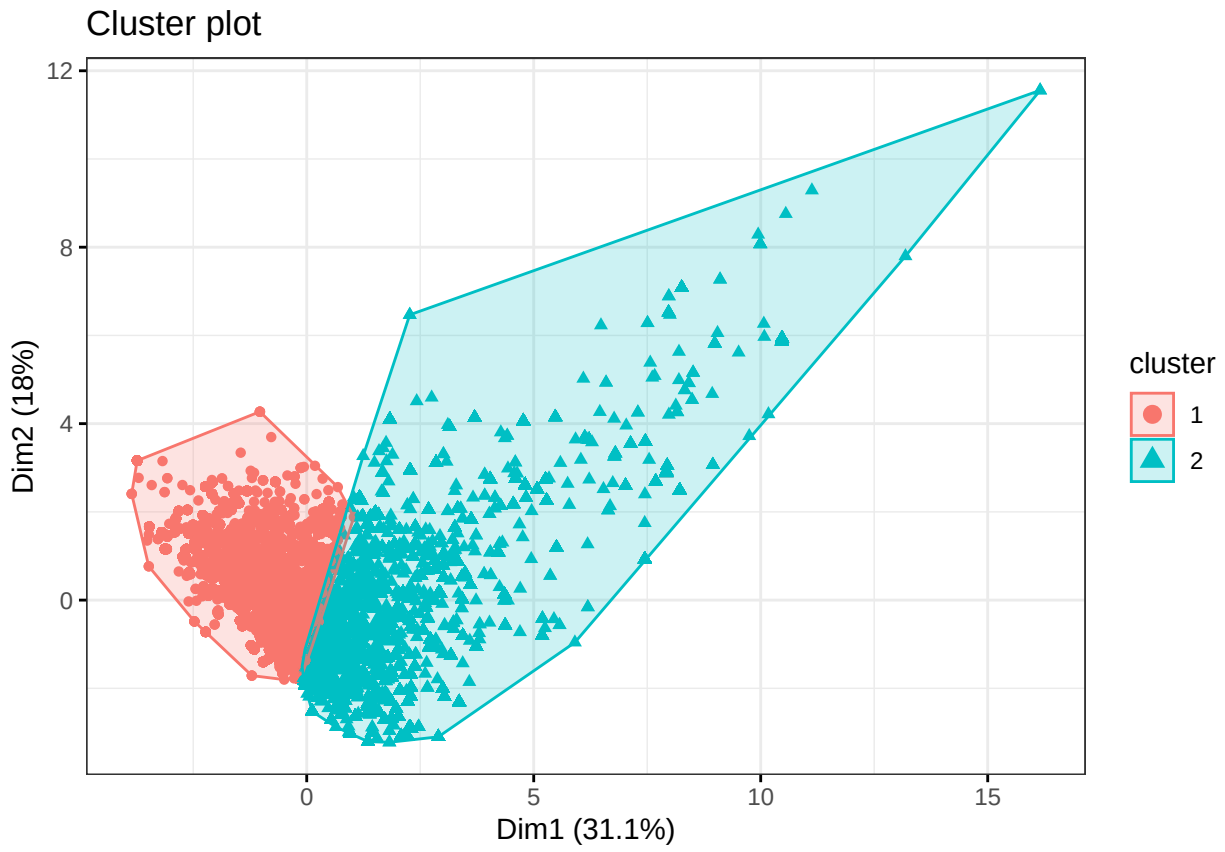
```
kmeansize <- kmean$size
kmeansize
```

```
## [1] 35993 20217
```

We have 2 clusters, with size 36053 and 20157.

Visually, the clusters look like so. We can see that the clusters have a little bit of overlap, but overall, it looks good.

```
fviz_cluster(kmean, data = clusterData, geom = "point", ellipse.type = "convex",
             ggtheme = theme_bw())
```



Cluster Intepretation

```
kmeaninfo <- data.frame(kmean$centers, kmean$size)
kmeaninfo
```

```
##   danceability     energy   loudness speechiness acousticness instrumentalness
## 1   -0.4119872  0.4312601  0.3184585   0.3595382   -0.2504674      -0.04796315
## 2    0.7334746 -0.7677868 -0.5669624  -0.6400979    0.4459154       0.08539039
##     liveness    valence      tempo kmean.size
## 1  0.2037292 -0.3638954  0.2409096      35993
## 2 -0.3627059  0.6478551 -0.4288994      20217
```

Here are the explanation of the traits, taken from Spotify's API documentation

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.

Danceability: Describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

Energy: Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy.

Instrumentalness: Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal." The closer the instrumentalness

value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

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.

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. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.

Tempo: The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

Valence: A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

We can see that cluster 1 have songs that are mixed louder, have higher energy and liveness, but lower valence

Maybe we can call this cluster "Sad Workout Songs"?

