Taylor Style

20nguyen.hk

2024-04-06

Introduction

The selected dataset comprises comprehensive data on Taylor Swift's discography, including three subsets: `taylor_album_songs`, `taylor_all_songs`, and `taylor_albums`. These subsets provide a detailed look at Taylor Swift's studio albums, EPs, singles, and re-releases, covering audio features, album release dates, Metacritic scores, and user scores. This dataset originates from the Genius lyrics database and Spotify's API, offering a rich field for exploring the relationship between musical elements and album reception. Dataset link here (https://tinyurl.com/yc2axk9f).

Data Dictionary

`taylor_album_songs.csv`

variable	class	description
album_name	character	Album name
ep	logical	Is it an EP
album_release	double	Album release date
track_number	integer	Track number
track_name	character	Track name
artist	character	Artists
featuring	character	Artists featured
bonus_track	logical	Is it a bonus track
promotional_release	double	Date of promotional release
single_release	double	Date of single release
track_release	double	Date of track release
danceability	double	Spotify danceability score. A value of 0.0 is least danceable and 1.0 is most danceable.
energy	double	Spotify energy score. Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
key	integer	The key the track is in.

variable	class	description
loudness	double	Spotify loudness score. The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track.
mode	integer	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
speechiness	double	Spotify speechiness score. 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	double	Spotify acousticness score. A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
instrumentalness	double	Spotify instrumentalness score. Predicts whether a track contains no vocals. 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	double	Spotify liveness score. 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	double	Spotify valence score. 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).
tempo	double	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.

variable	class	description
time_signature	integer	An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of “3/4”, to “7/4”.
duration_ms	integer	The duration of the track in milliseconds.
explicit	logical	Does the track have explicit lyrics.
key_name	character	The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C#/D♭, 2 = D, and so on. If no key was detected, the value is -1.
mode_name	character	Modality of the track.
key_mode	character	The key of the track.
lyrics	list	Track lyrics. These values are all NA. To get the lyrics in nested tibbles, `install.packages("taylor")` and use the source data.

taylor_all_songs.csv

variable	class	description
album_name	character	Album name
ep	logical	Is it an EP
album_release	double	Album release date
track_number	integer	Track number
track_name	character	Track name
artist	character	Artists
featuring	character	Artists featured
bonus_track	logical	Is it a bonus track
promotional_release	double	Date of promotional release
single_release	double	Date of single release
track_release	double	Date of track release
danceability	double	Spotify danceability score. A value of 0.0 is least danceable and 1.0 is most danceable.

variable	class	description
energy	double	Spotify energy score. Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
key	integer	The key the track is in.
loudness	double	Spotify loudness score. The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track.
mode	integer	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
speechiness	double	Spotify speechiness score. 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	double	Spotify acousticness score. A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
instrumentalness	double	Spotify instrumentalness score. Predicts whether a track contains no vocals. 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	double	Spotify liveness score. 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	double	Spotify valence score. 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).

variable	class	description
tempo	double	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.
time_signature	integer	An estimated time signature. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of “3/4”, to “7/4”.
duration_ms	integer	The duration of the track in milliseconds.
explicit	logical	Does the track have explicit lyrics.
key_name	character	The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C#/D♭, 2 = D, and so on. If no key was detected, the value is -1.
mode_name	character	Modality of the track.
key_mode	character	The key of the track.
lyrics	list	Track lyrics. These values are all NA. To get the lyrics in nested tibbles, `install.packages("taylor")` and use the source data.

`taylor_albums.csv`

variable	class	description
album_name	character	Album name
ep	logical	Is it an EP
album_release	double	Album release date
metacritic_score	integer	Metacritic score
user_score	double	User score

Reasons to choose the dataset

Taylor Swift has revolutionized the music industry with her timeless and relatable songwriting, winning numerous prestigious prizes and becoming one of our time’s most influential and beloved artists. In light of this, we hope that analyzing a dataset about Taylor Swift’s songs can provide valuable insights into the strategies behind her success. The dataset offers comprehensive features related to her albums and songs, so we expect to unveil interesting undiscovered patterns that contribute to her widespread appeal. This knowledge can provide valuable lessons in music production, marketing, and cultural impact, which benefits aspiring artists and music industry professionals.

Importing neccesary packages

```
suppressMessages({
  library(tidyverse)
  library(ggcorrplot)
  library(dplyr)
  library(corrplot)
  library(ggplot2)
  library(gridExtra)
  library(cluster)
  library(RColorBrewer)
})
```

Question 1: How do features such as `danceability`, `energy`, `valence`, and `tempo` correlate with Taylor Swift's albums' critical and public reception?

Introduction

In light of Taylor Swift's success, it can be beneficial to analyze how the characteristics of her songs influence their success. By analyzing features such as lyrics' explicitness, tempo and key, we hope to uncover patterns in Taylor Swift's songs that contribute to her music's popularity.

To simplify unnecessarily complexity, we propose excluding these columns. We believe they are not particularly interesting or relevant to how people score the albums. Our focus is on studying the statistics of Taylor Swift's songs based on audio features and how these features influence album scoring by listeners.

Column Name	Data Type	Description
ep	logical	Is it an EP?
album_release	double	Album release date
artist	character	Artists
featuring	character	Artists featured
bonus_track	logical	Is it a bonus track?
promotional_release	double	Date of promotional release
single_release	double	Date of single release
track_release	double	Date of track release

The overall dataset is as follows:

- Three columns represent the index
- 11 numerical features
- 6 categorical features

- 2 target numerical variables

Approach

The features incorporated in our analysis are split into two types: numerical and categorical features.

For the former, we utilize a correlation matrix. This is a very common way to reveal relationships between numerical or binary variables, which is exactly our goal.

For the latter, faceted box plots are used instead as using a correlation matrix is not recommended for non-binary categorical variables. Here, we also use bar charts to visualize class distribution, since class imbalance can impact our observations.

Analysis

Data preprocessing

Firstly, we need to read the two csv files. The first file contain the information about audio features (numerical and categorical) for each song whereas the second file contain the information about the album name, release date and scores:

```
album_song_df = read.csv("taylor_album_songs.csv")
album_df = read.csv("taylor_albums.csv")
```

Dropping unnecessary columns:

```
album_df_subset <- album_df[c("album_name", "metacritic_score", "user_score")]
album_song_df_subset <- album_song_df[c("album_name", "track_number",
                                        "track_name", "danceability",
                                        "energy", "key", "loudness",
                                        "mode", "speechiness",
                                        "acousticness", "instrumentalness",
                                        "liveness", "valence", "tempo",
                                        "time_signature", "duration_ms",
                                        "explicit", "key_name",
                                        "mode_name", "key_mode")]
```

We further remove rows with NaN values since we believe it is best to have a clean dataset. We don't think there is a reasonable way to interpolate these missing values, especially since they are subjective (for example, the user scores and Metacritic scores).

```
album_df <- na.omit(album_df)
```

Assuming that metacritic score and user score are calculated as their definition, which are averaged over the songs and acts as a mean, we can use it to represent the score for each song in the same album. We remove all NaN rows since there is no reasonable way to interpolate these scores based on the other albums. And the number of NaN rows is insignificant.

```
album_song_with_scores_df <- merge(album_song_df_subset, album_df_subset,
                                   by = "album_name", all.x = TRUE)

album_song_with_scores_df <- na.omit(album_song_with_scores_df)
```

