Assignment3_2023_Gia Bao Tran

2023-10-31

```r
#run needed packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(dplyr)
library(readr)
library(ggplot2)

#import data from csv file and assign NA to blank data
spotify <- read.csv('spotify-2023.csv', header=T, na.strings="")

#glance of the data frame
head(spotify)
```

```
##                             track_name    artist.s._name artist_count
## 1 Seven (feat. Latto) (Explicit Ver.)  Latto, Jung Kook            2
## 2                                 LALA       Myke Towers            1
## 3                              vampire    Olivia Rodrigo            1
## 4                         Cruel Summer      Taylor Swift            1
## 5                       WHERE SHE GOES         Bad Bunny            1
## 6                             Sprinter Dave, Central Cee            2
##   released_year released_month released_day in_spotify_playlists
## 1          2023              7           14                  553
## 2          2023              3           23                 1474
## 3          2023              6           30                 1397
## 4          2019              8           23                 7858
## 5          2023              5           18                 3133
## 6          2023              6            1                 2186
##   in_spotify_charts   streams in_apple_playlists in_apple_charts
## 1               147 141381703                 43             263
## 2                48 133716286                 48             126
## 3               113 140003974                 94             207
## 4               100 800840817                116             207
## 5                50 303236322                 84             133
## 6                91 183706234                 67             213
##   in_deezer_playlists in_deezer_charts in_shazam_charts bpm key  mode
## 1                  45               10              826 125   B Major
## 2                  58               14              382  92  C# Major
## 3                  91               14              949 138   F Major
## 4                 125               12              548 170   A Major
## 5                  87               15              425 144   A Minor
## 6                  88               17              946 141  C# Major
##   danceability_. valence_. energy_. acousticness_. instrumentalness_.
## 1             80        89       83             31                  0
## 2             71        61       74              7                  0
## 3             51        32       53             17                  0
## 4             55        58       72             11                  0
## 5             65        23       80             14                 63
## 6             92        66       58             19                  0
##   liveness_. speechiness_.
## 1          8             4
## 2         10             4
## 3         31             6
## 4         11            15
## 5         11             6
## 6          8            24
```

```
#check data type of each collumn
str(spotify)
```

```
## 'data.frame':    953 obs. of  24 variables:
##  $ track_name          : chr  "Seven (feat. Latto) (Explicit Ver.)" "LALA" "vampire" "Cruel Summer"
##  $ artist.s._name      : chr  "Latto, Jung Kook" "Myke Towers" "Olivia Rodrigo" "Taylor Swift" ...
##  $ artist_count        : int  2 1 1 1 1 2 2 1 1 2 ...
##  $ released_year       : int  2023 2023 2023 2019 2023 2023 2023 2023 2023 2023 ...
##  $ released_month      : int  7 3 6 8 5 6 3 7 5 3 ...
##  $ released_day        : int  14 23 30 23 18 1 16 7 15 17 ...
##  $ in_spotify_playlists: int  553 1474 1397 7858 3133 2186 3090 714 1096 2953 ...
##  $ in_spotify_charts   : int  147 48 113 100 50 91 50 43 83 44 ...
##  $ streams             : chr  "141381703" "133716286" "140003974" "800840817" ...
##  $ in_apple_playlists  : int  43 48 94 116 84 67 34 25 60 49 ...
##  $ in_apple_charts     : int  263 126 207 207 133 213 222 89 210 110 ...
##  $ in_deezer_playlists : chr  "45" "58" "91" "125" ...
##  $ in_deezer_charts    : int  10 14 14 12 15 17 13 13 11 13 ...
##  $ in_shazam_charts    : chr  "826" "382" "949" "548" ...
##  $ bpm                 : int  125 92 138 170 144 141 148 100 130 170 ...
##  $ key                 : chr  "B" "C#" "F" "A" ...
##  $ mode                : chr  "Major" "Major" "Major" "Major" ...
##  $ danceability_.      : int  80 71 51 55 65 92 67 67 85 81 ...
##  $ valence_.           : int  89 61 32 58 23 66 83 26 22 56 ...
##  $ energy_.            : int  83 74 53 72 80 58 76 71 62 48 ...
##  $ acousticness_.      : int  31 7 17 11 14 19 48 37 12 21 ...
##  $ instrumentalness_.  : int  0 0 0 0 63 0 0 0 0 0 ...
##  $ liveness_.          : int  8 10 31 11 11 8 8 11 28 8 ...
##  $ speechiness_.       : int  4 4 6 15 6 24 3 4 9 33 ...
```

```r
#Remove commas in number collumns if any
spotify$in_deezer_playlists <- as.numeric(gsub(",","",spotify$in_deezer_playlists))
spotify$in_shazam_charts <- as.numeric(gsub(",","",spotify$in_shazam_charts))

# Set streams data as numeric
spotify$streams <- as.numeric(spotify$streams)
```

```
## Warning: NAs introduced by coercion
```

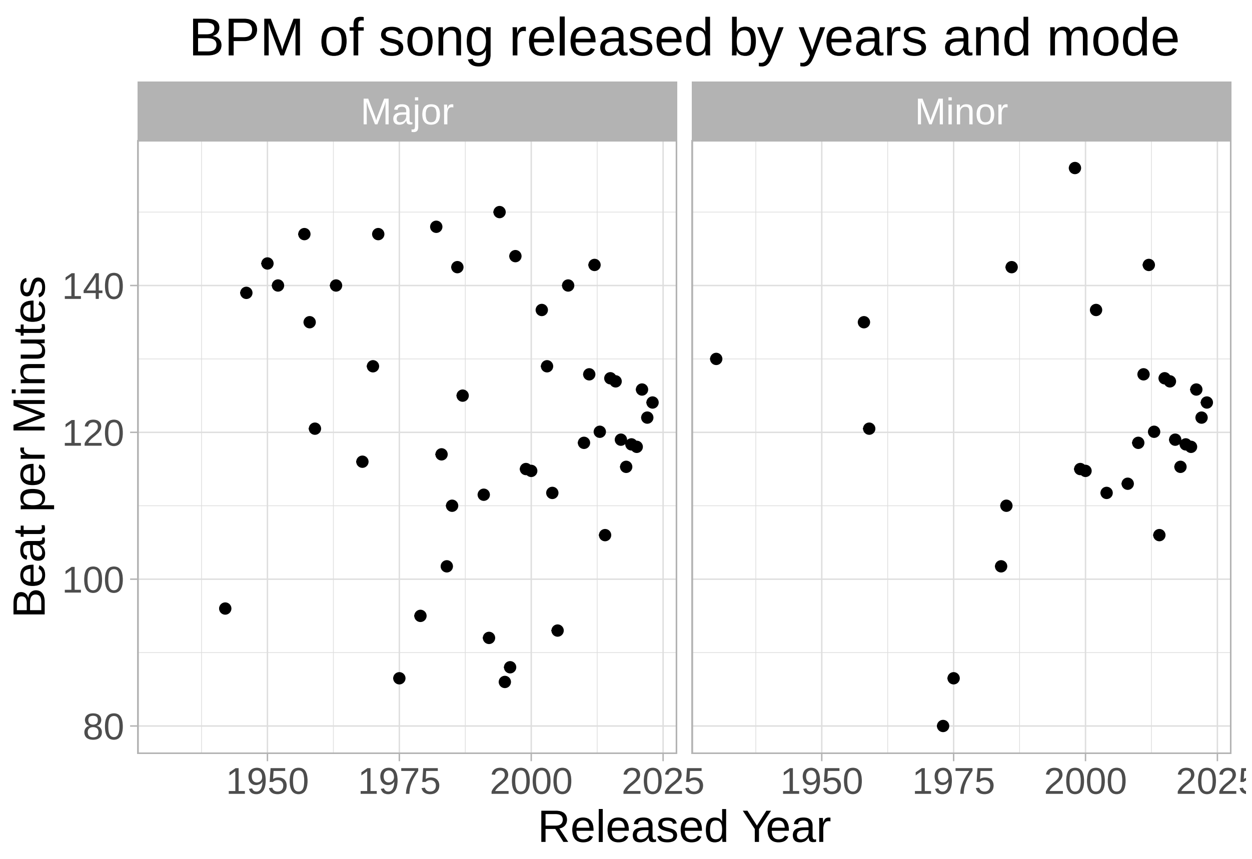
Visualisation 1

Data overview

```r
## Graph 1
# BPM of song released by years and mode
ov1 <- spotify %>%
  group_by(released_year) %>%
  mutate(n=mean(bpm)) %>%
  group_by(released_year,mode, n) %>%
  summarise(n1=n())
```

