Final Project

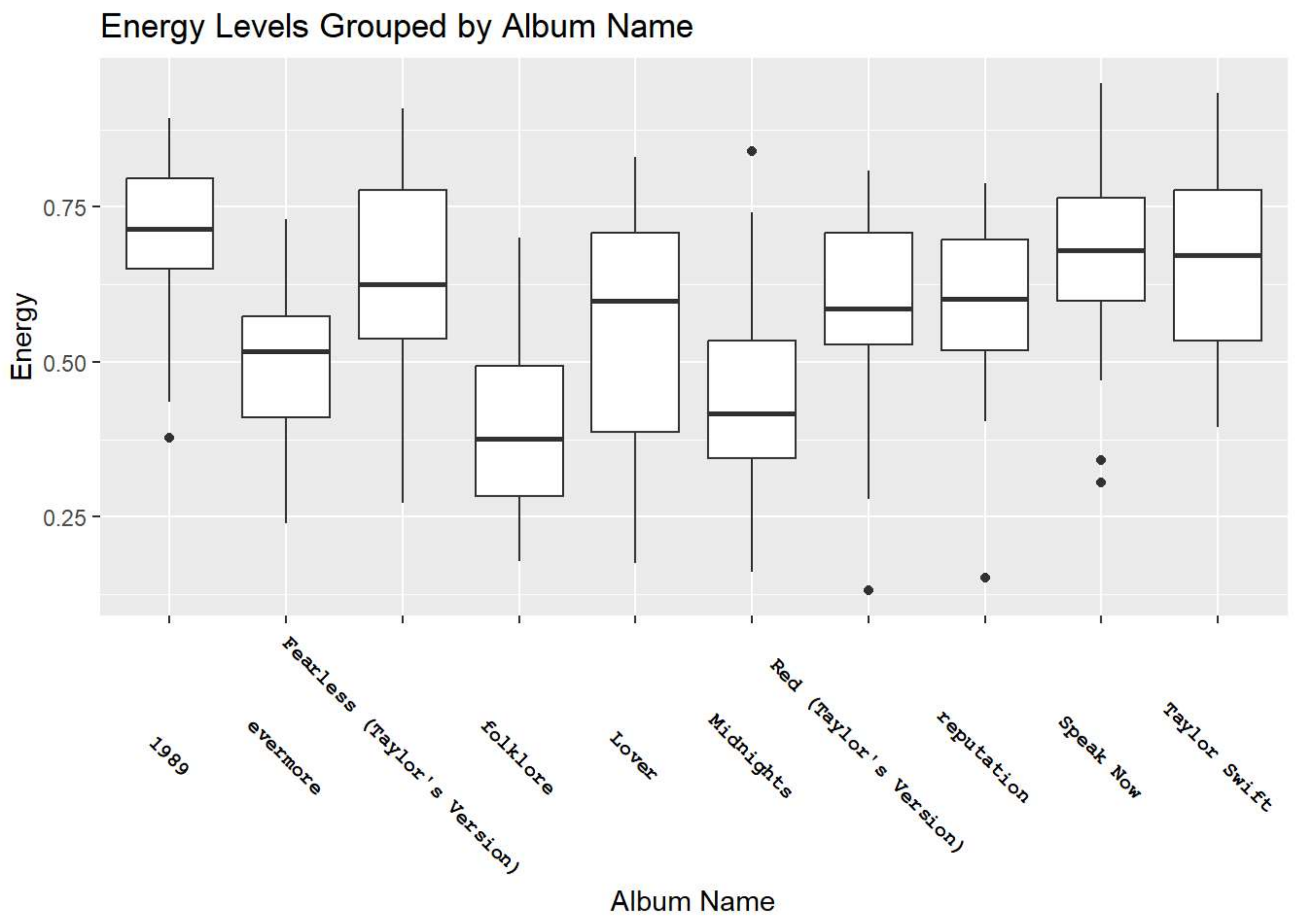
Muhammad Salman Al Farisi

2024-11-04

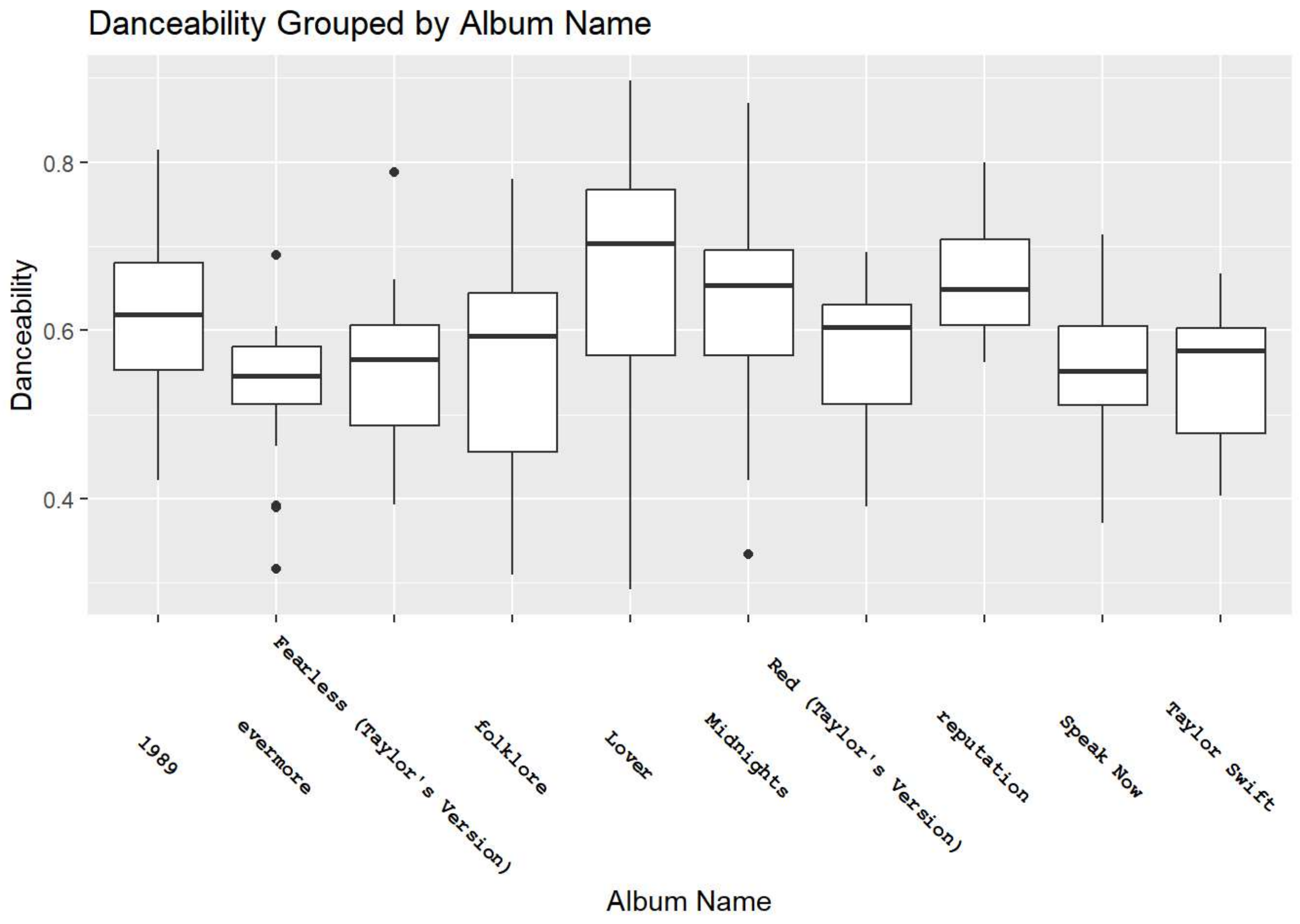
```
library(tidyverse)
library(stringr)
library(lubridate)
library(readxl)
library(ggplot2)
library(dplyr)
library(tidyr)
library(viridis)
library(gridExtra)
library(reshape2)
library(ggridges)
```

```
taylor_album_songs <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytue
sday/master/data/2023/2023-10-17/taylor_album_songs.csv')
taylor_all_songs <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesd
ay/master/data/2023/2023-10-17/taylor_all_songs.csv')
taylor_albums <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/
master/data/2023/2023-10-17/taylor_albums.csv')
```

