Spotify-data-EDA

Gaurav Surtani

2023-11-04

Read data from CSV file:

```r
data <- read.csv("spotify-2023.csv")
```

Know the dimensions:

```r
dim(data)

## [1] 953  24
```

Check the new dimensions of the cleaned data:

```r
spotify_data <- na.omit(data)
dim(spotify_data)

## [1] 953  24
```

Exploring Spotify Music Trends

Introduction

I'm interested in understanding what musical attributes and trends are associated with popularity and success on Spotify. Specifically, I want to explore how release date, tempo, danceability, energy, etc. relate to metrics like streams and playlist additions. This could shed light on what current listeners value in music.
I have chosen the Spotify dataset as it provides a comprehensive list of the most famous songs of 2023. I am curious to explore various aspects of popular music and how songs perform across different streaming platforms.

Data

Summary

The data comes from a CSV file containing details on songs released in 2022-2023 that appeared on Spotify charts and playlists. It has 950+ rows and 23 columns, with each row representing a song.

Key variables i want to focus on in this dataset is as belows: - release_date: date song was released - streams: total streams on Spotify - playlist_adds: number of Spotify playlists song was added to - bpm: beats per minute - danceability: Spotify danceability score - energy: Spotify energy score - key: song key - mode: major/minor

The data was web scraped and compiled in January 2023.

- **Data Source**: The dataset is sourced from Kaggle Link and Spotify and contains information about popular songs in 2023.
- **Data Collection**: The data was collected through a combination of sources, including Spotify's internal databases, music charts, and streaming statistics, through web scraping or API calls to gather additional information done in Kaggle
- **Cases**: Each row in the dataset represents a unique song. It provides detailed information about each song's attributes, popularity, and presence on various music platforms.
- **Variables**: (Referenced from Kaggle variable list)
  - `track_name`: Name of the song.
  - `artist(s)_name`: Name of the artist(s) of the song.
  - `artist_count`: Number of artists contributing to the song.
  - `released_year`: Year when the song was released.
  - `released_month`: Month when the song was released.
  - `released_day`: Day of the month when the song was released.
  - `in_spotify_playlists`: Number of Spotify playlists the song is included in.
  - `in_spotify_charts`: Presence and rank of the song on Spotify charts.
  - `streams`: Total number of streams on Spotify.
  - `in_apple_playlists`: Number of Apple Music playlists the song is included in.
  - `in_apple_charts`: Presence and rank of the song on Apple Music charts.
  - `in_deezer_playlists`: Number of Deezer playlists the song is included in.
  - `in_deezer_charts`: Presence and rank of the song on Deezer charts.
  - `in_shazam_charts`: Presence and rank of the song on Shazam charts.
  - `bpm`: Beats per minute, a measure of song tempo.
  - `key`: Key of the song.
  - `mode`: Mode of the song (major or minor).
  - `danceability_%`: Percentage indicating how suitable the song is for dancing.
  - `valence_%`: Positivity of the song's musical content.
  - `energy_%`: Perceived energy level of the song.
  - `acousticness_%`: Amount of acoustic sound in the song.
  - `instrumentalness_%`: Amount of instrumental content in the song.
  - `liveness_%`: Presence of live performance elements.
  - `speechiness_%`: Amount of spoken words in the song.
- **Type of Study**: This dataset is observational, as it provides information about songs and their attributes without any controlled experiments.

Exploratory Data Analysis:

Visualizations:

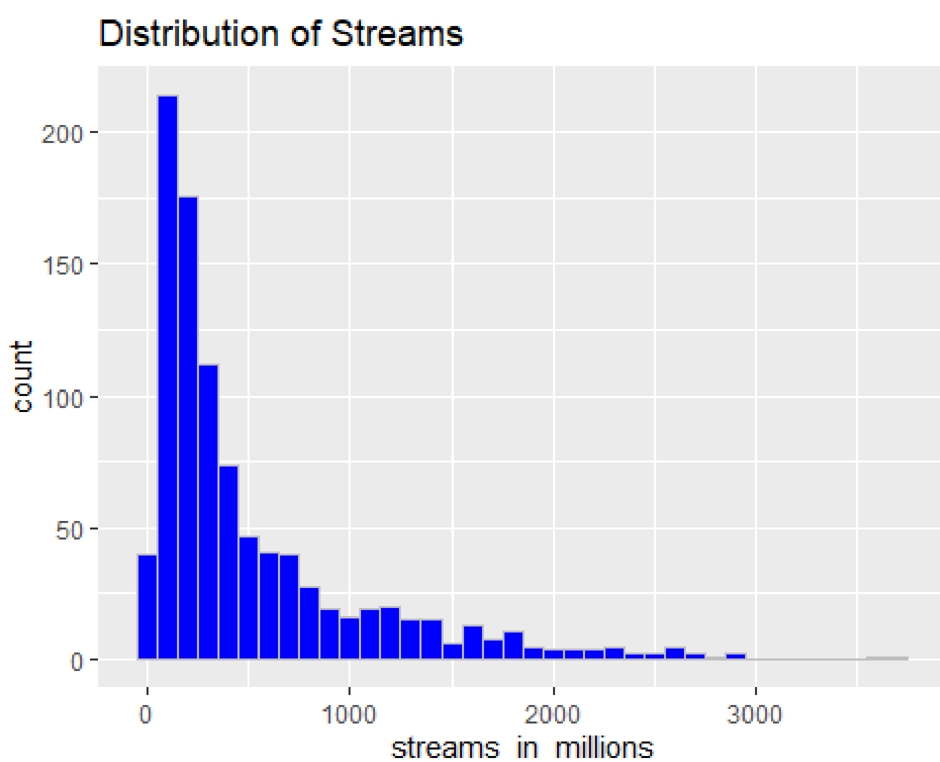
Histograms:

I chose to start with Histograms of Streams and Playlist Additions because:
- They provide an overview of the data distribution. You can immediately see if the data is skewed, has a normal distribution, or has multiple modes.
- They help identify outliers.If there's a long tail, that could suggest the presence of outliers. We may have to remove some outliers at the end because they might possibly skew the results towards them.
- They are a precursor to data cleaning. If you spot any anomalies, such as unexpected spikes that don't correspond to the real-world behavior of the data, this might indicate errors or noise in the data collection process that need to be addressed.

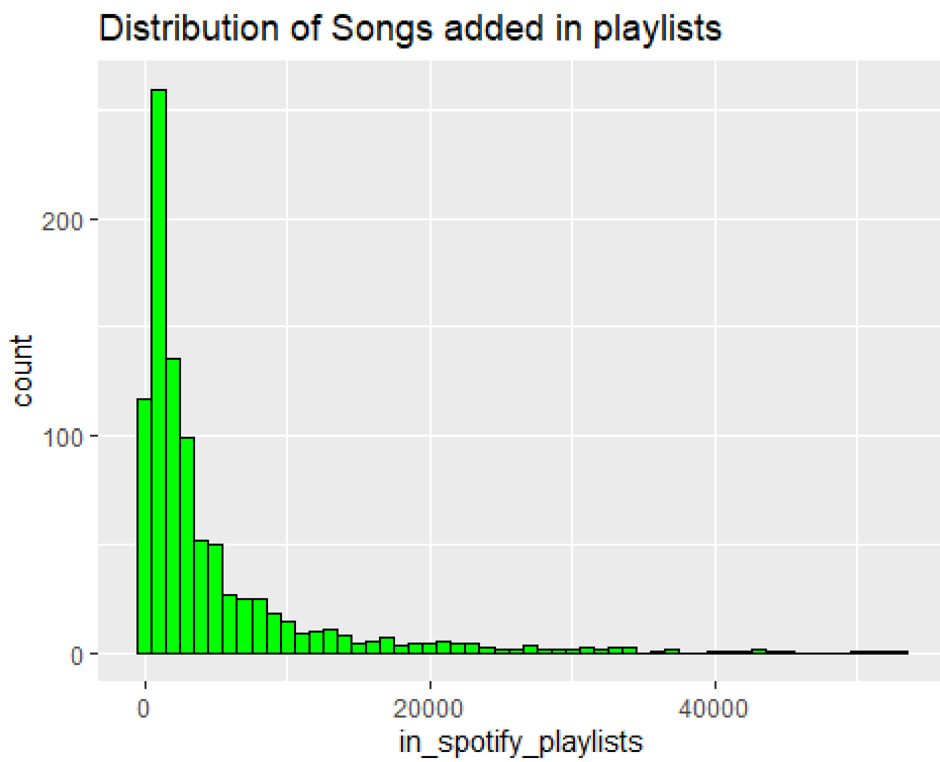
1. Histograms for Streams in millions and Playlist Adds:

We scale down to the stream to in millions,convert to int and clean-up NA's to improve data clarity

```
spotify_data$streams <- as.numeric(as.character(spotify_data$streams))

## Warning: NAs introduced by coercion

spotify_data <- spotify_data[!is.na(spotify_data$streams), ]
spotify_data$streams_in_millions <- spotify_data$streams/1000000
summary(spotify_data$streams_in_millions)

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.003  141.636  290.531  514.137  673.869 3703.895

ggplot(spotify_data, aes(x = streams_in_millions)) +
  geom_histogram(binwidth = 100, fill = "blue", color = "gray") +
  labs(title = "Distribution of Streams")
```



```r
ggplot(spotify_data, aes(x = in_spotify_playlists)) +
  geom_histogram(binwidth = 1000, fill = "green", color = "black") +
  labs(title = "Distribution of Songs added in playlists")
```

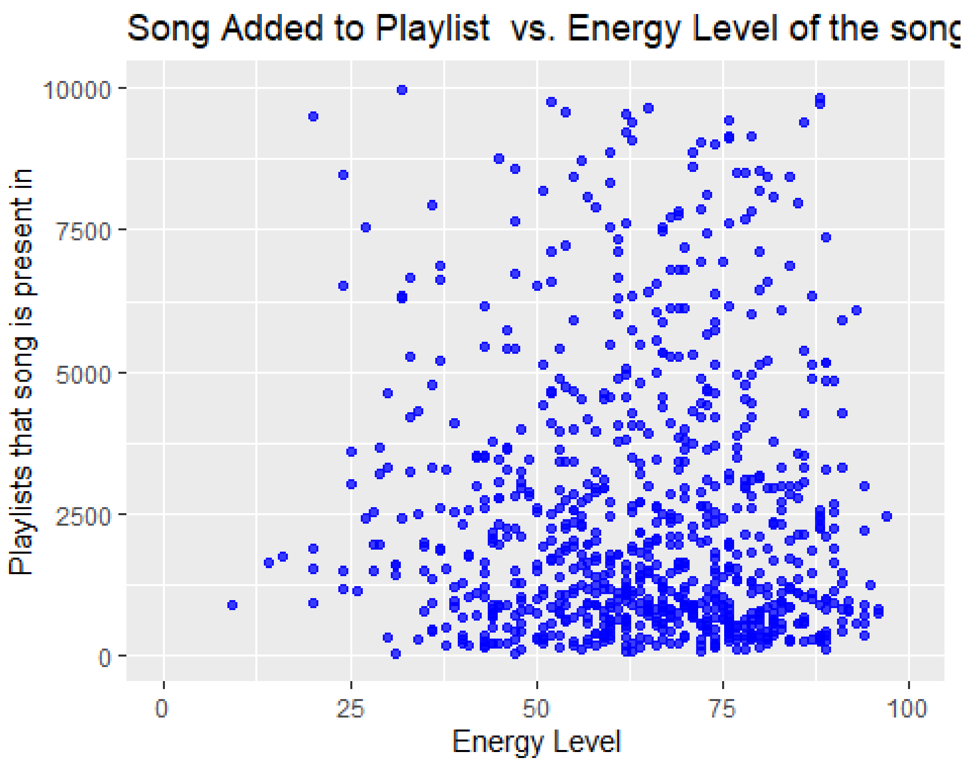


1. **Distribution of Streams**:
   - The histogram for `Streams` is **right-skewed**. Most songs have a relatively small number of streams (in millions), while a few songs have a very high number of streams.
   - There is a noticeable peak in the distribution, which again suggests that a higher number of songs have fewer streams.
   - The long tail to the right indicates that while most songs don't achieve extremely high stream counts, there are a select few that do.
2. **Distribution of Playlist Adds**:
   - The histogram for `Songs added in playlists` shows a **heavily right-skewed** distribution. This indicates that a large number of songs have a relatively small number of playlist additions, while only a few songs have a very high number of additions.
   - The peak at the left suggests that the most common number of playlist additions is low, near zero.
   - There are a few outliers with a very high number of playlist adds, but these are exceptional.
   - The distribution suggests that it is relatively rare for songs to be added to a large number of playlists.