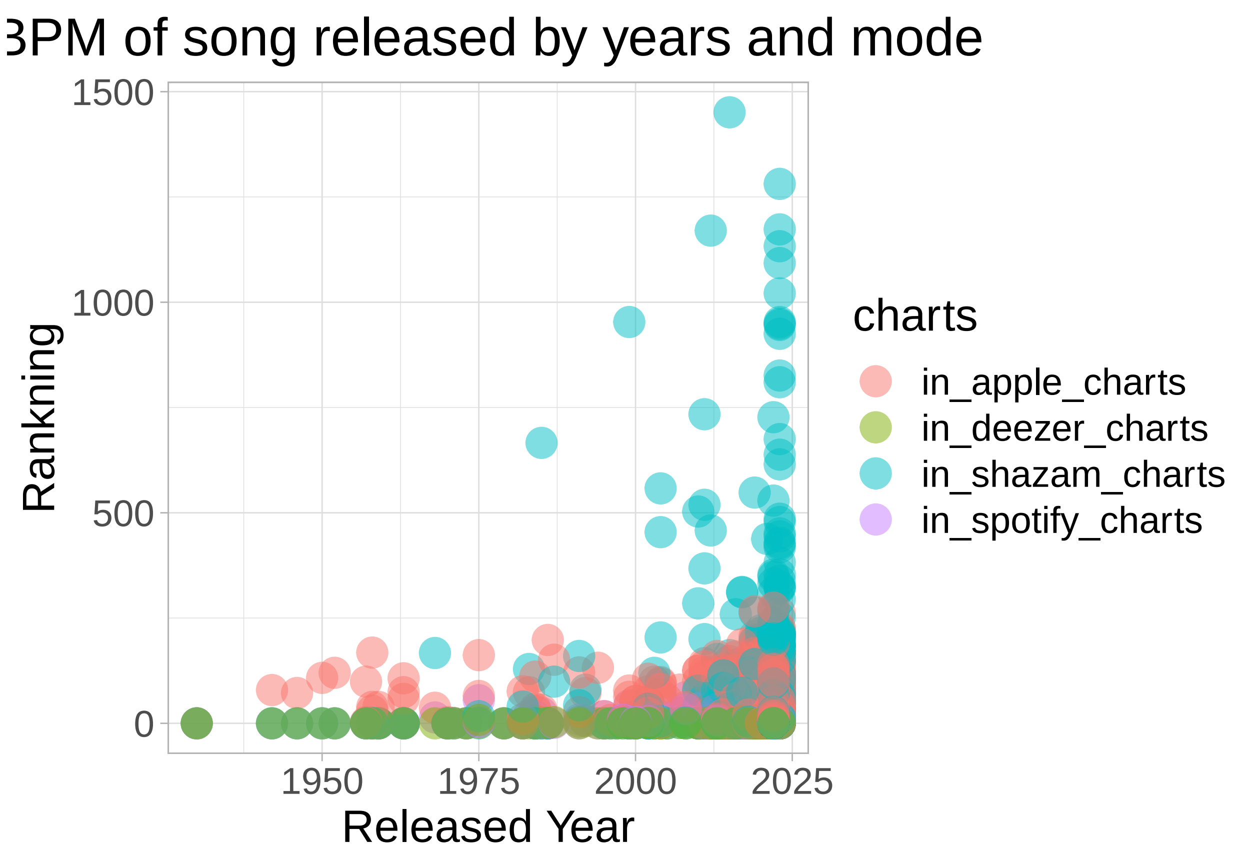
```
## `summarise()` has grouped output by 'released_year', 'mode'. You can override
## using the `.groups` argument.
```

```r
ggplot(ov1, aes(x = released_year, y = n)) + geom_point() + facet_wrap(~mode, drop=TRUE)+
  xlab("Released Year") + ylab("Beat per Minutes") +
ggtitle("BPM of song released by years and mode") + theme_light() + theme(plot.title = element_text(hju
```

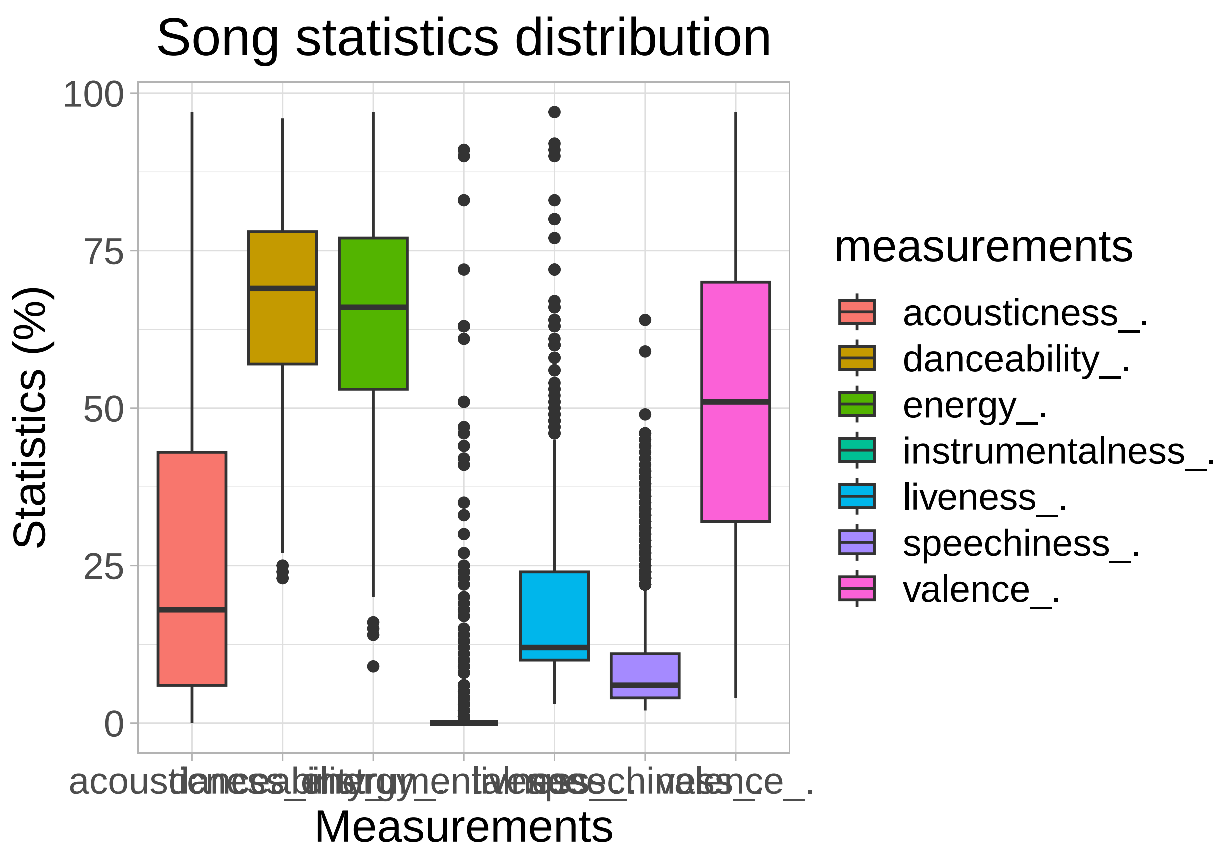


```r
# ranking of top 1000 Songs by platform charts through year
ov2 <- spotify %>%
  pivot_longer(cols = c(`in_spotify_charts`, `in_apple_charts`,
                        `in_shazam_charts`, `in_deezer_charts`),
               names_to = "charts",
               values_to = "ranking")

ggplot(data=ov2, aes(x=released_year, y=ranking, color=charts)) + geom_point(alpha=0.5, size=5) + xlab(
ggtitle("BPM of song released by years and mode") + theme_light() + theme(plot.title = element_text(hju
```

```
## Warning: Removed 50 rows containing missing values (‘geom_point()‘).
```



```
# box plot of statistics
ov3 <- spotify %>%
  pivot_longer(cols = c(`danceability_.`: `speechiness_.`),
               names_to = "measurements",
               values_to = "statistics")
ggplot(ov3, aes(x = measurements, y = statistics, fill= measurements)) + geom_boxplot() + xlab("Measuren
ggtitle("Song statistics distribution") + theme_light()+
theme(plot.title = element_text(hjust = 0.5)) + theme(text = element_text(size = 17))
```