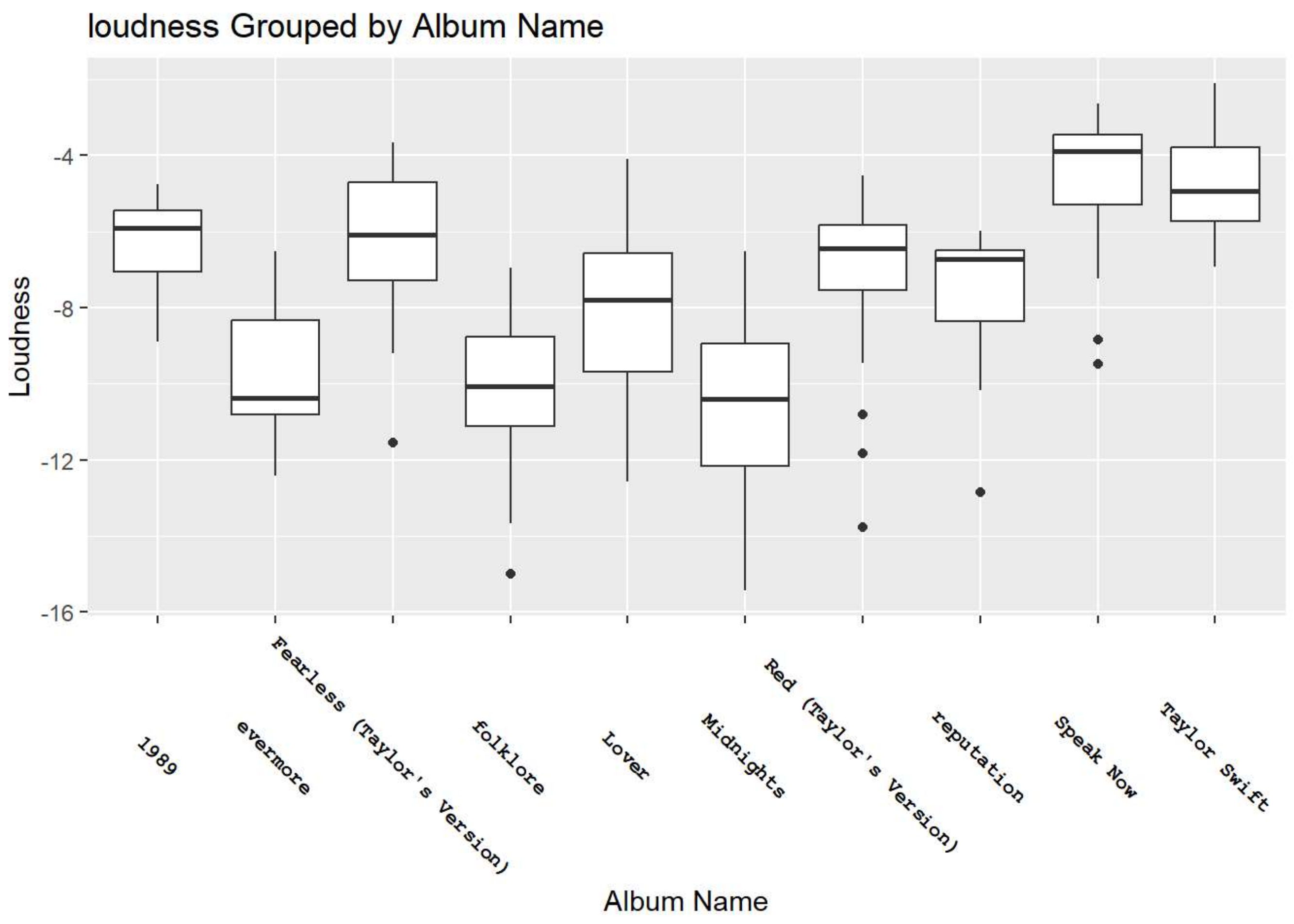
```
data = taylor_album_songs

ggplot(data, aes(x = album_name, y = energy)) +
  geom_boxplot() +
  labs(title = "Energy Levels Grouped by Album Name",
       x = "Album Name",
       y = "Energy") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



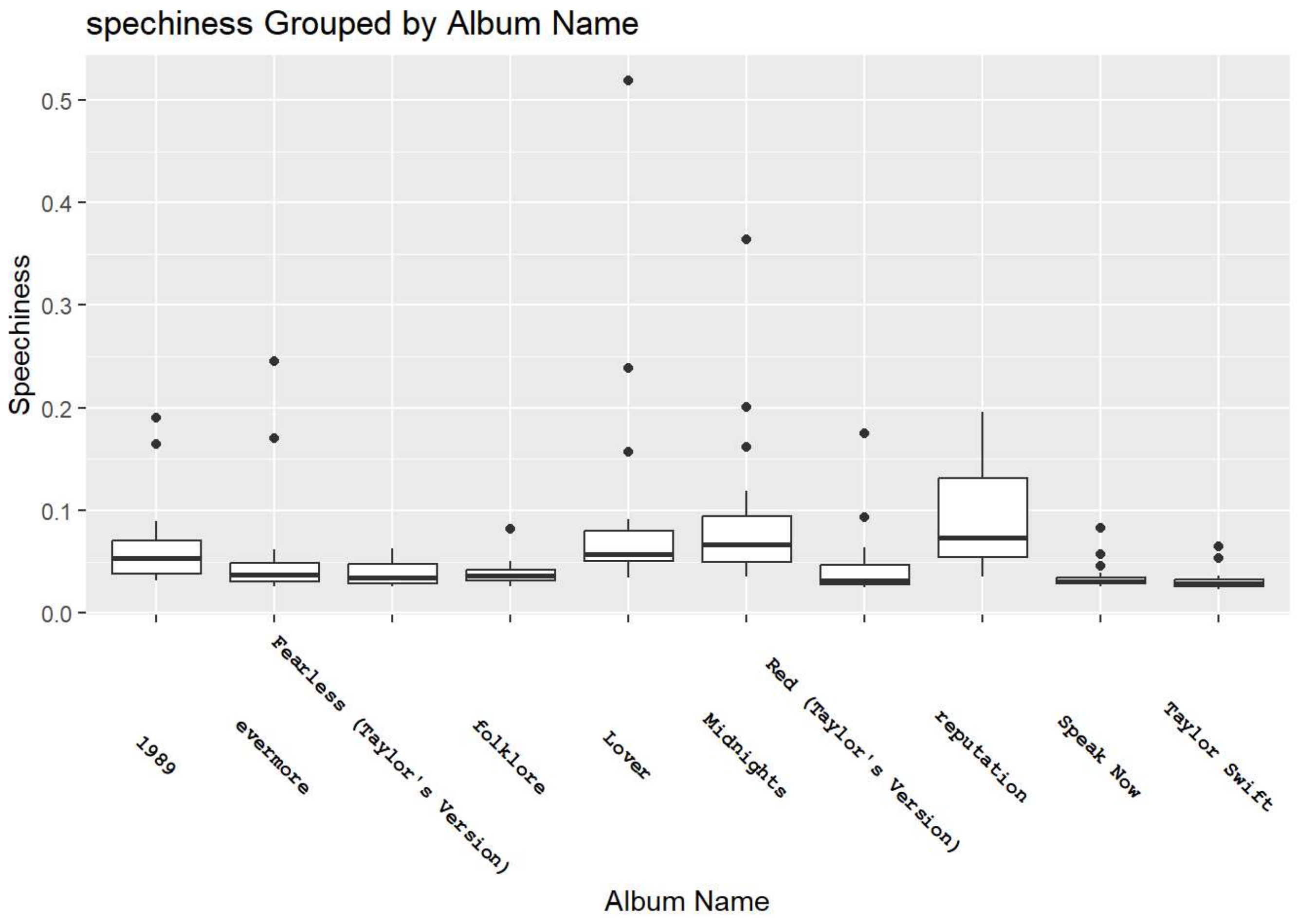
```
ggplot(data, aes(x = album_name, y = danceability)) +
  geom_boxplot() +
  labs(title = "Danceability Grouped by Album Name",
       x = "Album Name",
       y = "Danceability") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



