Spotify_Project

Uday Gadge

2022-12-01

Spotify Data Analysis

Introduction

Spotify is a widely used music player application used by billions of users across the world. The goal of this project is to do an analysis of nine popular genres of music and how it evolved over time.

Data

I extracted the data through the spotify API. I collected 30 most popular artists per genre and their music collection. This gives an access to different genres of music across time.

Getting an overview of the data

Loading the necessary libraries and file

```
head(df)
```

```
## # A tibble: 6 x 41
##    ...1 id           name  popul~1 genre follo~2 artis~3 album~4 album~5 album~6
##   <dbl> <chr>        <chr>   <dbl> <chr>   <dbl> <chr>   <chr>   <chr>   <chr>
## 1     1 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## 2     2 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## 3     3 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## 4     4 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## 5     5 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## 6     6 06HL4z0CvFA~ Tayl~     100 pop    6.27e7 Taylor~ 3lS1y2~ album   2022-1~
## # ... with 31 more variables: album_release_year <dbl>,
## #   album_release_date_precision <chr>, danceability <dbl>, energy <dbl>,
## #   key <dbl>, loudness <dbl>, mode <dbl>, speechiness <dbl>,
## #   acousticness <dbl>, instrumentalness <dbl>, liveness <dbl>, valence <dbl>,
## #   tempo <dbl>, track_id <chr>, analysis_url <chr>, time_signature <dbl>,
## #   disc_number <dbl>, duration_ms <dbl>, explicit <lgl>, track_href <chr>,
## #   is_local <lgl>, track_name <chr>, track_preview_url <chr>, ...
```

```
colnames(df)
```

```
##  [1] "...1"                          "id"
##  [3] "name"                          "popularity"
##  [5] "genre"                         "followers.total"
##  [7] "artist_name"                   "album_id"
##  [9] "album_type"                    "album_release_date"
## [11] "album_release_year"            "album_release_date_precision"
## [13] "danceability"                  "energy"
## [15] "key"                           "loudness"
## [17] "mode"                          "speechiness"
## [19] "acousticness"                  "instrumentalness"
## [21] "liveness"                      "valence"
## [23] "tempo"                         "track_id"
## [25] "analysis_url"                  "time_signature"
## [27] "disc_number"                   "duration_ms"
## [29] "explicit"                      "track_href"
## [31] "is_local"                      "track_name"
## [33] "track_preview_url"             "track_number"
## [35] "type"                          "track_uri"
## [37] "external_urls.spotify"         "album_name"
## [39] "key_name"                      "mode_name"
## [41] "key_mode"
```

selecting the columns needed for the analysis

```
df <- df %>% mutate(artist_id = id) %>%
  select(track_id,artist_id,album_id,genre,track_name,artist_name,album_name,album_release_date, album_
```

```
head(df)
```

```
## # A tibble: 6 x 28
##   track_id artis~1 album~2 genre track~3 artis~4 album~5 album~6 album~7 follo~8
##   <chr>    <chr>   <chr>   <chr> <chr>   <chr>   <chr>   <chr>     <dbl>   <dbl>
## 1 4g2c7No~ 06HL4z~ 3lS1y2~ pop   Lavend~ Taylor~ Midnig~ 2022-1~    2022  6.27e7
## 2 199E1RR~ 06HL4z~ 3lS1y2~ pop   Maroon  Taylor~ Midnig~ 2022-1~    2022  6.27e7
## 3 02Zkkf2~ 06HL4z~ 3lS1y2~ pop   Anti-H~ Taylor~ Midnig~ 2022-1~    2022  6.27e7
## 4 6ADDIJx~ 06HL4z~ 3lS1y2~ pop   Snow O~ Taylor~ Midnig~ 2022-1~    2022  6.27e7
## 5 7gVWKBc~ 06HL4z~ 3lS1y2~ pop   You're~ Taylor~ Midnig~ 2022-1~    2022  6.27e7
## 6 6Nl7Kyv~ 06HL4z~ 3lS1y2~ pop   Midnig~ Taylor~ Midnig~ 2022-1~    2022  6.27e7
## # ... with 18 more variables: track_number <dbl>, popularity <dbl>,
## #   duration_ms <dbl>, danceability <dbl>, energy <dbl>, loudness <dbl>,
## #   key <dbl>, mode <dbl>, speechiness <dbl>, acousticness <dbl>,
## #   instrumentalness <dbl>, liveness <dbl>, valence <dbl>, tempo <dbl>,
## #   time_signature <dbl>, disc_number <dbl>, key_mode <chr>, key_name <chr>,
## #   and abbreviated variable names 1: artist_id, 2: album_id, 3: track_name,
## #   4: artist_name, 5: album_name, 6: album_release_date, ...
```

```
colnames(df)
```

```
##  [1] "track_id"           "artist_id"          "album_id"
##  [4] "genre"              "track_name"         "artist_name"
##  [7] "album_name"         "album_release_date" "album_release_year"
## [10] "followers.total"    "track_number"       "popularity"
```

```
## [13] "duration_ms"        "danceability"       "energy"
## [16] "loudness"           "key"                "mode"
## [19] "speechiness"        "acousticness"       "instrumentalness"
## [22] "liveness"           "valence"            "tempo"
## [25] "time_signature"     "disc_number"        "key_mode"
## [28] "key_name"
```

Let us calculate the average popularity of different genres

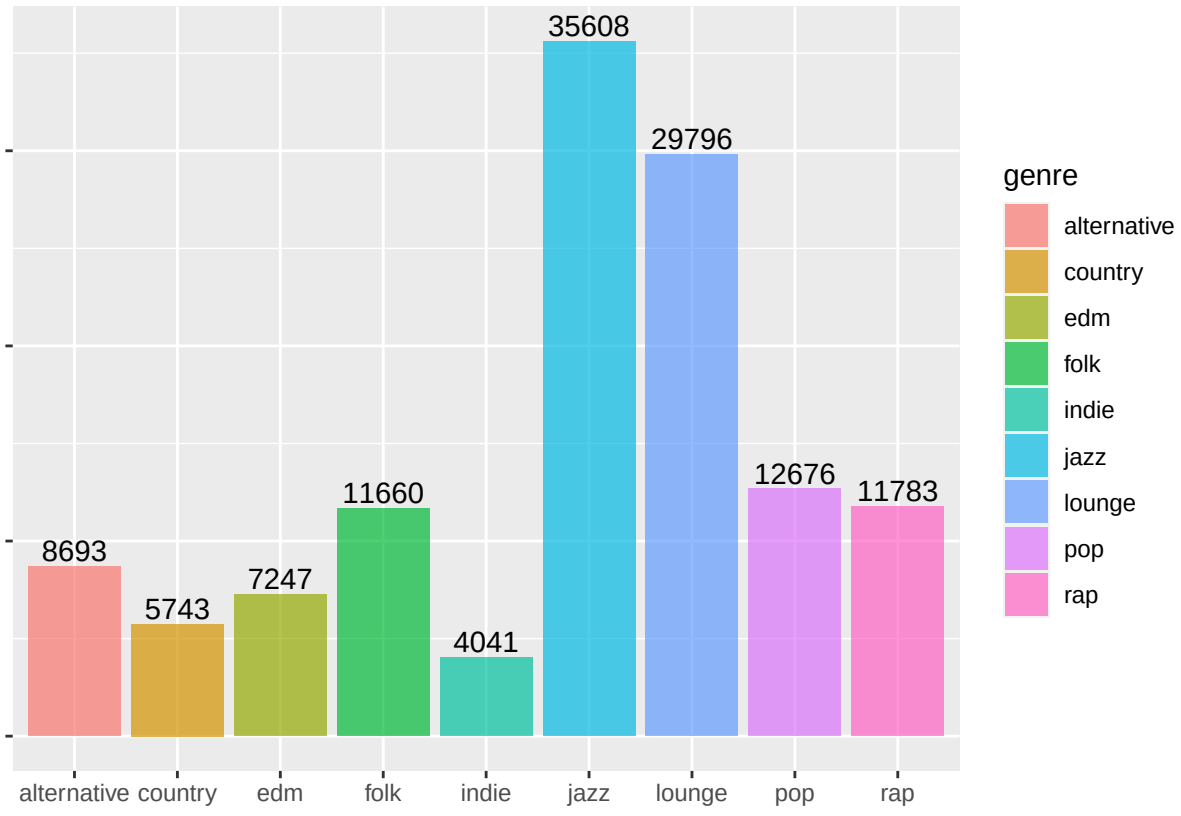
```
df %>% dplyr::group_by(genre) %>% dplyr::summarise(popularity = mean(popularity))
```

```
## # A tibble: 9 x 2
##   genre       popularity
##   <chr>            <dbl>
## 1 alternative       75.8
## 2 country           72.1
## 3 edm               78.0
## 4 folk              69.2
## 5 indie             75.9
## 6 jazz              67.6
## 7 lounge            61.7
## 8 pop               88.7
## 9 rap               86.0
```

```
df <- df %>% dplyr::mutate(decade = round(df$album_release_year/10)*10)
```

Overall distribution of different genres in the data we have.

```
df %>% dplyr::group_by(genre) %>% dplyr::tally() %>%
  ggplot(aes(x = genre,y = n,fill = genre)) +
  geom_bar(stat = 'identity',alpha = 0.7)+
  geom_text(aes(label = n),position=position_dodge(width=0.9), vjust=-0.25)+
  xlab("") + ylab("") + theme(axis.text.y=element_blank())
```



Next, let us check how popularity and the number of songs have evolved with time for the genres we have.

