



**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

**MH3511 Data Analysis with Computer
Group Project**

Numbers in Spotify and YouTube: Views, Likes and Streams

Done by:	Matriculation Number
SCSE ZANE YEE SUN	U2221504B
SCSE IAIN RODERICK TAY RONG YU	U2221382K
SCSE QIAN JIANHENG OSCAR	U2220109K
SCSE YAU JUN HAO	U2220332F
SCSE LEE CHENG YAO	U2221387J

Abstract:

The way that people listen to music has changed drastically in this digital age. Streaming giants like Spotify have become many users' go-to platforms for instant music gratification due to their large libraries of music. However, YouTube, a platform that is traditionally known for its video streaming popularity, has notably become a significant factor in music discovery and consumption, especially in the form of music videos. This intriguing dynamic poses a question to determine if there is a correlation between a song's performance on YouTube and its performance on Spotify. By measuring statistics from an artist's top songs, we aim to find any relationship between these two popularity measurements.

Table of Contents

1. Introduction	3
2. Data Description and Variable Selection	3
3. Description and Cleaning of Dataset	4
3.1. Summary statistics for the main variable of interest, Stream	4
3.2. Summary statistics for other variables	4
3.2.1 Danceability	5
3.2.2 Energy	5
3.2.3 Loudness	5
3.2.4 Speechiness	6
3.2.5 Acousticness	6
3.2.6 Valence	6
3.2.7 Tempo	7
3.2.8 Duration(second)	7
3.2.9 Views	7
3.2.10 Likes	8
3.2.11 Comments	8
3.2.12 Album_type	8
3.2.13 Key	9
3.2.14 Licensed	9
3.2.15 official_video	9
4. Statistical Analysis	10
4.1 Correlations between $\ln(\text{Stream})$ and other Continuous Variables	10
4.2 Statistical Tests	11
4.2.1 YouTube Views vs Spotify Streams	11
4.2.2 Relation between Streams and Music Video	12
4.2.3 Relation between Streams and Tempo	13
4.2.4 The single most important song characteristic that is affecting streams	13
4.3 Multiple Linear Regression	15
5. Conclusion and Discussion	16
6. Appendix	18
7. References	28

1. Introduction

Music has seen a significant shift in recent times from old record DVDs to now digital music being streamed on multiple platforms such as Spotify and YouTube. Spotify, primarily being an audio and media service provider, provides more than 602 million users with millions of songs from creators all over the world. On the other hand, YouTube, an American online video-sharing and social media platform offers a broad range of content including music videos. This poses an interesting question if there is a correlation between a song's streams on Spotify with its corresponding YouTube music video.

In our project, a dataset containing the statistics for the Top 10 songs of various Spotify artists and their respective YouTube music videos as of 7th February 2023. Based on this dataset, we aim to answer the following questions around artists' songs:

1. Does the views of YouTube music videos have a correlation to the number of streams on Spotify?
2. Do songs on Spotify without music videos on Youtube gain as much popularity as those with music videos?
3. Does a song's Spotify streams depend on the Tempo of the song?
4. Is there a single song characteristic that is more important in affecting the streams of the song?

This report will cover the data descriptions and analysis using R language. For each of our research objectives, we performed statistical analysis and concluded the most appropriate approach, together with explanations and elaborations.

2. Data Description and Variable Selection

The dataset, titled "Spotify and Youtube", is obtained from the online community of data scientists and machine learning engineers, Kaggle. The original CSV titled "Spotify_Youtube.csv" consists of 20718 songs and 26 variables for each song.

Before proceeding to data analysis, we first performed a preliminary data cleaning to ensure that:

- There are no NULL values for each of the columns pertaining to Spotify;
- Normalise all variables to a standardised format;
- Remove redundant variables, e.g. Url_spotify, Url_youtube;

After all the preparation, 20718 observations (songs) with 17 variables are retained for analysis:

1. Artist
2. Album_type
3. Danceability
4. Energy
5. Key (explain with ref to kaggle dataset, each number correspond to 1 key)
6. Loudness
7. Speechiness
8. Acousticness

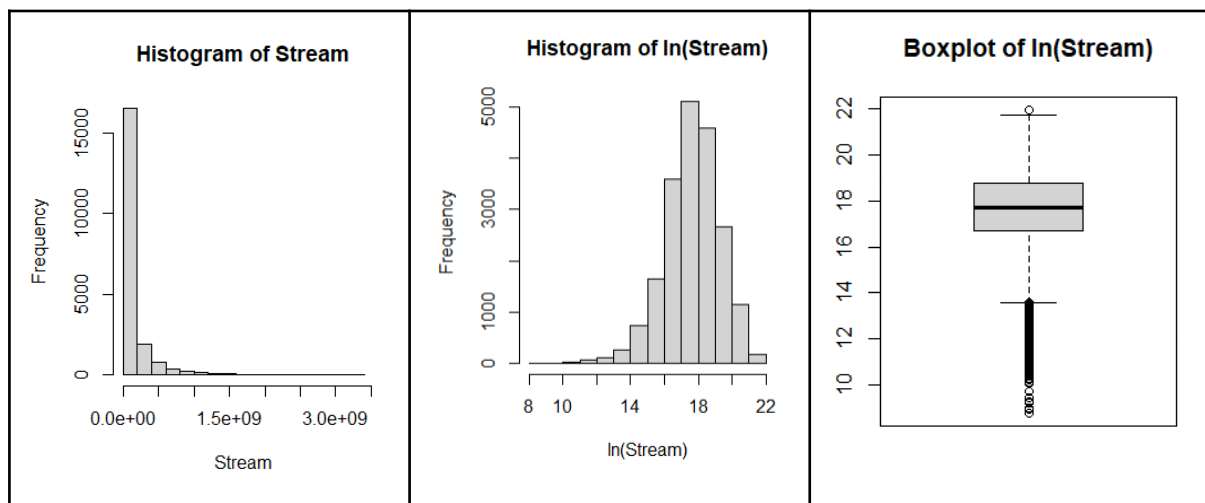
9. Valence
10. Tempo
11. Duration_ms
12. Views
13. Likes
14. Comments
15. Licensed
16. official_video
17. Stream

3. Description and Cleaning of Dataset

In this section, we shall look into the data in more detail. Each variable is investigated individually to look for possible outliers, and/or to perform a transformation to avoid highly skewed data.

3.1. Summary statistics for the main variable of interest, *Stream*

The following plots show the overall distribution of the variable *Stream*.

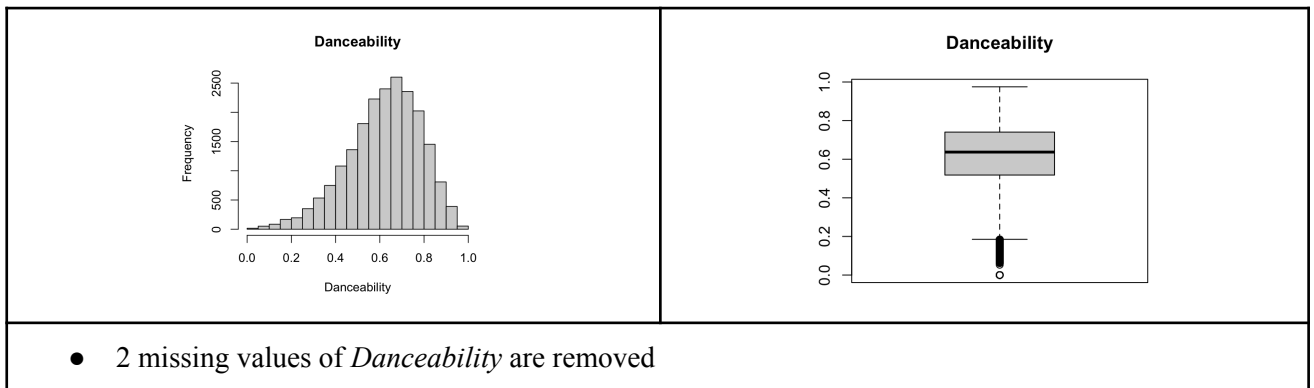


It appears that the variable *Stream* is highly skewed, hence we apply a log-transformation (base e) to the variable. The log-transformed data appears to have some outlying values at the left tail.