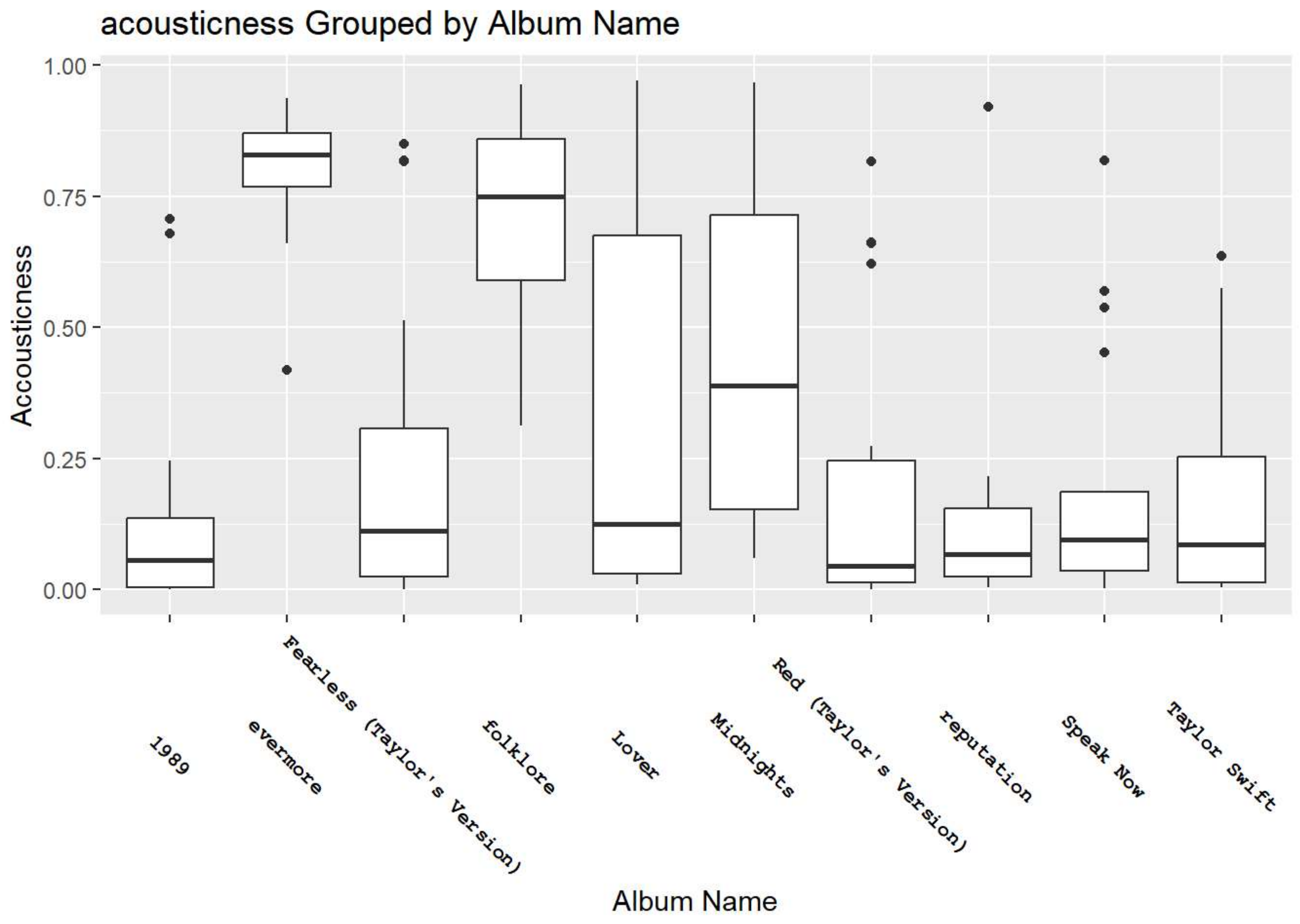
```
ggplot(data, aes(x = album_name, y = loudness)) +
  geom_boxplot() +
  labs(title = "loudness Grouped by Album Name",
       x = "Album Name",
       y = "Loudness") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



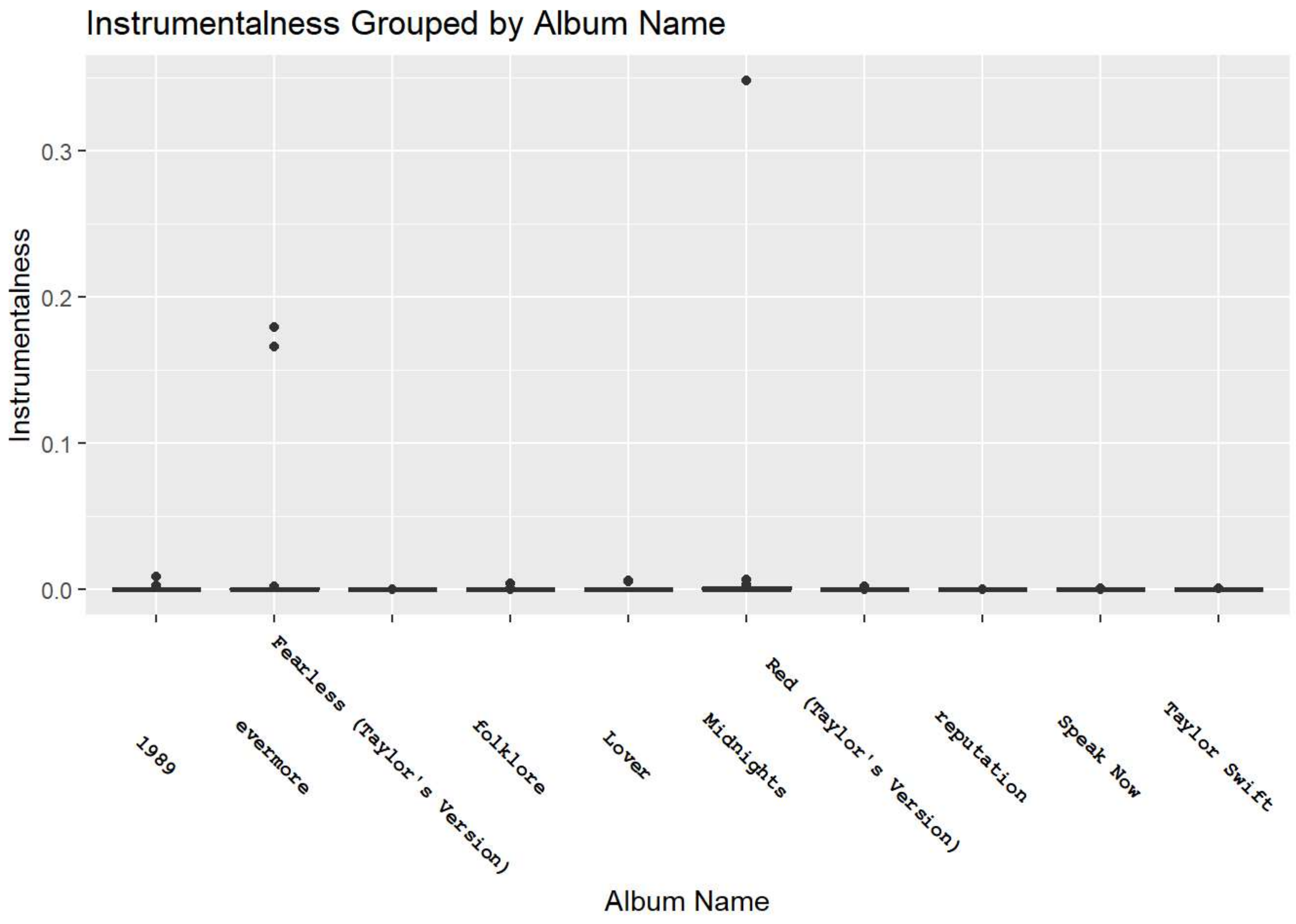
```
ggplot(data, aes(x = album_name, y = speechiness)) +
  geom_boxplot() +
  labs(title = "spechiness Grouped by Album Name",
       x = "Album Name",
       y = "Speechiness") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