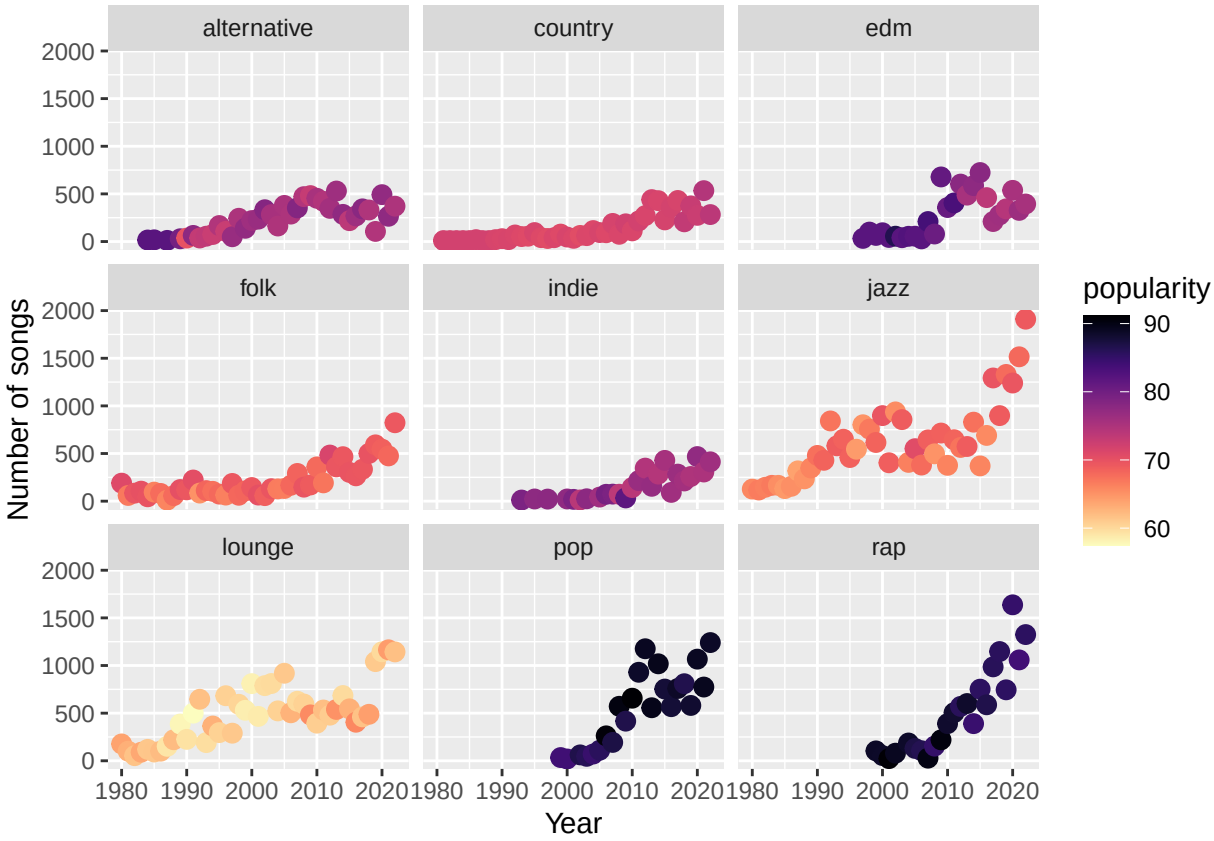
```
library(viridis)
```

```
## Warning: package 'viridis' was built under R version 4.2.2
```

```
## Loading required package: viridisLite
```

```
df %>% dplyr::group_by(genre, album_release_year) %>%
  dplyr::summarize(count = n(), popularity = mean(popularity)) %>%
  filter(album_release_year >= 1980) %>%
  ggplot(aes(album_release_year,count)) +
  geom_point(aes(color = popularity), size = 3)+ xlab("Year") +ylab("Number of songs") +
  scale_color_viridis(option = "A", direction = -1) + facet_wrap(~genre)
```

```
## `summarise()` has grouped output by 'genre'. You can override using the
## `.groups` argument.
```



Clearly pop and rap music are more popular than other genres and it's as expected. Some of the other inferences we can make are:

- lounge music is making a come back as the 2000s music isn't particularly popular compared to the earlier works.
- The same can be said about alternative music.
- EDM music these days is not as popular as the music of 2000s

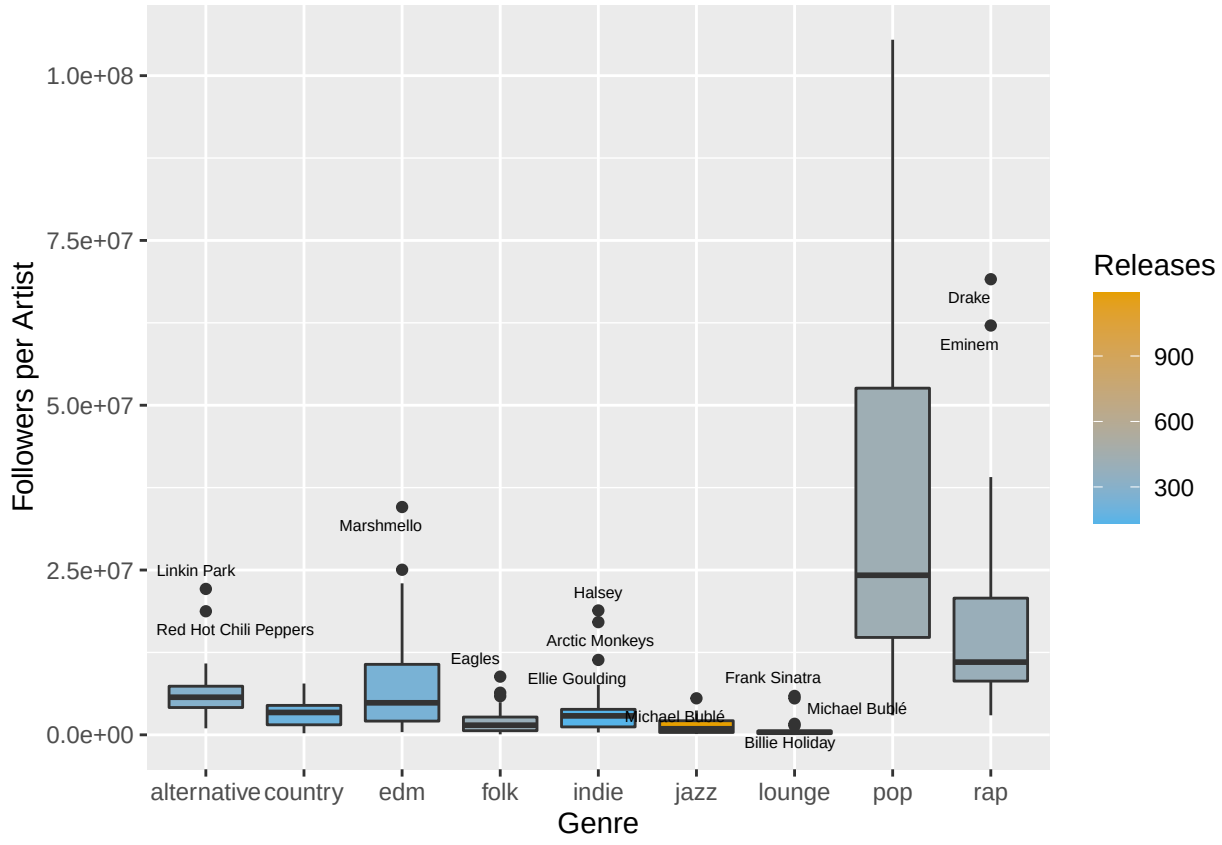
Checking for the followers for genres

```
library(ggrepel)
artists_followers <- df %>% dplyr::group_by(artist_id, artist_name, genre) %>%
  dplyr::summarize(followers = mean(followers.total),songs = dplyr::n()) %>%
  dplyr::group_by(genre) %>%
  dplyr::mutate(Releases = mean(songs),quant75 = quantile(followers, probs = 0.75)) %>%
  dplyr::mutate(outlier = ifelse(followers > 1.6*quant75,"Yes","No"))
```

```
## `summarise()` has grouped output by 'artist_id', 'artist_name'. You can
## override using the `.groups` argument.
```

```
outliers <- artists_followers %>% filter(followers > 2.5*quant75)

p <- artists_followers %>%
  dplyr::mutate(genre = reorder(genre, followers, FUN = median)) %>%
  ggplot(aes(genre, followers, fill = Releases)) + geom_boxplot() +
```

```
  geom_text_repel(data = outliers, aes(genre,followers,label=artist_name),size = 2)+
  xlab("Genre") +
  ylab("Followers per Artist")

p + scale_fill_gradient(low = "#56B4E9",high ="#E69F00")
```



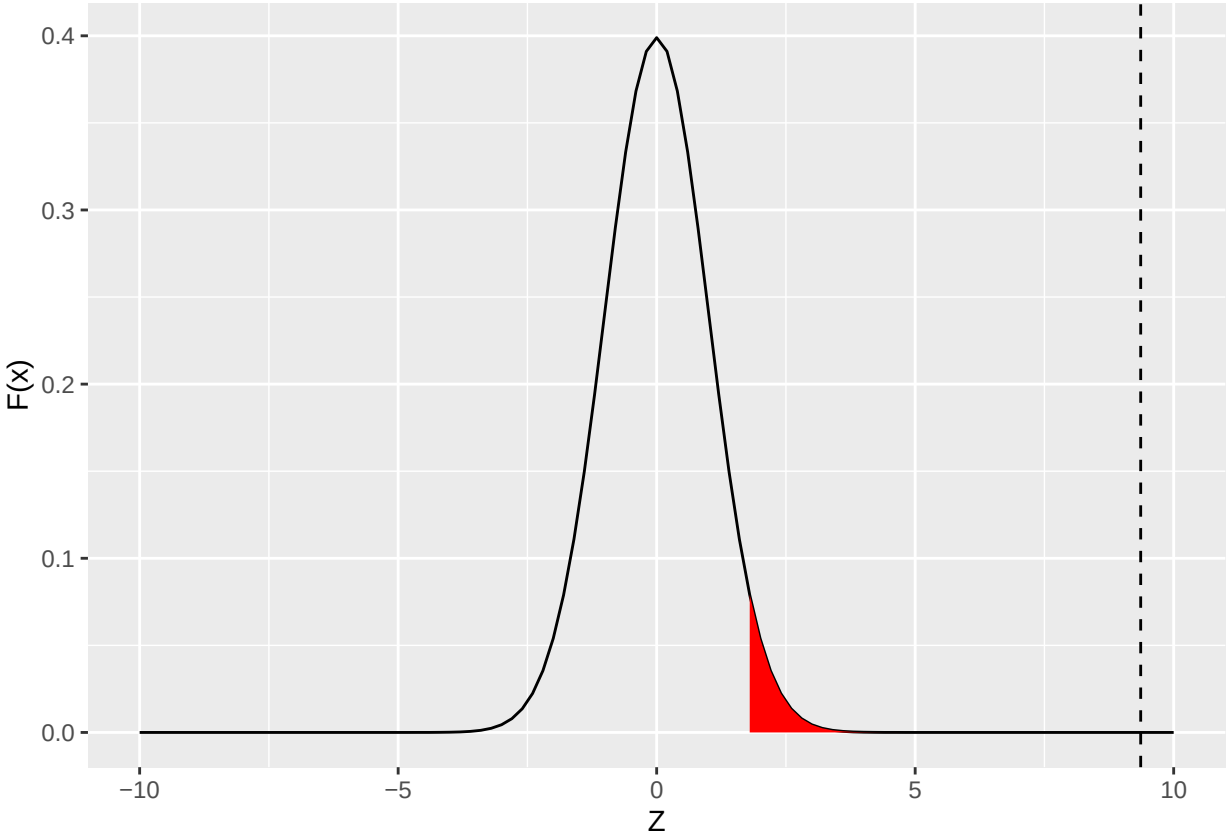
- Clearly Pop music artists have more followers.
- Drake and Eminem have considerably more followers than other rap artists.
- Jazz musicians release way more albums than other musicians.