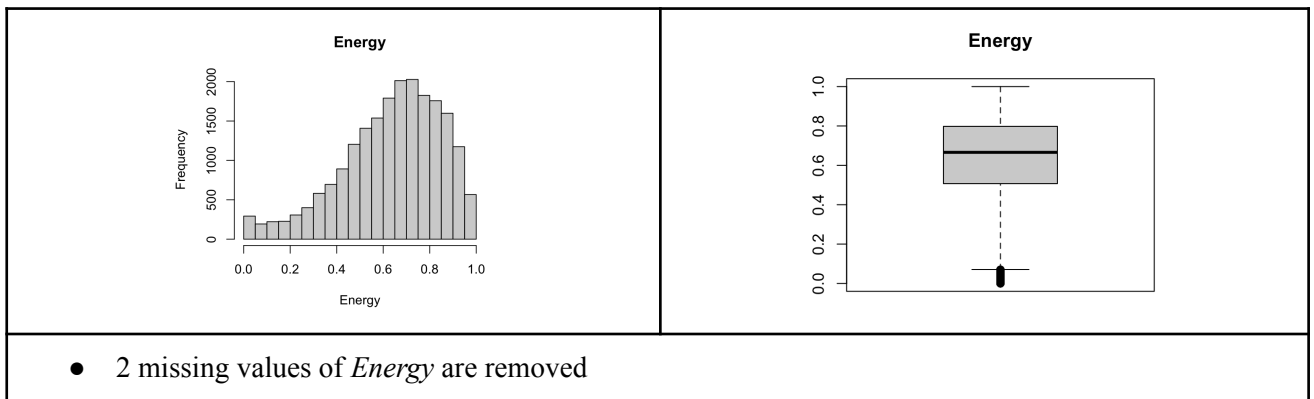
3.2. Summary statistics for other variables

The histogram, the boxplot, the transformation applied and the outliers removed from the variables are tabulated in the following subsections.

