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

Khoi Trinh

2023-02-20

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

First, we need some data

Data description

The data I chose is a dataset of tracks from Spotify. There are 586,672 observations; each of with has 20 variables. These include 12 numeric variables, and 7 factor variables.

Data source

The data I chose came from this kaggle URL: Spotify Dataset

Misc data processing:

The data included some variables that are binary, but were recorded as numeric. These included "explicit" which is a flag if a song has explicit lyrics, and "mode" which is a flag for major mode (1) or minor mode (0); both of these were converted to factor.

Additionally, there is the "key" factor which has values from 0 to 11; coinciding to the 12 keys in Western music: A, A#, B, C, C#, D, D#, E, F, G, G#. While this is not a binary feature, it should not be considered numeric, and will also be converted to factor.

While most of numeric data ranges from 0 to 1; tempo and length were not, so scaling was done to ensure the data is normalized.

Fortunately, the data does not have any missing values, so no imputations were needed. Clustering and the subsequent analysis can be performed.

The data had almost 600,000 observations, out of those, the top 500 observations will be chosen for the purpose of this experiment.

Time To Create Some Clusters

One last data processing step

Scale the data, and use that to create our clusters.

K means

K means clustering are done here with k = 2. This value of k was suggested using the NbClust() method from the `NbClust` package. One of the lines in the results states "According to the majority rule, the best number of clusters is 2"

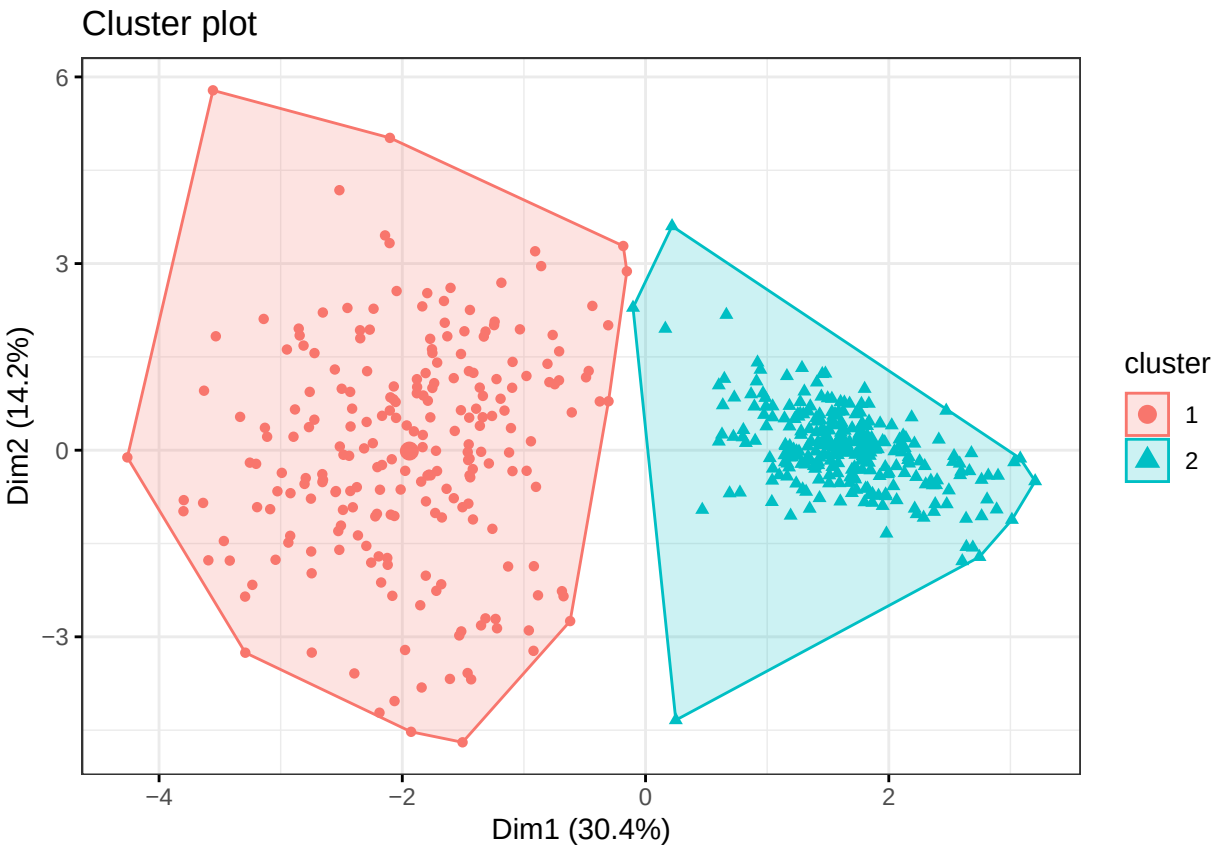
For this clustering method, the kmeans() function from the stats package is used.

Clusters Analysis

First, let's see the size of the clusters

We have 2 clusters, with size 229 and 271.

Visually, the clusters look like so



Cluster Intepretation

```
##   popularity duration_ms danceability      energy   loudness speechiness
## 1  0.4445902   0.7527152   -0.6728745  0.09532898  0.5946238  -1.0524128
## 2 -0.3756869  -0.6360583    0.5685913 -0.08055475 -0.5024681   0.8893083
##   acousticness instrumentalness    liveness    valence       tempo
## 1    0.7661786        0.5725982  0.04803213 -0.3470702  0.10613458
## 2   -0.6474351       -0.4838561 -0.04058804  0.2932807 -0.08968568
##   time_signature kmean.size
## 1      0.2514930        229
## 2     -0.2125162        271
```

From the above data frame, cluster 1 represent songs that have high loudness, so they were mixed louder than average, as well as both high acousticness and instrumentalness; and low in valence. This group can be considered as "Loud Angry Songs".

Cluster 2 have songs that are high in speechiness, danceability, and valence. This group can be considered as "Happy Songs To Dance To".