# The Sound of Success: Analyzing Spotify's Most-Streamed Songs and Artists

Erica Rettig, Hao Jiang, and Nishant Sai Challa

University of Victoria

STAT 321: Data Management and Presentation

Arjun Banik

April 5, 2025

## Introduction and Motivation

With Spotify among the most powerful platforms influencing worldwide music trends, the advent of streaming services has significantly altered the music business. By examining Spotify’s most popular songs and artists, we can gain insights into music consumption patterns, artist success, and the factors driving streaming popularity.

Analyzing this data helps to understand listener behavior, artist consistency, and the factors that influence streaming success, given Spotify's broad user base and global influence. Combining data visualization and statistical modeling, this analysis applies both descriptive and predictive analytics to explore the relationships between different artist attributes (such daily streams, total streams, and song roles) and their streaming popularity. Our objectives are to:

- Identify and visualize patterns within Spotify's top-streamed artists and songs.
- Predict streaming success using statistical modeling to identify key predictive factors.
- Group artists into success patterns using K-means clustering based on streaming data.

## Data Cleaning and Wrangling

Kaggle provided the databases used for this analysis, which included comprehensive records of Spotify's most streamed songs and artists (Magsi, 2023). The Artists dataset consists of 2,995 records and includes one categorical variable, Artist, and four numerical variables: Streams, As lead, Daily, Solo, and As feature. The Songs dataset includes 2,501 observations and consists of two categorical variables, song_title and artist, and two numerical variables, daily (average streams per day in 2023) and streams. The data cleaning and wrangling steps are as follows (see Appendix A for SQL code):

- **Separation of variable:** The combined "song title and artist" column was split into separate columns (artist and song_title) using substring code.

- **Cleaning steps:** Removed commas from numeric variables to ensure accurate numerical computations. Missing values were replaced with zeros, ensuring no assumptions about missing data skewed the analysis. We decided to use zero since mean or median cannot be used to realistically show stream and listener counts.
- **Data Transformation:** all variables were converted to the same unit of measurement.

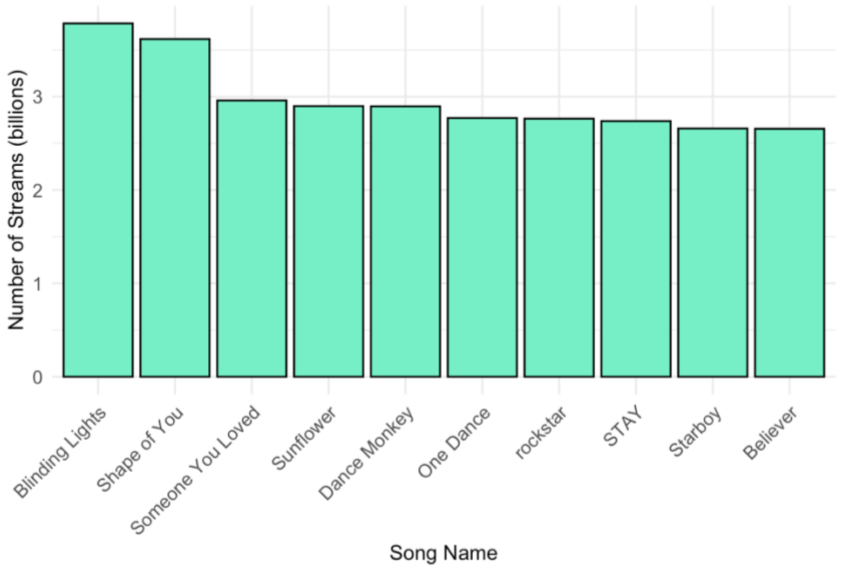
## Data Visualization

To visualize Spotify's top-streamed songs and artists, bar charts and a heat map were used to highlight key trends (see Appendix B for R code). Bar charts are highly effective in visualizing categorical data and comparing streaming numbers, whereas heat maps provide a clear and immediate way to interpret the strength of multiple correlations.

The top 10 most-streamed songs are illustrated in Figure 1. “Blinding Lights” by The Weeknd emerges as the most-streamed song, followed by “Shape of You” by Ed Sheeran, with both tracks surpassing 3.5 billion streams. The remaining eight songs in the top 10 each surpass 2.5 billion but fall short of 3 billion streams.

**Figure 1**

*Top 10 Most Streamed Songs of All Time on Spotify*



Similarly, the top 10 most-streamed artists are shown in Figure 2. Drake is the top-streamed artist, accumulating over 85 billion streams, with the next closest being Bad Bunny, who has just under 70 billion streams. Interestingly, the top 10 most-streamed artists differ from the top 10 most-streamed artists per day. Taylor Swift ranks as the top daily-streamed artist with around 85 million streams per day, followed by Olivia Rodrigo with around 72 million streams per day (Figure 3). Only four artists appear in both the top 10 most-streamed artists and the top 10 daily-streamed artists: Taylor Swift, Drake, Bad Bunny, and The Weeknd. Similarly, only four artists in the top 10 most -streamed artists have a song in the top 10 most -streamed songs: Drake, The Weeknd, Ed Sheeran, and Post Malone.