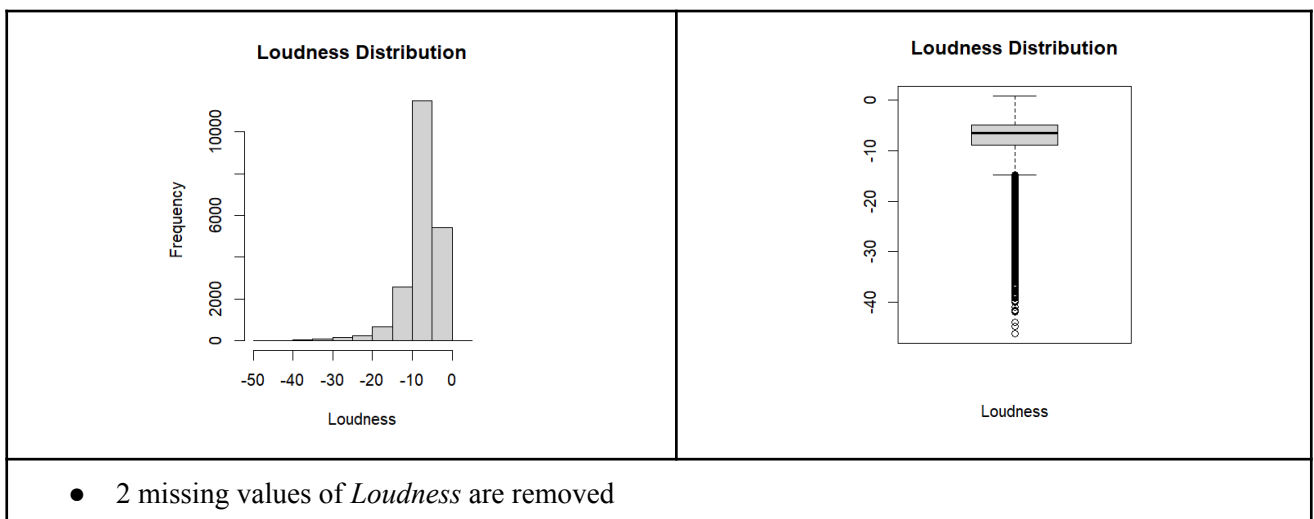
3.2.1 Danceability



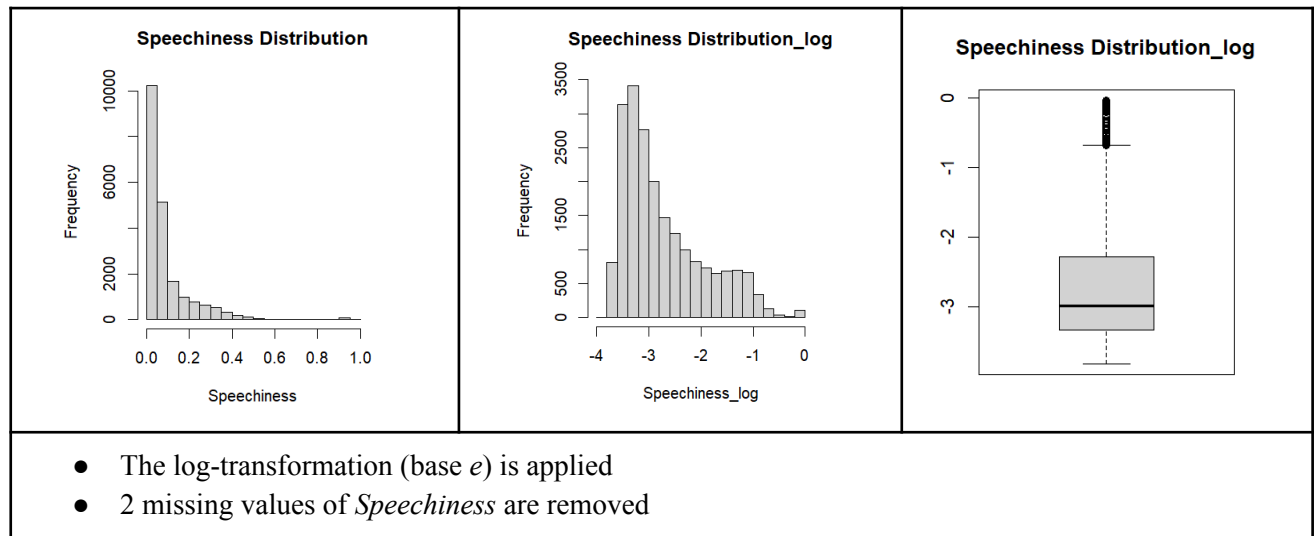
3.2.2 Energy



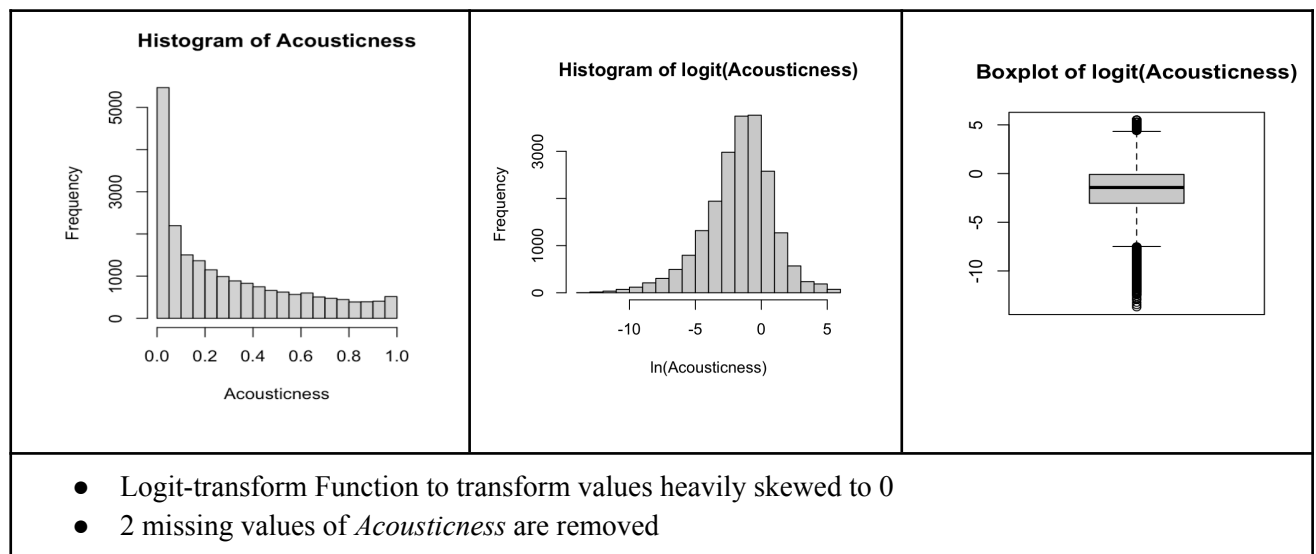
3.2.3 Loudness



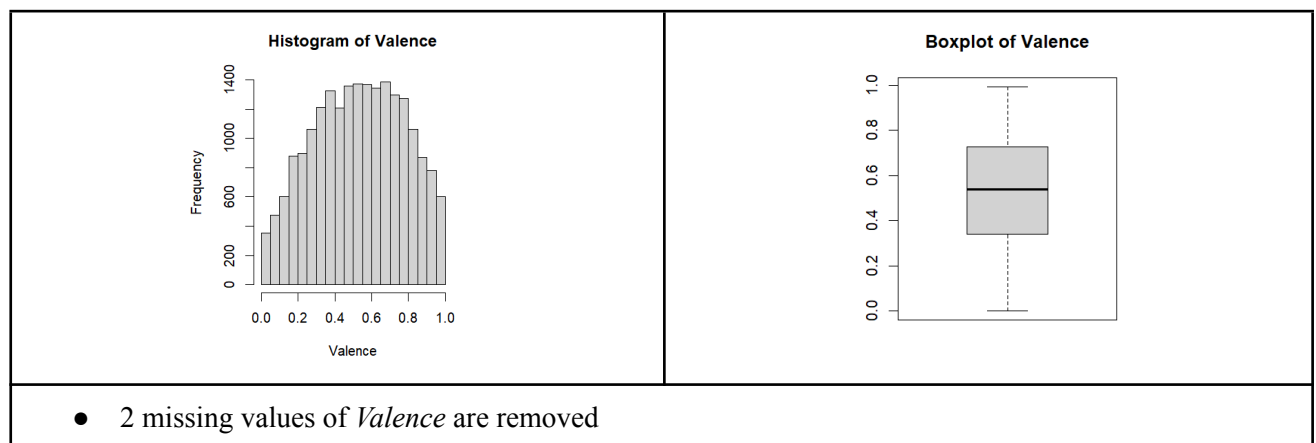
3.2.4 Speechiness