Hypothesis testing

Using Central limit theorem, we can perform a z- test to check if pop music artists have significantly higher followers than the rest.

```
mu <- mean(artists_followers$followers)
sd <- sd(artists_followers$followers)
x_bar <- mean((artists_followers %>% filter(genre == "pop"))$followers)

n <- dim(artists_followers %>% filter(genre == "pop"))[1]

z <- (x_bar - mu)/(sd/(sqrt(n)))

cri <- qnorm(0.95)
```

```
rejection_z_test <- function(x){
  y <- dnorm(x,0,1)
  y[x<cri] <- NA
  y
}

ggplot(data.frame(x = c(-10, 10)), aes(x)) +
  stat_function(fun = dnorm,geom = "line")+
  stat_function(fun = function(x){rejection_z_test(x)}, geom = "area", fill = "red")+
  geom_vline(xintercept = z, linetype = 'dashed') + xlab('Z')+ylab('F(x)')
```



Our Z value is significantly higher than the critical value of Z(alpha = 0.05). So we reject the null hypothesis and accept alternative hypothesis that pop artists have significantly higher followers than other genre artists.

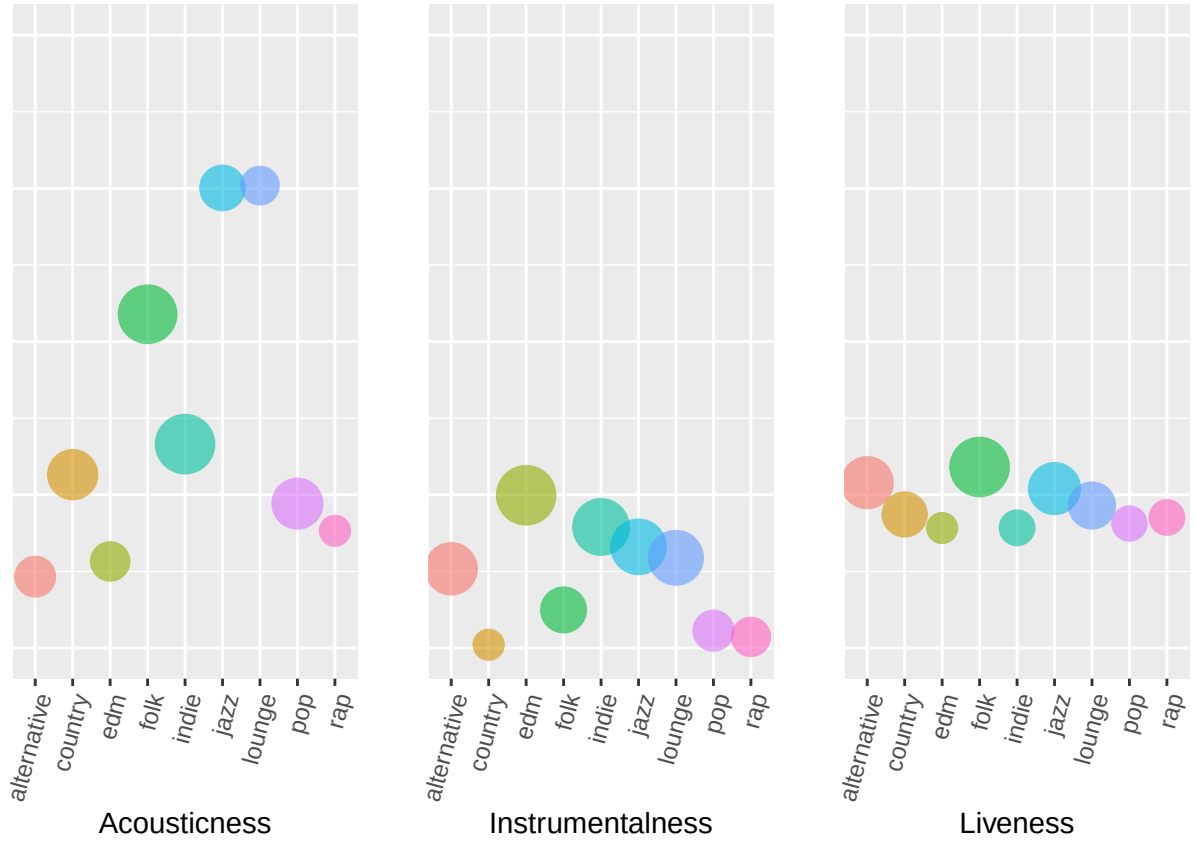
```
head(artists_followers)
```

```
## # A tibble: 6 x 8
## # Groups:   genre [5]
##   artist_id              artist_name genre follo~1 songs Relea~2 quant75 outlier
##   <chr>                  <chr>       <chr>   <dbl> <int>   <dbl>   <dbl> <chr>
## 1 00FQb4jTyendYWaN8pK0wa Lana Del R~ pop    1.89e7   593    423.  5.26e7 No
## 2 02kJSzxNuaWGqwubyUba0Z G-Eazy      indie  5.11e6   271    135.  3.87e6 No
## 3 05fG473iIaoy82BF1aGhL8 Slipknot    alte~  8.89e6   347    290.  7.38e6 No
## 4 05YVYeV4HxYp5rrWalvuE1 Grover Was~ jazz   2.76e5   535   1187.  2.14e6 No
## 5 06HL4z0CvFAxyc27GXpf02 Taylor Swi~ pop    6.27e7  1265    423.  5.26e7 No
```

```
## 6 07YZf4WDAMNwqr4jfgOZ8y Jason Deru~ edm    1.16e7   294    242.  1.07e7 No
## # ... with abbreviated variable names 1: followers, 2: Releases
```

Acousicness, Instrumentalness and liveness can define the nature of a song. We can visualize it for genre by looking at the mean value and variance of these features.

```
nature_columns <- c("acousticness","instrumentalness","liveness")

p1 <- df %>% dplyr::group_by(genre) %>%
  dplyr::summarise(mu = mean(na.omit(acousticness)), variance = var(na.omit(acousticness))) %>%
  ggplot(aes(genre,mu)) +
  geom_point(aes(color = genre, size = variance), alpha = 0.6, show.legend = FALSE)+
  scale_size_continuous(range = c(5,10))+ylim(c(0,1)) + xlab("Acousticness")+ ylab("") +
  theme(axis.text.y=element_blank(),axis.ticks.y=element_blank(),axis.text.x=element_text(angle=75, hju

p2 <- df  %>% dplyr::group_by(genre) %>%
  dplyr::summarise(mu = mean(na.omit(instrumentalness)), variance = var(na.omit(instrumentalness))) %>%
  ggplot(aes(genre,mu)) +
  geom_point(aes(color = genre, size = variance), alpha = 0.6, show.legend = FALSE)+
  scale_size_continuous(range = c(5,10))+ylim(c(0,1)) + xlab("Instrumentalness")+ylab("") +
  theme(axis.text.y=element_blank(),axis.ticks.y=element_blank(),axis.text.x=element_text(angle=75, hju

p3 <- df %>% dplyr::group_by(genre) %>%
  dplyr::summarise(mu = mean(na.omit(liveness)), variance = var(na.omit(liveness))) %>%
  ggplot(aes(genre,mu)) +
  geom_point(aes(color = genre, size = variance), alpha = 0.6, show.legend = FALSE)+
  scale_size_continuous(range = c(5,10))+ylim(c(0,1)) + xlab("Liveness")+ylab("") +
  theme(axis.text.y=element_blank(),axis.ticks.y=element_blank(),axis.text.x=element_text(angle=75, hju


grid.arrange(p1,p2,p3, ncol = 3)
```



- Country music has very less Instrumentality. It doesn't have a lot of variance either. - EDM doesn't have a lot of liveness. - Jazz and Lounge are way more acoustic than other genres.

We can take a further look into the other features we have to get more insight into these genres and also how time has affected them.