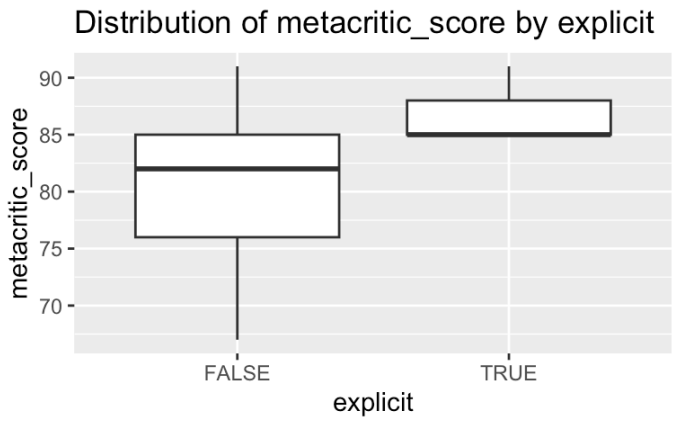
The constructed dataframe is now as follows: - Three columns represent the index - 11 numerical features - 6 categorical features - 2 target numerical variables

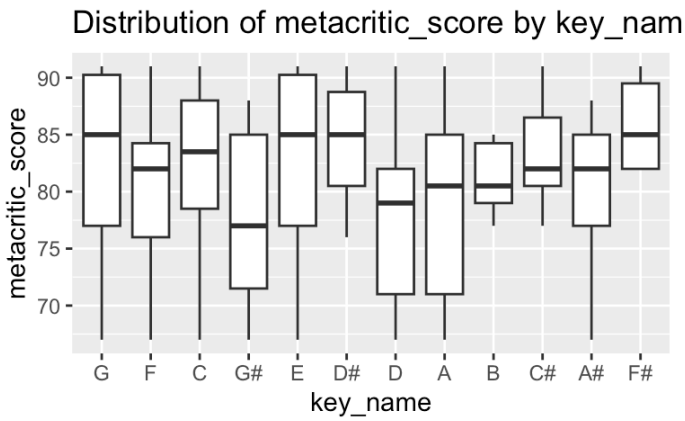
```
# 3 first index columns

# 11 numerical columns
numerical_feats = c("danceability", "energy", "loudness", "speechiness",
                    "acousticness", "instrumentalness", "liveness",
                    "valence", "tempo", "time_signature", "duration_ms")

# 6 categorical columns
categorical_feats = c("explicit", "key", "key_name", "mode",
                      "mode_name", "key_mode")

# 2 target numerical columns: metacritic_score VS user_score
```

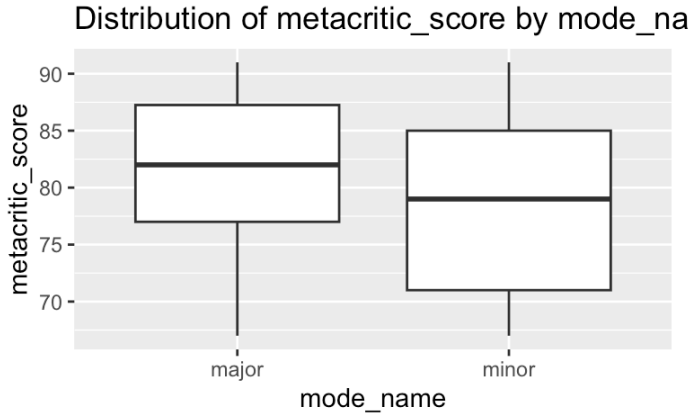
Box plots for categorical variables

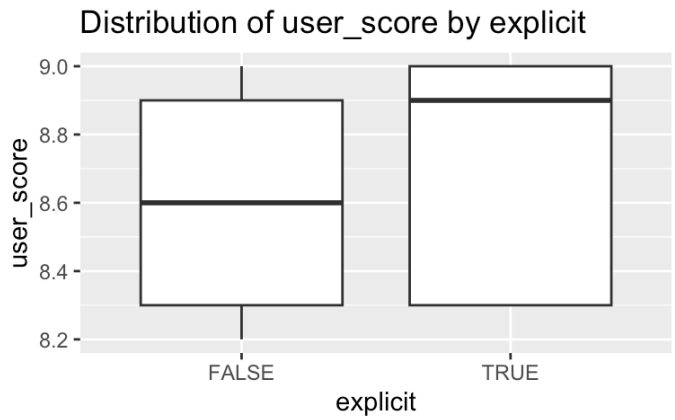
```
# Define the list of categorical columns
# categorical_cols <- c("key", "mode", "explicit", "key_name", "mode_name")
categorical_cols <- c("explicit", "key_name", "mode_name") # remove "key" and "mode"
because they're just "key_name" and "mode_name" but in numerical form.

# Create facet grid plots for each score
for (score in c("metacritic_score", "user_score")) {
  # Initialize a list to store individual plots
  plots <- list()

  # Loop through each categorical column
  for (col in categorical_cols) {
    # Convert the column to factor with explicit levels
    album_song_with_scores_df[[col]] <-
      factor(album_song_with_scores_df[[col]],
             levels = unique(album_song_with_scores_df[[col]]))

    # Create a facet grid plot for the current categorical column
    p <- ggplot(album_song_with_scores_df, aes_string(x = col, y = score)) +
      geom_boxplot() +
      labs(x = col, y = score) +
      ggtitle(paste("Distribution of", score, "by", col))

    # Add the plot to the list
    plots[[col]] <- p
  }

  # Combine plots into a single facet grid
  grid.arrange(grobs = plots, nrow = 2, ncol = 2)
  # Adjust nrow and ncol as needed
}
```

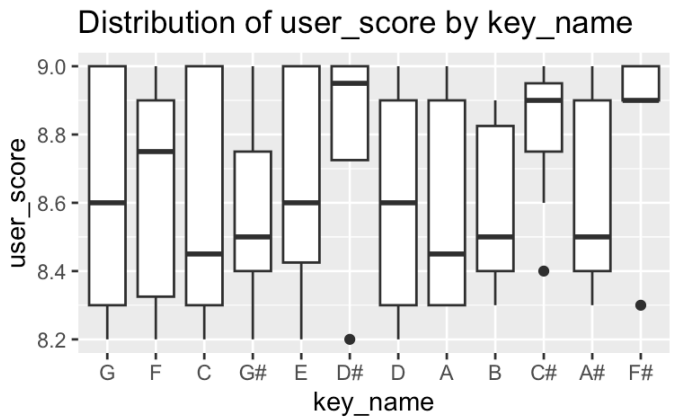
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## ℹ Please use tidy evaluation idioms with `aes()`.
## ℹ See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