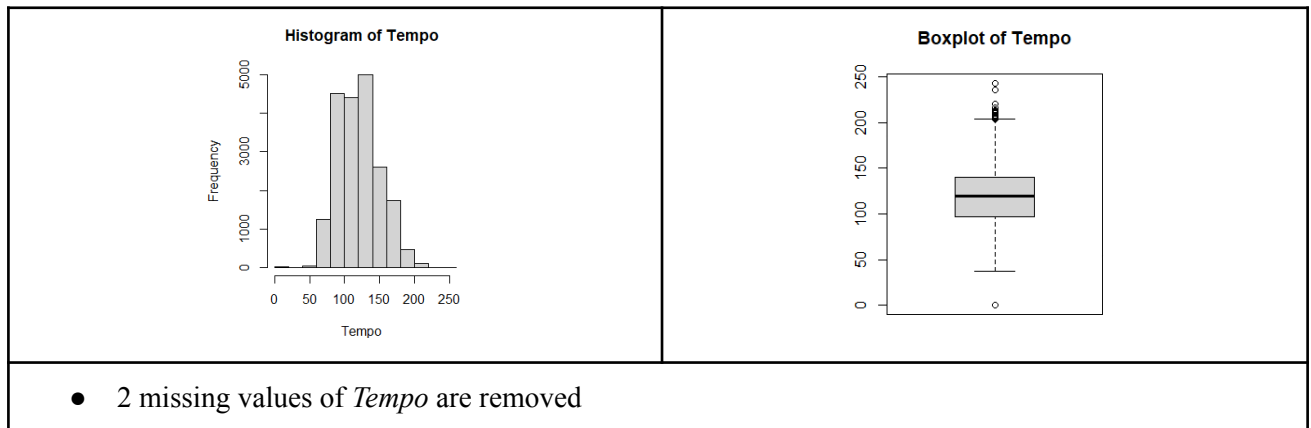
3.2.5 Acousticness



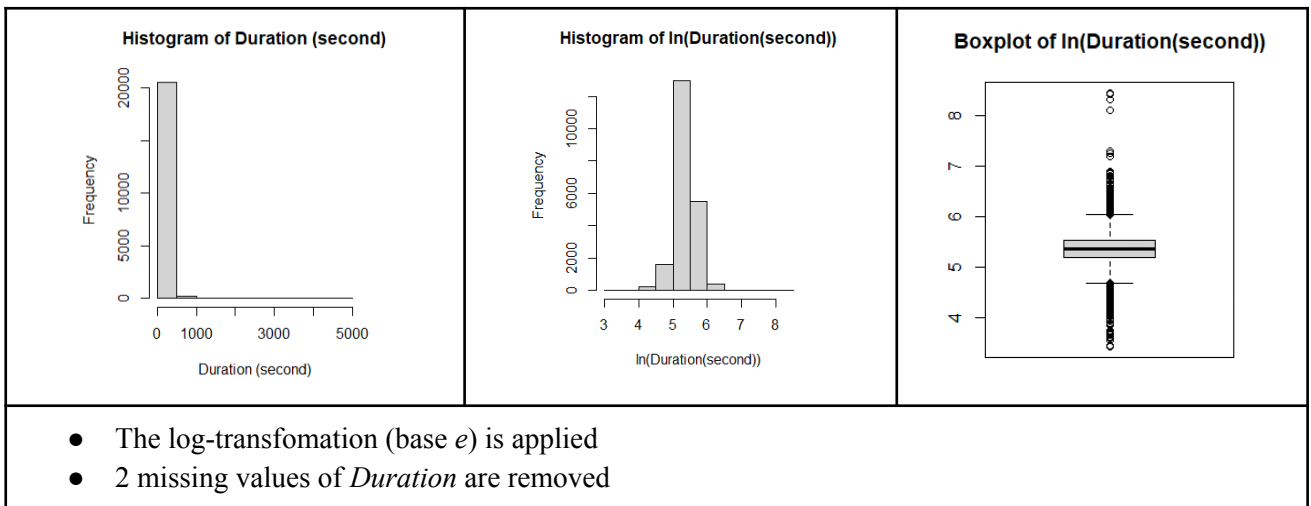
3.2.6 Valence



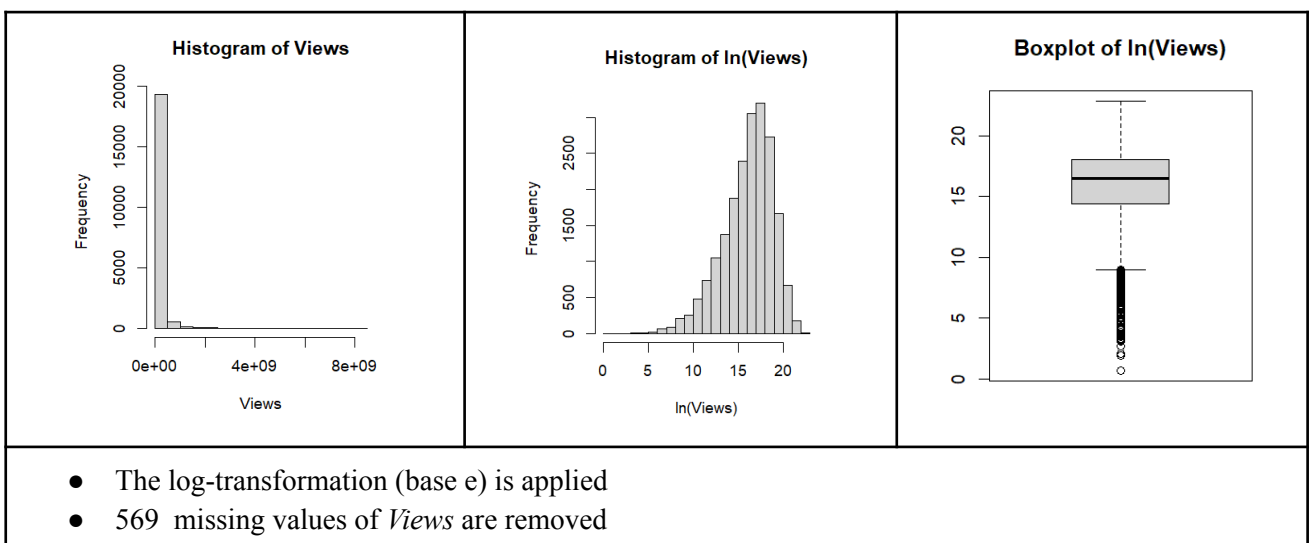
3.2.7 Tempo



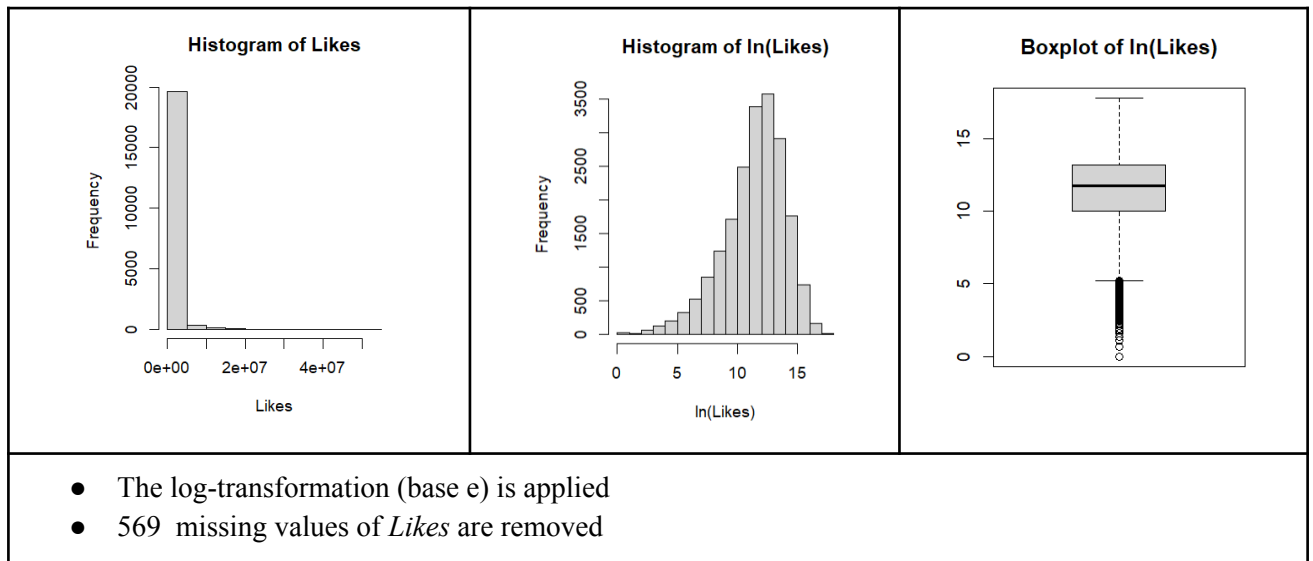
3.2.8 Duration(second)