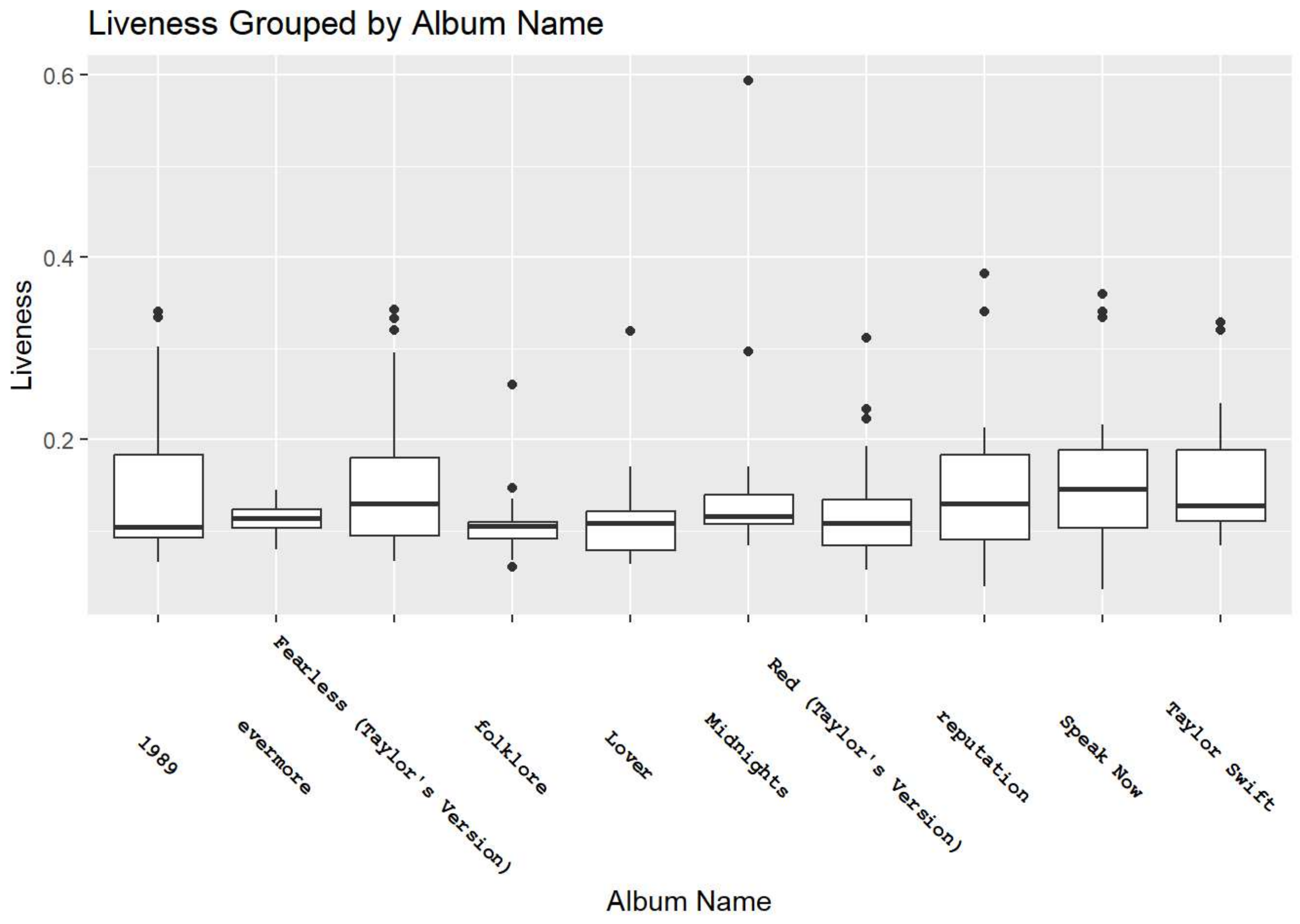
```
ggplot(data, aes(x = album_name, y = acousticness)) +
  geom_boxplot() +
  labs(title = "acousticness Grouped by Album Name",
       x = "Album Name",
       y = "Accousticness") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



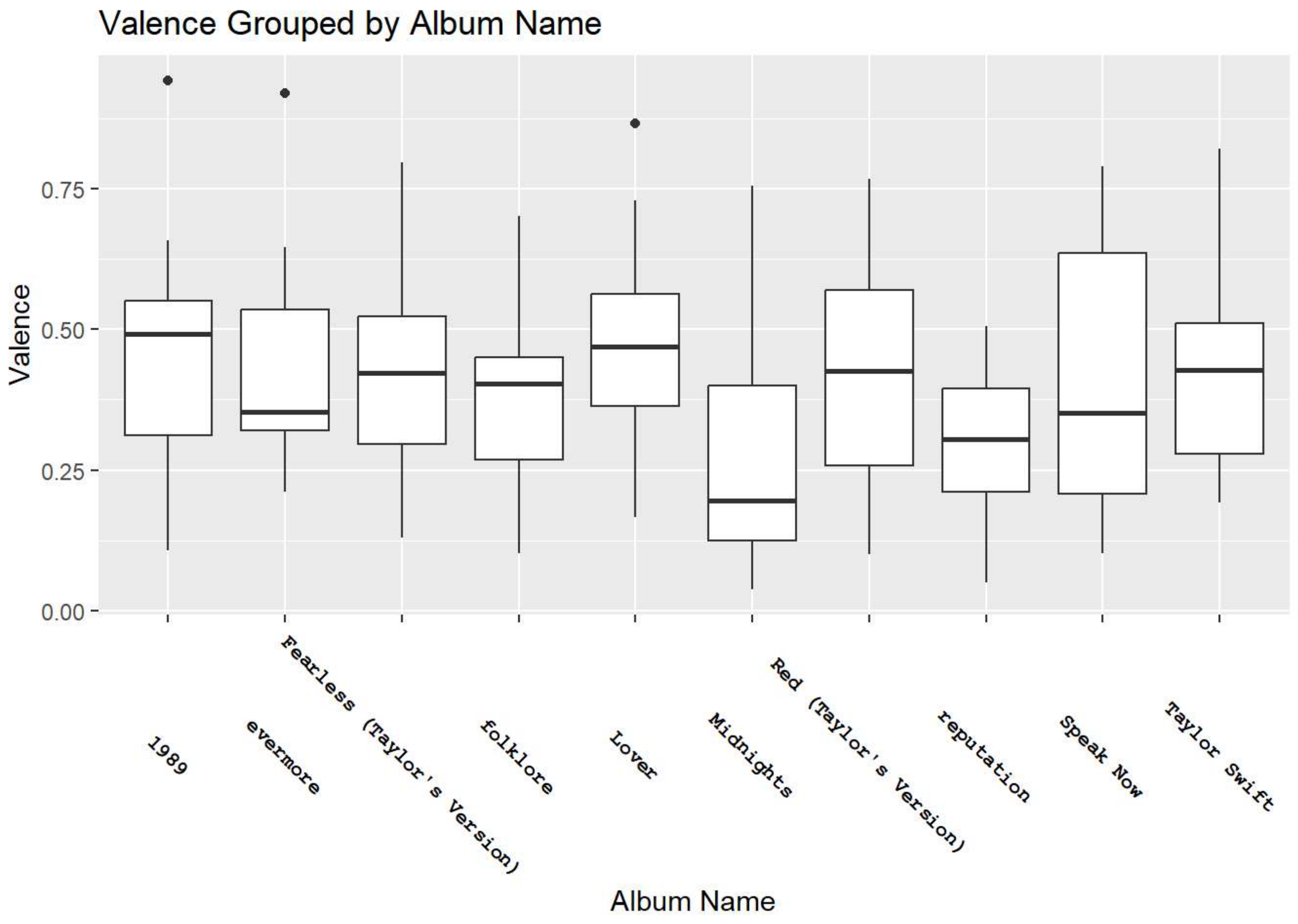
```
ggplot(data, aes(x = album_name, y = instrumentalness)) +
  geom_boxplot() +
  labs(title = "Instrumentalness Grouped by Album Name",
       x = "Album Name",
       y = "Instrumentalness") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