Visualisation 2

Song distribution and categories analysis

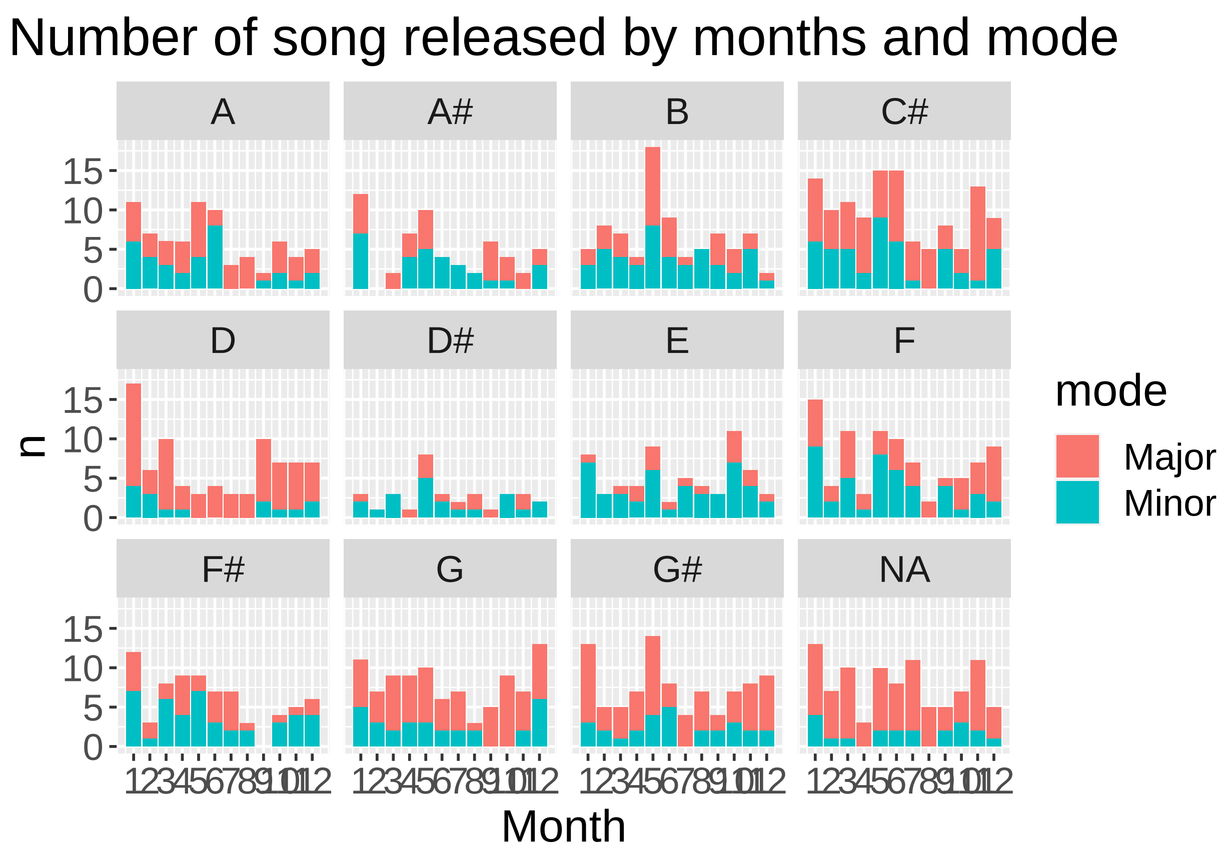
```r
#Number of song released by months and its mode and key distribution.

graph1_1 <- spotify %>%
  group_by(released_month, mode, key) %>%
  summarise(n=n())
```

```
## ‘summarise()‘ has grouped output by ’released_month’, ’mode’. You can override
## using the ‘.groups‘ argument.
```

```r
month.abb[graph1_1$released_month]
```