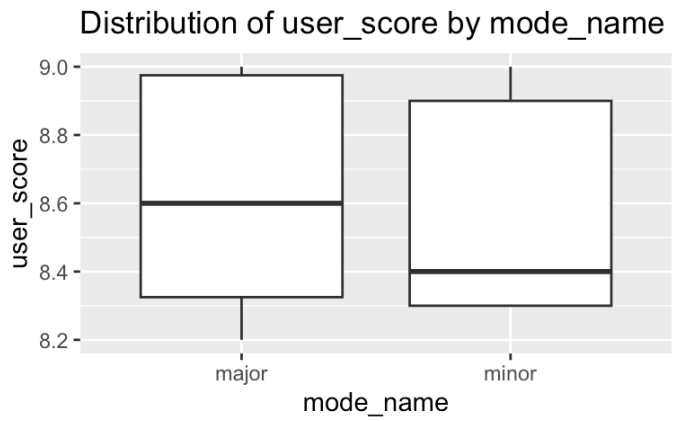






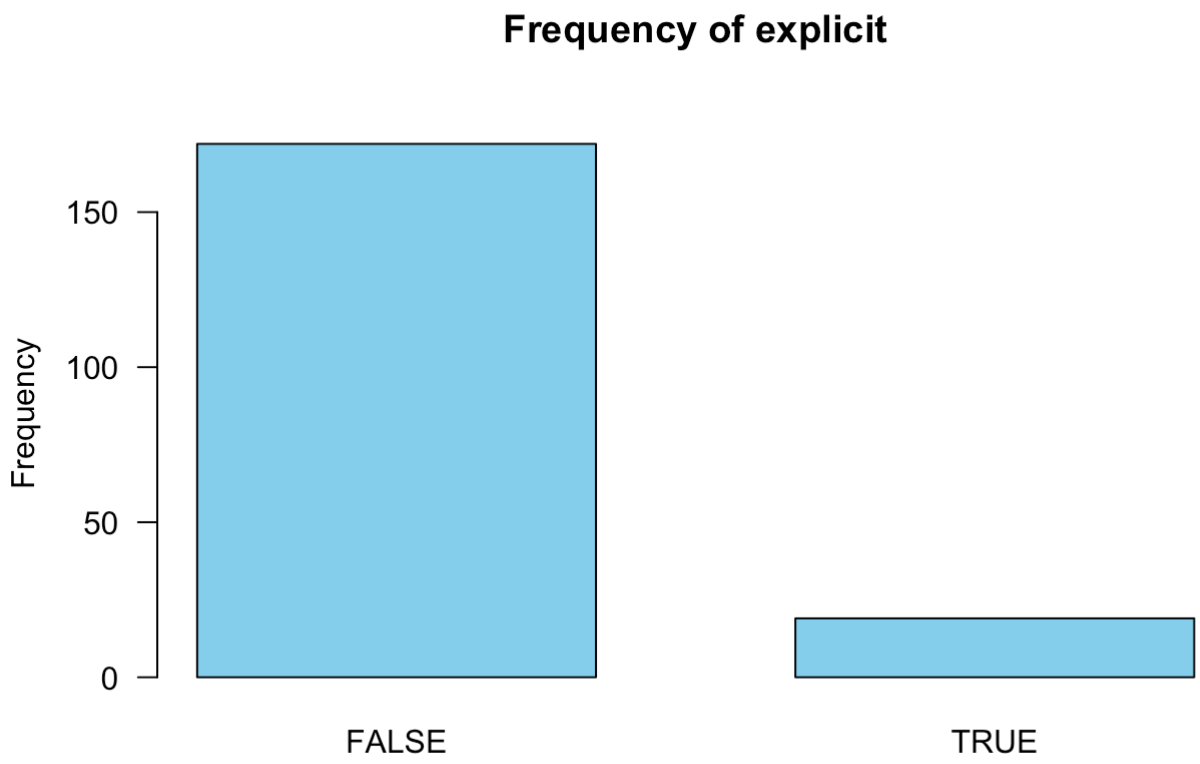


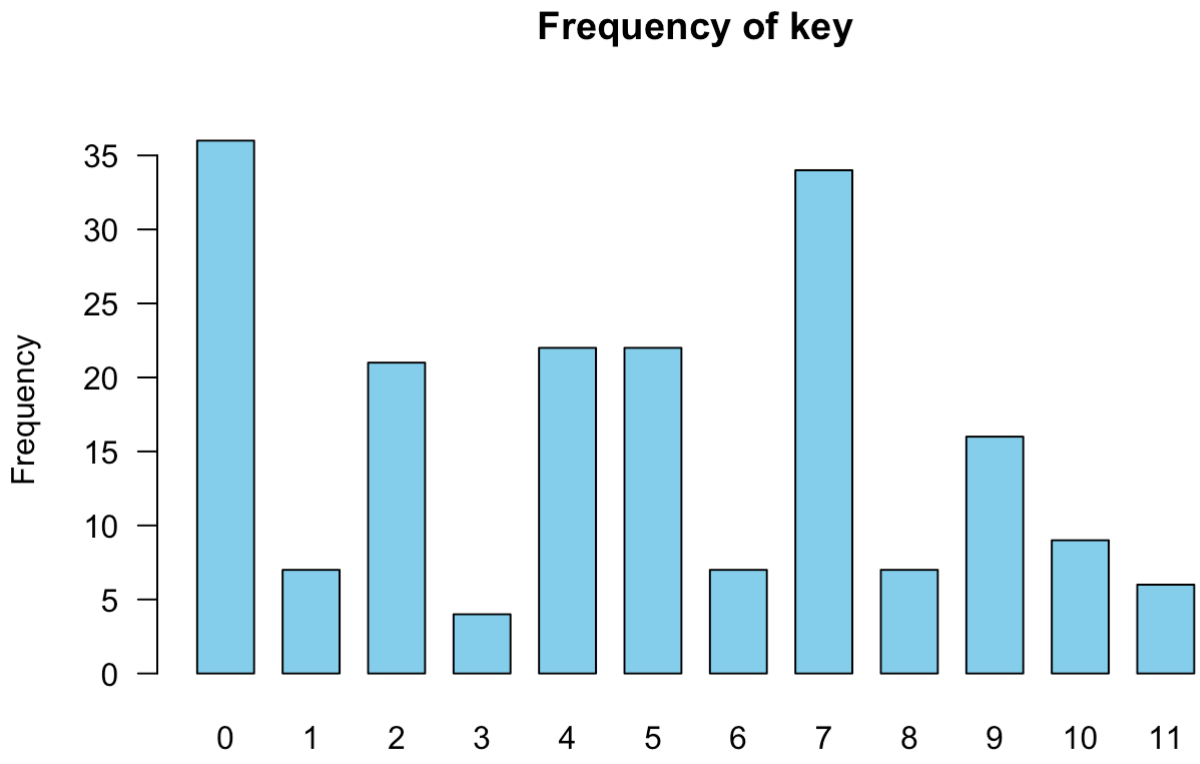


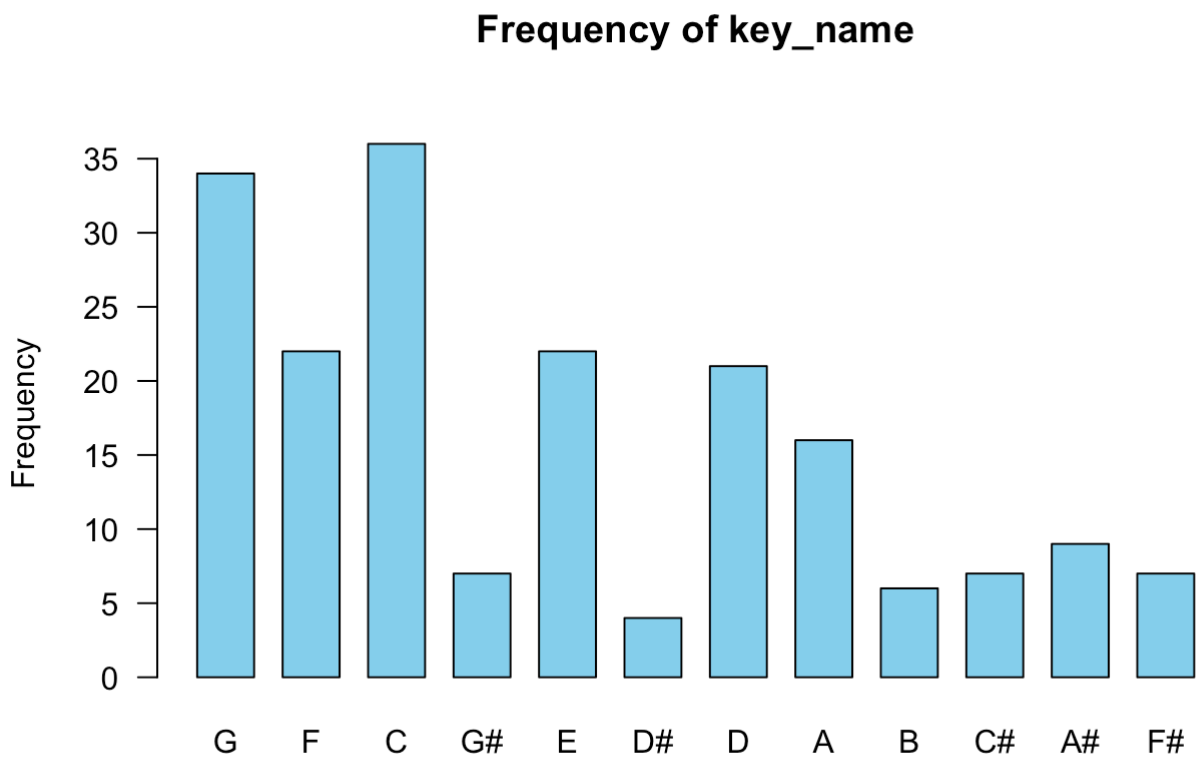


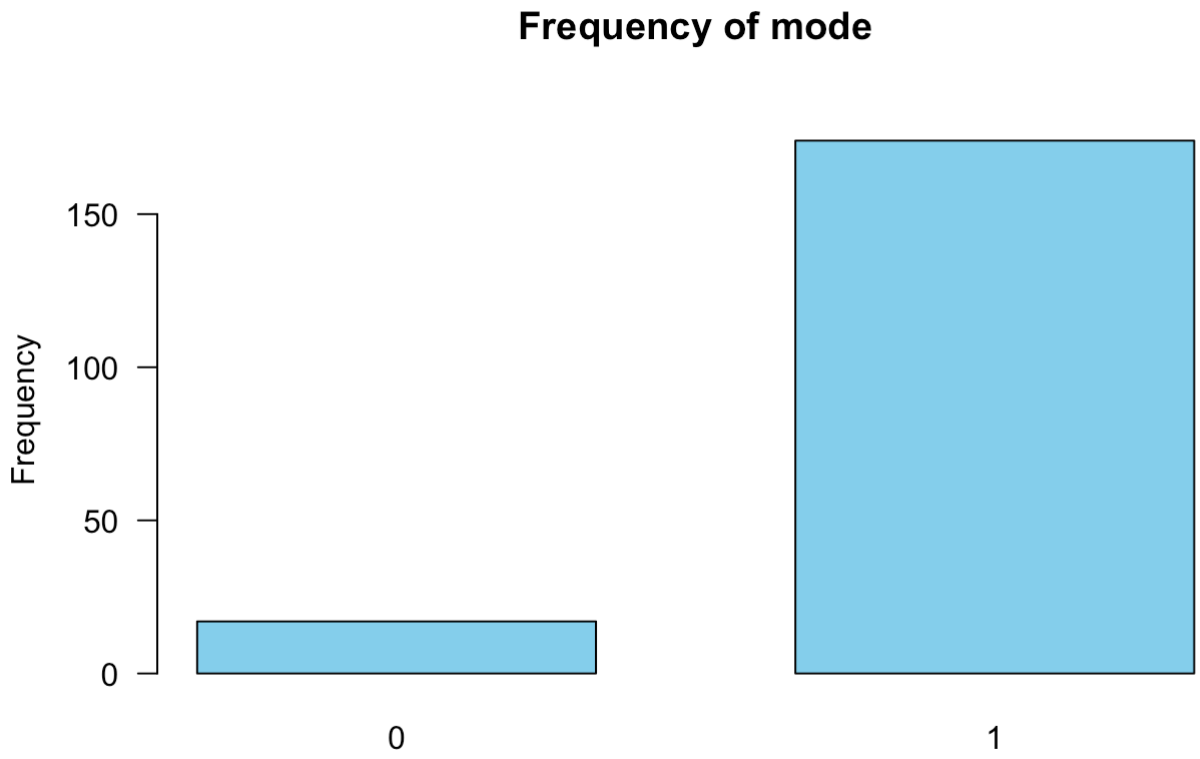
Bar plots for categorical variable distribution

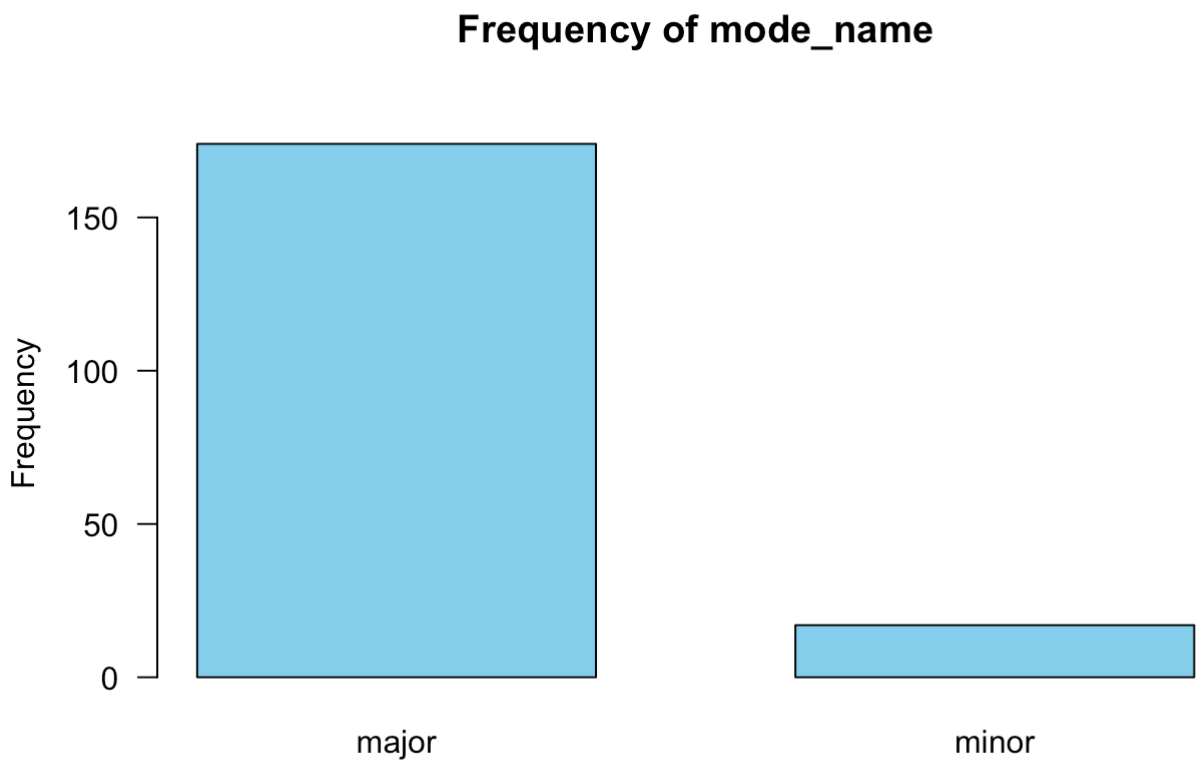
```
categorical_feats_df <- album_song_with_scores_df[categorical_feats]

categorical_feats_df <- na.omit(categorical_feats_df)


# Convert categorical variables to factors
categorical_feats_df <- lapply(categorical_feats_df, factor)

# Get frequency tables for each categorical variable
frequency_tables <- lapply(categorical_feats_df, table)

# Plot bar charts for each categorical variable
for (i in seq_along(frequency_tables)) {
  # Set up plotting area
  par(mar = c(5, 5, 6, 2))  # Adjust margins for x-axis label

  # Plot bar chart with wider spacing between categories and
  # horizontal x-axis labels
  barplot(frequency_tables[[i]],
          main = paste("Frequency of", names(frequency_tables)[i]),
          ylab = "Frequency",
          col = "skyblue",
          border = "black",
          space = 0.5,  # Adjust the spacing between bars
          las = ifelse(names(frequency_tables)[i] == "key_mode", 2, 1))
  # Rotate x-axis labels vertically if it's "key_mode"
}
```