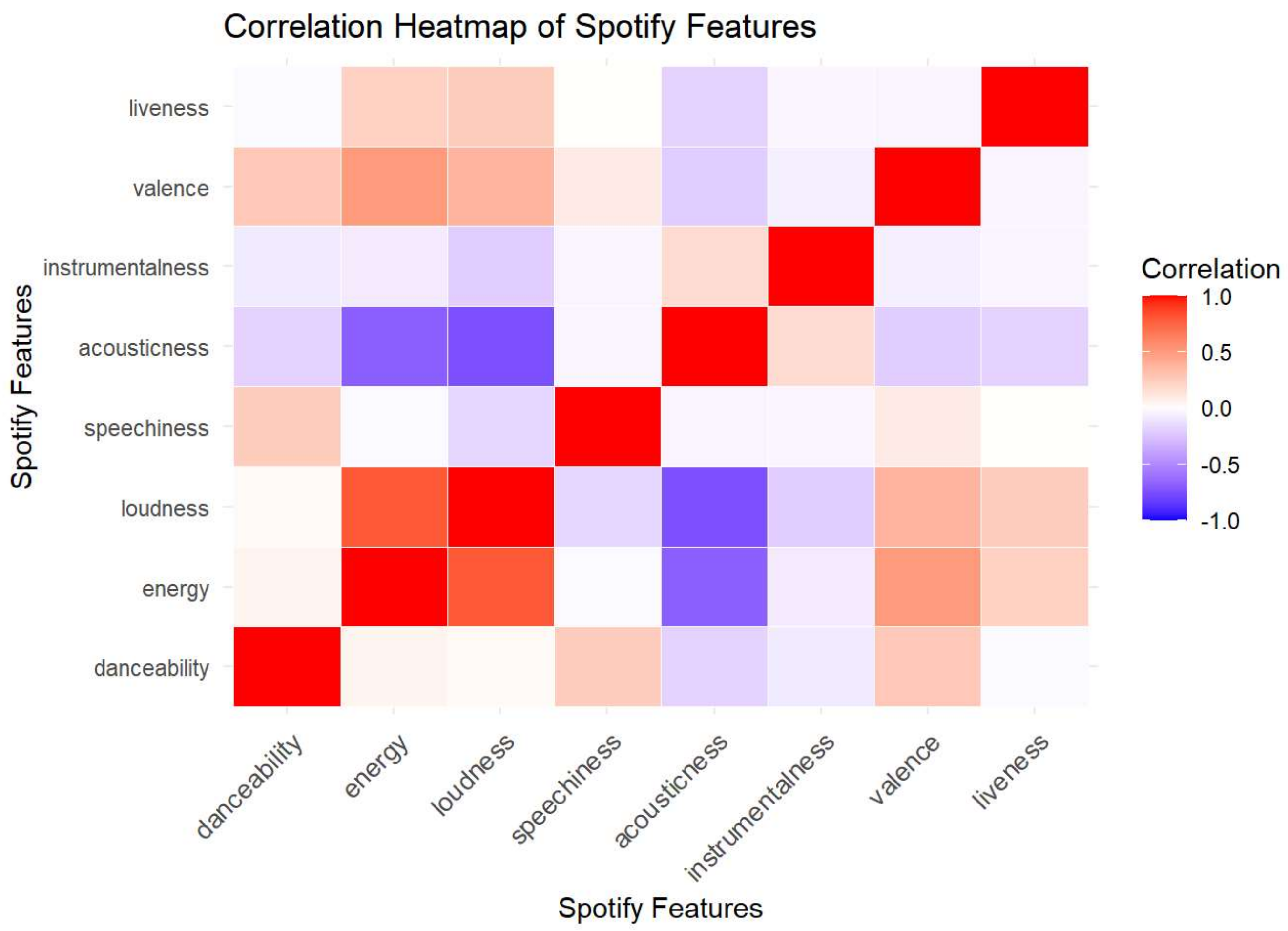
```
ggplot(data, aes(x = album_name, y = liveness)) +
  geom_boxplot() +
  labs(title = "Liveness Grouped by Album Name",
       x = "Album Name",
       y = "Liveness") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



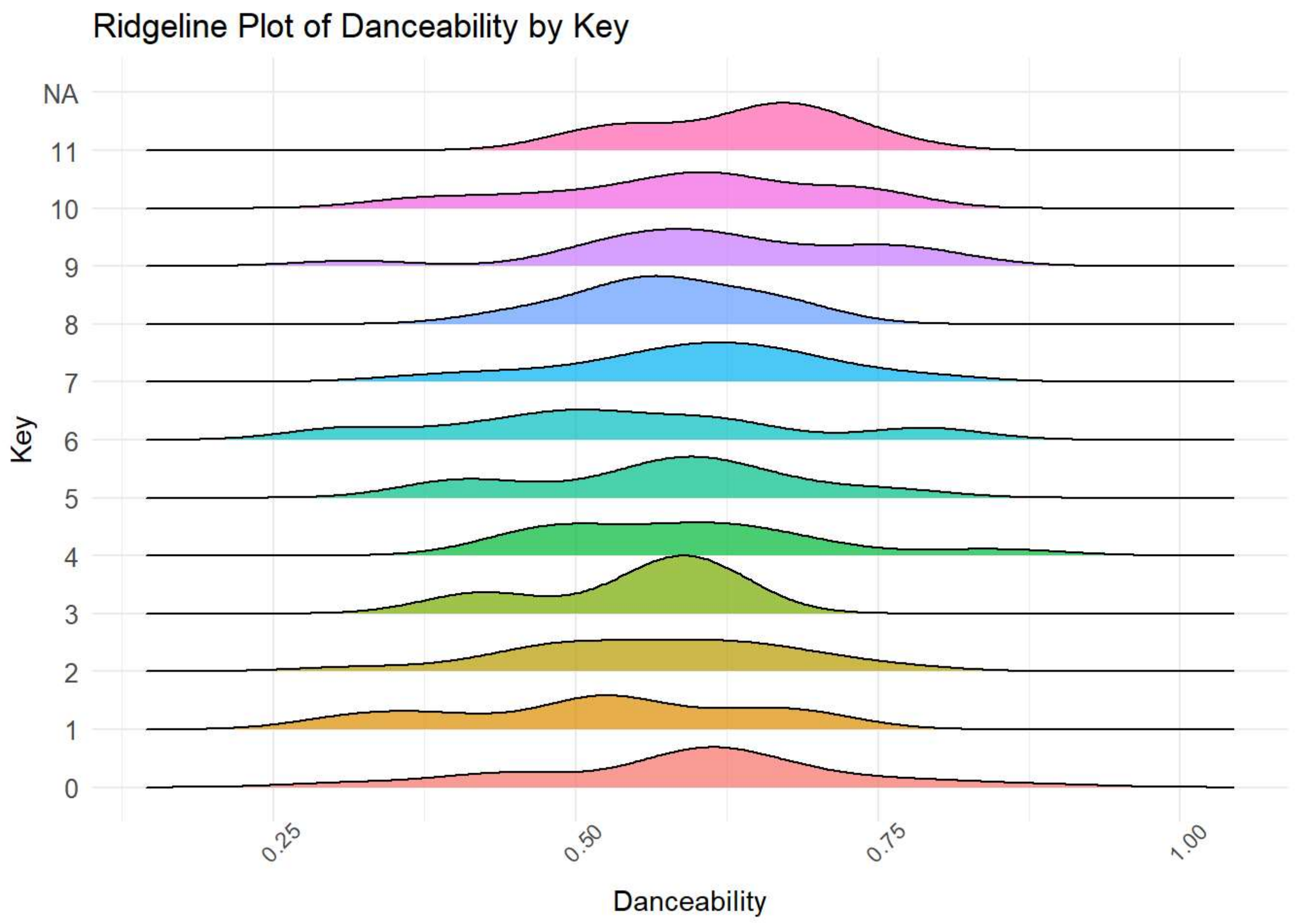
```
ggplot(data, aes(x = album_name, y = valence)) +
  geom_boxplot() +
  labs(title = "Valence Grouped by Album Name",
       x = "Album Name",
       y = "Valence") +
  theme(axis.text.x = element_text(angle = -45, vjust = 0.5))+
  theme(axis.text.x = element_text(size = 8, family = "mono", face = "bold", color = "black"))
```



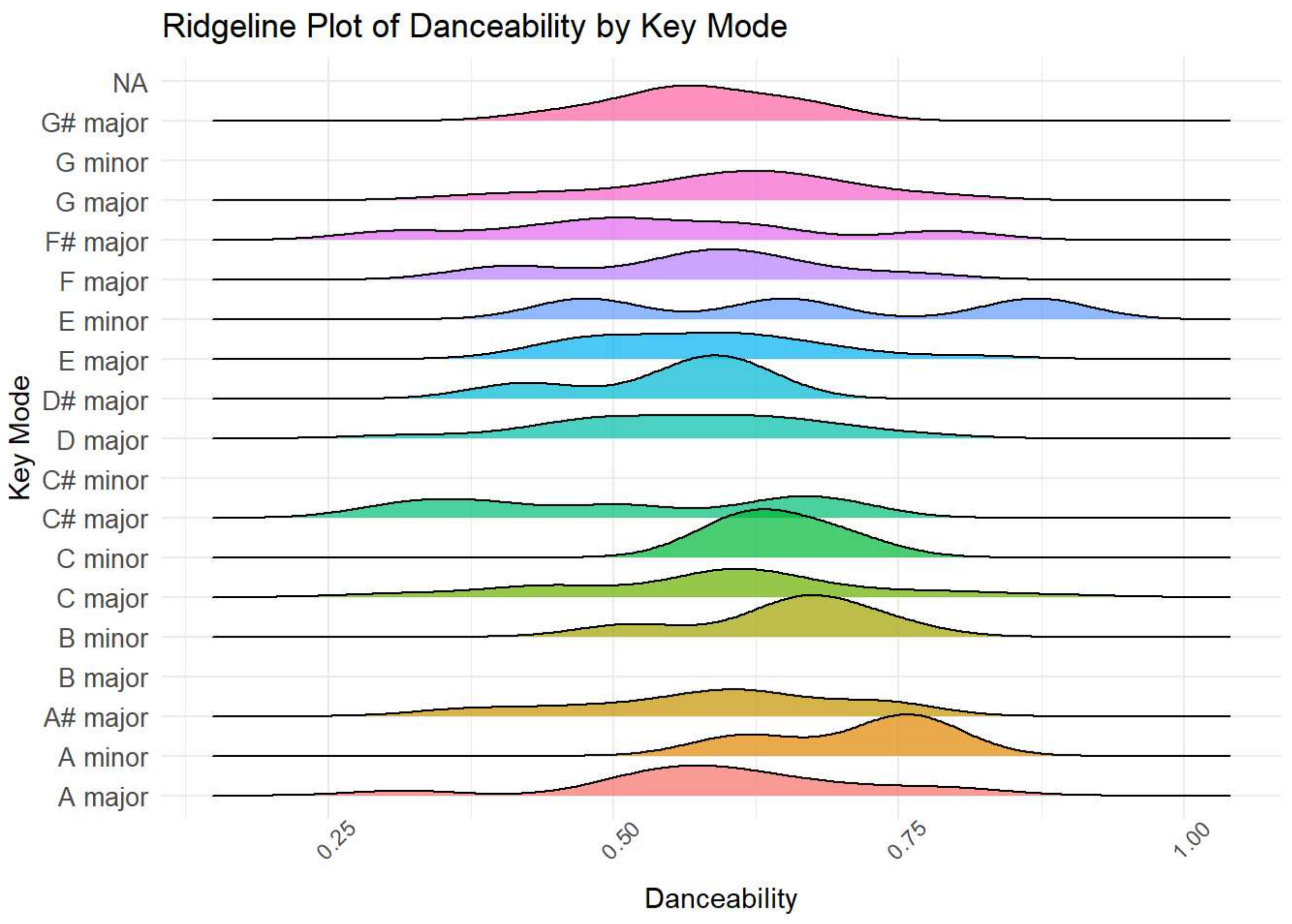
```
# Load your dataset
data = taylor_album_songs