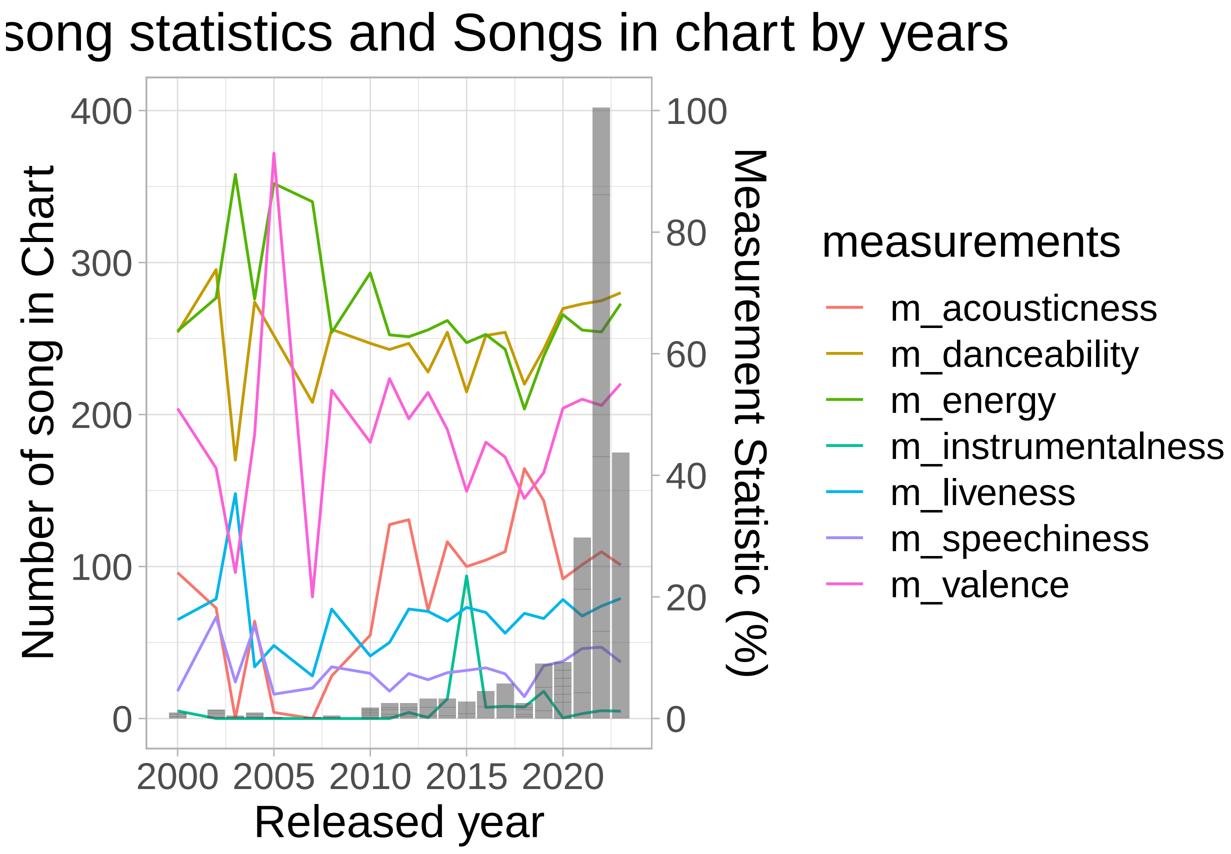
```
##  [1] "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan"
## [13] "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan" "Jan"
## [25] "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb"
## [37] "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Feb" "Mar" "Mar" "Mar" "Mar"
## [49] "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Mar"
## [61] "Mar" "Mar" "Mar" "Mar" "Mar" "Mar" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr"
## [73] "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr" "Apr"
##  [85] "Apr" "Apr" "Apr" "Apr" "May" "May" "May" "May" "May" "May" "May" "May"
##  [97] "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May"
## [109] "May" "May" "May" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun"
## [121] "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun" "Jun"
## [133] "Jun" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul"
## [145] "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Jul" "Aug" "Aug" "Aug"
## [157] "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug" "Aug"
## [169] "Aug" "Aug" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep"
## [181] "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Sep" "Oct" "Oct" "Oct"
## [193] "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct"
## [205] "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Oct" "Nov" "Nov" "Nov" "Nov" "Nov"
## [217] "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Nov"
## [229] "Nov" "Nov" "Nov" "Nov" "Nov" "Nov" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec"
## [241] "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec" "Dec"
## [253] "Dec" "Dec" "Dec" "Dec" "Dec"
```

```r
ggplot(graph1_1, aes(x = released_month, y = n, fill=mode)) + geom_bar(stat='identity') + labs(x=NULL)+
  theme(text=element_text(size=8)) + facet_wrap(~key, drop=TRUE) +
  scale_x_continuous(breaks = seq_along(month.abb)) + xlab('Month') +
  ggtitle("Number of song released by months and mode") + theme(plot.title = element_text(hjust = 0.5))
```



```r
  # dancebility and number of song overtime (from 2000 to 2023)

spotify$track_name <- as.character(spotify$track_name)
```

```r
graph1_2 <- spotify %>%
  filter(released_year >= 2000) %>%
  group_by(released_year) %>%
  summarise(n=n(),
            m_danceability=mean(danceability_.),
            m_valence=mean(valence_.),
            m_energy=mean(energy_.),
            m_acousticness=mean(acousticness_.),
            m_instrumentalness=mean(instrumentalness_.),
            m_liveness=mean(liveness_.),
            m_speechiness=mean(speechiness_.)) %>%
 pivot_longer(cols = c(`m_danceability`: `m_speechiness`),
               names_to = "measurements",
               values_to = "statistics")

ggplot(data = graph1_2) +
  geom_line(aes(x=released_year,y=statistics*4, color = measurements),
stat='identity') + ylab('Number of song in Chart') +
  geom_bar(aes(x=released_year, y= n/7, alpha=0.5), stat = 'identity') +
  scale_y_continuous(sec.axis=sec_axis(~.*0.25,name="Measurement Statistic (%)",breaks = seq(0, 100, by
ggtitle('Average song statistics and Songs in chart by years') + theme_light()+
theme(plot.title = element_text(hjust = 0.5)) + guides(alpha='none') + theme(text = element_text(size =
```