**Figure 2**

*Top 10 Most Streamed Artists of All Time on Spotify*

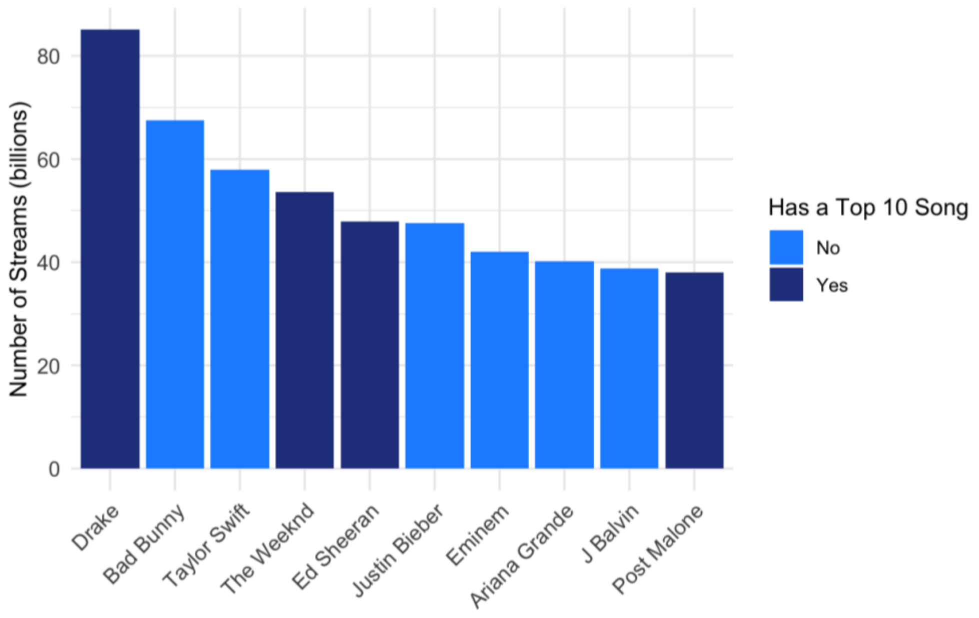
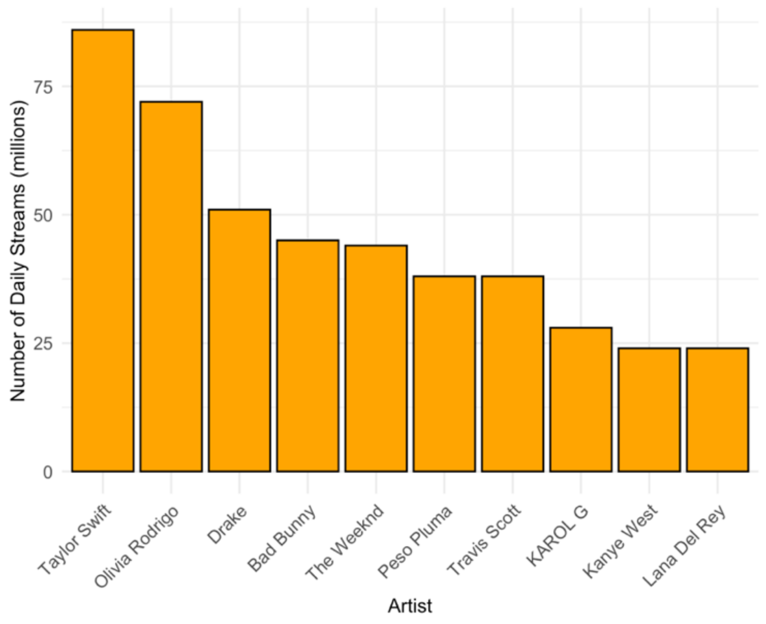


**Figure 3**

*Top 10 Most Streamed Artists Per Day on Spotify in 2023*

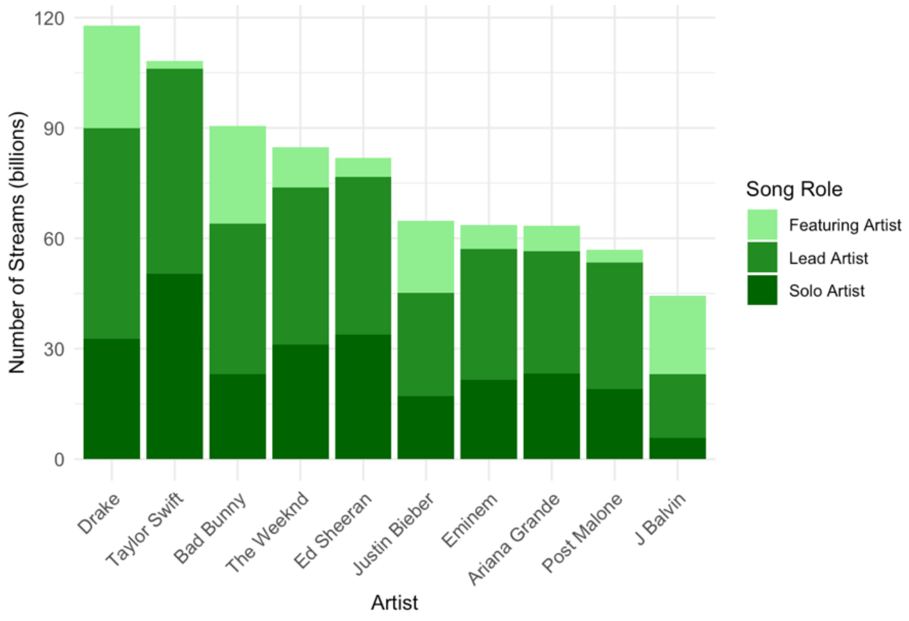


The discrepancy between overall and daily streams suggests that total popularity does not always equate to consistent daily listening. For instance, Drake's higher total streams suggest that his music has been consistently popular over a long period, whereas Taylor Swift has gained more recent attention. These charts also reveal that the distribution of streams is highly uneven, with only a few artists dominating the charts. Within the top 10 alone, there is a difference of 40 billion streams between the top artist and the tenth. This may reflect the structure of the music industry, where only a small number of top-tier artists capture the majority of streams. Additionally, with fewer than 50% of the top 10 artists having a song in the top 10, artist popularity doesn’t always guarantee exceptional song performance.

Figure 4 provides a breakdown of each artist’s streams based on their role in the song. In this context, Drake remains the top artist, amassing nearly 120 billion total streams across featured, lead, and solo roles. Taylor Swift ranks as the second-most streamed artist when considering song roles, with around 105 billion streams. From this chart, it can also be concluded that Taylor Swift is the most-streamed solo artist, while Drake is the top lead artist and top featured artist.