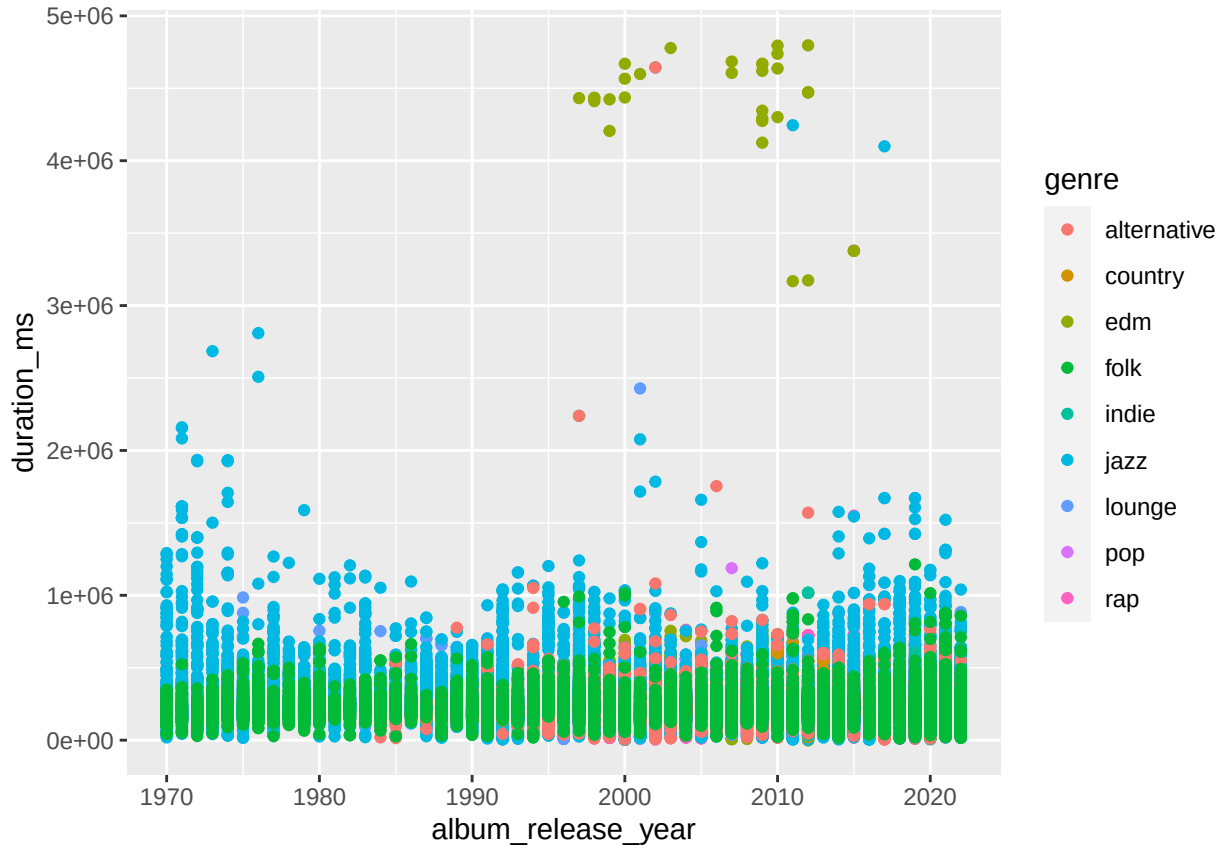
```
colnames(df)
```

```
##  [1] "track_id"           "artist_id"          "album_id"
##  [4] "genre"              "track_name"         "artist_name"
##  [7] "album_name"         "album_release_date" "album_release_year"
## [10] "followers.total"    "track_number"       "popularity"
## [13] "duration_ms"        "danceability"       "energy"
## [16] "loudness"           "key"                "mode"
## [19] "speechiness"        "acousticness"       "instrumentalness"
## [22] "liveness"           "valence"            "tempo"
## [25] "time_signature"     "disc_number"        "key_mode"
## [28] "key_name"           "decade"
```

```
further_analysis <- c("energy","danceability","key","duration_ms","loudness","mode", "valence","tempo",
```

```
df %>% dplyr::filter(album_release_year >= 1970) %>%
  ggplot(aes(album_release_year,duration_ms,color = genre)) + geom_point()
```

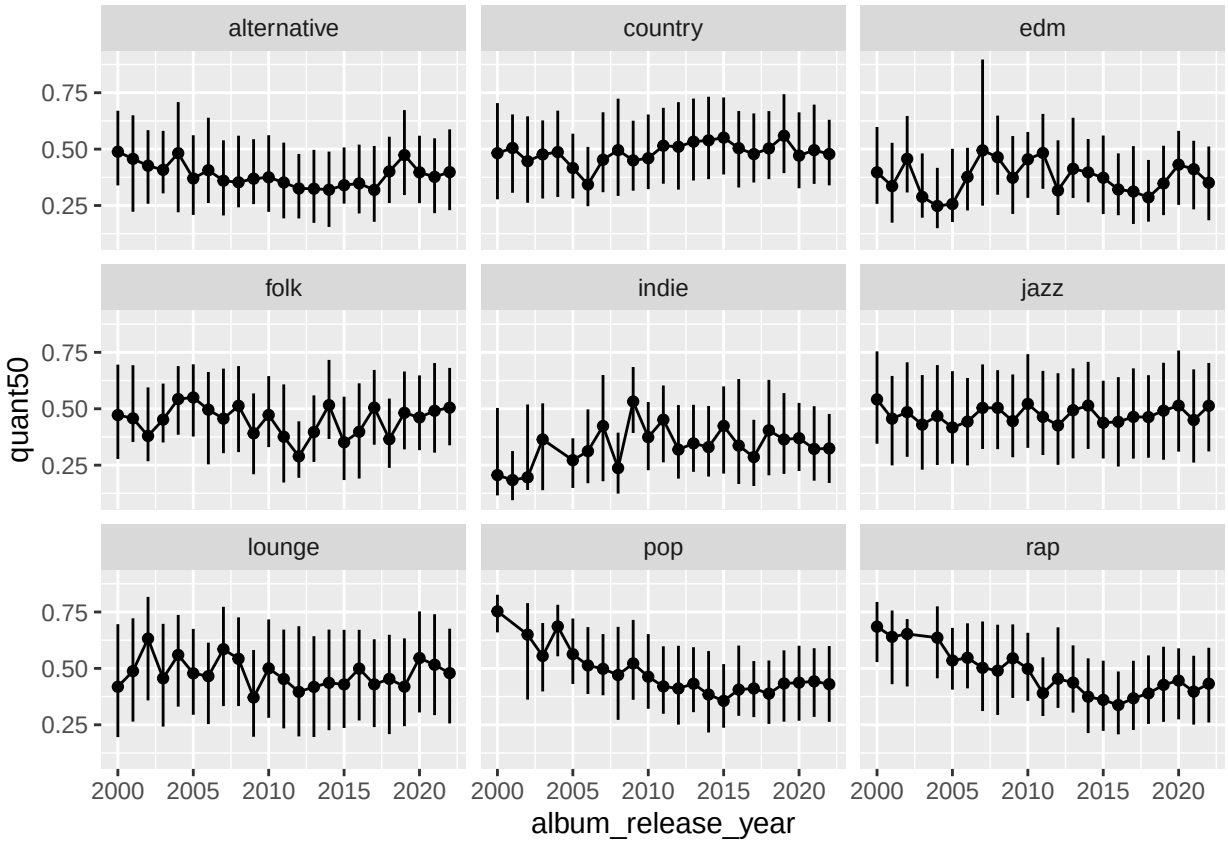
```
## Warning: Removed 1 rows containing missing values (geom_point).
```




```
valence <- df %>% dplyr::filter(album_release_year >= 2000) %>%
  dplyr::group_by(genre,album_release_year) %>%
  dplyr::summarize(quant25 = quantile(valence, probs = 0.25,na.rm = TRUE),
            quant50 = quantile(valence, probs = 0.50,na.rm = TRUE),
            quant75 = quantile(valence, probs = 0.75,na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'genre'. You can override using the
## `.groups` argument.
```

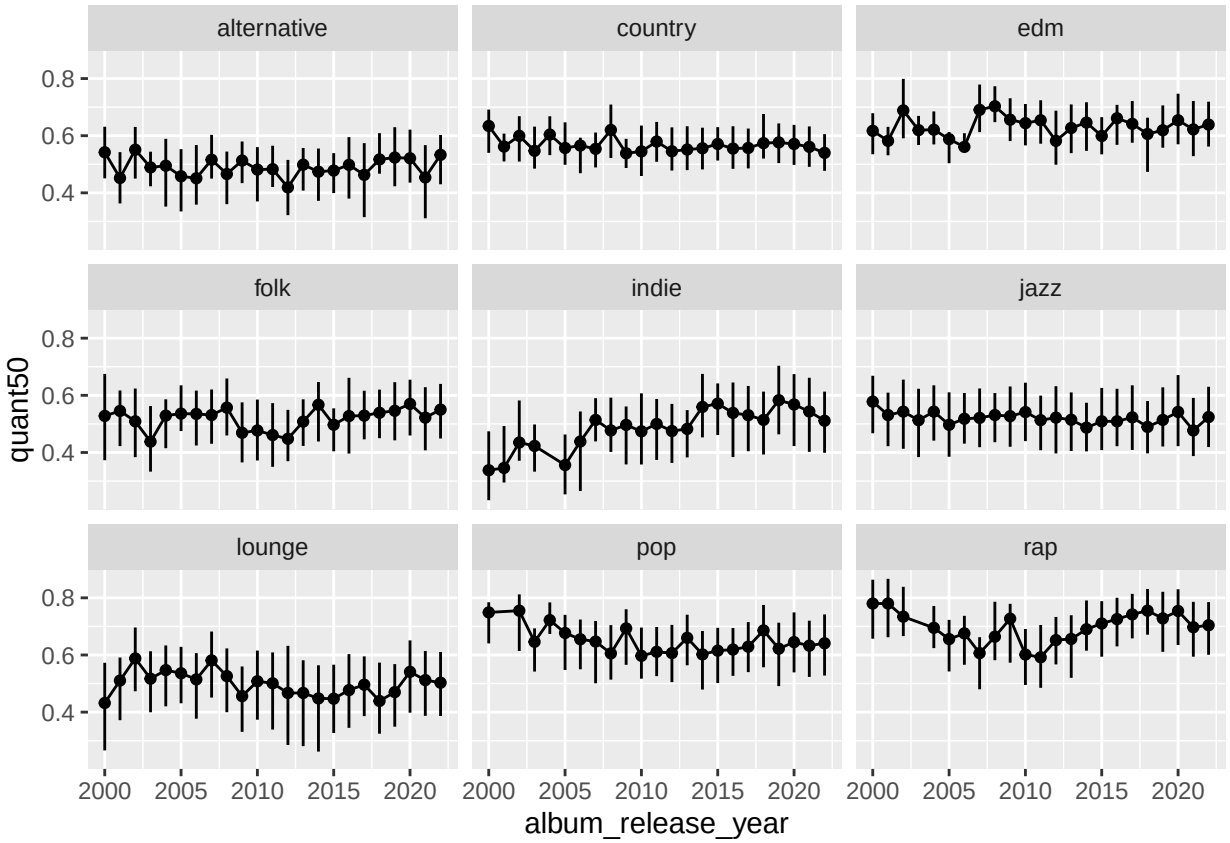
```
 valence %>% ggplot(aes(album_release_year,quant50)) + geom_segment(aes(x = album_release_year,
                                          xend = album_release_year,
                                          y = quant25,
                                          yend = quant75)) +
   geom_point()+geom_line()+facet_wrap(~genre)
```




```
danceability <- df %>% dplyr::filter(album_release_year >= 2000) %>%
  dplyr::group_by(genre,album_release_year) %>%
  dplyr::summarize(quant25 = quantile(danceability, probs = 0.25,na.rm = TRUE),
            quant50 = quantile(danceability, probs = 0.50,na.rm = TRUE),
            quant75 = quantile(danceability, probs = 0.75,na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'genre'. You can override using the
## `.groups` argument.
```