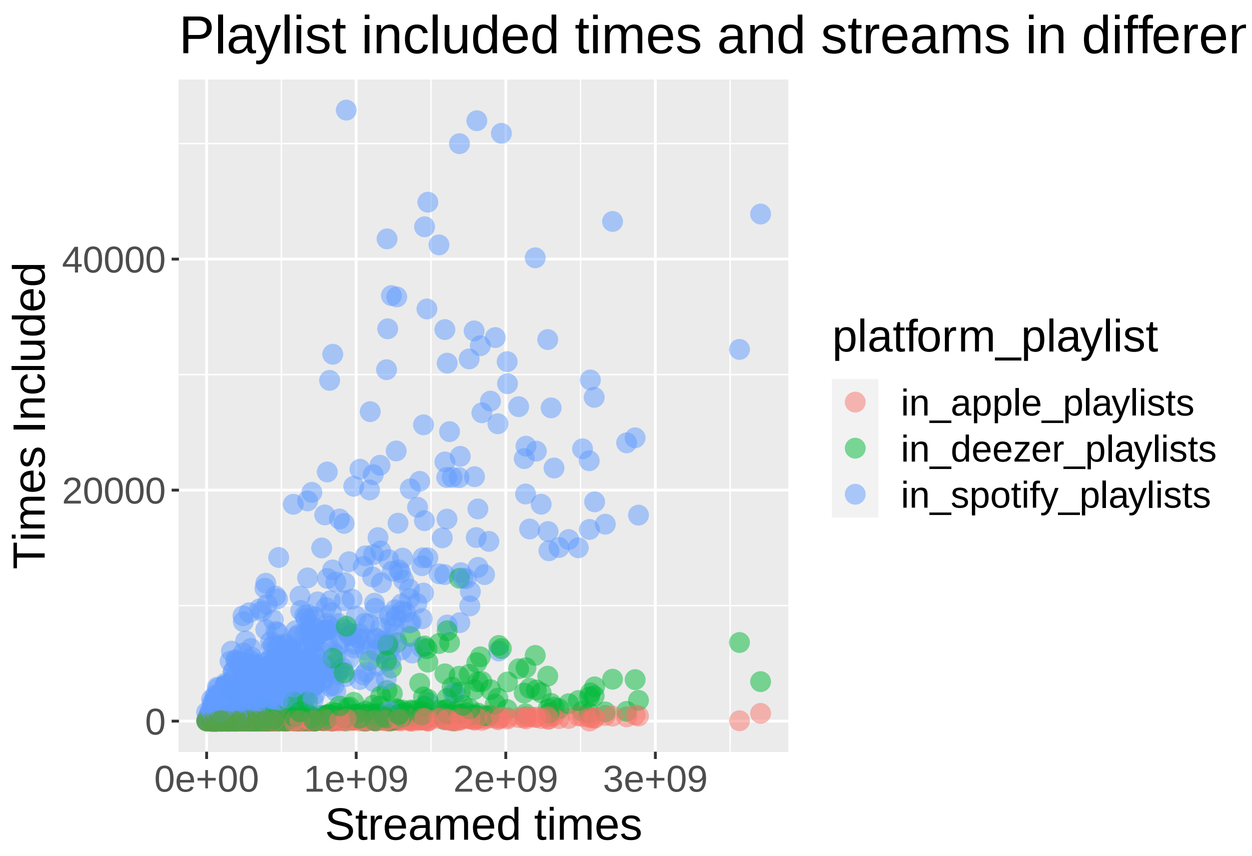


Visualisation 3

streaming platforms analysis

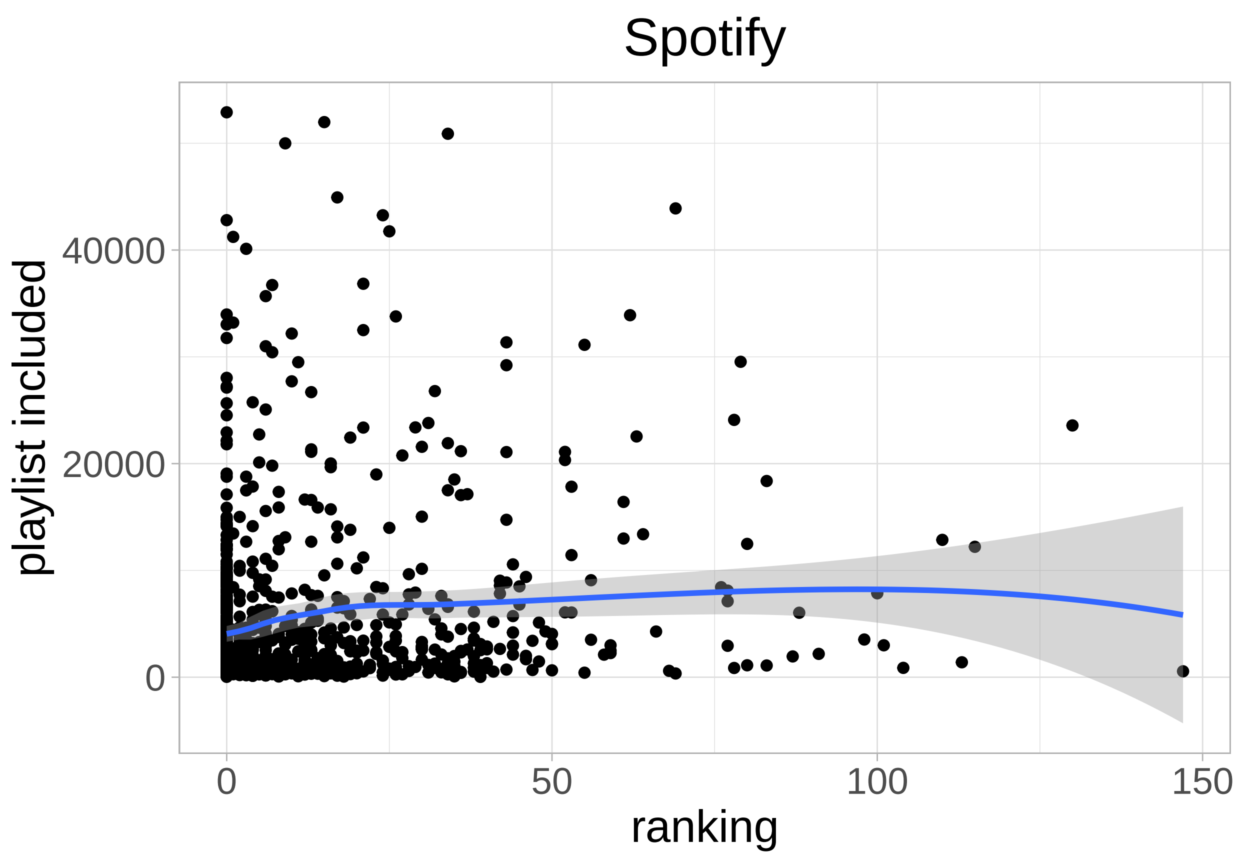
```r
# Bubble plot of different
graph2_1 <- spotify %>%
  pivot_longer(cols = c(`in_spotify_playlists`,`in_apple_playlists`,
                        `in_deezer_playlists`),
               names_to = "platform_playlist",
               values_to = "included_times")

ggplot(graph2_1, aes(x=streams, y=included_times, color=platform_playlist)) + geom_point(alpha=0.5, siz
```

```
## Warning: Removed 3 rows containing missing values (`geom_point()`).
```



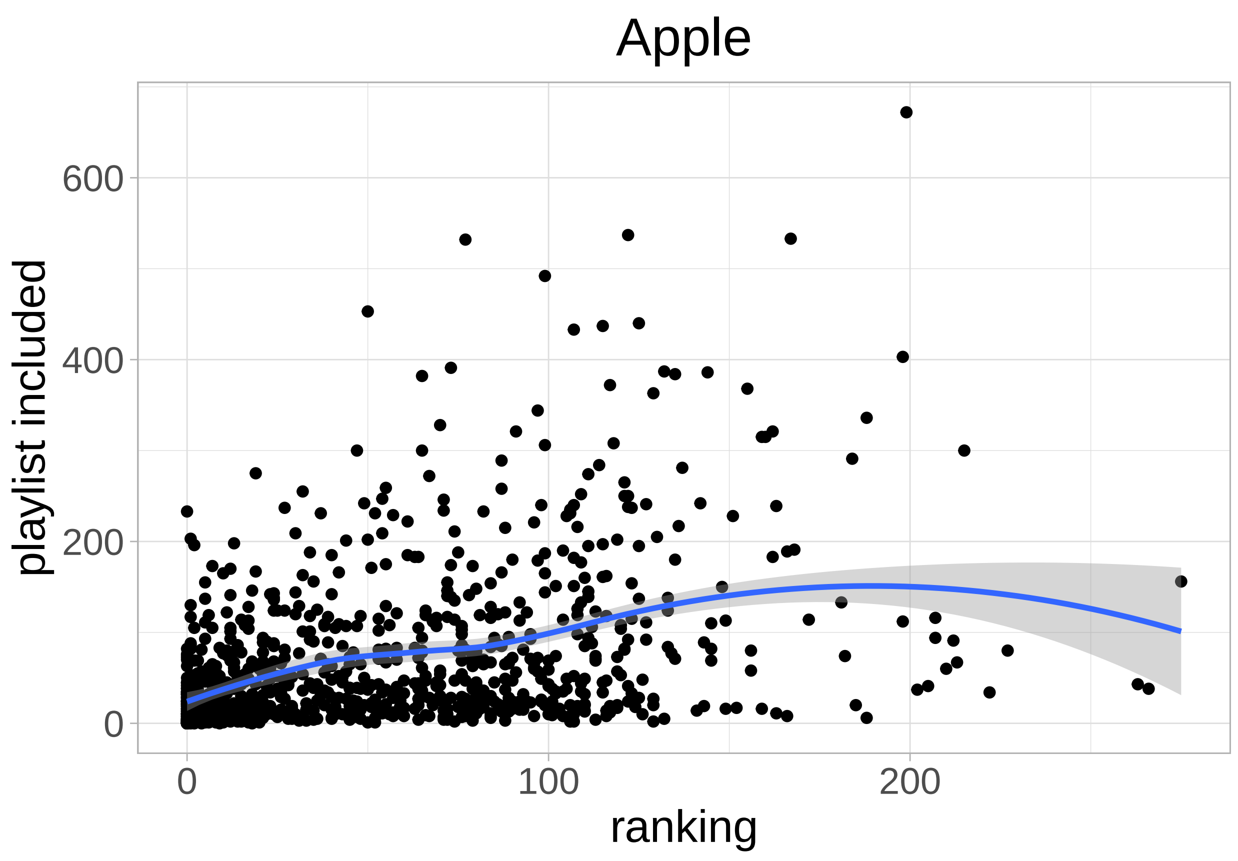
```r
# Relationship between chart and playlist of different platforms

ggplot(spotify, aes(x=in_spotify_charts, y=in_spotify_playlists)) + geom_point() + geom_smooth() + ggti
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