**Figure 4**

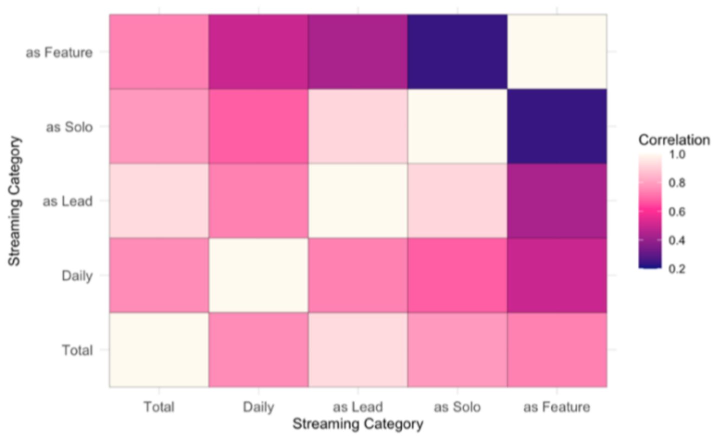
*Top 10 Most Streamed Artists on Spotify: Streams as Lead, Solo, and Featured Artist*



A heat map (Figure 5) reveals strong correlations between total and lead streams (r = 0.93), indicating that lead roles drive an artist's overall success. Solo and featured streams are also correlated with total streams (r = 0.79 and r = 0.74, respectively), while total and daily streams show a moderate correlation (r = 0.76). The weakest correlation (r = 0.24) is between solo and featured streams, suggesting solo success does not strongly predict collaborative success. Correlation strength is classified as strong (0.7–0.9), moderate (0.4–0.6), or weak (<0.4) based on Akoglu (2018).

**Figure 5**

*Correlation Heatmap of Streaming Categories*



## Statistical Analysis/Modeling

### Descriptive Statistics

The streaming data exhibits strongly right-skewed distributions, with most artists clustered at lower stream counts and a few outliers dominating overall plays (see Appendix C for histograms). Boxplots confirm the significant impact of these outliers. Among artists, total streams range from roughly 2.1 billion to as high as 85 billion, with a median near 3.9 billion; daily streams have a median of about 2 million but exceed 80 million in some cases. In terms of streaming share, most artists capture only a small part of total streams, although a small number of artists dominate the streams (see Appendix D for R code used for Statistical Analysis).

### Transformations

Logarithmic transformations were applied to the skewed streaming data to compress large ranges and reduce the impact of extreme values (see Appendix E for histograms). For percentage-based variables, we used an arcsine square-root transformation to stabilize their variance, which is helpful when dealing with proportions that can be near zero or one.

### Linear Regression

Linear regression was used to capture how an artist's single highest-streaming song relates to their overall streaming success. Linear regression is appropriate because both the dependent variable and the main predictor are continuous, and the log scale helps handle large, skewed streaming counts. This regression results in the model equation: log(Total Streams) = 0.7015 + 1.0535 x log(top song streams).

The key takeaway from the linear regression is that an artist's top-streaming track is a strong, positive predictor of that artist's total log-transformed streams, with a slope of approximately 1.05 and high statistical significance ($p < 0.001$). However, the model explains about 21% of the variation ($R^2 \approx 0.216$), indicating that while top song performance matters, other factors also influence an artist's overall stream counts. Diagnostic checks show no severe violations of linear model assumptions (see Appendix F). A few data points deviate significantly from the overall trend, indicating potential influential observations that might affect the model fit. Overall, the main finding is that a single high-impact track can be a meaningful indicator of total streaming success, but it does not fully account for all variance in total streams.

**Logistic Regression**

The logistic model predicts whether a given song ranks in the top 10 (binary outcome) based on artist-level features such as daily streams and total streams. Logistic regression is well-suited for this task because it directly models the probability of a binary event (top 10 vs. not top 10) and provides interpretable coefficients that show how changes in each predictor affect the odds of being in the top 10. Some assumptions were that observations are independent of each other, and the size of the data set was large enough to have statistical power.

This model does provide some ability to distinguish top-10 songs from others (AUC $\approx$ 0.71), but none of the predictors are statistically significant (see Appendix G). The specific artist-level features used here do not strongly explain which songs end up in the top 10.  Additional or alternative predictors may be necessary to better capture what truly drives a song's rank.

### K-means Clustering

This analysis involved transforming each artist's streaming metrics to see whether artists can be grouped into distinct segments based on their streaming behavior. An elbow plot suggested three clusters as a suitable choice (Appendix H Figure 1). After fitting the k-means model, the resulting average silhouette width was about 0.368, indicating a weak cluster separation (Appendix H Figure 2). The observations had relatively low silhouette widths (<0.25), suggesting those points do not fit well into their assigned clusters and may be considered outliers. The PCA projection showed partial overlap among the clusters (Appendix H Figure 3). Overall, while three clusters provide some separation among artists based on their streaming characteristics, the moderate silhouette scores and numerous low-width observations imply that the data may not naturally fall into well-defined, highly cohesive groups under this simple approach. Clusters can be further refined or interpreted by examining outliers and incorporating additional variables if needed.

## Conclusion

High artist rankings do not guarantee top-performing songs, and lead roles strongly influence an artist's total success. Artists achieve streaming success through various roles, some excel solo, while others benefit from a mix of lead and featured performances. Despite this, artist-level predictors do not significantly forecast whether a song will place in the top 10, suggesting that additional factors, such as genre, collaborators, and daily streams, may be necessary to enhance prediction. Finally, a three-cluster solution provides a reasonable initial partition of artists, though further refinement could be pursued with more in-depth data.

## References

Akoglu, H. (2018). User's guide to correlation coefficients. *Turkish Journal of Emergency Medicine*, *18*(3), 91–93. https://doi.org/10.1016/j.tjem.2018.08.001

DataCamp. (2025). *Reshaping Data in R*. https://www.datacamp.com/doc/r/reshape

Magsi, M. A. (2023). *Spotify dataset series: Most streamed songs, most streamed artists of all time, and top artists by monthly listeners*. Kaggle. https://www.kaggle.com/datasets/meeratif/spotify-most-streamed-songs-of-all-time

MySQL. (n.d.). *MySQL 8.4 reference manual: ALTER TABLE statement*. Oracle. https://dev.mysql.com/doc/refman/8.4/en/alter-table.html

MySQL. (n.d.). *MySQL 8.4 reference manual: String functions*. Oracle. https://dev.mysql.com/doc/refman/8.4/en/string-functions.html

OpenAI. (2024). ChatGPT (GPT-4) [Large language model]. (Debugging) https://chatgpt.com/