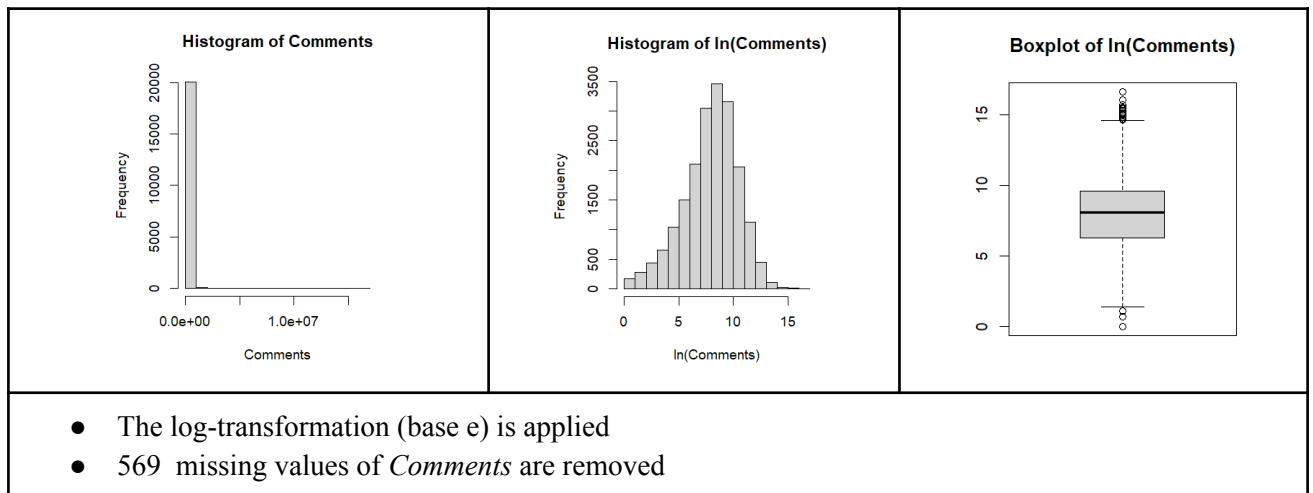
3.2.9 Views



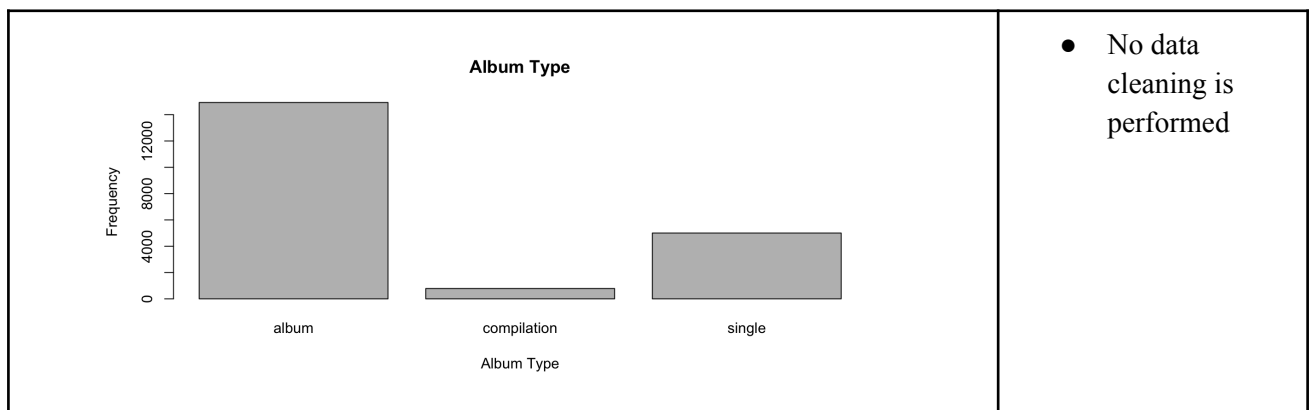
3.2.10 Likes



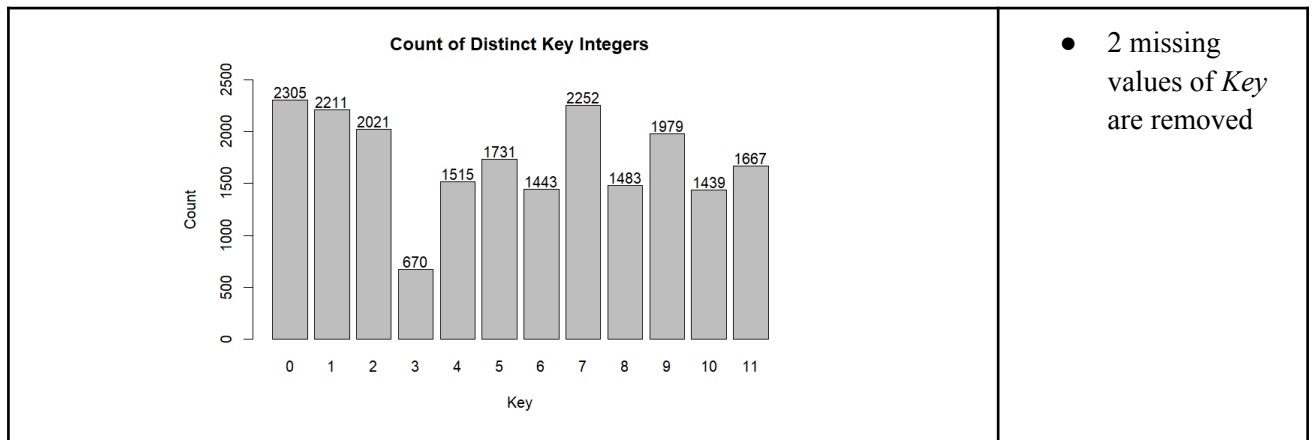
3.2.11 Comments



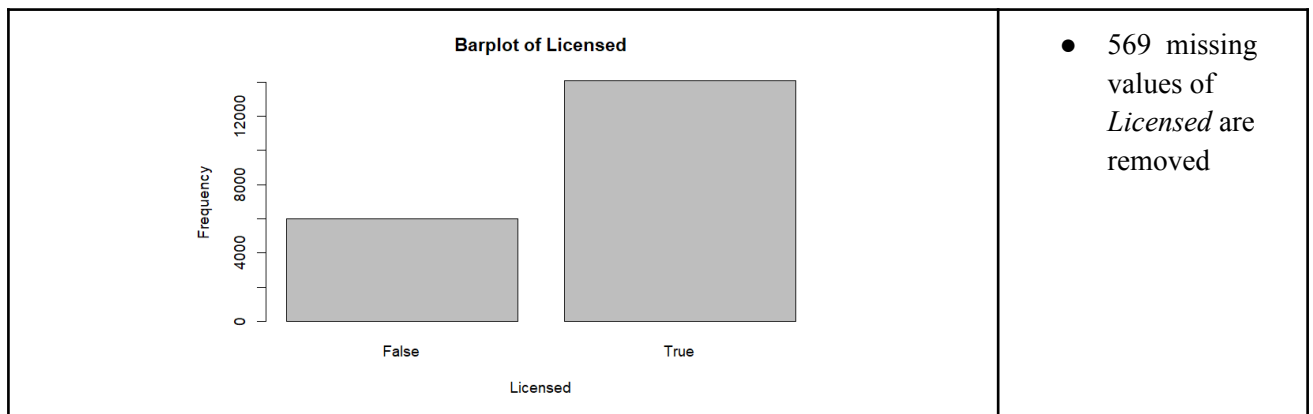
3.2.12 Album_type



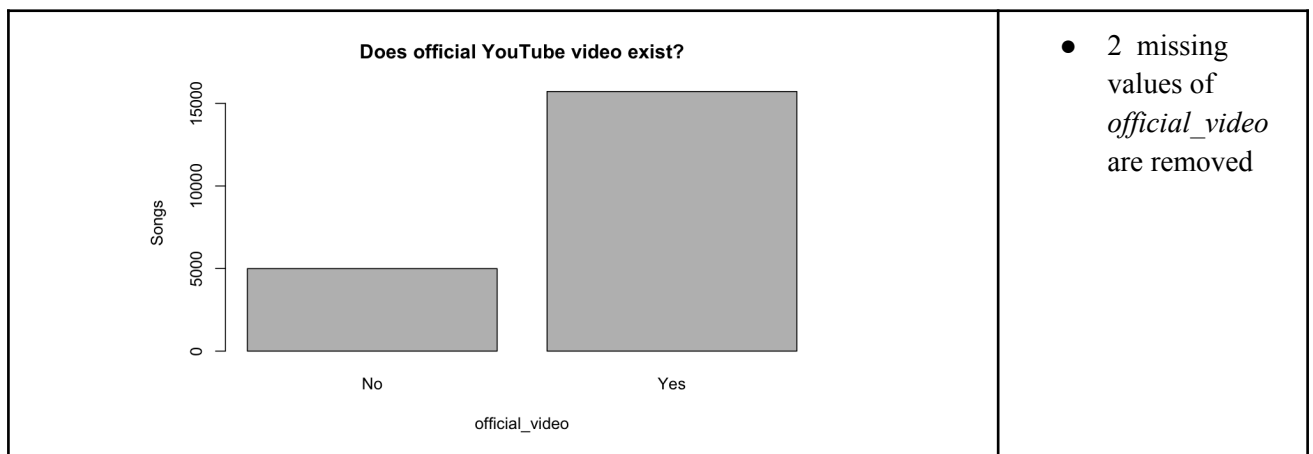
3.2.13 Key



3.2.14 Licensed



3.2.15 official_video

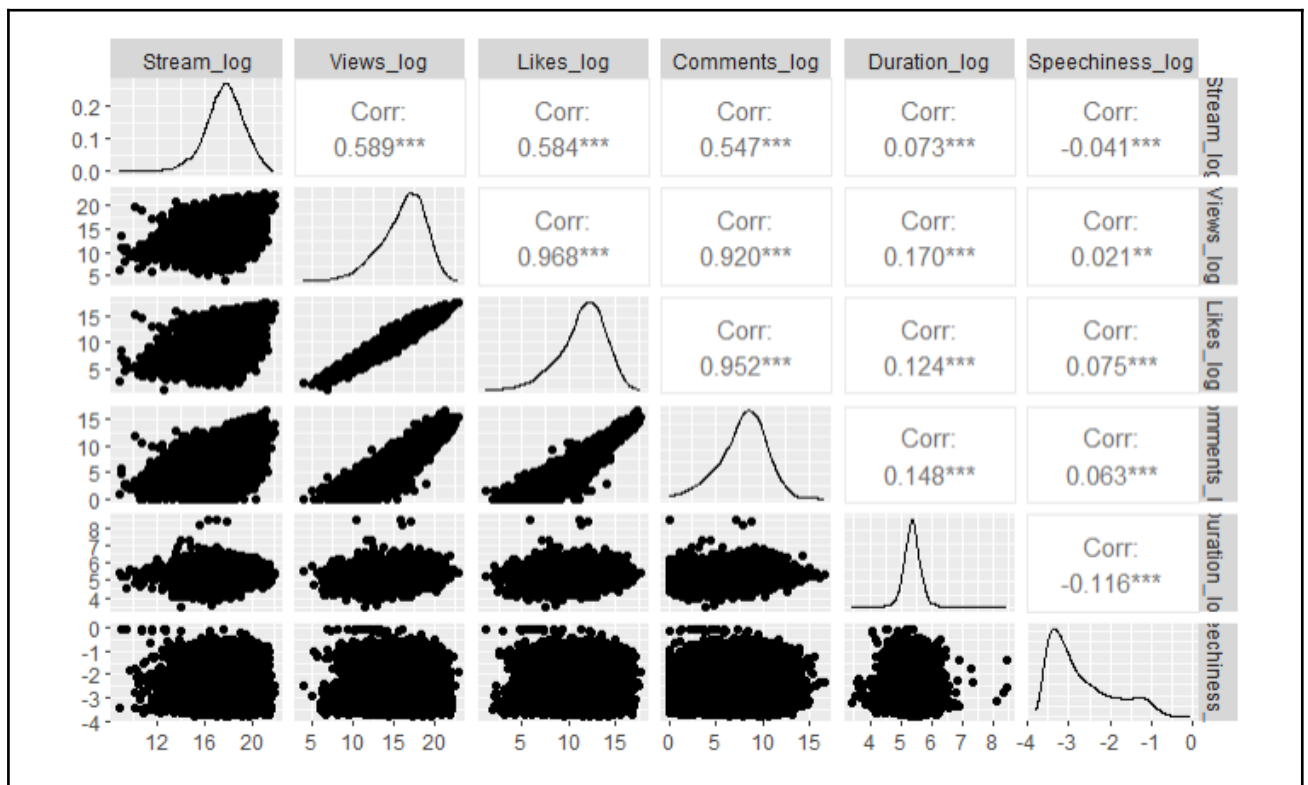


4. Statistical Analysis

4.1 Correlations between $\ln(\text{Stream})$ and other Continuous Variables

Scatter plots and correlation coefficients are useful in studying the possible linear relationships between a song's stream count and other variables.