```
ggplot(spotify, aes(x=in_apple_charts, y=in_apple_playlists)) + geom_point() + geom_smooth() + ggtitle(
```

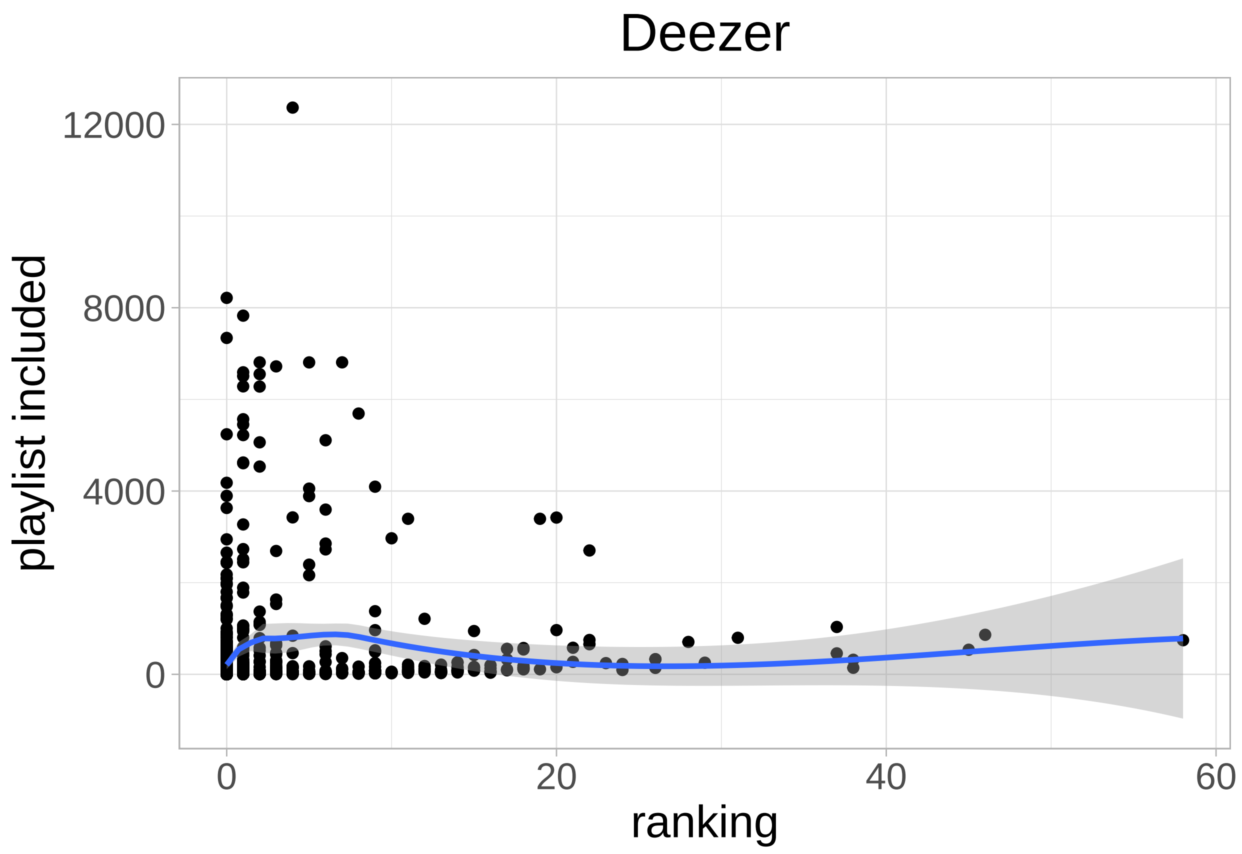
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



```
ggplot(spotify, aes(x=in_deezer_charts, y=in_deezer_playlists)) + geom_point() + geom_smooth() + ggtitl
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at -0.29

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 2.29

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 4.2235e-15

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 4

## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## -0.29

## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 2.29
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 4.2235e-15
```

```
## Warning in predLoess(object$y, object$x, newx = if (is.null(newdata)) object$x
## else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 4
```



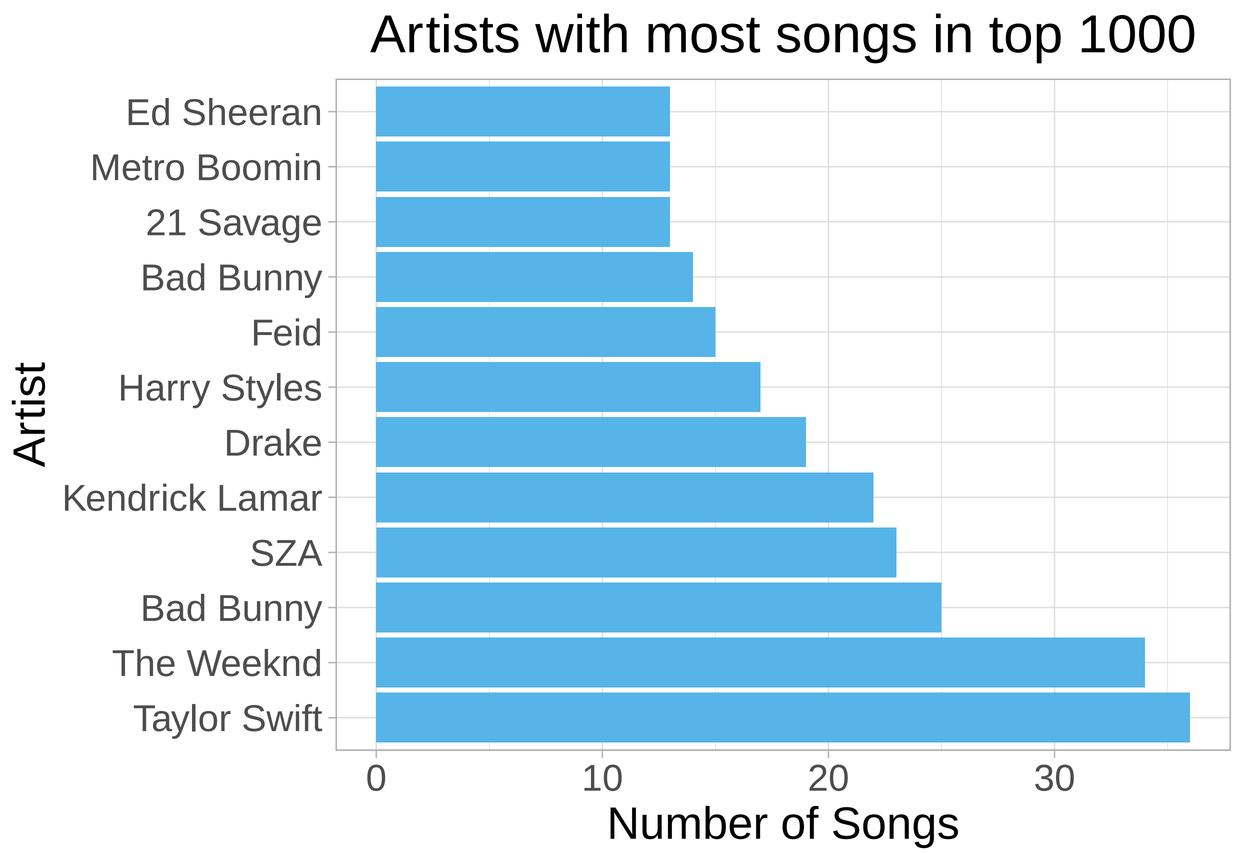
Visualisation 4

Artist analysis