## Appendix A

## SQL Code for Data Cleaning and Wrangling

```
CREATE DATABASE FINAL_PROJECT;

 USE FINAL_PROJECT;

ALTER TABLE spotify most streamed RENAME TO spotify_most_streamed;

ALTER TABLE spotify_most_streamed

ADD artist VARCHAR(255),

ADD song_title VARCHAR(255);

SET SQL_SAFE_UPDATES = 0;

UPDATE spotify_most_streamed

SET artist = SUBSTRING_INDEX(Artist and Title, ' - ', 1),

song_title = SUBSTRING_INDEX(Artist and Title, ' - ', -1)

WHERE Artist and Title IS NOT NULL;

SET SQL_SAFE_UPDATES = 1; -- Re-enable Safe Mode after updating

SET SQL_SAFE_UPDATES = 0;

UPDATE spotify_most_streamed

SET Streams = REPLACE(Streams, ',', '' ),

Daily = REPLACE(Daily, ',', '');

 UPDATE spotify_most_streamed

SET Streams = 0

 WHERE Streams IS NULL OR TRIM(Streams) = '';

UPDATE spotify_most_streamed

SET Daily = 0

WHERE Daily IS NULL OR TRIM(Daily) = '';
```

```
ALTER TABLE spotify_most_streamed

MODIFY COLUMN Streams BIGINT,

MODIFY COLUMN Daily BIGINT;

SET SQL_SAFE_UPDATES = 1;

ALTER TABLE spotify_most_streamed

 DROP COLUMN Artist and Title

SELECT

song_title,

artist,

Streams,

 ROUND((Streams * 100.0) / (SELECT SUM(Streams) FROM spotify_most_streamed), 2) AS
percentage_of_total_streams FROM spotify_most_streamed

ORDER BY Streams DESC;
```

```
SET SQL_SAFE_UPDATES = 0;

ALTER TABLE artists1

MODIFY COLUMN Daily BIGINT,

 MODIFY COLUMN Streams BIGINT;

UPDATE artists1

SET

Daily = Daily * 1000000,

Streams = Streams * 1000000;

UPDATE artists1

SET Streams = REPLACE(Streams, ',', ''),
Daily = REPLACE(Daily, ',', ''),

As lead = REPLACE(As lead, ',', ''),

 Solo = REPLACE(Solo, ',', ''),

 As feature = REPLACE(As feature, ',', '');

UPDATE artists1

SET Streams = 0

WHERE Streams IS NULL OR TRIM(Streams) = '';

UPDATE artists1

SET Daily = 0

WHERE Streams IS NULL OR TRIM(Daily) = '';

UPDATE artists1

SET As lead = 0

WHERE Streams IS NULL OR TRIM(As lead) = '';

UPDATE artists1

SET Solo = 0

WHERE Streams IS NULL OR TRIM(Solo) = '';

UPDATE artists1

 SET As feature=0

WHERE Streams IS NULL OR TRIM( As feature) = '';

ALTER TABLE artists1

MODIFY COLUMN Streams BIGINT DEFAULT 0,

MODIFY COLUMN Daily DECIMAL(10,3) DEFAULT 0,

MODIFY COLUMN As lead DECIMAL(10,3) DEFAULT 0,

MODIFY COLUMN Solo DECIMAL(10,3) DEFAULT 0,

MODIFY COLUMN As feature DECIMAL(10,3) DEFAULT 0;
```

```
SELECT

Artist,

SUM(Streams) AS total_streams,

ROUND((SUM(Streams) * 100.0) / (SELECT SUM(Streams) FROM artists1), 2) AS
percentage_of_total_streams

FROM artists1

GROUP BY Artist;
```

## Appendix B

## R Code for Data Visualization

```
### Libraries ###
library(tidyverse)
library(reshape2)

### Load Cleaned Data ###

setwd("~/STAT321/Project")
songs = read.csv("most streamed songs data perctange of total streams1.csv")  # Top 1000 Streamed Songs
artists = read.csv("artists _cleaned 2.csv")        # Top 1000 Streamed Artists

### Bar Plots ###

# Top 10 Songs

top_10_songs = songs %>%.              # Create df with only top 10 most streamed songs
  arrange(desc(Streams)) %>%
  head(10) %>%
  mutate(Streams_bil = Streams/1000000000)       # Scale by a billion for better visualization

ggplot(top_10_songs, aes(x=reorder(song_title, -Streams), y = Streams_bil)) + # Order by no. streams (desc)
  geom_bar(stat = "identity", fill = "aquamarine2", color ="black") +
  labs(title = "Top 10 Most Streamed Songs on Spotify",
       x = "Song Name", y = "Number of Streams (millions)") + theme_minimal() +
  theme(
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10))                         # Increase tick label size for better readability

# Top 10 Artists Overall

top_10_artists = artists %>%              # Create df with only top 10 most streamed artists
  arrange(desc(Streams)) %>%
  head(10) %>%
  mutate(Streams_bil = Streams/1000000000, Daily_bil = Daily/1000000000) %>%       # scale by a billion
  mutate(has_top_10_song = ifelse(Artist %in% top_10_songs[,2], "Yes", "No"))   # add column that
indicates if the artist also has a song in the top 10

ggplot(top_10_artists, aes(x = reorder(Artist, -Streams), y = Streams_bil,
                           fill = has_top_10_song)) +
  geom_bar(stat = "identity") +         # Create bar chart, ordered by number of streams (desc) and filled to
indicate Y/N to having a top 10 song
  scale_fill_manual(values = c("Yes" = "royalblue4", "No" = "dodgerblue")) +
  theme_minimal() +
  theme(axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10)) +
  labs(title = "Top 10 Artists", x = "Artist", fill =
       "Has a Top 10 Song", y = "Number of Streams (millions)")

# Top 10 Daily Streamed Artists

top_10_per_day = artists %>%                # Create df of top 10 artists based on streams per day
  arrange(desc(Daily)) %>%
  head(10) %>%
  mutate(Daily_mil = Daily/1000000)        # scale by a million

ggplot(top_10_per_day, aes(x = reorder(Artist, -Daily), y = Daily_mil)) +      # Create bar chart, ordered by
no. streams (desc)
  geom_bar(stat = "identity", fill = "orange", color = "black") + theme_minimal() +
  labs(title = "Top 10 Most Streamed Artists Per Day on Spotify",
     x = "Artist", y = "Number of Daily Streams (millions)") +
  theme(
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10)
  )

# Top 10 Artist Streams Breakdown

top_10_artists_long = top_10_artists %>%
  pivot_longer(cols = c(As.lead, Solo, As.feature),       # Convert columns for artist roles into a long format
             names_to = "artist_type",                    # Create a new column 'artist_type' to hold the roles
             values_to = "no_streams")   %>%              # Store the corresponding stream counts in no_streams
  mutate(no_streams_bil = no_streams/1000)                # scale to be in billions (og is in millions)

ggplot(top_10_artists_long,
     aes(x = reorder(Artist, -no_streams_bil),                 # Create bar chart, ordered by no. streams (desc)
         y = no_streams, fill = artist_type)) +    # Fill no. streams with a breakdown of the streams by artist
type
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("As.feature" =  "lightgreen","As.lead"= "forestgreen","Solo" =  "darkgreen"),
                 labels = c("As.feature" = "Featuring Artist", "As.lead" = "Lead Artist",
                                              "Solo" = "Solo Artist")) +
  theme_minimal() +
  labs(title = "Top 10 Artists: Comparison of Streams as Lead, Solo, and Featured Artist",
     x = "Artist", y = "Number of Streams (millions)", fill = "Song Role") +
  theme(
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10))

### Heatmap ###
```

```
# Correlation of artist categories

artist_cor = artists %>%
  rename("as Feature" = As.feature, "as Lead" = As.lead, "Total" = Streams,
         "Daily" = Daily, "as Solo" = Solo) %>%              # Rename categories for better interpretation
  select(-Artist) %>%                                        # Remove artist column, not needed
  cor() %>%                         # Create correlation matrix
  melt()                           # Convert the correlation matrix to long format for plotting (DataCamp,
2025)

ggplot(artist_cor, aes(x = Var1, y =Var2, fill = value)) +        # Create heatmap of artist categories
correlations
  geom_tile(color = "black") +
  scale_fill_gradient2(low = "navy", high = "floralwhite", mid = "deeppink",
                       midpoint = 0.6,                        # Shift midpoint as there are more high
correlations
                       limit = c(0.2, 1),                     # Adjust to fit the range of correlations
                       name = "Correlation") +
  theme_minimal() + labs(title = "Correlation Heatmap of Streaming Categories",
                         x = "Streaming Category", y = "Streaming Category") +
  theme(
    axis.text.x = element_text(size = 10.5),
    axis.text.y = element_text(size = 10.5))
```