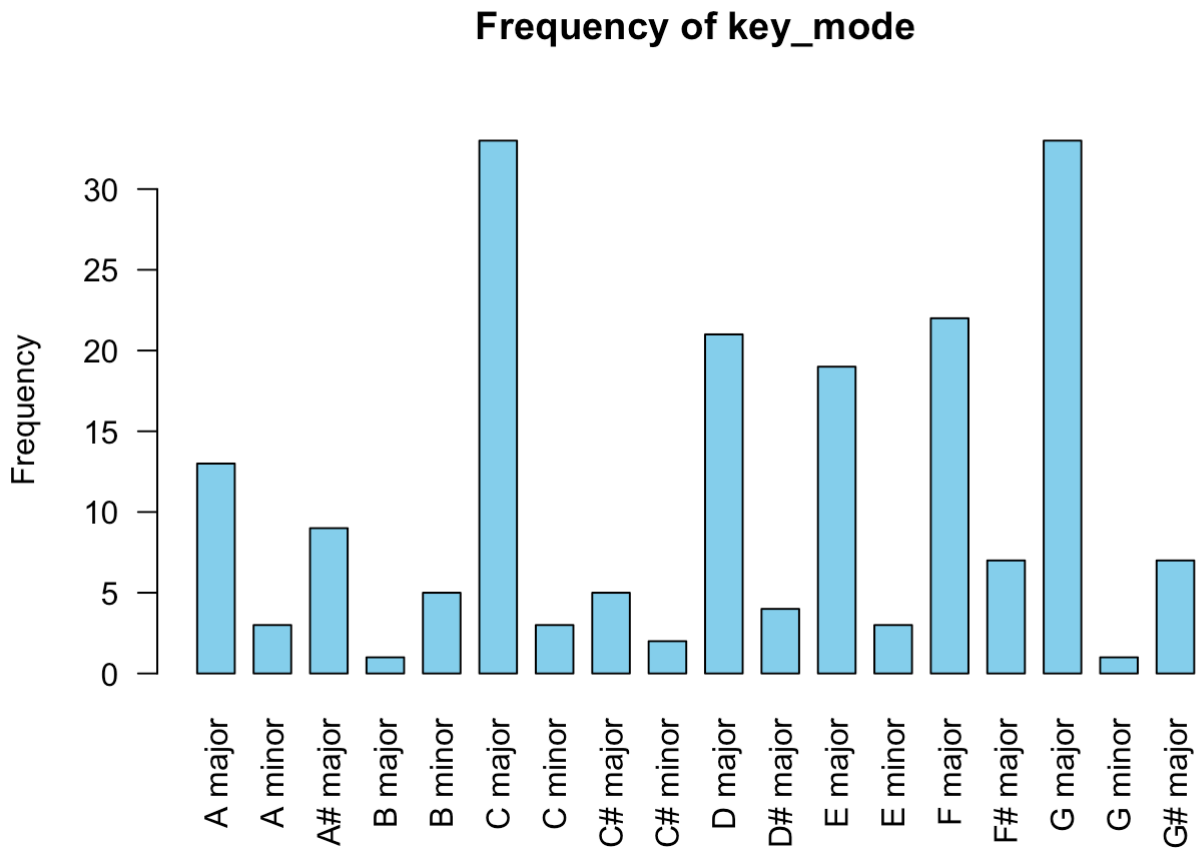

Frequency of explicit
Frequency
150
100
50
0
FALSE
TRUE


Frequency of key
Frequency
35
30
25
20
15
10
5
0
0
1
2
3
4
5
6
7
8
9
10
11


Frequency of key_name
Frequency
0
5
10
15
20
25
30
35
G
F
C
G#
E
D#
D
A
B
C#
A#
F#