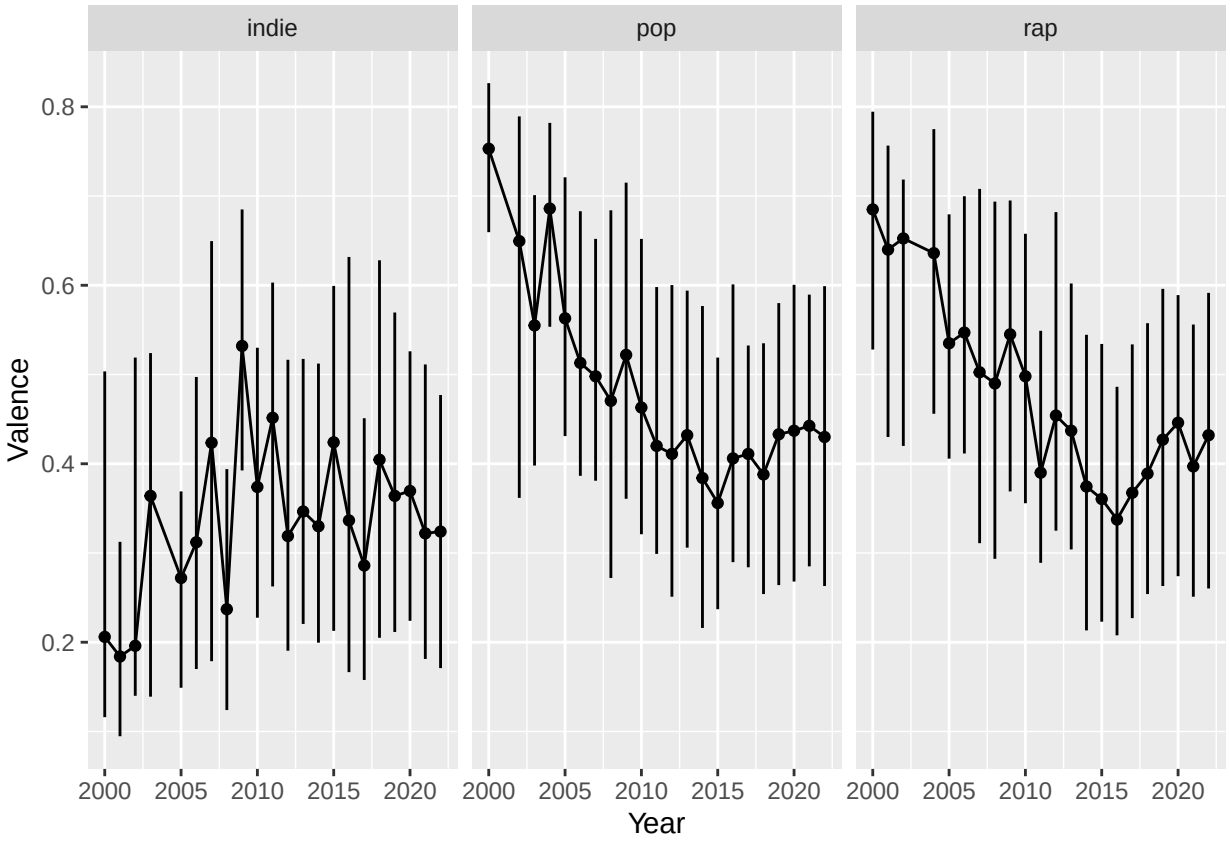
```
danceability %>% ggplot(aes(album_release_year,quant50)) + geom_segment(aes(x = album_release_year,
                                          xend = album_release_year,
                                          y = quant25,
                                          yend = quant75)) +
   geom_point()+geom_line()+facet_wrap(~genre)
```




```
valence <- df %>%
  dplyr::filter(album_release_year >= 2000 & genre %in% c("indie","pop","rap")) %>%
  dplyr::group_by(genre,album_release_year) %>%
  dplyr::summarize(quant25 = quantile(valence, probs = 0.25,na.rm = TRUE),
            quant50 = quantile(valence, probs = 0.50,na.rm = TRUE),
            quant75 = quantile(valence, probs = 0.75,na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'genre'. You can override using the
## `.groups` argument.
```

```
 valence %>% ggplot(aes(album_release_year,quant50)) + geom_segment(aes(x = album_release_year,
                                          xend = album_release_year,
                                          y = quant25,
                                          yend = quant75)) +
   xlab("Year")+ylab("Valence")+
   geom_point()+geom_line()+facet_wrap(~genre,ncol = 3)
```

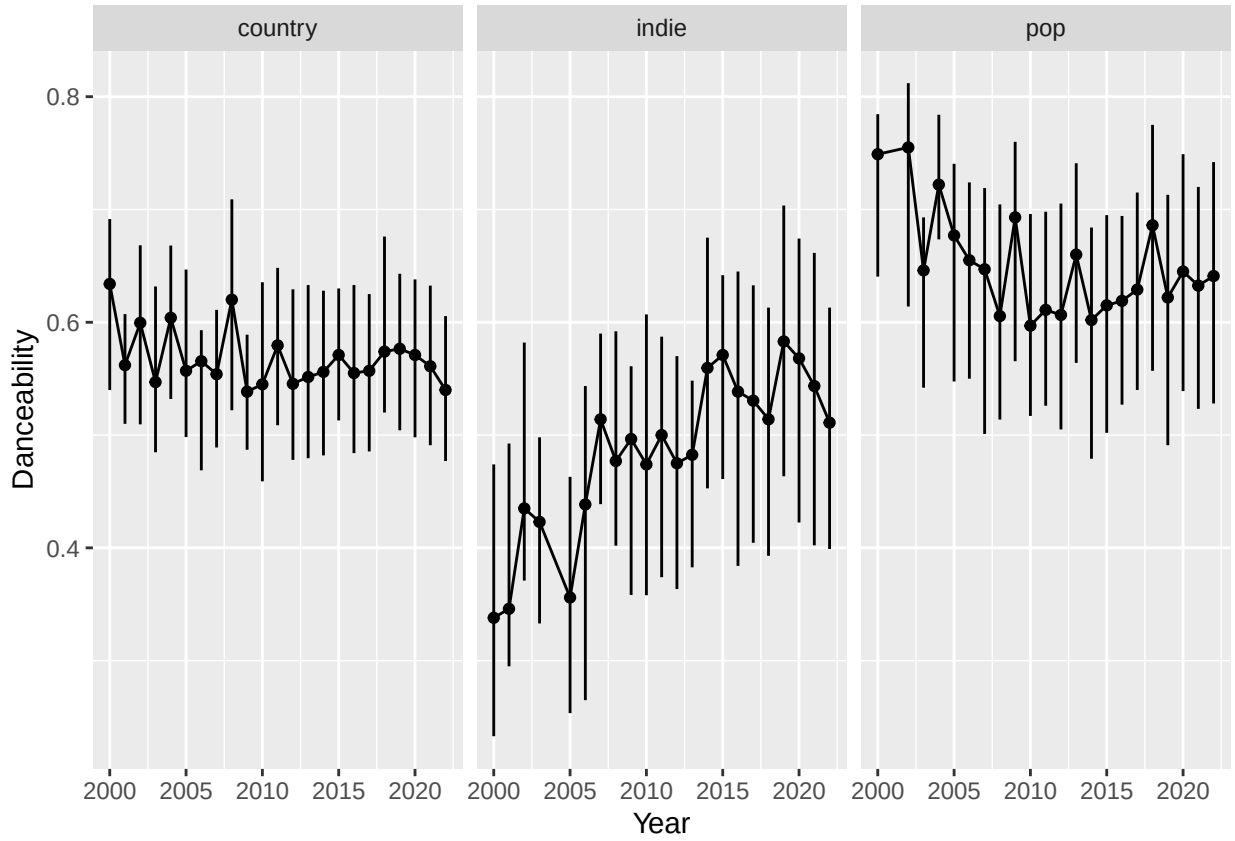


Valence is a measure of positivity in a song and there is a clear downward trend for pop and rap music while indie is becoming more positive.

```
danceability <- df %>%
  dplyr::filter(album_release_year >= 2000 & genre %in% c("indie","pop","country")) %>%
  dplyr::group_by(genre,album_release_year) %>%
  dplyr::summarize(quant25 = quantile(danceability, probs = 0.25,na.rm = TRUE),
            quant50 = quantile(danceability, probs = 0.50,na.rm = TRUE),
            quant75 = quantile(danceability, probs = 0.75,na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'genre'. You can override using the
## `.groups` argument.
```

```
danceability %>% ggplot(aes(album_release_year,quant50)) + geom_segment(aes(x = album_release_year,
                                        xend = album_release_year,
                                        y = quant25,
                                        yend = quant75)) +
  xlab("Year")+ylab("Danceability")+
  geom_point()+geom_line()+facet_wrap(~genre)
```



Pop music was way more danceable in the 2000s when pop music was dominated by beyonce, britney spears where as now it's dominated by taylor swift, ed sheeran etc.,