## Appendix C

## Histograms of Artist Streams

**Figure C1**

*Distribution of Artist Streams*

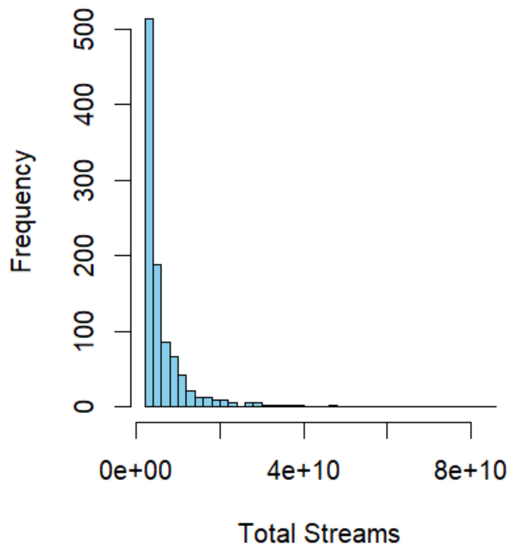
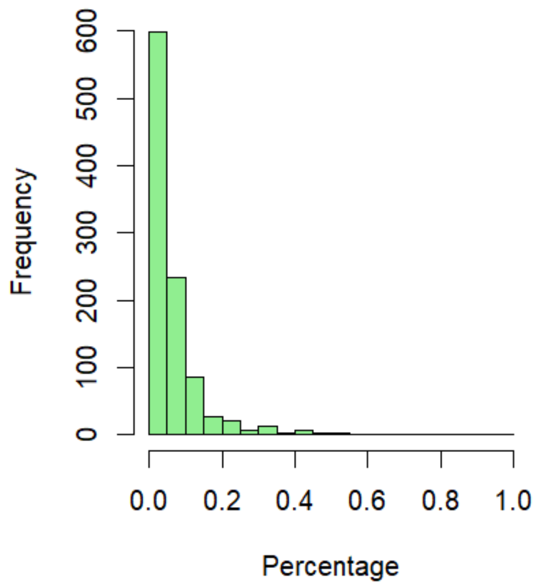