# Select only the Spotify features of interest for correlation
spotify_features <- data[, c("danceability", "energy", "loudness", "speechiness",
                             "acousticness", "instrumentalness", "valence", "liveness")]

# Calculate the correlation matrix
cor_matrix <- cor(spotify_features, use = "complete.obs")

# Melt the correlation matrix for ggplot2
melted_cor <- melt(cor_matrix)

# Create the heatmap using ggplot2
ggplot(melted_cor, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name = "Correlation") +
  theme_minimal() +
  labs(title = "Correlation Heatmap of Spotify Features",
       x = "Spotify Features",
       y = "Spotify Features") +
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 10, hjust = 1))
```



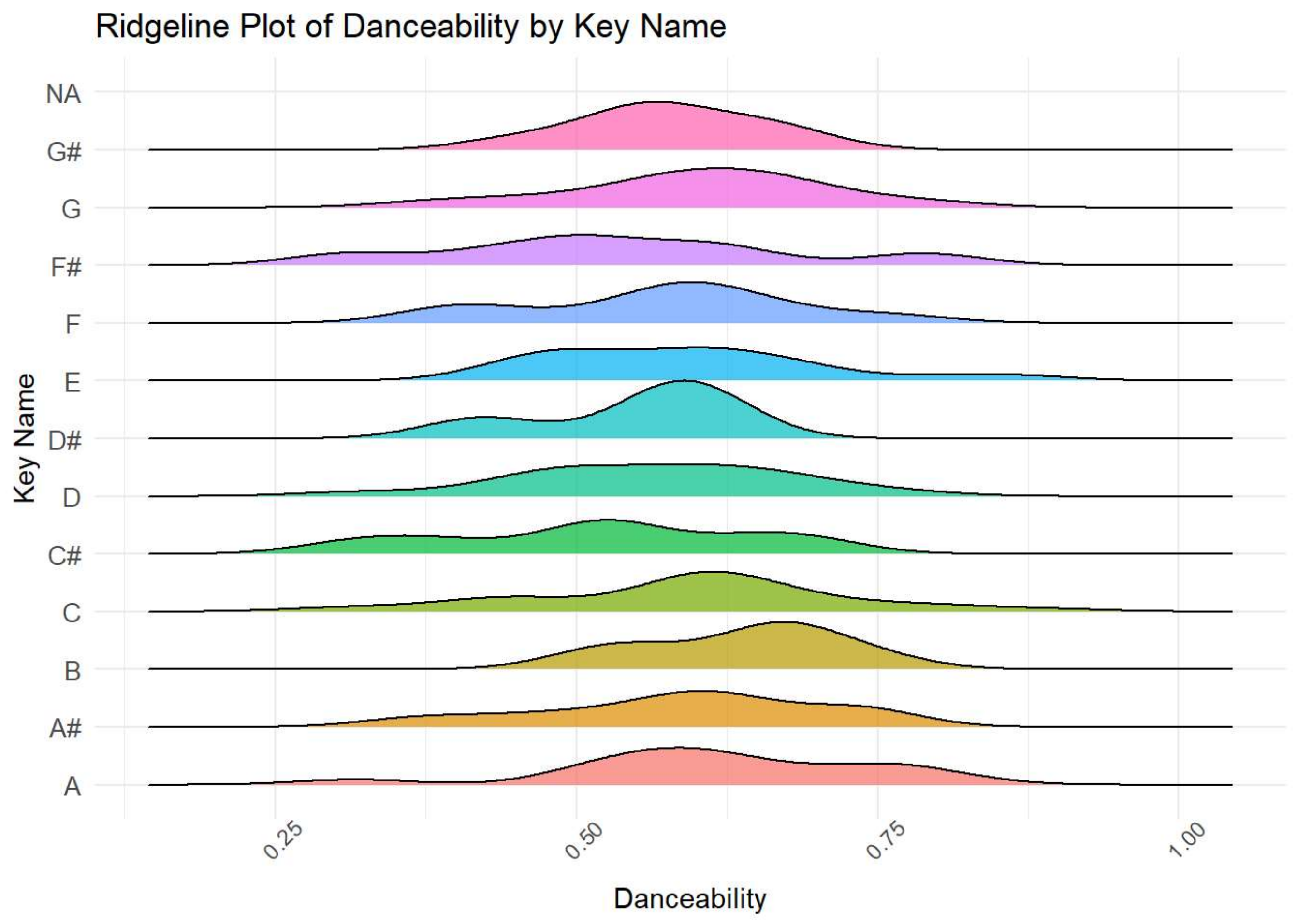
```r
# 1. Danceability vs Song Features
ggplot(data, aes(x = danceability, y = as.factor(key), fill = as.factor(key))) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Danceability by Key",
       x = "Danceability",
       y = "Key") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