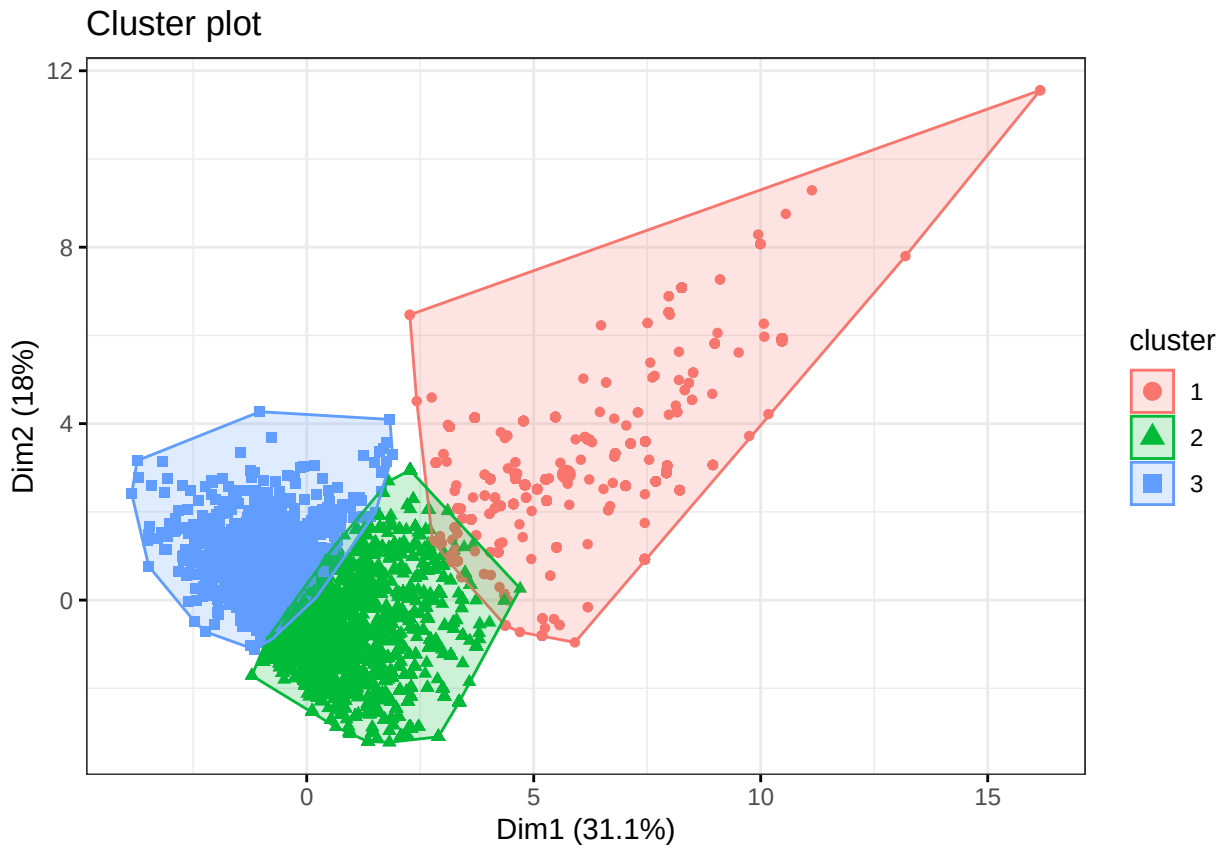
Cluster 2 have songs that are higher in valence, acousticness, danceability, but lower energy(?)

Maybe we can call this cluster "Happy Coffehouse Acoustic Songs To Dance To"???

Let's increase the number of clusters, to see if we get any more clear separation.

```
kmean2 <- kmeans(clusterData,3, nstart=10)
```

```
fviz_cluster(kmean2, data = clusterData, geom = "point", ellipse.type = "convex",
             ggtheme = theme_bw())
```




```
kmeaninfo2 <- data.frame(kmean2$centers, kmean2$size)
kmeaninfo2
```

```
##   danceability     energy    loudness speechiness acousticness instrumentalness
## 1    0.2863125 -3.7874760 -2.91568780  -0.8446513   4.35916164       0.53308803
## 2    0.5825141 -0.1484980 -0.05771523  -0.4862819  -0.06122882      -0.08447799
## 3   -0.6305388  0.4235014  0.26672088   0.5692154  -0.24421672       0.05079510
##     liveness    valence      tempo kmean2.size
## 1 -0.3793901 -0.2876731 -0.5326758        1877
## 2 -0.2612448  0.5737209 -0.3979591       27799
## 3  0.3005374 -0.5807230  0.4546128       26534
```

We can do the same analysis as with the 2 clusters. See if you can try it for yourself!