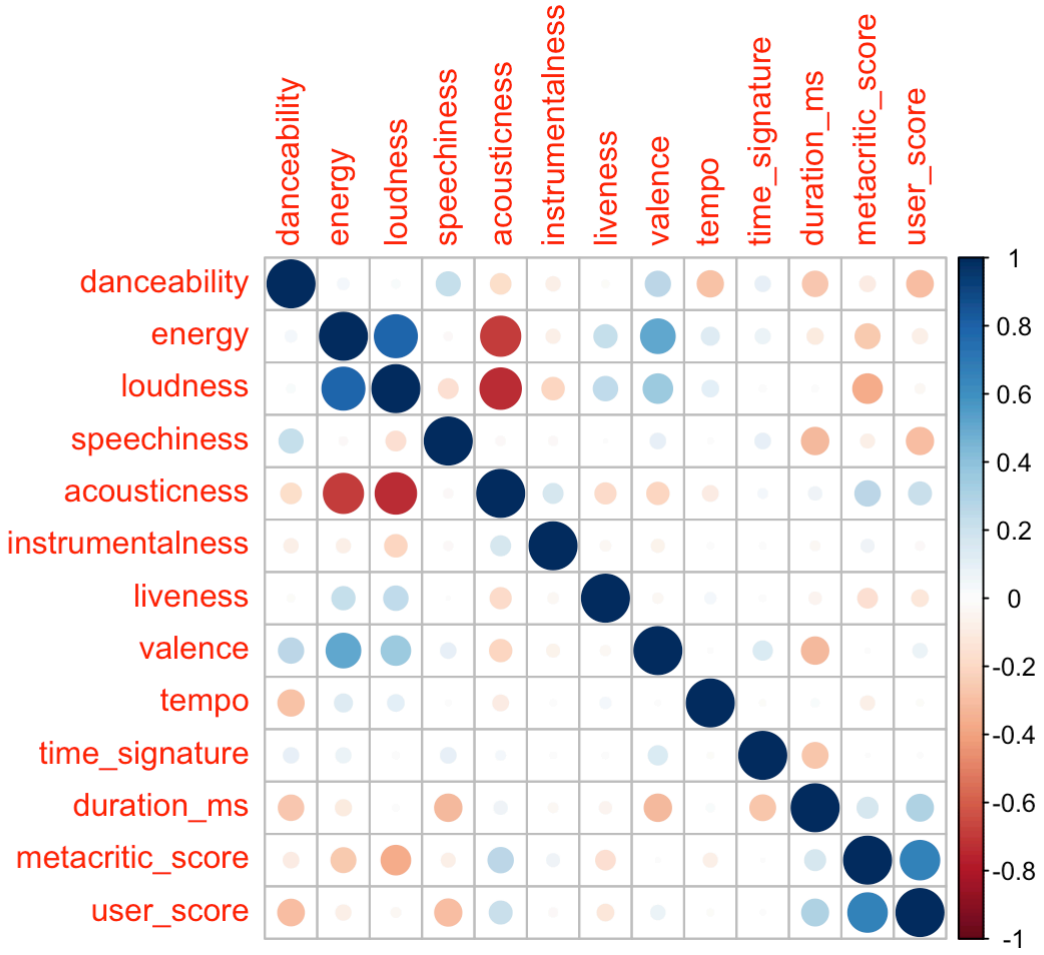

Frequency of mode
Frequency
0
50
100
150
0
1


Frequency of mode_name
Frequency
150
100
50
0
major
minor


Frequency of key_mode
Frequency
30
25
20
15
10
5
0
A major
A minor
A# major
B major
B minor
C major
C minor
C# major
C# minor
D major
D# major
E major
E minor
F major
F# major
G major
G minor
G# major

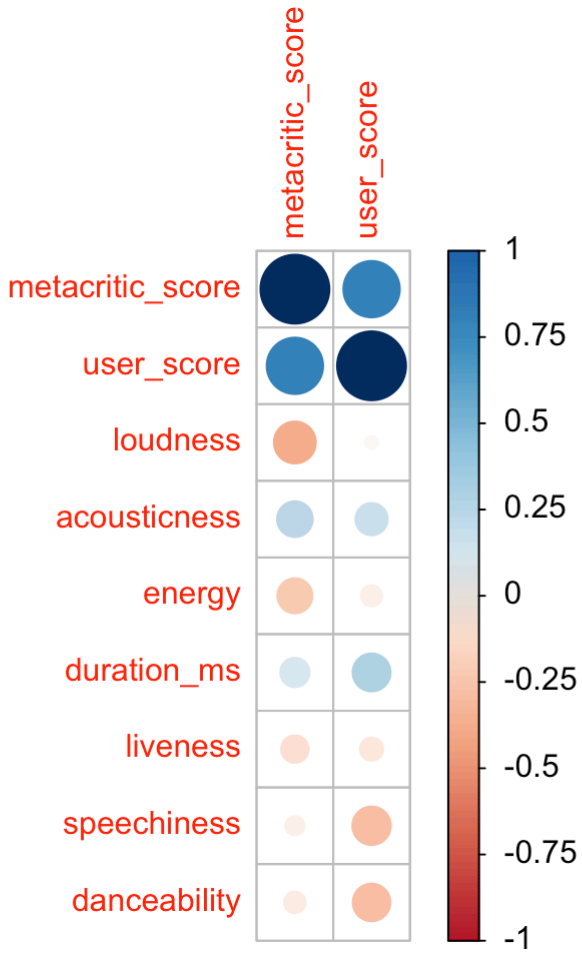
Correlation matrix for numerical variables

```
numerical_feats_and_scores_df <- album_song_with_scores_df[c("danceability",
                                                             "energy",
                                                             "loudness",
                                                             "speechiness",
                                                             "acousticness",
                                                             "instrumentalness",
                                                             "liveness",
                                                             "valence",
                                                             "tempo",
                                                             "time_signature",
                                                             "duration_ms",
                                                             "metacritic_score",
                                                             "user_score")]

numerical_feats_and_scores_df <- na.omit(numerical_feats_and_scores_df)

# Compute the correlation matrix
correlation_matrix <- cor(numerical_feats_and_scores_df)

# Visualize the correlation matrix using corrplot
corrplot(correlation_matrix, method = "circle")
```



Since a number of variables were showing negligible correlation to the scores, we decided to limit the set of features shown to the 5 most correlated to each score, totaling 7 after removing duplicates.

```
# Filter out the most correlated columns with respect to "metacritic_score"
metacritic_correlation <- abs(correlation_matrix["metacritic_score", ])
most_correlated_metacritic <- names(sort(metacritic_correlation,
                                         decreasing = TRUE)[2:7])
# Excluding self-correlation

# Filter out the most correlated columns with respect to "user_score"
user_correlation <- abs(correlation_matrix["user_score", ])
most_correlated_user <- names(sort(user_correlation, decreasing = TRUE)[2:7])

highly_correlated = unique(c(c("metacritic_score", "user_score"), most_correlated_met
acritic, most_correlated_user))
salient_numerical_feats_and_scores_df = numerical_feats_and_scores_df[highly_correlat
ed]

# Compute the correlation matrix
correlation_matrix <- cor(salient_numerical_feats_and_scores_df)[, c(1, 2)]

# Visualize the correlation matrix using corrplot
corrplot(correlation_matrix, method = "circle", cl.pos = "n")

colorlegend(xlim=c(3,4), ylim=c(0.5,9.5), colorRampPalette(brewer.pal(8, "RdBu"))(20
0), c(seq(-1,1,.25)), align="l", vertical=TRUE, addlabels=TRUE)
```



Discussion

A crucial insight is that songs in the major mode, typically associated with a cheerful and upbeat mood, consistently receive higher ratings from both users and Metacritic. In contrast, songs in the minor mode, often characterized by a more melancholic and somber tone, tend to score lower.

Surprisingly, songs with explicit lyrics also outperform those without. This is more evident in both scores, although the difference is much more pronounced looking at the Metacritic score.

In terms of keys, songs in C#, D# and F# tend to get the highest scores with little variance, while those in keys such as A, C and G often score lower and have more variance.

It should be noted that data imbalance is present regarding all three attributes mentioned above, as evidenced by the supplementing bar charts. As such, the above claims should not be taken at face value.

In terms of numerical features, there are no cases where the opinions of Metacritic reviewers and those of users oppose. However, each group places different weights on different attributes, positive or negative.

Certain attributes often lead to a lower score. Metacritic has a noticeable bias against loudness, energy and liveness compared to users; in particular, users are virtually indifferent towards loudness. On the other hand, users tend to score “speech” and “danceable” songs more harshly compared to Metacritic itself.

In terms of favourable attributes, both groups show a preference for longer and more acoustic songs. However, while Metacritic places more emphasis on acousticness, users are more interested in the duration of a song.

Question 2

Introduction

The question seeks to investigate the changes of Taylor Swift’s musical style throughout her career. Swift’s debut album in 2006 marked the beginning of a prolific career, with her ability to consistently produce hit songs keeping her in the spotlight until now. As she has matured both personally and professionally, it’s intriguing to explore how, if at all, her music has transformed alongside her and remained successful for almost 20 years. By examining and comparing audio features across her albums, we can identify potential trends and changes in her music. Particularly, we have identified valence, mode, and acousticness to show considerable discrepancies that reveal interesting insights into the change in her music style.

Approach

Our approach includes the use of three types of graph: (1) a relative frequency bar plot overlaid by (2) line chart, (3) a lollipop plot. These plots are suitable for highlighting differences in selected audio feature values across albums. Comparing between songs will be too fine-grained, so we compare across albums by aggregating individual song scores through counts or averages.