Here we will investigate the correlation of $\log(\text{Stream})$ with $\log(\text{Views})$, $\log(\text{Likes})$, $\log(\text{Comments})$, $\log(\text{Duration(second)})$, $\log(\text{Speechiness})$.

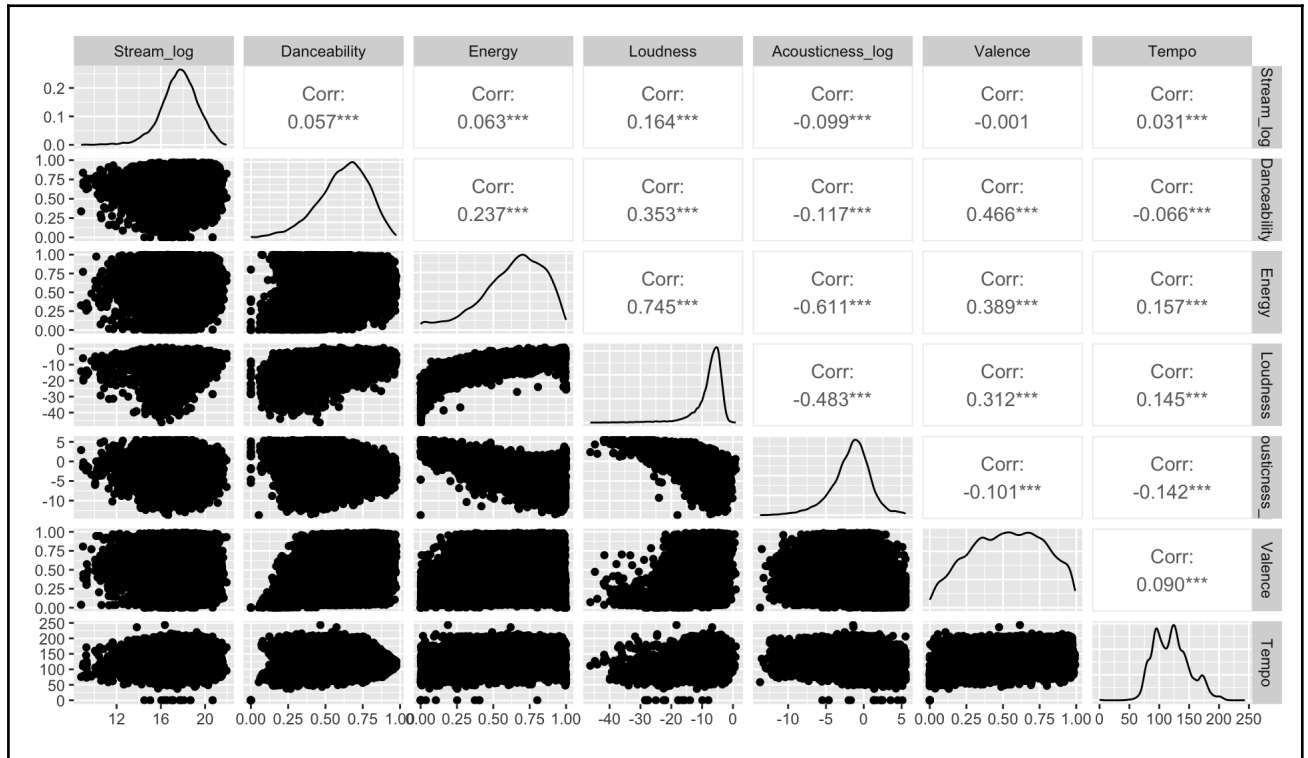


From the plots, it appears that $\log(\text{Stream})$ is more correlated with $\log(\text{Views})$, $\log(\text{Comments})$ and $\log(\text{Likes})$ than other variables.

Among the variables, there are a few interesting observations from this tabulation:

- $\log(\text{Views})$ and $\log(\text{Likes})$ are highly positively correlated (= 0.968)
- $\log(\text{Views})$ and $\log(\text{Comments})$ are highly positively correlated (= 0.920)
- $\log(\text{Likes})$ and $\log(\text{Comments})$ are highly positively correlated (= 0.952)

Next, we investigated the correlation of $\log(\text{Stream})$ with, Danceability , Energy , Loudness , $\log(\text{Acousticness})$, Valence , Tempo .



From the plots, it appears that $\log(\text{Stream})$ has a weak correlation with each audio feature, which is expected as there are many different factors influencing the likeability of the song, thus affecting its stream numbers.

Among the variables, there are a few interesting observations from this tabulation:

- *Loudness* and *Energy* are highly correlated ($= 0.745$)
- *Loudness* and $\log(\text{Acousticness})$ are inversely correlated ($= -0.483$)
- *Danceability* and *Valence* are correlated ($= 0.466$)
- *Energy* and $\log(\text{Acousticness})$ are inversely correlated ($= -0.611$)

4.2 Statistical Tests

4.2.1 YouTube Views vs Spotify Streams

In this section, we try to answer the question “Do songs tend to accumulate more Views on YouTube compared to Streams on Spotify?”

Since the size of the dataset is large (>30), we can assume CLT. The mean of Views and Streams is normally distributed.

Hence, we will perform a pairwise t-test to investigate the two hypotheses:

$$H_0: \mu_1 = \mu_2 \quad \text{against} \quad H_1: \mu_1 > \mu_2$$

```

Paired t-test

data: clean_data$log_Stream and clean_data$log_views
t = 98.315, df = 19531, p-value < 2.2e-16
alternative hypothesis: true mean difference is greater than 0
95 percent confidence interval:
 1.53182      Inf
sample estimates:
mean difference
 1.557885

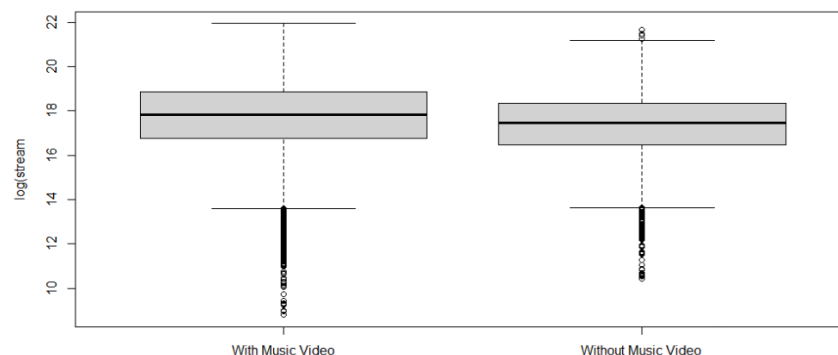
```

The p-value obtained is less than $2.2e-16$, with $\alpha = 0.05$, we reject H_0 in favour H_1 of as there is enough evidence to conclude that the mean of Stream is larger than mean of Views

4.2.2 Relation between Streams and Music Video

In this section, we try to answer the question “Do songs on Spotify without music videos on YouTube gain as much popularity as those with music videos?”