ScatterPlots:

1. Song Added to Playlist vs. Energy Level of the song:

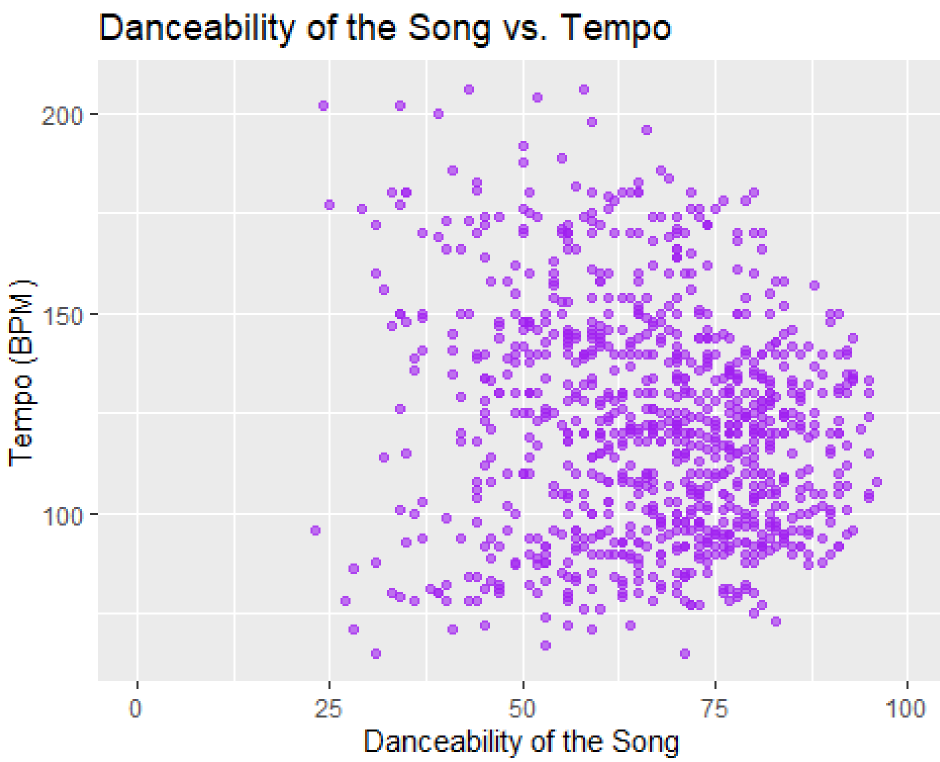
```
spotify_data_lessthan10000_playlist <- spotify_data %>%
  filter(in_spotify_playlists <= 10000)

ggplot(spotify_data_lessthan10000_playlist, aes(x = energy_., y =
in_spotify_playlists)) +
  geom_point(alpha = 0.75, color = "blue") +
  labs(title = "Song Added to Playlist  vs. Energy Level of the song",
       x = "Energy Level",
       y = "Playlists that song is present in")+
  xlim(0,100)
```



Song Added to Playlist vs. Energy Level of the song: - The plot shows a wide spread of points, suggesting *there isn't a clear linear relationship* between the energy level of songs and the number of playlists they appear in. - While songs of all energy levels appear to have a chance of being added to a range of playlists, there is a concentration of songs with lower playlist presence, indicating that most songs, regardless of energy, tend to have a lower number of playlist adds. - There are some songs with high energy levels that also have a higher number of playlist adds, but these are not the majority, indicating that high energy alone does not guarantee a higher presence in playlists.

2. Danceability of the Song vs. Tempo:

```r
ggplot(spotify_data, aes(x = danceability_., y = bpm )) +
  geom_point(alpha = 0.6, color = "purple") +
  labs(title = "Danceability of the Song vs. Tempo",
       x = "Danceability of the Song",
       y = "Tempo (BPM)")+
  xlim(0, 100)
```