**Figure C2**

*Distribution of Percentage of Artist Streams*



**Appendix D**

**R Code for Statistical Analysis**

```
library(tidyverse)
library(psych)
library(broom)
library(ggplot2)
library(dplyr)
library(pROC)
library(factoextra)
library(cluster)
```

```
artists_cleaned <- read_csv("artists _cleaned 2.csv") artists_cleaned_percentage <-
read_csv("artists _cleaned percentage_of_total_streams2.csv") streamed_songs <-
read_csv("most streamed songs data perctange of total streams1.csv")
```

```
#descriptive analysis
#1. Inspect Data Structure & Summary Statistics
# Artists Cleaned
summary(artists_cleaned)
describe(artists_cleaned)
# Artists Cleaned Percentage
summary(artists_cleaned_percentage)
describe(artists_cleaned_percentage)
# Streamed Songs
summary(streamed_songs)
describe(streamed_songs)
#2. Visualizations
# Set up plotting area for artists_cleaned (2x2 grid)
par(mfrow = c(2, 2))
hist(artists_cleaned$total_streams, breaks = 30, col = "skyblue",
main = "Distribution of Total Streams (Artists)", xlab = "Total Streams")
hist(artists_cleaned$Daily, breaks = 30, col = "lightgreen",
main = "Distribution of Daily Streams (Artists)", xlab = "Daily Streams")
boxplot(artists_cleaned$`As lead`, main = "Boxplot: As lead (Artists)", col = "orange")
boxplot(artists_cleaned$Solo, main = "Boxplot: Solo (Artists)", col = "purple")
# Plot for artists_cleaned_percentage (1x2 grid)
par(mfrow = c(1, 2))
hist(artists_cleaned_percentage$total_streams, breaks = 30, col = "skyblue",
main = "Total Streams (Artists %)", xlab = "Total Streams")
hist(artists_cleaned_percentage$percentage_of_total_streams, breaks = 30, col = "lightgreen",
main = "Percentage of Total Streams", xlab = "Percentage")
```

```
# As the data is heavily skewed, we will transform it
```

```
#1. Appling Transformations
# For artists_cleaned:
# new log-transformed columns for heavy-skewed variables.
artists_cleaned <- artists_cleaned %>%
mutate(
log_total_streams = log(total_streams),
log_Daily = log(Daily + 1), # Adding 1 to avoid log(0)
log_As_lead = log(`As lead` + 1),
log_Solo = log(Solo + 1),
log_As_feature = log(`As feature` + 1)
)
# For artists_cleaned_percentage:
# Since percentage_of_total_streams is in percentage format (e.g., 0.1 for 0.1%), convert to
proportion.
artists_cleaned_percentage <- artists_cleaned_percentage %>%
mutate(
log_total_streams = log(total_streams),
arcsine_perc = asin(sqrt(percentage_of_total_streams / 100))
)
# For streamed_songs:
# Transform Streams (with offset) and apply arcsine square root to
percentage_of_total_streams after conversion.
streamed_songs <- streamed_songs %>%
mutate(
log_Streams = log(Streams + 1),
arcsine_perc = asin(sqrt(percentage_of_total_streams / 100))
)
```

```
#data after transformation
#Histogram for log_total_streams
ggplot(artists_cleaned, aes(x = log_total_streams)) + geom_histogram(bins = 30, fill =
"skyblue", color = "black") + labs(title = "Histogram of Log Total Streams (Artists)", x = "Log
Total Streams", y = "Frequency")
#Histogram for log_Streams (Songs)
ggplot(streamed_songs, aes(x = log_Streams)) + geom_histogram(bins = 30, fill = "orchid",
color = "black") + labs(title = "Histogram of Log Streams (Songs)", x = "Log Streams", y =
"Frequency")
```

```
# Linear regression
# Use the log-transformed Streams from streamed_songs to get each artist's top song
top_song <- streamed_songs %>%
group_by(artist) %>%
summarise(top_log_Streams = max(log_Streams, na.rm = TRUE))
# Joining using the artist name
```

```
artist_model_data <- artists_cleaned %>%
left_join(top_song, by = c("Artist" = "artist"))
# Model: log_total_streams (dependent) as a function of top_log_Streams (independent)
lm_model <- lm(log_total_streams ~ top_log_Streams, data = artist_model_data)
summary(lm_model)
# Diagnostic Plots for Model Adequacy
par(mfrow = c(2, 2))
plot(lm_model)
# Outlier Detection using Cook's Distance
cooksd <- cooks.distance(lm_model)
influential <- which(cooksd > (4/length(cooksd)))
cat("Influential observations based on Cook's distance:\n", influential, "\n")
```

```
# Logistic regression
artists_cleaned <- artists_cleaned %>%
mutate(
log_total_streams = log(total_streams),
log_Daily = log(Daily + 1)
)
streamed_songs <- streamed_songs %>%
arrange(desc(Streams)) %>%
mutate(
rank = row_number(),
top10 = ifelse(rank <= 10, 1, 0)
)
song_model_data <- streamed_songs %>%
left_join(artists_cleaned %>% select(Artist, log_Daily, log_total_streams),
by = c("artist" = "Artist")) %>%
left_join(artists_cleaned_percentage %>% select(Artist, arcsine_perc),
by = c("artist" = "Artist"))
song_model_data <- song_model_data %>%
rename(arcsine_perc = arcsine_perc.y)
song_model_data_clean <- song_model_data %>%
filter(!is.na(log_Daily) & !is.na(log_total_streams) & !is.na(arcsine_perc))
# Logistic Regression Model
logit_model <- glm(top10 ~ log_Daily + log_total_streams + arcsine_perc,
data = song_model_data_clean, family = binomial)
summary(logit_model)
# Diagnostic Plots for Model Adequacy
par(mfrow = c(2, 2))
plot(logit_model)
# Outlier Detection using Cook's Distance
cooksd <- cooks.distance(logit_model)
influential <- which(cooksd > (4 / length(cooksd)))
cat("Influential observations (Cook's Distance):", influential, "\n")
```

```
# Model Evaluation: ROC Curve and AUC
roc_obj <- roc(song_model_data_clean$top10, fitted(logit_model))
plot(roc_obj, main = "ROC Curve for Top 10 Song Prediction")
auc_val <- auc(roc_obj)
cat("AUC:", auc_val, "\n")
```

```
#K-means cluster
artists_cluster <- artists_cleaned %>%
mutate(
#raw proportions
prop_lead = `As lead` / total_streams,
prop_solo = Solo / total_streams,
prop_feature = `As feature` / total_streams,
arcsine_prop_lead = asin(sqrt(prop_lead)),
arcsine_prop_solo = asin(sqrt(prop_solo)),
arcsine_prop_feature = asin(sqrt(prop_feature))
) %>%
# Selecting the transformed streams and transformed proportions
select(log_Daily, log_total_streams, arcsine_prop_lead, arcsine_prop_solo,
arcsine_prop_feature)
# Scale the Data for Clustering
artists_cluster_scaled <- scale(artists_cluster)
# Determine Optimal Number of Clusters (Elbow Method)
wss <- sapply(1:10, function(k) {
kmeans(artists_cluster_scaled, centers = k, nstart = 25)$tot.withinss
})
plot(1:10, wss, type = "b", pch = 19, frame = FALSE,
xlab = "Number of Clusters (K)",
ylab = "Total Within-Cluster Sum of Squares",
main = "Elbow Method for Optimal K")
#4. Perform K-means Clustering
set.seed(123)
k <- 3
kmeans_result <- kmeans(artists_cluster_scaled, centers = k, nstart = 25)
cluster_assignments <- kmeans_result$cluster
artists_cleaned$cluster <- cluster_assignments
# Model Adequacy: Silhouette Analysis
sil <- silhouette(cluster_assignments, dist(artists_cluster_scaled))
fviz_silhouette(sil) +
labs(title = "Silhouette Plot for K-means Clustering")
avg_sil_width <- mean(sil[, "sil_width"])
cat("Average Silhouette Width:", avg_sil_width, "\n")
# Outlier Detection
outliers <- which(sil[, "sil_width"] < 0.25)
cat("Potential outlier observations (silhouette width < 0.25):", outliers, "\n")
```

```
# Cluster Visualization
#
pca_res <- prcomp(artists_cluster_scaled, center = TRUE, scale. = TRUE)
pca_df <- data.frame(PC1 = pca_res$x[,1],
PC2 = pca_res$x[,2],
cluster = factor(cluster_assignments))
ggplot(pca_df, aes(x = PC1, y = PC2, color = cluster)) +
geom_point(size = 2) +
labs(title = "K-means Clustering (PCA Projection)",
x = "Principal Component 1", y = "Principal Component 2") +
theme_minimal()
```