A Welch two-sample t-test will be conducted to determine whether the $\log(\text{Stream})$ is different for songs with or without music videos. The following plot illustrates the distribution of $\log(\text{Stream})$ grouped by music video.



Looking at the boxplot, we see that both groups seem to be normally distributed. Hence, the one-tailed Welch two-sample t-test is appropriate for testing:

$$H_0: \mu_1 = \mu_2 \quad \text{against} \quad H_1: \mu_1 > \mu_2$$

We first performed an F-test to check if the variance of $\log(\text{Stream})$ of the 2 groups is equal or not and concluded that variances are not equal according to the result shown below.

```

F test to compare two variances

data: official_video$log_Stream and non_official_video$log_Stream
F = 1.1215, num df = 15244, denom df = 4286, p-value = 3.573e-06
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 1.068678 1.176187
sample estimates:
ratio of variances
 1.121511

```

We then conducted a one-tailed t-test assuming variance to be not equal.

```
welch Two Sample t-test

data: official_video$log_Stream and non_official_videos$log_Stream
t = 14.785, df = 7214, p-value < 2.2e-16
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 0.3605478      Inf
sample estimates:
mean of x mean of y
 17.73451  17.32882
```

The p-value obtained is 2.2e-16. Therefore, we rejected H_0 at $\alpha = 0.05$. Thus, we can conclude that songs on Spotify with music videos are more popular than songs without.

4.2.3 Relation between Streams and Tempo

In this section, we try to answer the question “Does a song’s Spotify streams depend on the Tempo of the song?”.

A Pearson’s product-moment correlation test is conducted to check if $\log(\text{Stream})$ depends on Tempo, we then test:

$$H_0: \rho = 0 \quad \text{against} \quad H_1: \rho \neq 0$$

Where ρ is the correlation coefficient between $\log(\text{Stream})$ and Tempo.

The correlation test is conducted with the following results:

```
Pearson's product-moment correlation

data: data$log_Stream and data$Tempo
t = 4.1966, df = 19530, p-value = 2.722e-05
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.01599804 0.04402128
sample estimates:
cor
0.03001556
```

The p-value obtained is 2.722e-05. Therefore, we rejected H_0 at $\alpha = 0.05$. Thus, we can conclude that $\rho \neq 0$, and that $\log(\text{Stream})$ and Tempo are correlated.

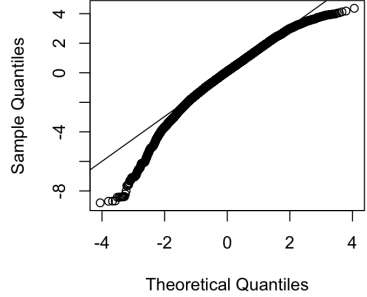
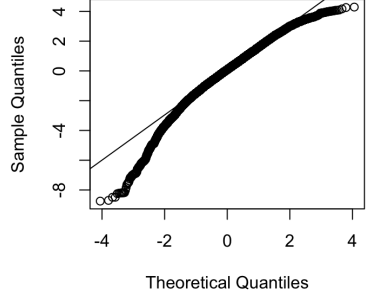
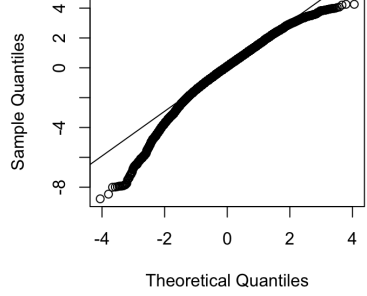
4.2.4 The single most important song characteristic that is affecting streams

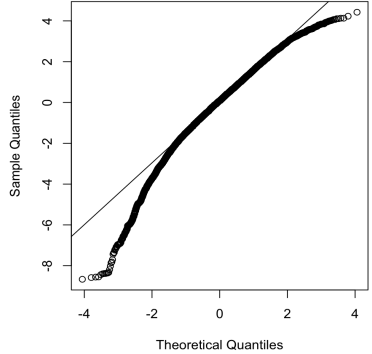
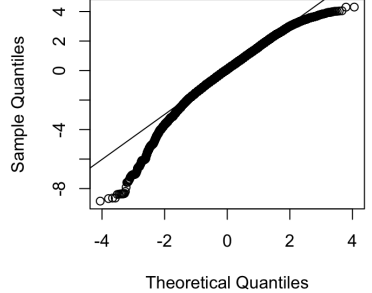
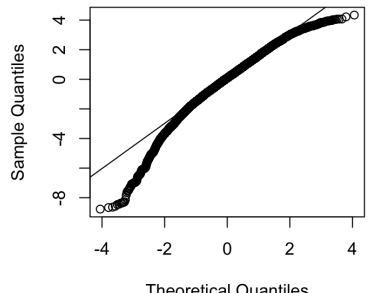
In this section, we try to answer the question “Is there a single song characteristic that is more important in affecting the streams of the song?” We now will perform a single linear regression analysis to determine which of the 5 musical characteristics could be used to model $\log(\text{stream})$ linearly.

$$\log(\text{Stream}) = \beta_0 + \beta_1 * X + \varepsilon$$

where X could be any one of Danceability, Energy, Loudness, $\log(\text{Acousticness})$, Valence or Tempo. The summary of the analysis is listed in the table below.

By comparing the R-squared and the residual plot, Loudness is determined to be the single most important song characteristic to model the log(Stream) using a simple linear model.

Variable (X)	Fitted Model, with Y being log(Stream)	p-value	R-squared	QQ-plot of residuals
Danceability	$\hat{Y} = 17.29 + 0.569X$	4.587e-16	0.003270	<p>Normal Q-Q Plot</p> 
Energy	$\hat{Y} = 17.33 + 0.4885X$	1.966e-19	0.004028	<p>Normal Q-Q Plot</p> 
Loudness	$\hat{Y} = 18.09 + 0.05841X$	6.834e-122	0.02700	<p>Normal Q-Q Plot</p> 

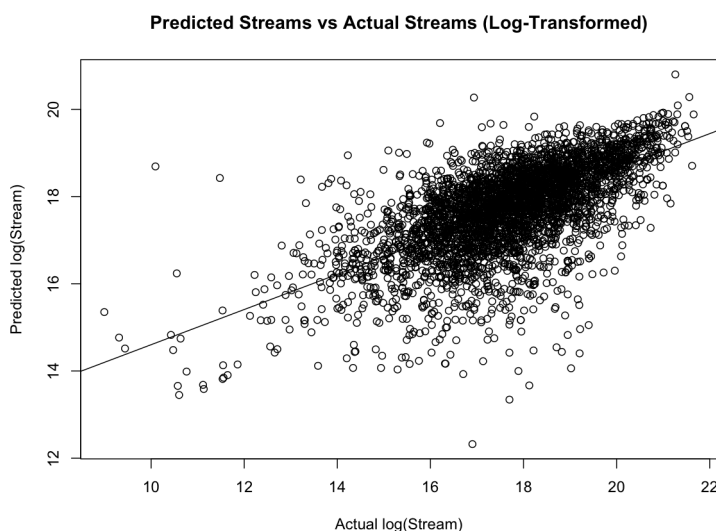
log(Acousticness)	$\hat{Y} = 17.53 - 0.06347X$	8.194e-45	0.009749	<p>Normal Q-Q Plot</p> 
Valence	$\hat{Y} = 17.64 - 0.004937X$	0.9169	5.405e-07	<p>Normal Q-Q Plot</p> 
Tempo	$\hat{Y} = 17.43 + 0.001724X$	1.101e-05	0.0009593	<p>Normal Q-Q Plot</p> 