```
## GRAPH 1

# artist with most songs
graph3_1 <- spotify %>%
  separate_rows(artist.s._name, sep = ",") %>%
  group_by(artist.s._name) %>%
  summarise(n=n()) %>%
  mutate(rank = min_rank(desc(n))) %>%
  arrange(rank) %>%
  filter(rank<=10)
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): input string 119 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 211 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 232 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 237 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 259 is invalid UTF-8
```

```
ggplot(graph3_1, aes(x=reorder(artist.s._name,-n), y=n)) + geom_bar(stat='identity', fill="#56B4E9")+
ggtitle("Artists with most songs in top 1000")+ coord_flip() + theme_light()+
theme(plot.title = element_text(hjust = 0.5)) + xlab("Artist")+ylab("Number of Songs") + theme(text = el
```



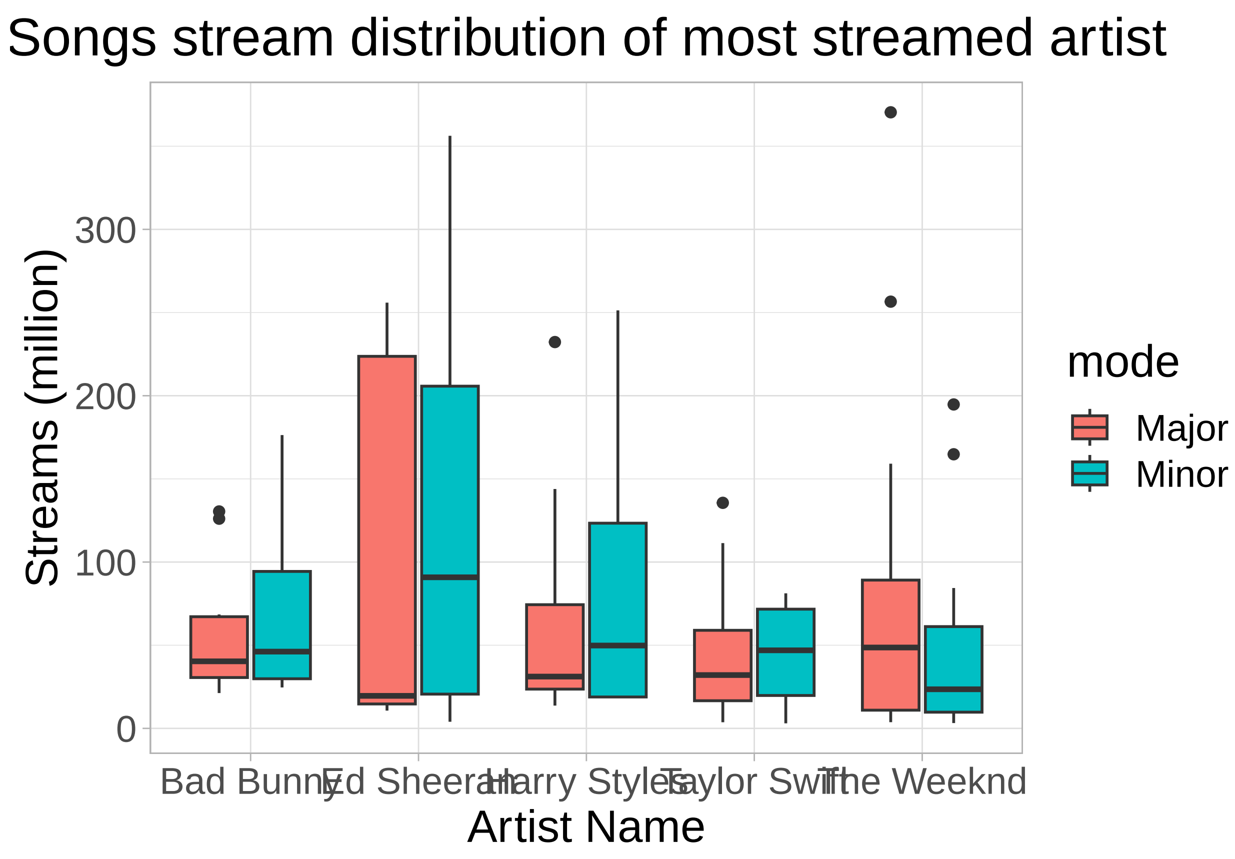
```
##GRAPH 2

    # Filter out most streamed artists
graph3_3a <- spotify[-c(575), ]

graph3_3a <- graph3_3a %>%
  separate_rows(artist.s._name, sep = ",") %>%
  group_by(artist.s._name) %>%
  summarise(n = sum(streams)) %>%
  arrange(desc(n)) %>%
  mutate(rank = dense_rank(desc(n)))
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): input string 119 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 211 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 232 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 237 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 259 is invalid UTF-8
```

```
graph3_3a <- filter(graph3_3a, rank <= 5)

    # Collect the name from above table and plot
graph3_3 <- spotify[-c(575), ]

graph3_3 <- spotify %>%
  separate_rows(artist.s._name, sep = ",") %>%
  filter(artist.s._name == 'The Weeknd'|
           artist.s._name == 'Bad Bunny'|
           artist.s._name == 'Ed Sheeran'|
           artist.s._name == 'Taylor Swift'|
           artist.s._name == 'Harry Styles')
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): input string 119 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 211 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 232 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 237 is invalid UTF-8