## Appendix E

## Histograms of Song and Artist Streams Following Transformation

**Figure E1**

*Distribution of Artist Streams After Logarithmic Transformation*

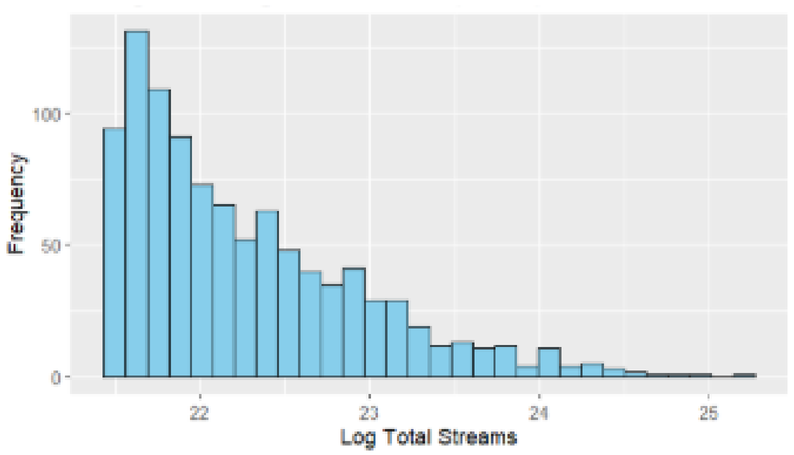
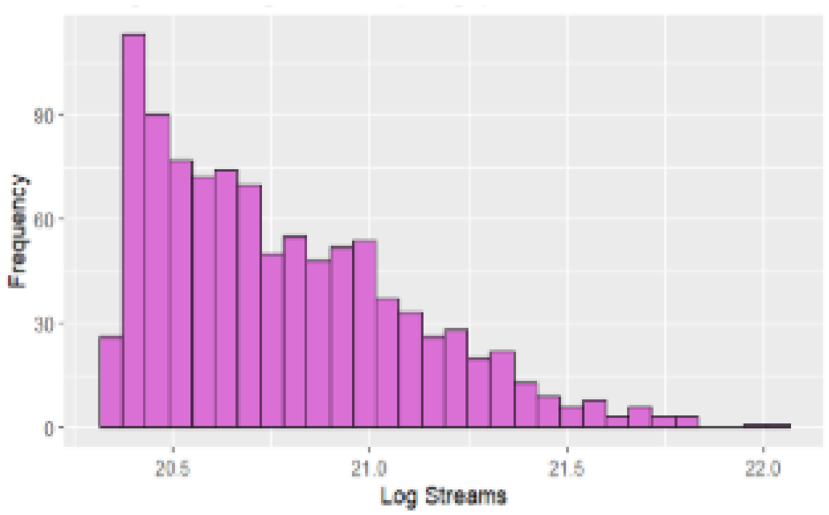