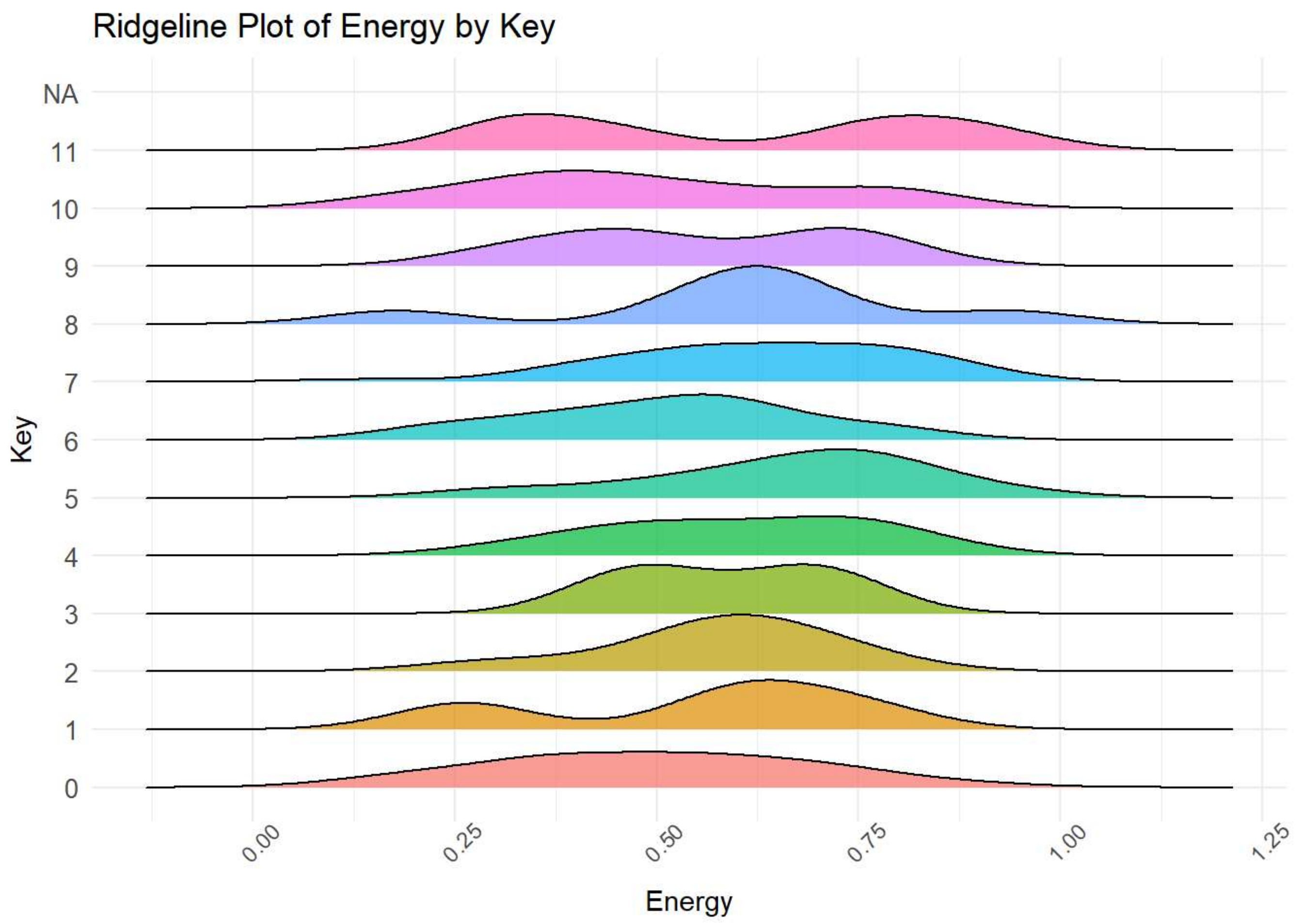
```
ggplot(data, aes(x = danceability, y = as.factor(key_mode), fill = as.factor(key_mode))) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Danceability by Key Mode",
       x = "Danceability",
       y = "Key Mode") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



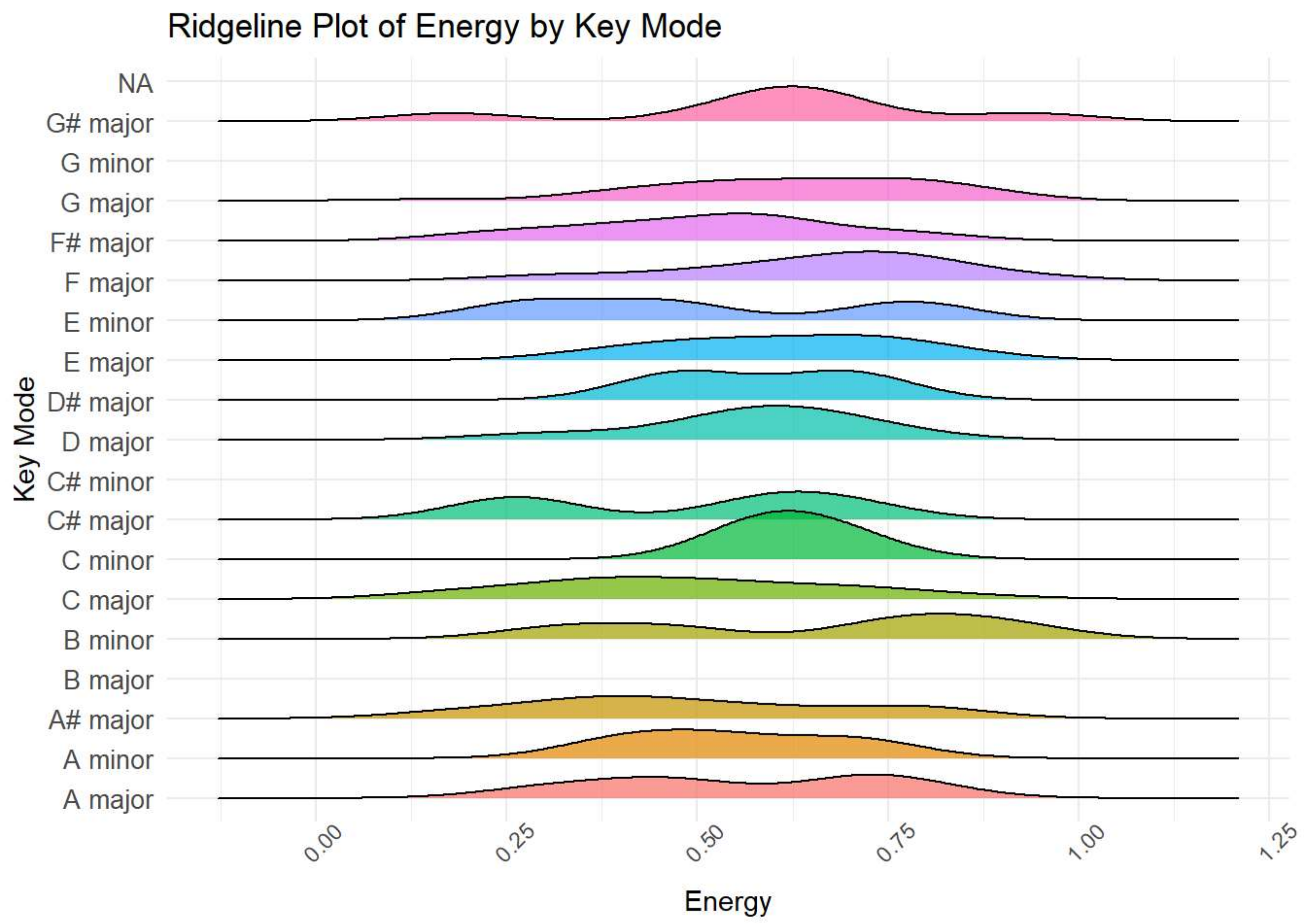
```
ggplot(data, aes(x = danceability, y = key_name, fill = key_name)) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Danceability by Key Name",
       x = "Danceability",
       y = "Key Name") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