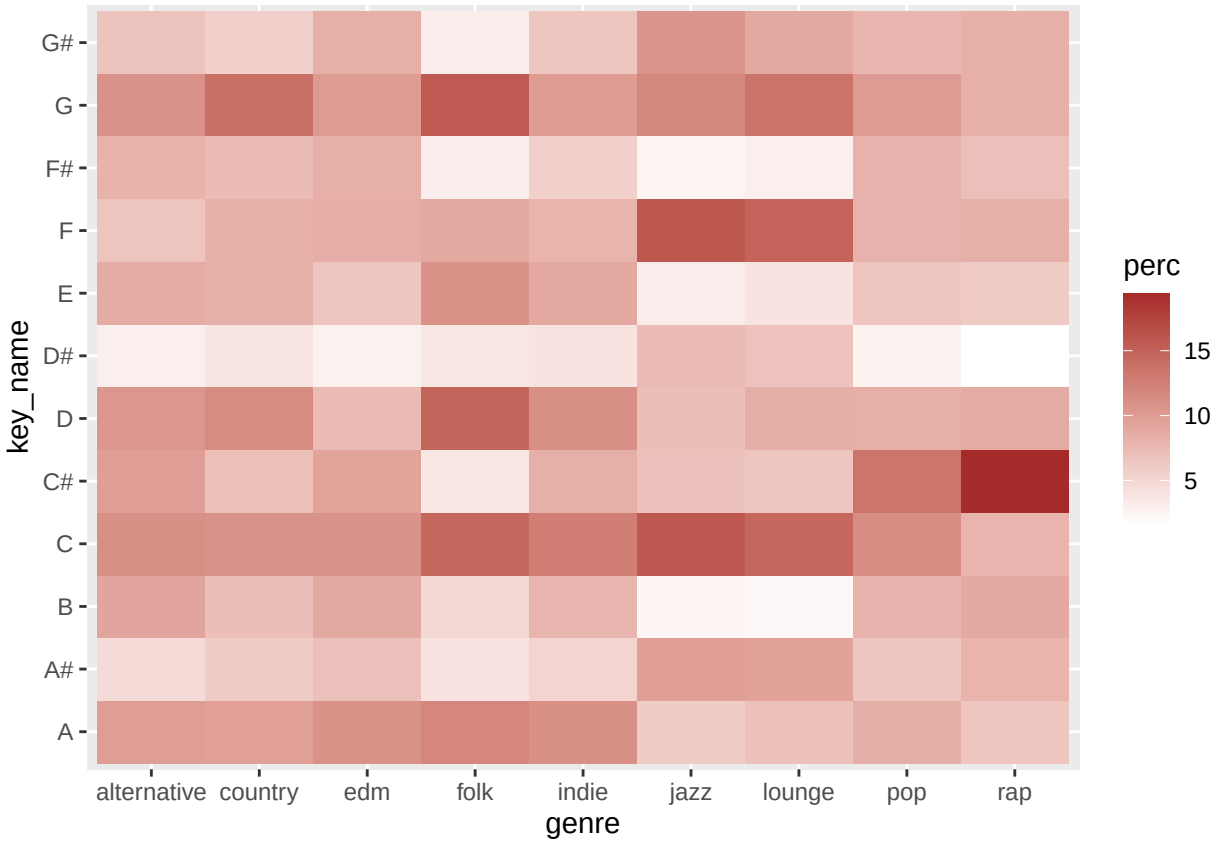
Indie music became more danceable during 2015 due to tik-tok.

```
unique(df %>% select(key,key_mode,key_name,mode))
```

```
## # A tibble: 25 x 4
##      key key_mode key_name  mode
##    <dbl> <chr>    <chr>    <dbl>
##  1    10 A# major A#           1
##  2     7 G major  G            1
##  3     4 E major  E            1
##  4     9 A major  A            1
##  5     2 D major  D            1
##  6     0 C major  C            1
##  7     4 E minor  E            0
##  8     8 G# major G#           1
##  9     5 F major  F            1
## 10     6 F# major F#           1
## # ... with 15 more rows
```

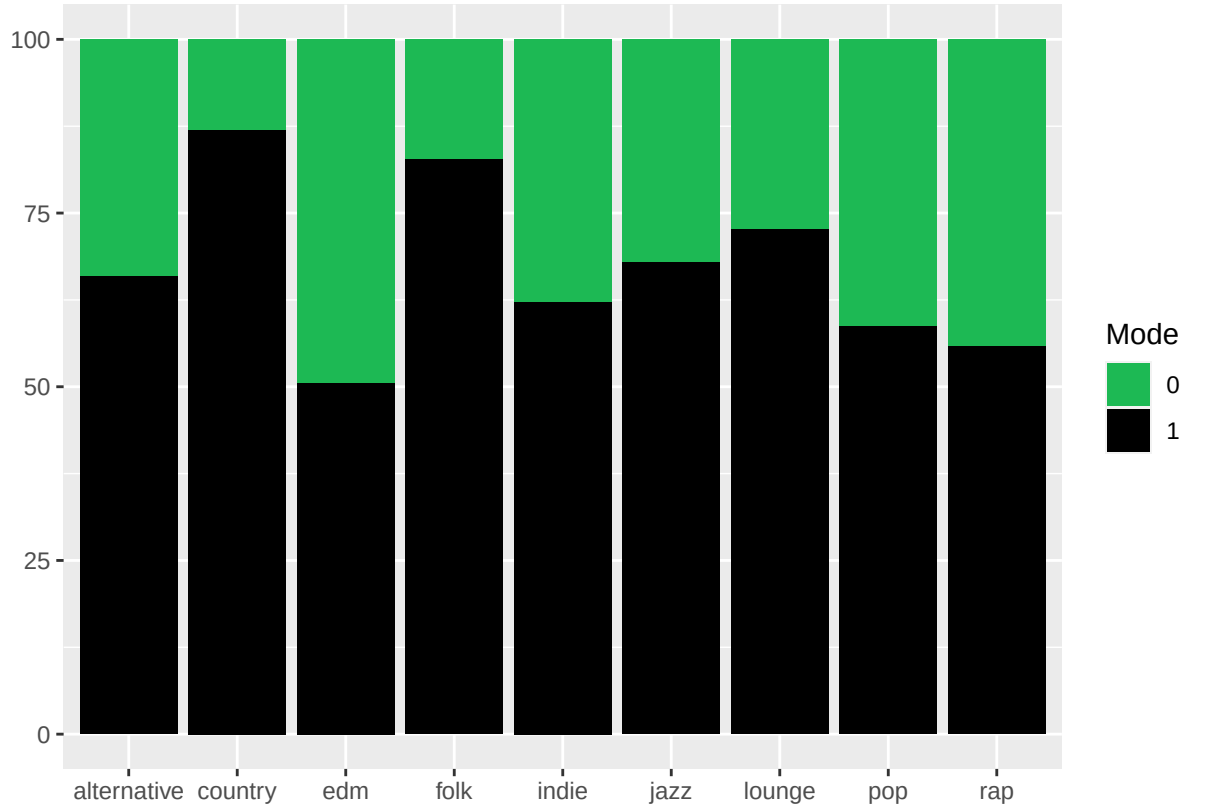
Key in which the music is played plays an important role. This can perhaps give us an insight into what genres are similar in the key they use and if any genre is very particular.

```
df %>% filter(key_mode != "NA NA") %>% dplyr::group_by(genre,key_name) %>%
  dplyr::tally() %>% dplyr::group_by(genre) %>% dplyr::mutate(perc = (n/sum(n))*100) %>%
  ggplot(aes(genre,key_name,fill=perc)) + geom_raster() +
  scale_fill_gradient(low = "white",high = "brown")
```



- rap music is mostly played at C# key - Jazz and longue are similar to each other.

```
df %>% dplyr::filter(key_mode != "NA NA") %>%
  dplyr::group_by(genre,mode) %>% dplyr::tally() %>%
  dplyr::group_by(genre) %>% dplyr::mutate(Mode = as.factor(mode),perc = (n/sum(n))*100) %>%
  ggplot(aes(genre,perc,fill=Mode)) + geom_bar(stat = 'identity')+
  scale_fill_manual(values=c("#1DB954",
                             "Black"))+
  xlab("")+ylab("")
```




```
colnames(df)
```

```
##  [1] "track_id"           "artist_id"          "album_id"
##  [4] "genre"              "track_name"         "artist_name"
##  [7] "album_name"         "album_release_date" "album_release_year"
## [10] "followers.total"    "track_number"       "popularity"
## [13] "duration_ms"        "danceability"       "energy"
## [16] "loudness"           "key"                "mode"
## [19] "speechiness"        "acousticness"       "instrumentalness"
## [22] "liveness"           "valence"            "tempo"
## [25] "time_signature"     "disc_number"        "key_mode"
## [28] "key_name"           "decade"
```

Although these features can help us understand music genres to a certain extent, we can look for a way where we can use distance metrics.

Distances

```
song_features_num <- c("danceability","energy","loudness","acousticness","instrumentalness",
                       "liveness","valence","tempo")

song_features_cat <- c("key_name","key_mode","mode")
```

```
identifiers <- c("track_id","track_name","genre")

data <- df %>% select(any_of(c(identifiers,song_features_num,song_features_cat)))
```

It is important to scale the numerical data in any distance metrics.

```
data <- data %>% dplyr::group_by(genre) %>% sample_n(1000)

for (col in song_features_num){
  data[,col] = (data[,col] - min(data[,col]))/(max(data[,col])-min(data[,col]))
}
```

```
modes <- data %>%
  dplyr::group_by(mode) %>%
  dplyr::summarize(mode_prob = (n())/(dim(data)[1]))
keys <- data %>%
  dplyr::group_by(key_name) %>%
  dplyr::summarize(key_name_prob = (n())/(dim(data)[1]))
key_modes <- data %>%
  dplyr::group_by(key_mode) %>%
  dplyr::summarize(key_mode_prob = (n())/(dim(data)[1]))
modes
```

```
## # A tibble: 2 x 2
##    mode mode_prob
##   <dbl>     <dbl>
## 1     0     0.326
## 2     1     0.674
```

```
data <- left_join(data,modes,by="mode")
data <- left_join(data,keys,by="key_name")
data <- left_join(data,key_modes,by="key_mode")
```

```
class(data)
```

```
## [1] "grouped_df" "tbl_df"     "tbl"        "data.frame"
```

```
class(df)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
song_features_cat <- c("key_name_prob", "key_mode_prob", "mode_prob")
```

```
a <- data[,c(song_features_num,song_features_cat)]
class(a)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Euclidean distance:

$$dis = \sqrt{\sum (x_i - y_i)^2}$$

This doesn't include the categorical features, so we can update it with the distance:

$$dis = \sqrt{\sum_{num} (x_i - y_i)^2 + \sum_{cat} (\delta_{ij} P_i P_j + (1 - \delta_{ij})(1 - P_i P_j))}$$

```
distance_prob <- function(v1,v2){
  sqrt(sum((v1[1:8]-v2[1:8])^2) +
         sum((v1[9:11] == v2[9:11])*v1[9:11]*v2[9:11])+
         sum((v1[9:11] != v2[9:11])*(1-v1[9:11]*v2[9:11])))
}
```

We can use this distance to calculate the mean of distances within a genre and average distances between the combinations of two clusters.