## Warning in gregexpr(pattern, x, perl = TRUE): input string 259 is invalid UTF-8
```

```
ggplot(graph3_3, aes(x = artist.s._name, y = streams*0.0000001, fill=mode)) + geom_boxplot() + xlab("Ar
ggtitle("Songs stream distribution of most streamed artist") + theme_light()+
theme(plot.title = element_text(hjust = 0.5)) + theme(text = element_text(size = 17))
```

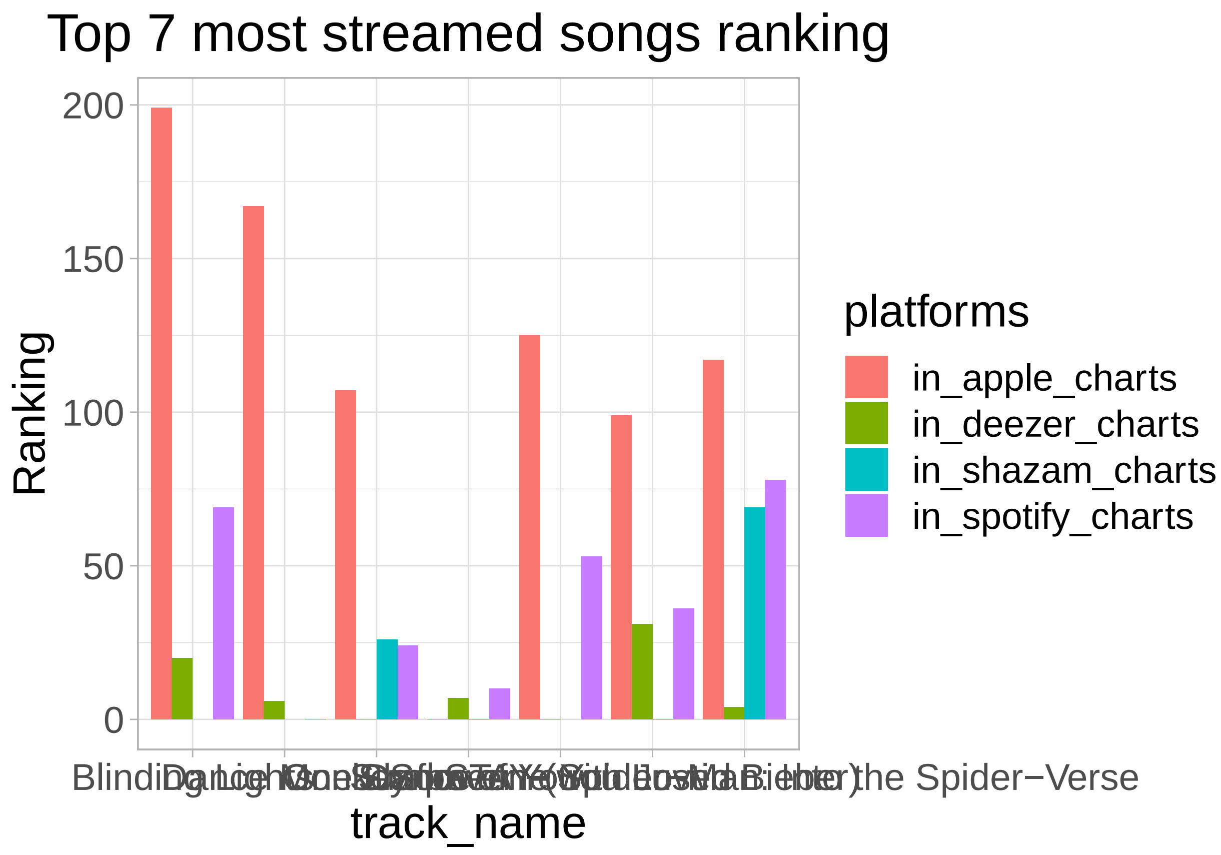


```
# number of artist with relation to streams
```

Visualisation 5

mixed variables analysis

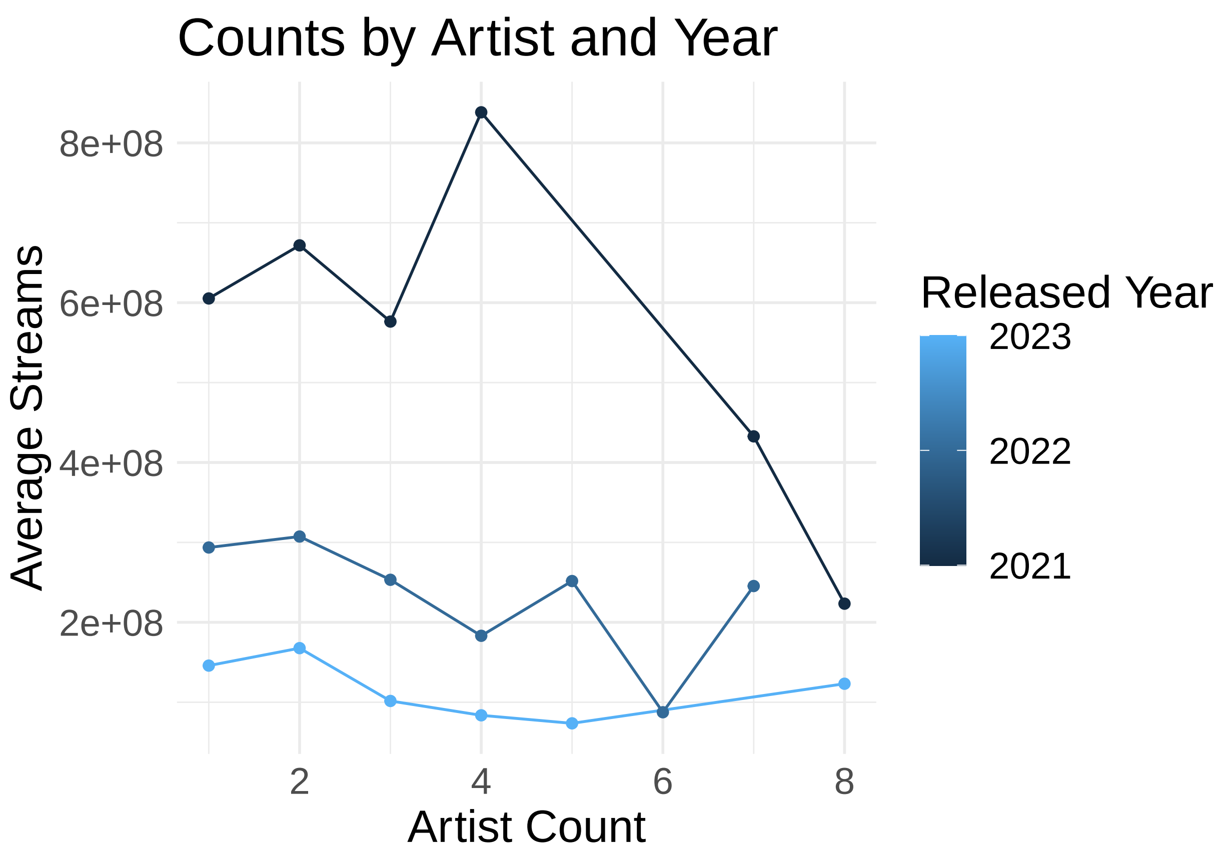
```
# GRAPH 1
# Most streamed song ranking in different charts
graph5_1 <- spotify %>%
  mutate(rank = min_rank(desc(streams))) %>%
  filter(rank <= 7) %>%
  arrange(rank)

graph5_1 <-graph5_1%>%
    pivot_longer(cols = c(`in_spotify_charts`,`in_apple_charts`,
                          `in_deezer_charts`, `in_shazam_charts`),
                 names_to = "platforms",
                 values_to = "ranking")


ggplot(graph5_1, aes(x = track_name, y = ranking, fill=platforms))+
geom_bar(position=position_dodge(), stat='identity') +
theme_light() + ggtitle('Top 7 most streamed songs ranking') + ylab('Ranking') +  theme(plot.title = el
## Warning: Removed 3 rows containing missing values ('geom_bar()').
```



```
## GRAPH 2
# Average stream time relationship with artist count and release year (2021-2023)

graph5_2 <- spotify[-c(575), ]

graph5_2 <- graph5_2 %>%
  group_by(artist_count,released_year) %>%
  summarise(n=mean(streams)) %>%
  filter(released_year>=2021)
```

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

```
graph5_2$artist_count <- as.numeric(graph5_2$artist_count)

graph5_2 <- graph5_2 %>% arrange(artist_count)

ggplot(graph5_2, aes(x = artist_count, y = n))+
geom_line(aes(color = released_year, group = released_year)) + geom_point(aes(color = released_year))+
  scale_color_continuous(name = "Released Year", breaks = 2021:2023) + theme(text = element_text(size =
```


Counts by Artist and Year
Average Streams
8e+08
6e+08
4e+08
2e+08
2
4
6
8
Artist Count
Released Year
2023
2022
2021