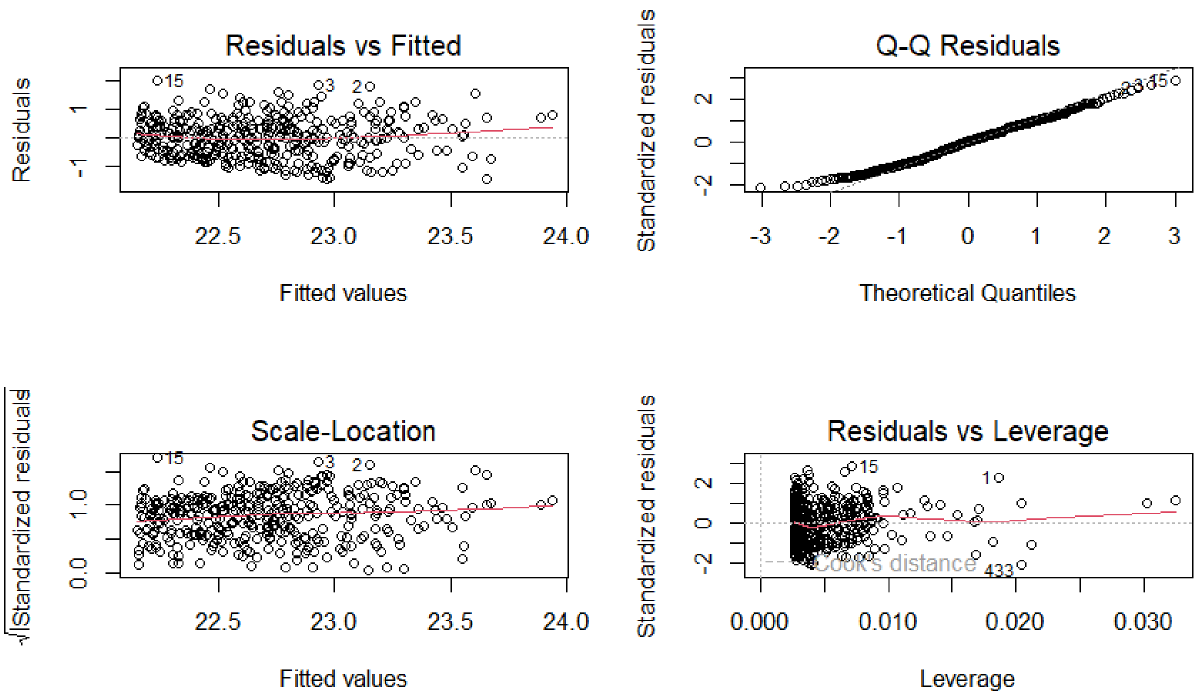
**Figure E2**

*Distribution of Song Streams After Logarithmic Transformation*



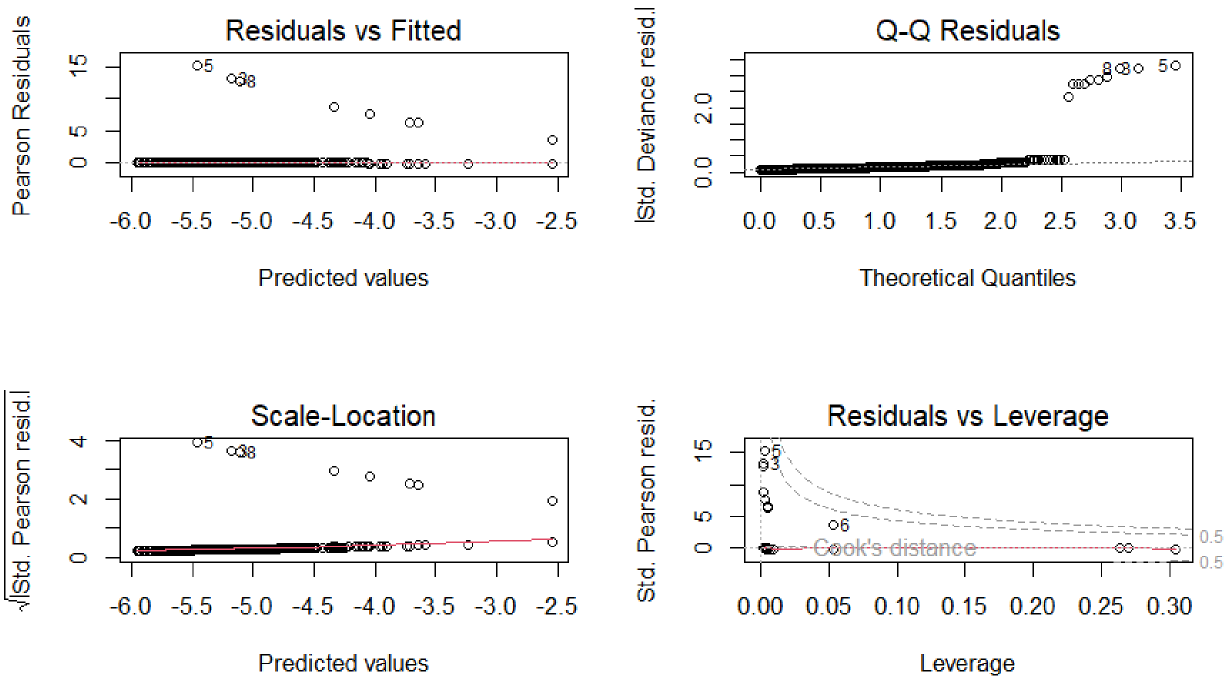
## Appendix F

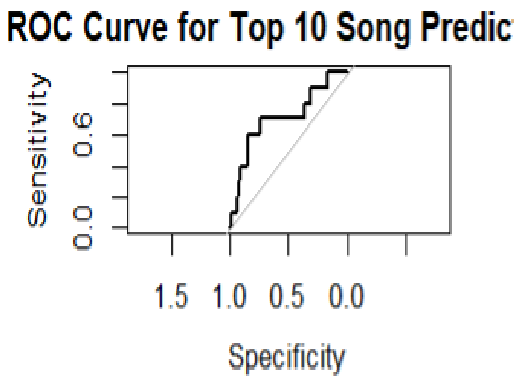
### Model Adequacy Plots for Linear Regression



## Appendix G

### Model Adequacy Plots for Logistic Regression





## Appendix H

## K-means Clustering Plots

**Figure H1**

*Elbow Curve to Determine Optimal K*

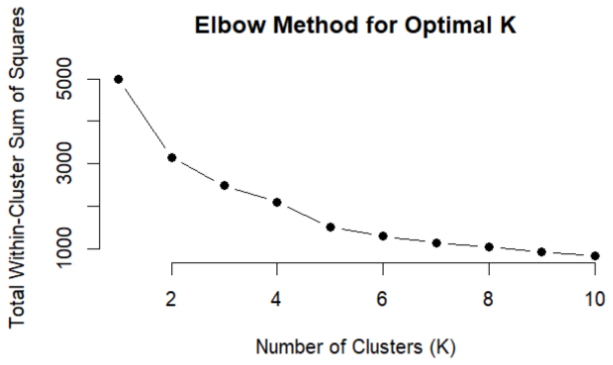


**Figure H2**

*Silhouette Plot for K-means Clustering*

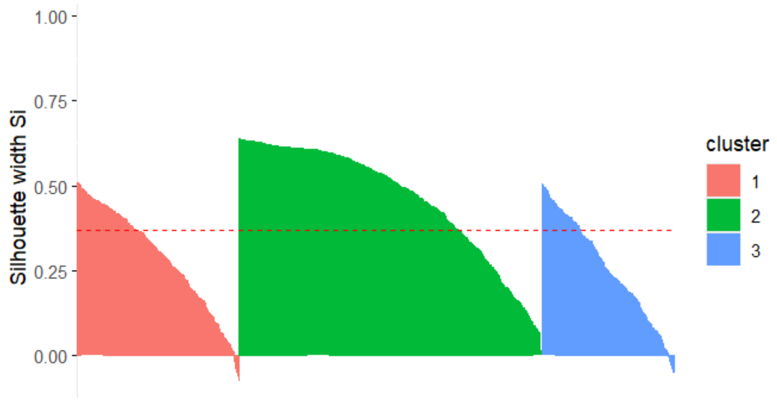
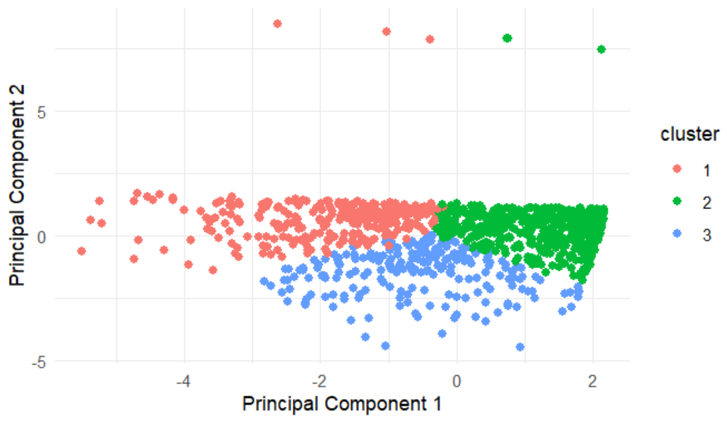


**Figure H3**

*PCA Projection of K-means Clustering*


Principal Component 2
Principal Component 1
cluster