```
genre_df1 <- data.frame(genre1 = unique(data$genre))
genre_df2 <- data.frame(genre2 = unique(data$genre))
genre_df <- crossing(genre_df1,genre_df2)
```

```
genre_df$distance <- NA
genre_df
```

```
## # A tibble: 81 x 3
##    genre1      genre2      distance
##    <chr>       <chr>       <lgl>
##  1 alternative alternative NA
##  2 alternative country     NA
##  3 alternative edm         NA
##  4 alternative folk        NA
##  5 alternative indie       NA
##  6 alternative jazz        NA
##  7 alternative lounge      NA
##  8 alternative pop         NA
##  9 alternative rap         NA
## 10 country     alternative NA
## # ... with 71 more rows
```

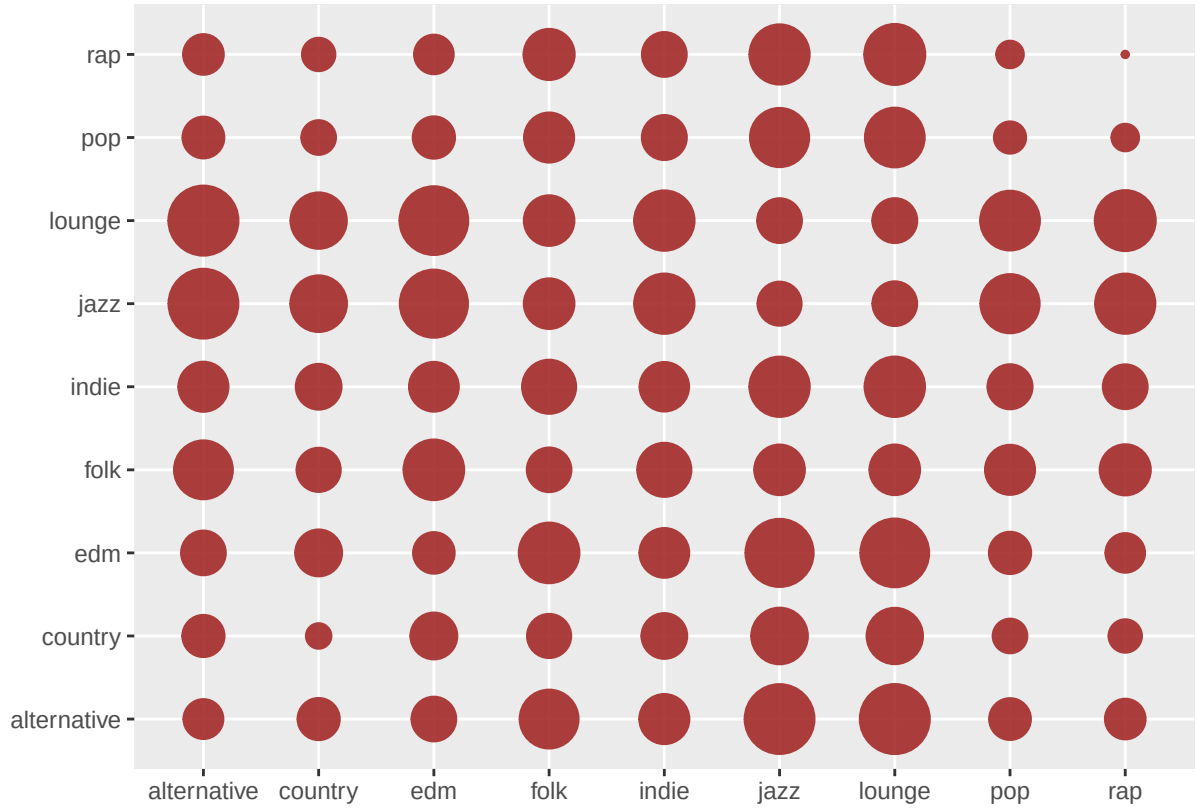
```
for (i in 1:dim(genre_df)[1]){
  #print(i)
  genre_1 <- genre_df$genre1[[i]]
  genre_2 <- genre_df$genre2[[i]]
  a <- data[data$genre == genre_1,c(song_features_num,song_features_cat)]
  b <- data[data$genre == genre_2,c(song_features_num,song_features_cat)]
  #print(genre_1)
  #print(genre_2)
  mat <- proxy::dist(a,b,method = distance_prob)
  if (genre_1 == genre_2){
    val <- mean(mat[upper.tri(mat)])
  }
```

```
  else{
    val <- mean(mat)
  }
  genre_df$distance[[i]] = val
}
```

```
genre_df
```

```
## # A tibble: 81 x 3
##    genre1      genre2      distance
##    <chr>       <chr>          <dbl>
##  1 alternative alternative     1.72
##  2 alternative country         1.73
##  3 alternative edm             1.75
##  4 alternative folk            1.83
##  5 alternative indie           1.78
##  6 alternative jazz            1.92
##  7 alternative lounge          1.92
##  8 alternative pop             1.73
##  9 alternative rap             1.73
## 10 country     alternative     1.73
## # ... with 71 more rows
```

```
genre_df %>% ggplot(aes(genre1,genre2,size=distance)) +
  geom_point(color = 'brown',alpha = 0.9)+
  scale_size_continuous(range = c(1,12)) + xlab("")+ylab("")+
  theme(legend.position = "none")
```



- rap and country music are very close within the genre. - Jazz and lounge music has similar patterns and are very different from EDM and Alternative which are similar as well.

Principal Component Analysis (PCA)

PCA is a dimensionality reduction which creates linear combinations of features that explain maximum variance in a dataset. However, it doesn't incorporate categorical features.

```
pc <- prcomp(data[,song_features_num],center = TRUE)
```

```
attributes(pc)
```

```
## $names
## [1] "sdev"     "rotation" "center"   "scale"    "x"
##
## $class
## [1] "prcomp"
```

```
pc$center
```

```
##     danceability           energy         loudness     acousticness
##        0.5670202        0.5640173        0.8546884        0.3707150
## instrumentalness         liveness          valence            tempo
##        0.1111840        0.2303781        0.4590175        0.5506435
```

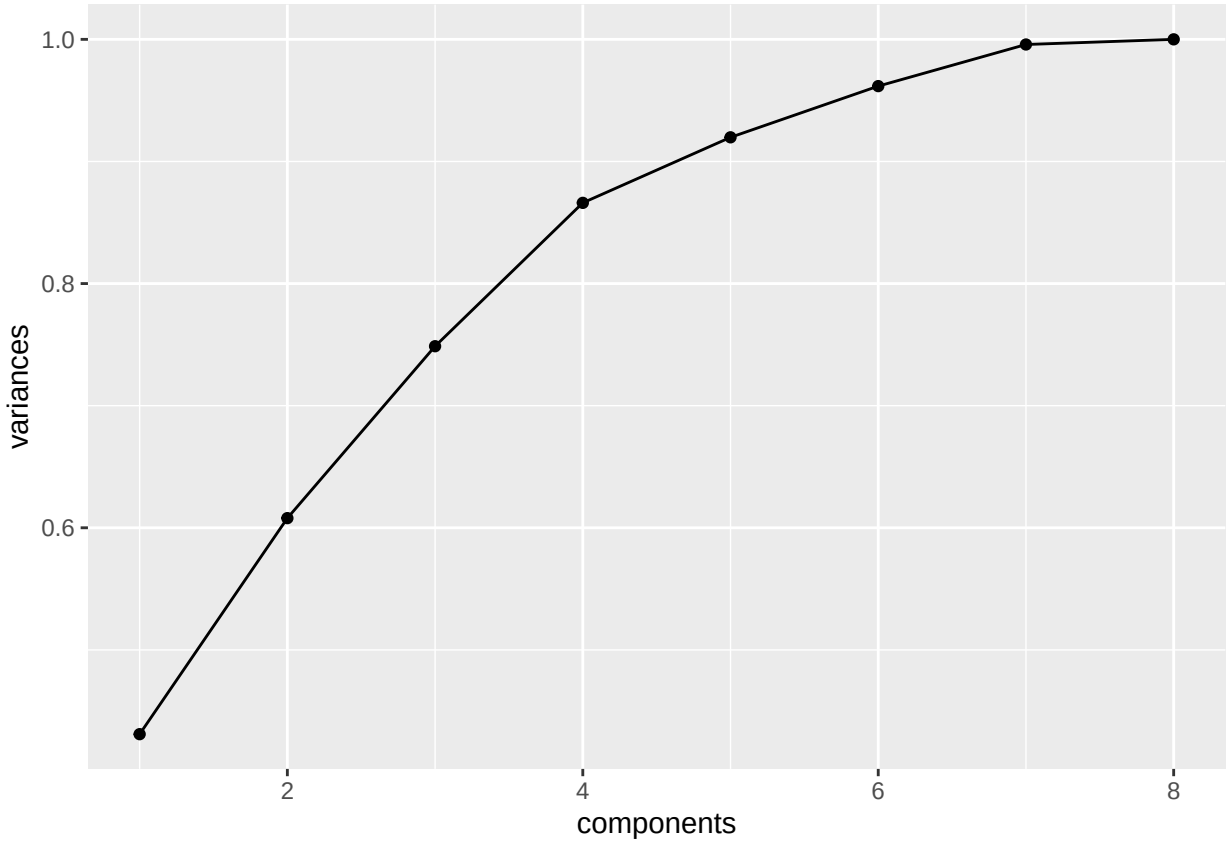
```
print(pc)
```

```
## Standard deviations (1, .., p=8):
## [1] 0.4254349 0.2699501 0.2415361 0.2194750 0.1478247 0.1309194 0.1192295
## [8] 0.0420898
##
## Rotation (n x k) = (8 x 8):
##                           PC1         PC2         PC3          PC4         PC5
## danceability      0.11782722 -0.24834477 -0.25981893 -0.217160884 -0.59659670
## energy            0.53928914  0.04061463 -0.08852978  0.212948472  0.45150766
## loudness          0.13826430 -0.01998253  0.01511262  0.004235549  0.11645938
## acousticness     -0.80669676 -0.16906407 -0.12258556  0.137470235  0.29989455
## instrumentalness -0.08420102  0.75356025 -0.64391382  0.051935744 -0.07121994
## liveness          0.01323177 -0.01832429  0.09274779  0.935069589 -0.30571091
## valence           0.10584473 -0.58256843 -0.69517856  0.104704686  0.10335428
## tempo             0.08330244 -0.01088835 -0.05385734  0.023718442  0.47696974
##                           PC6          PC7         PC8
## danceability      0.24678747 -0.625067563  0.04728380
## energy           -0.35791913 -0.504164432  0.26112459
## loudness         -0.08191827 -0.189958223 -0.96118668
## acousticness     -0.13820395 -0.421175707  0.01750273
## instrumentalness  0.01630154  0.005069589 -0.04869389
## liveness          0.14629718 -0.007135908 -0.04023562
## valence          -0.08700494  0.368277524 -0.03597659
## tempo             0.86944835 -0.077011086  0.01037743
```

```
summary(pc)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     0.4254 0.2700 0.2415 0.2195 0.14782 0.13092 0.11923
## Proportion of Variance 0.4358 0.1754 0.1405 0.1160 0.05261 0.04127 0.03423
## Cumulative Proportion  0.4358 0.6112 0.7517 0.8676 0.92024 0.96151 0.99573
##                           PC8
## Standard deviation     0.04209
## Proportion of Variance 0.00427
## Cumulative Proportion  1.00000
```