The relative frequency bar plot is chosen to compare the distribution of songs in minor versus major modes for each album. This plot type excels at showcasing proportional differences within categories, making it the most suitable choice for visualizing the balance between minor and major modes across albums. The valence, ranged from 0 to 1, is converted to a percentage to align with the relative frequency plot’s scale. Line graph is the best choice to show average valence because it is very simple (not overcrowding the relative frequency bar plot), yet effective at depicting trend over time.

For the second plot, a lollipop chart is utilized to compare average acousticness scores across albums. This chart type is particularly suited for highlighting variations in a single metric over different categories—in this case, albums. By plotting average acousticness as ‘lollipops,’ we can easily spot deviations and trends, and

including the average trend line further serves to segment Swift's albums into periods of higher and lower acousticness.

Code

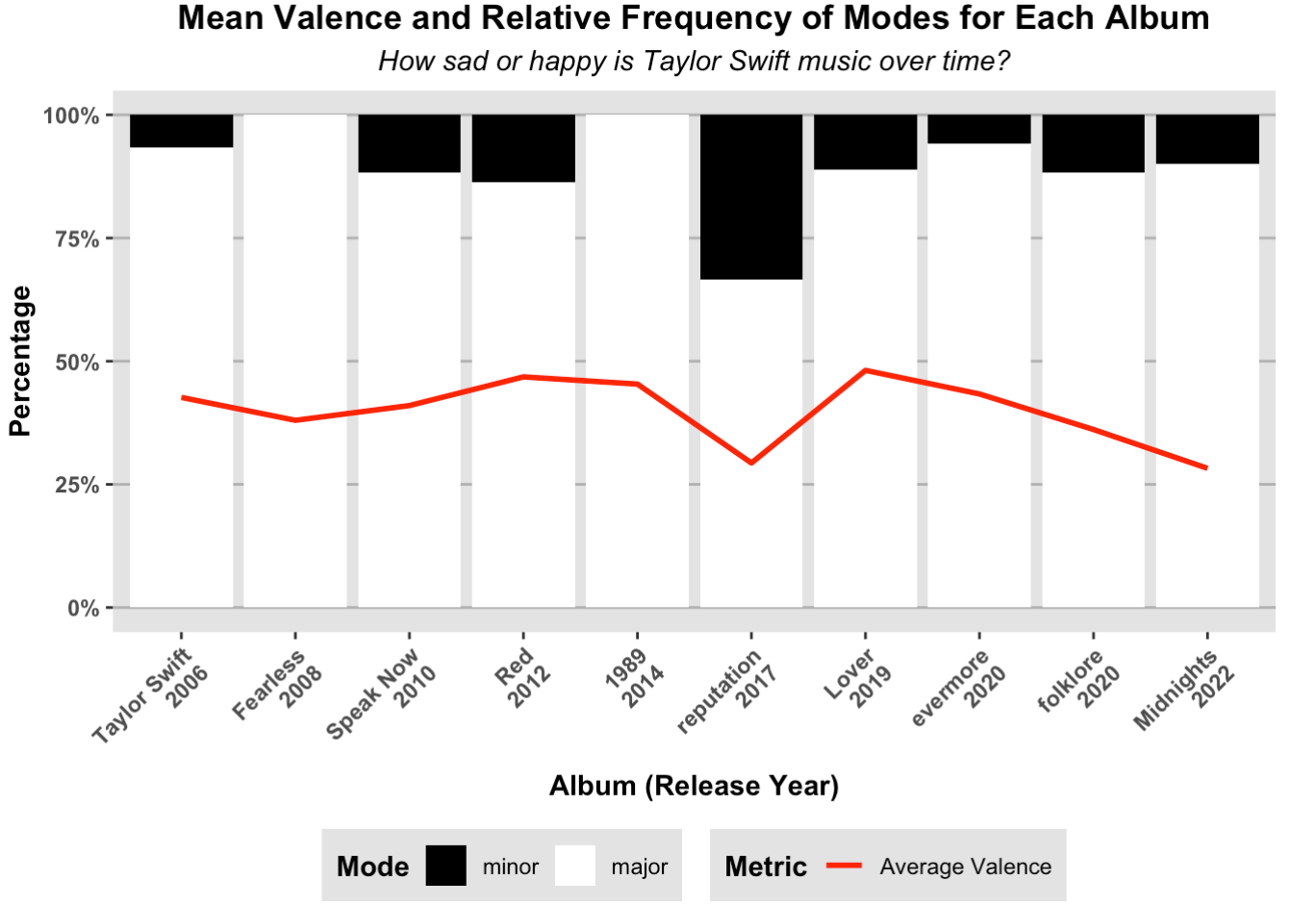
```
library(tidyverse)
library(dplyr)
library(ggplot2)
all_songs = read.csv("taylor_all_songs.csv")
album_df = read.csv("taylor_albums.csv")
#remove Extended Play (ep) and re-recorded albums (Taylor's Version). We will analyze
the remaining 10 main albums.
albums_filtered <- filter(album_df, ep==FALSE & !endsWith(album_name, "(Taylor's Vers
ion)"))
#filter songs not in albums_filtered
all_songs_filtered <- all_songs |>
  filter(album_name %in% albums_filtered$album_name)
# Filter out NA entries in mode_name and add a column for the album's release year
all_songs_filtered <- all_songs_filtered |>
  filter(!is.na(mode_name)) |>
  mutate(album_release_year = as.numeric(format(as.Date(album_release), "%Y")))

#theme for 1st chart
theme_piano <- function() {
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, face = "italic"),
    panel.grid.major.y = element_line(color = "grey70"),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank(),
    legend.position = "bottom",
    legend.background = element_rect(fill = "grey90"),
    legend.title = element_text(face = "bold"),
    axis.text = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text.x = element_text(angle = 45, hjust = 1),
    panel.background = element_rect(fill="grey90")
  )
}

#theme for 2nd chart
theme_clean <- function() {
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    panel.grid.major.y = element_line(color = "grey60",linewidth = 0.2, linetype = "d
otted"),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank(),
    legend.position = "bottom",
    axis.text = element_text(face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text.x = element_text(angle = 45, hjust = 1),
    panel.background = element_rect(fill="grey90")
  )
}
```

```
# Group by album, calculate total number of songs and count of each mode
album_modes_rf <- all_songs_filtered |>
  # total number of songs in album
  group_by(album_name, album_release) |>
  mutate(total_count = n()) |>
  # mode counts
  group_by(album_name, album_release, mode_name, total_count) |>
  summarise(mode_count = n()) |>
  # relative freq of mode
  mutate(relative_freq = mode_count / total_count) |>
  ungroup()
```