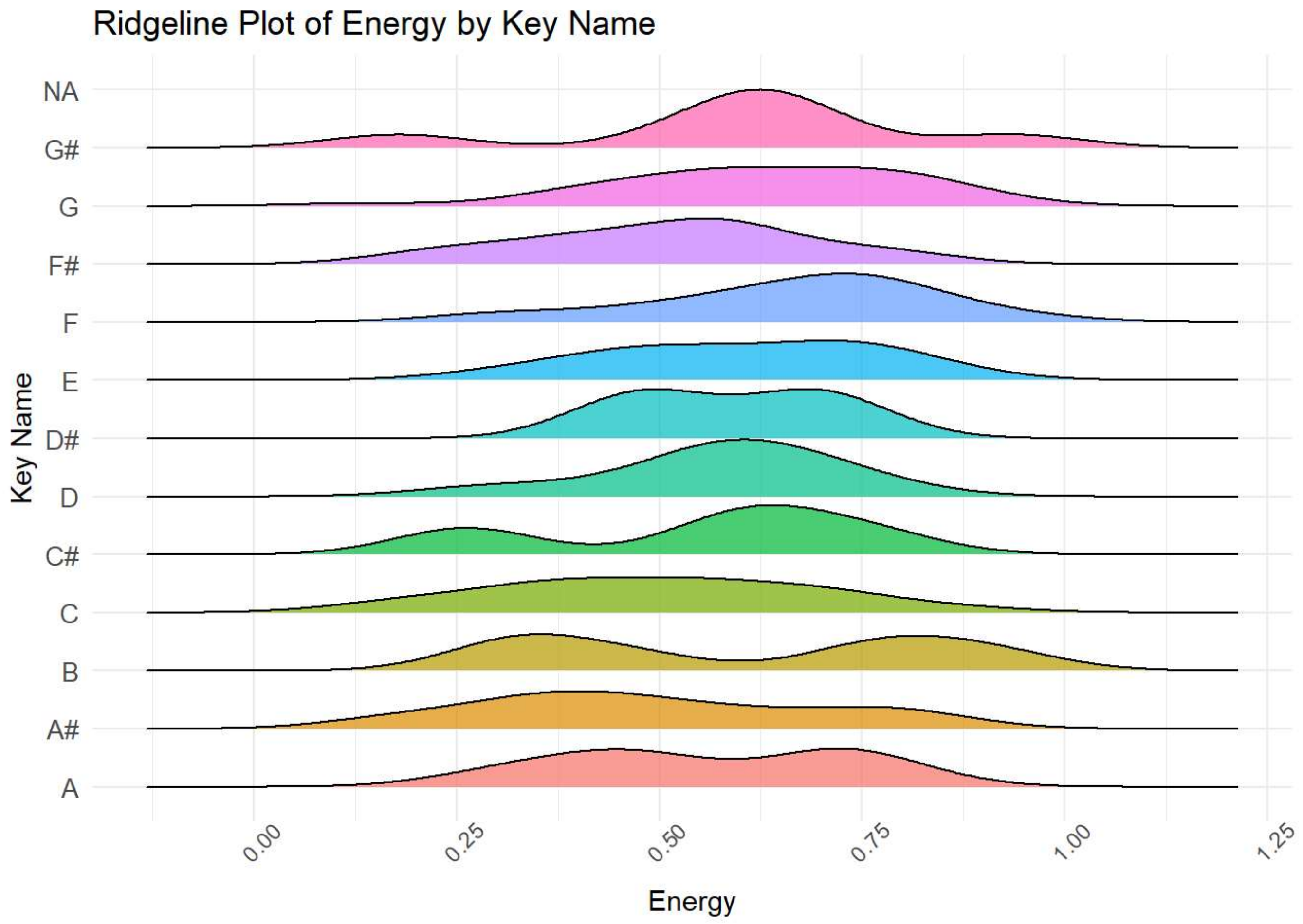
```r
# 2. Energy vs Song Features
ggplot(data, aes(x = energy, y = as.factor(key), fill = as.factor(key))) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Energy by Key",
       x = "Energy",
       y = "Key") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



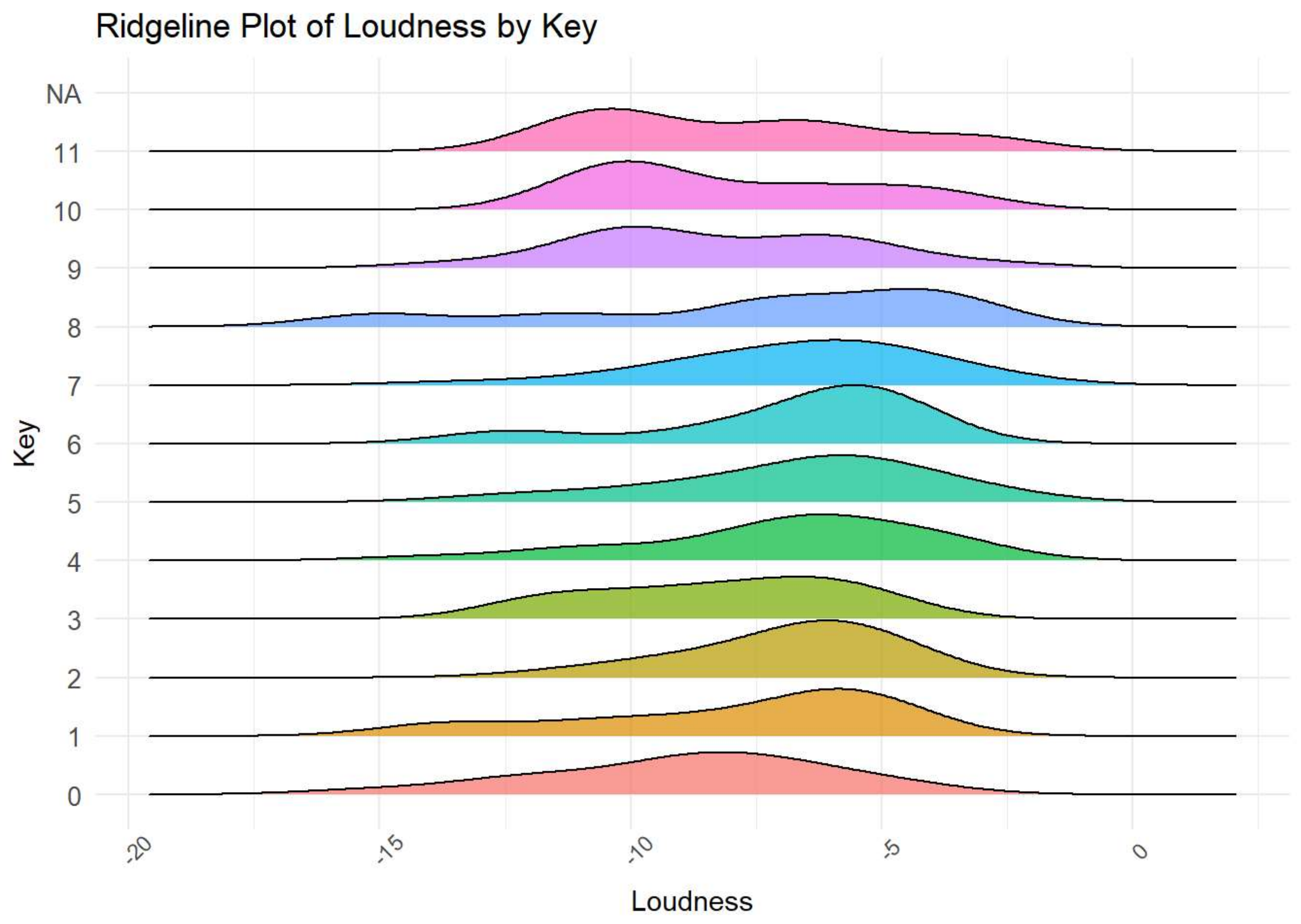
```
ggplot(data, aes(x = energy, y = as.factor(key_mode), fill = as.factor(key_mode))) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Energy by Key Mode",
       x = "Energy",
       y = "Key Mode") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



```
ggplot(data, aes(x = energy, y = key_name, fill = key_name)) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Energy by Key Name",
       x = "Energy",
       y = "Key Name") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



```r
# 3. Loudness vs Song Features
ggplot(data, aes(x = loudness, y = as.factor(key), fill = as.factor(key))) +
  geom_density_ridges(alpha = 0.7, scale = 1) +
  labs(title = "Ridgeline Plot of Loudness by Key",
       x = "Loudness",
       y = "Key") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, vjust = 1),
        axis.text.y = element_text(size = 10))
```



```
# You can continue similarly for the rest of the combinations of song features (key, key_mode, k
ey_name)
# and Spotify features (speechiness, acousticness, instrumentalness, liveness, valence)
```