```
components <- seq(1,8)
variances <- c(0.4310, 0.6079, 0.7487, 0.8661, 0.91977, 0.96170, 0.99578, 1.00000)
data.frame(components,variances) %>% ggplot(aes(components,variances)) + geom_point()+
  geom_line()
```

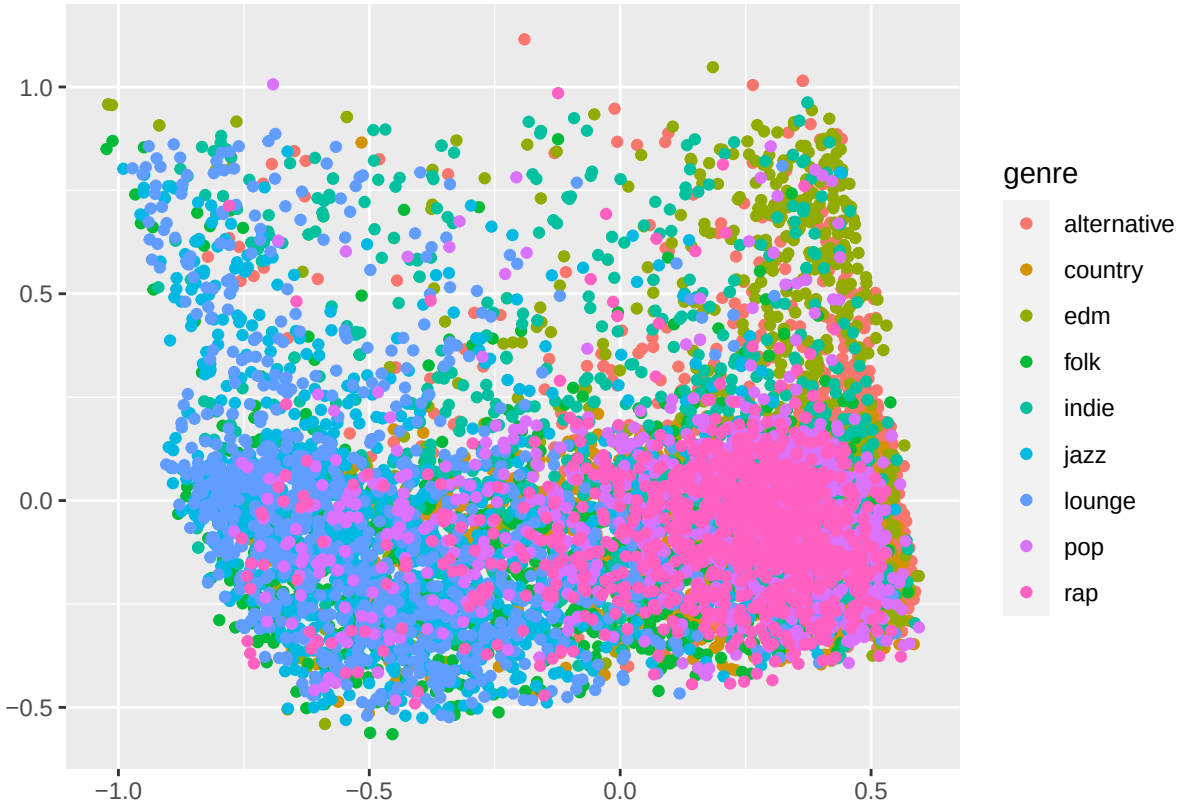


The first two principal components explain about 60% of the variance.

```
pca_df <- data.frame(predict(pc,data[,song_features_num]))
data_pca <- cbind(data,pca_df)
```

Let us take a look at the first two principal components although it only explains 60% of the variance.

```
data_pca %>% ggplot(aes(PC1,PC2,color = genre)) + geom_point() + xlab("") + ylab("")
```



The scatter plot isn't very clear. There is a better way to represent this. We can think of genres as a cluster and get the centroid of each genre and the average of distances to this centroid as a radius.

```
data_pca <- data_pca %>% dplyr::group_by(genre) %>% dplyr::mutate(PC1_mean = mean(PC1),
                                                                  PC2_mean = mean(PC2)) %>%
  dplyr::mutate(distance = (PC1-PC1_mean)^2+(PC2-PC2_mean)^2)
```

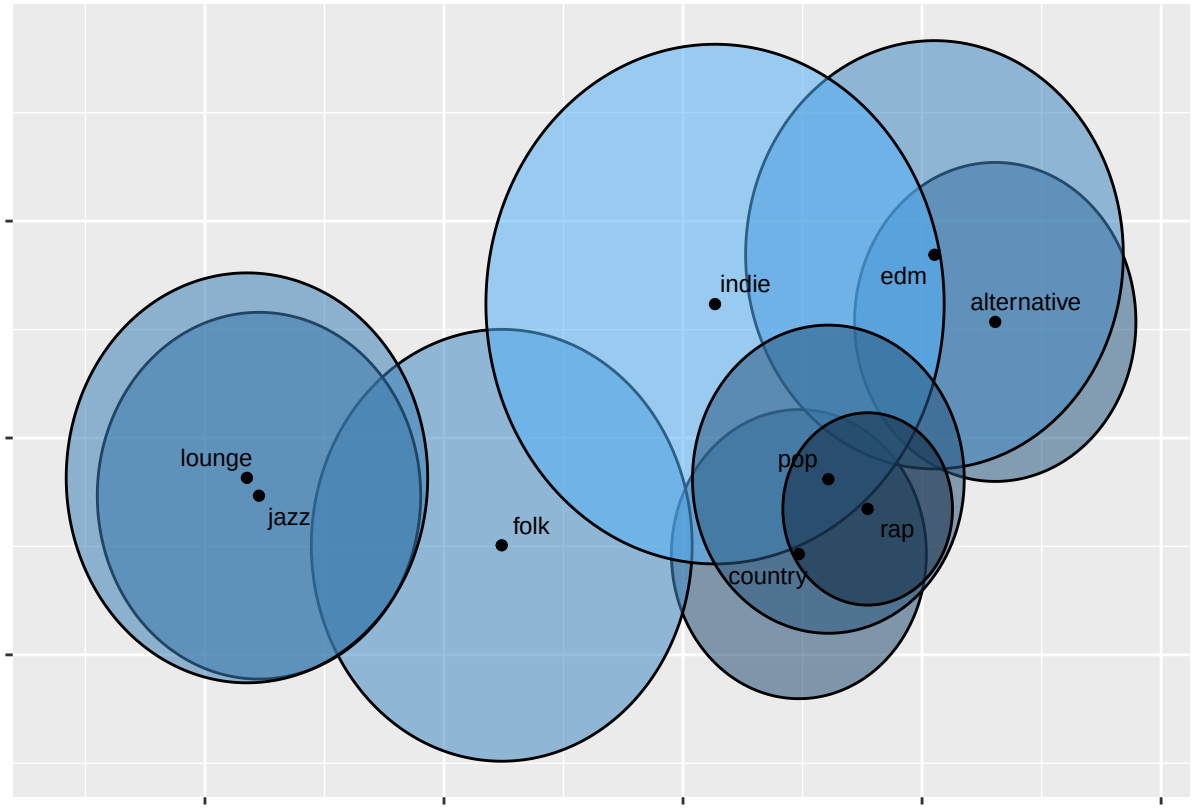
```
head(data_pca)
```

```
## # A tibble: 6 x 28
## # Groups:   genre [1]
##   track_id  track~1 genre dance~2 energy loudn~3 acous~4 instr~5 liven~6 valence
##   <chr>     <chr>   <chr>   <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 3PjetLEY~ Sea Gr~ alte~   0.346  0.886   0.925 8.84e-5 4.67e-4  0.0971   0.582
## 2 6ftHCe7Y~ I.M.Sin alte~   0.542  0.992   0.962 8.16e-6 0         0.0874   0.170
## 3 59UFxdRk~ You Ta~ alte~   0.605  0.767   0.902 1.72e-4 5.84e-5  0.0668   0.402
## 4 1SCXEzZQ~ Chocol~ alte~   0.443  0.865   0.871 2.31e-2 6.96e-5  0.403    0.641
## 5 0jk6J6B9~ Let Me~ alte~   0.467  0.649   0.910 1.14e-2 1.70e-5  0.212    0.751
## 6 7hEvciMv~ Seek a~ alte~   0.499  0.973   0.977 2.40e-5 2.11e-3  0.0608   0.554
## # ... with 18 more variables: tempo <dbl>, key_name <chr>, key_mode <chr>,
## #   mode <dbl>, mode_prob <dbl>, key_name_prob <dbl>, key_mode_prob <dbl>,
## #   PC1 <dbl>, PC2 <dbl>, PC3 <dbl>, PC4 <dbl>, PC5 <dbl>, PC6 <dbl>,
## #   PC7 <dbl>, PC8 <dbl>, PC1_mean <dbl>, PC2_mean <dbl>, distance <dbl>, and
## #   abbreviated variable names 1: track_name, 2: danceability, 3: loudness,
## #   4: acousticness, 5: instrumentalness, 6: liveness
```

```
genre_pca <- data_pca %>% dplyr::group_by(genre) %>% dplyr::summarise(PC1 = mean(PC1_mean),
                                                                      PC2 = mean(PC2_mean),
                                                                      R = mean(distance))
genre_pca
```

```
## # A tibble: 9 x 4
##   genre           PC1     PC2      R
##   <chr>         <dbl>   <dbl>  <dbl>
## 1 alternative  0.327   0.107  0.147
## 2 country      0.121  -0.107  0.133
## 3 edm          0.263   0.169  0.197
## 4 folk        -0.190  -0.0990 0.199
## 5 indie        0.0335  0.123  0.240
## 6 jazz        -0.443  -0.0532 0.169
## 7 lounge      -0.456  -0.0368 0.189
## 8 pop          0.152  -0.0380 0.142
## 9 rap          0.193  -0.0654 0.0887
```

```
library(ggforce)

genre_pca %>% ggplot()+geom_circle(aes(x0 = PC1,
                                       y0 = PC2,
                                       r = R,
                                       fill = R,
                                       alpha = 0.3))+
  geom_point(aes(PC1,PC2))+
  geom_text_repel(aes(PC1,PC2,label=genre),size = 3)+
  theme(axis.text.y=element_blank(),axis.text.x=element_blank(),legend.position = "none")+
  xlab("")+ylab("")
```



This does establish a lot of things we inferenced previously. Jazz and lounge are very similar so are edm and alternative to an extent. Rap music is very less variant. Surprisingly Pop and Country are closer to each other.