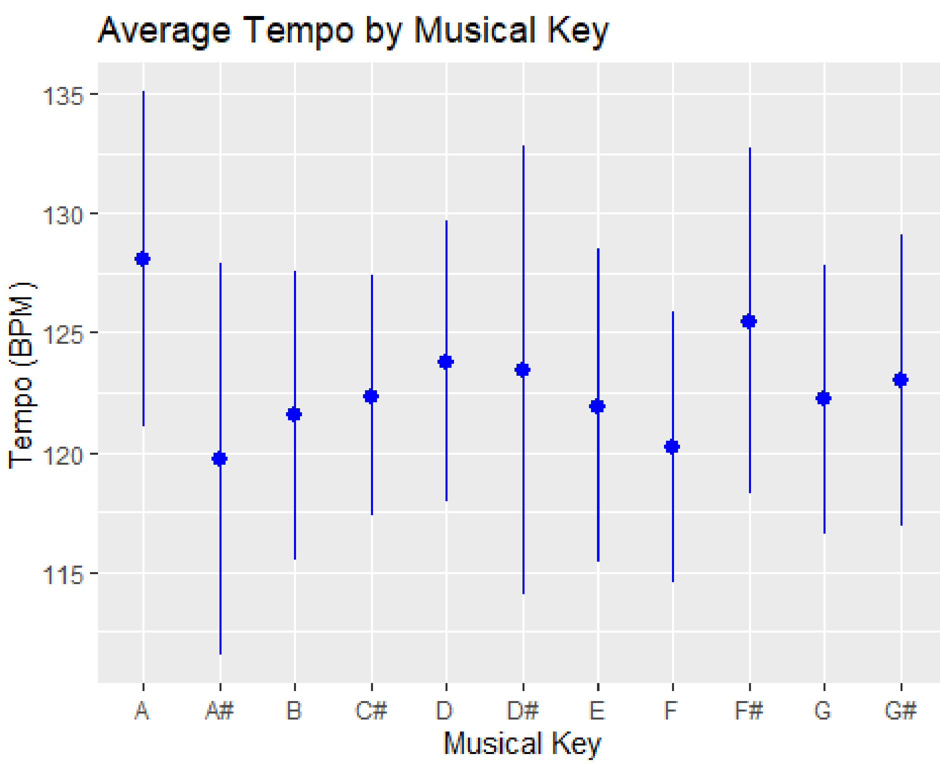
Danceability of the Song vs. Tempo: - This plot shows a broad and relatively uniform distribution of points across the range of danceability scores between 25 and 80. - There is danceability on bpm in range of 80 - 180. The spread of BPM across danceability scores suggests that songs with a wide range of tempos can be danceable, and high danceability is not confined to a narrow tempo range. - There doesn't seem to be a strong correlation between danceability and tempo based on this plot, indicating that tempo might not be a defining factor in danceability, or at least that danceable songs can come at a variety of tempos.

Point Range Plots:

1. Average Tempo by Musical Key:

Which songs have highest tempo based on the musical key. We answer these type of questions using this plot. By using this plot, you are able to present a clear and statistically grounded picture of how tempo varies by musical key in the songs from your Spotify data. It's a more nuanced view than simply plotting the raw data or the means without confidence intervals, as it takes into account the precision of your estimates.

```
spotify_data_filtered_emptyKey <- spotify_data %>%
  filter(!is.na(key) & !is.na(bpm) & bpm != 0 & key!='')

ggplot(spotify_data_filtered_emptyKey, aes(x = as.factor(key), y = bpm)) +
  geom_pointrange(stat = "summary", fun.data = "mean_cl_normal", color =
"blue") +
  labs(title = "Average Tempo by Musical Key",
       x = "Musical Key",
       y = "Tempo (BPM)")
```



The graph suggests a stable trend in song tempos across musical keys, with average BPMs closely grouped between 120 and 130. Variability within each key is moderate, and no particular key is associated with a distinctly faster or slower tempo. This indicates that a song's key is likely not a major factor in determining its tempo.

The `geom_pointrange` plot you have created is effectively a way to visualize the mean tempo (`bpm`) for songs in each musical key (`key`) along with the confidence intervals for those means.

What Questions This Plot Answers:

- **What is the average tempo for songs in each musical key?** You can compare the central tendency (mean bpm) across different keys.
- **How much variability is there in the tempo of songs within each key?** The length of the vertical lines (the point ranges) indicates the confidence interval for the mean, which reflects variability. A longer line means more variability; a shorter line means less.
- **Are there significant differences in tempo between keys?** If the confidence intervals for two keys don't overlap, it suggests a significant difference in the average tempos between those keys.
- **Are certain keys associated with faster or slower tempos?** This can be seen by the position of the point on the y-axis (tempo).

Conclusion I draw from the given analysis:

I calculated summary statistics and visualized distributions of key variables:

- Most songs are relatively recent, released in 2022 or 2023. Streams and playlist adds are right skewed, with most songs having <500M streams and <200 playlist adds.

- There is danceability on bpm in range of 80 - 180. The spread of BPM across danceability scores suggests that songs with a wide range of tempos can be danceable, and high danceability is not confined to a narrow tempo range.

- Songs released more recently tend to have fewer streams, likely because they've had less time to accumulate them. Songs with more playlist adds also tend to have more streams.

- While songs of all energy levels appear to have a chance of being added to a range of playlists, there is a concentration of songs with lower playlist presence, indicating that most songs, regardless of energy, tend to have a lower number of playlist adds.

———————————————————————————

Future Questions that I can ask from the complete dataset!

I'd like to test hypotheses about how song attributes relate to popularity:

1. Are songs with higher danceability scores streamed more?

2. Do songs with higher energy have more playlist adds?

3. Are songs in a major key streamed more than songs in a minor key?