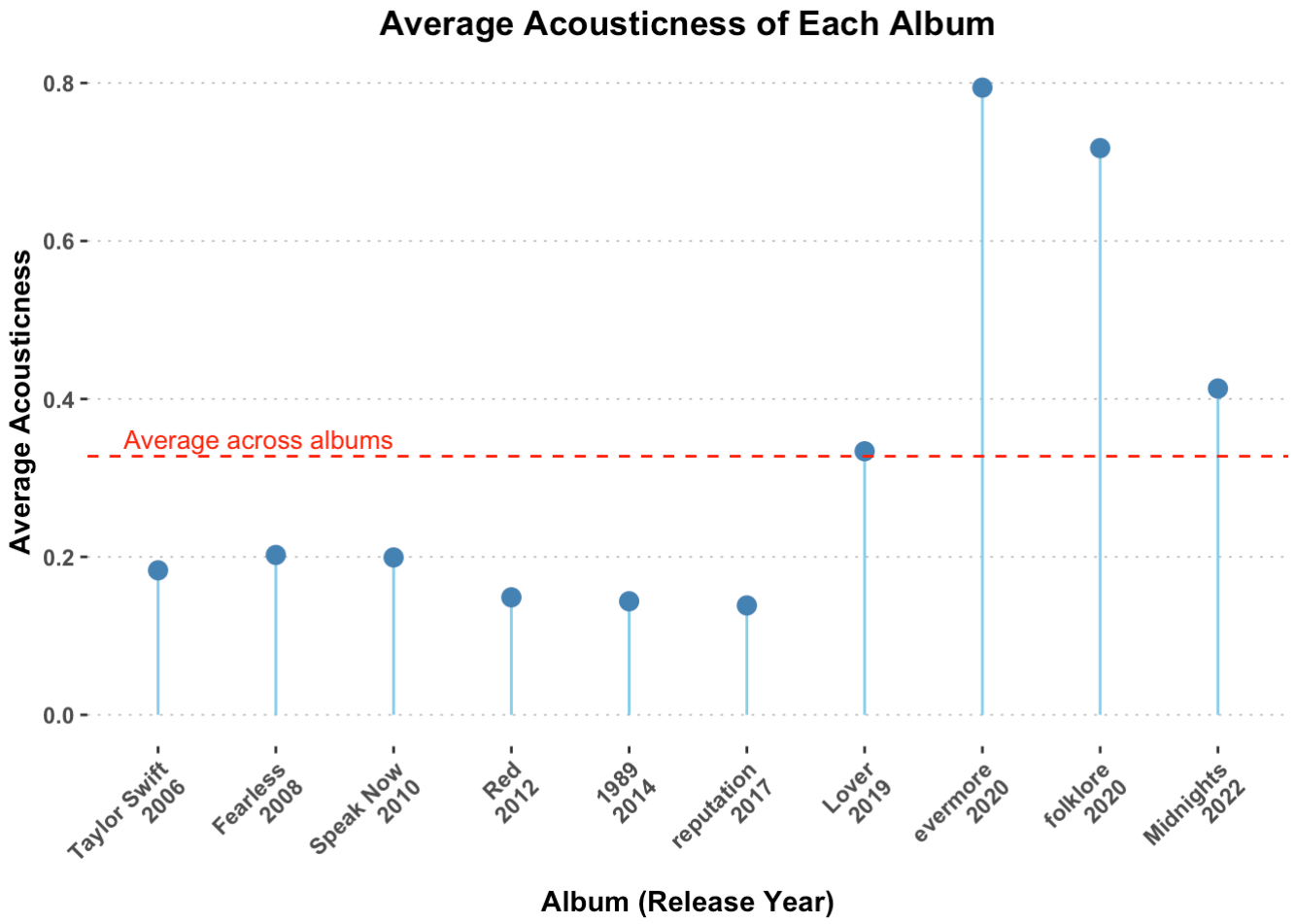
```
## `summarise()` has grouped output by 'album_name', 'album_release', 'mode_name'.
## You can override using the `.groups` argument.
```

```
#major below, minor above
album_modes_rf$mode_name <- factor(album_modes_rf$mode_name, levels = c("minor", "maj
or"))
# Calculate mean valence for each album
mean_valence_by_album <- all_songs_filtered |>
  group_by(album_name, album_release) |>
  summarise(mean_valence = mean(valence, na.rm = TRUE), .groups = 'drop')

# Join the mean valence data back with the mode frequency data
album_data_combined <- album_modes_rf |>
  left_join(mean_valence_by_album, by = c("album_name", "album_release")) |>
  arrange(album_release) |>
  mutate(album_release_year = as.numeric(format(as.Date(album_release), "%Y")))

# draw
ggplot(album_data_combined, aes(x=reorder(album_name, album_release_year))) +
  geom_bar(aes(fill=mode_name, y=relative_freq), stat="identity", position="fill") +
  geom_line(aes(y=mean_valence, group=1, color="Average Valence"), size=1) +
  # geom_point(aes(y=mean_valence), color="red", size=1.5) +
  scale_y_continuous(labels = scales::percent_format()) +
  theme_piano() +
  scale_fill_manual(values=c("major" = "white", "minor" = "black"), name="Mode") +
  scale_color_manual(values=c("Average Valence" = "red"), name="Metric", labels=c("Av
erage Valence"))+
  labs(x="Album (Release Year)", y="Percentage", fill="Mode",
       title="Mean Valence and Relative Frequency of Modes for Each Album",
       subtitle = "How sad or happy is Taylor Swift music over time?") +

  scale_x_discrete(labels=function(x) paste(x, album_data_combined$album_release_year
[match(x, album_data_combined$album_name)], sep="\n"))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# calculate the average acousticness for each album
average_acousticness <- all_songs_filtered |>
  filter(!is.na(acousticness)) |>
  group_by(album_name, album_release_year) |>
  summarise(average_acousticness = mean(acousticness, na.rm = TRUE), .groups = 'dro
p') |>
  arrange(album_release_year, album_name)

overall_average = mean(average_acousticness$average_acousticness, na.rm = TRUE)
# draw
ggplot(average_acousticness, aes(x=reorder(album_name, album_release_year), y=average
_acousticness)) +
  geom_segment(aes(xend=reorder(album_name, album_release_year), yend=0), color="skyb
lue") +
  geom_point(color="steelblue", size=3) + # Draws the dots
  geom_hline(yintercept = overall_average, linetype="dashed", color="red", size=0.5)
+  #trendline
  theme_clean() +
  theme(panel.background = element_rect(fill="white"))+
  annotate("text", x=3, y=overall_average+0.01, label="Average across albums", hjust=
1, vjust=0, color="red", size=3.5) +
  labs(x="Album (Release Year)", y="Average Acousticness",
       title="Average Acousticness of Each Album") +
  scale_x_discrete(labels=function(x) paste(x, average_acousticness$album_release_yea
r[match(x, average_acousticness$album_name)], sep="\n"))
```



Discussion

The first graph predominately shows Swift's preference for major modes in her songs, a common characteristic associated with a brighter sound. Despite this predominance of major mode usage, the valence (musical positiveness) scores are moderately low, with none surpassing the 50% mark. Notably, her album "Reputation" in 2017 diverges from this pattern, showing a consider increase in minor mode tracks commonly linked to sadder tone. This shift corresponds with a noticeable dip in the album's average valence score. Both these elements highlight that "Reputation" is Taylor Swift's saddest album.

In examining acousticness, a marked transformation post-'Reputation' is evident. Albums preceding this, including 'Reputation' itself, have an acousticness score not exceeding 0.2. In stark contrast, the albums released after 'Reputation' exhibit significantly elevated acousticness levels, with 'Evermore' showcasing a fourfold increase over the pre-'Reputation' threshold. This surge in acousticness has led to an upward adjustment of the overall average trend line, now exceeding 0.3.

In speculating the catalyst behind these observable changes, one might hypothesize the influence of external events on Swift's artistic direction. The phone call scandal with Kanye West in 2016, which cast a shadow over Swift's *reputation*, could highly likely be linked to the darker tones in 'Reputation' in 2017. The subsequent stylistic pivot towards a more acoustic or authentic sound may reflect an emotional response to the preceding turmoil, signaling a new chapter in Swift's musical narrative.