4.3 Multiple Linear Regression

Recognising the plethora of variables affecting Streams, we will build a Multi-Linear Regression (MLR) model (See Appendix). From our initial MLR model, we have identified variables that were more significant ($p < 0.05$) in predicting the value of $\log(Stream)$ and used them to construct a less erroneous model (*RMSE of 1.26*) with its respective coefficients. The model is as follows:

$$\begin{aligned} \log(\text{Stream}) = & 13.94 - 0.54(\text{Energy}) + 0.02(\text{Loudness}) - 0.16(\log(\text{Speechiness})) \\ & - 0.05(\log(\text{Acousticness})) - 0.17(\text{Valence}) - 0.15(\log(\text{Duration})) \\ & + 0.10(\log(\text{Views})) + 0.38(\log(\text{Likes})) - 0.05(\log(\text{Comments})) \\ & - 0.30(\text{Album_type}) - 0.64(\text{official_video}) + \epsilon \end{aligned}$$

```
> # Predict results of the test set using our trained model
> test$predicted_log_Stream = predict(linreg_significant,test)
>
> # Plot Actual and Predicted log(Streams)
> plot(test$log_Stream,test$predicted_log_Stream,xlab = "Actual log(Stream)", ylab = "Predicted log(Stream)", main = "Predicted Streams vs Actual Streams (Log-Transformed)")
>
> # Plot a best fit line
> abline(lm(test$predicted_log_Stream~test$log_Stream))
```



```
> # Calculate residuals
> residuals <- test$log_Stream - test$predicted_log_Stream
>
> # Square the residuals
> squared_residuais <- residuals^2
>
> # Calculate mean squared residuals
> mean_squared_residuais <- mean(squared_residuais)
>
> # Calculate RMSE
> rmse <- sqrt(mean_squared_residuais)
>
> print(rmse)
[1] 1.263613
```

5. Conclusion and Discussion

In the realm of audio entertainment, the intersection of music and video content has become a pivotal arena for attracting audiences and generating revenue. The Spotify and YouTube dataset, offering insights as recent as February 2023, presents a rich tapestry of information that underscores the dynamic characteristics of songs and the music industry. As the music and video streaming platforms continue to evolve, understanding the factors influencing viewership, engagement, and content performance is crucial for stakeholders seeking to adapt and thrive in this competitive landscape. In this report, we delve into the analysis of the Spotify YouTube dataset, aiming to shed light on key aspects related to music streams, music video views, and the evolving digital media ecosystem.

We conclude that:

- Songs tend to have a higher number of Views on YouTube than Streams on Spotify
- Songs on Spotify with music videos are more popular than songs without
- The number of Spotify streams of a song depends on its tempo

Additionally, we found that although loudness had the highest significance in affecting a song's Spotify streams, it is still considerably weak and does not fully represent the most important audio feature that can accurately model a song's streams. As such, we conclude that each audio feature alone does not have a significant impact on the number of streams of a song. The true factor could be a combination of audio features, semantic meaning, social media trends and artist fanbase.

Although the results are intriguing, it must be noted that this report is only based on the top 10 songs of each artist. By focusing on the top 10 songs of each artist, we may have missed out on a broader understanding of each artist's discography or the overall music landscape. Additionally, the top 10 songs may not be representative of an artist's entire body of work. A wider analysis will still be required to fully understand the factors affecting a song's popularity.

6. Appendix

```
> # # Installation of packages
> # install.packages("caTools")
> # install.packages("GGally")
> library(caTools)
Warning: package 'caTools' was built under R version 4.3.3
> library(tidyr)
Warning: package 'tidyr' was built under R version 4.3.3
> library(GGally)
Warning: package 'GGally' was built under R version 4.3.3Loading required
package: ggplot2
Warning: package 'ggplot2' was built under R version 4.3.3Registered S3 method
overwritten by 'GGally':
  method from
  +.gg ggplot2
>
> ##### For .Rmd file #####
> # Data set source:
> data = read.csv("Spotify_Youtube.csv", header = T) # <- uncomment if run on
.Rmd file
>
> ##### For .R file #####
> # Data set source:
> # data = read.csv("# insert filepath here #", header = T) # <- uncomment if run
on .R file
> # Drop unused columns
> column_to_drop = c("Url_spotify", "Track", "Album", "Uri", "Url_youtube",
"Title", "Channel", "Description")
> data = data[!(names(data) %in% column_to_drop)]
>
> ##### For .R file #####
> # check for na in the official_video column, if there is, assign it to "FALSE"
> # data[is.na(data$official_video),]$official_video = "FALSE" #<- uncomment if
run on .R file
>
> ##### For .Rmd file #####
> # check for empty string in the official_video column, if there is, assign it
to "FALSE"
> data[which(data$official_video == ""),]$official_video = "FALSE" #<- uncomment
if run on .Rmd file
>
> # convert official_video to boolean
> data$official_video = as.logical(data$official_video) # Cast as Boolean
(Logical)
>
> # Acousticness, Instrumentalness, Liveness
> # Since number of NA values is small compared to number of data points, remove
them
> # Instrumentalness, Liveness excluded from analysis
> clean_data = data %>% drop_na(c("Acousticness", "Instrumentalness",
"Liveness"))
>
> # Logit-transform Function to transform values heavily skewed to 0
> logit = function(x) {log(x / (1 - x))}
>
> # Logit-transform Acousticness
> clean_data$log_Acousticness = logit(clean_data$Acousticness)
>
> c("Danceability", "Energy", "Valence", "Tempo", "Duration(second)", "Views",
"Likes", "Comments", "Album_type", "Licensed")
```

```
[1] "Danceability"      "Energy"            "Valence"           "Tempo"
"Duration(second)"
[6] "Views"            "Likes"            "Comments"          "Album_type"
"Licensed"
>
> # Key, Loudness, Speech, Valence
> key_row_todrop <- clean_data[is.na(clean_data$Key) | clean_data$Key == -1 ,]
> loudness_row_todrop <- clean_data[is.na(clean_data$Loudness),]
> speech_row_todrop <- clean_data[is.na(clean_data$Speechiness) |
clean_data$Speechiness < 0 | clean_data$Speechiness > 1,]
> valence_row_todrop <- clean_data[is.na(clean_data$Valence) | clean_data$Valence
< 0.0 | clean_data$Valence > 1.0,]
> rows_todrop <- unique(rbind(key_row_todrop$X, loudness_row_todrop$X,
speech_row_todrop$X, valence_row_todrop$X))
> clean_data <- clean_data[!clean_data$X %in% rows_todrop,]
>
> # Logit-transform Speechiness
> clean_data$log_Speechiness = logit(clean_data$Speechiness)
> clean_data = subset(clean_data, !is.infinite(log_Speechiness)) # Remove
infinite
>
> # Views, Likes, Comments, Licensed
> # drop rows in data_sub with null values
> clean_data = clean_data %>% drop_na(c("Views", "Likes", "Comments",
"Licensed"))
>
> # log transform
> clean_data$log_Views = log1p(clean_data$Views)
> clean_data$log_Likes = log1p(clean_data$Likes)
> clean_data$log_Comments = log1p(clean_data$Comments)
> clean_data
>
> # Tempo, Duration_ms
> # Drop NA
> clean_data = clean_data %>% drop_na(c("Tempo", "Duration_ms", "Stream"))
> clean_data
>
> # convert ms to seconds
> clean_data$Duration_s <- clean_data$Duration_ms/1000
>
> # Log-Transform
> clean_data$log_Duration_s = log(clean_data$Duration_s)
>
> # Dependent Variable: Stream
> # Drop all NA in Streams
> clean_data = clean_data %>% drop_na(Stream)
>
> # Normalising Streams (Log-transform)
> clean_data$log_Stream = log(clean_data$Stream)
>
> # Check summary and types of variables
> summary(clean_data)
```

X	Artist	Album_type	Danceability
Energy	Key		
Min. : 0	Length:19532	Length:19532	Min. :0.0532
:0.0000203	Min. : 0.000		Min.
1st Qu.: 5212	Class :character	Class :character	1st Qu.:0.5200
Qu.:0.5090000	1st Qu.: 2.000		1st
Median :10432	Mode :character	Mode :character	Median :0.6390
:0.6660000	Median : 5.000		Median
Mean :10408			Mean :0.6216
:0.6356005	Mean : 5.294		Mean
3rd Qu.:15624			3rd Qu.:0.7420
Qu.:0.7970000	3rd Qu.: 8.000		3rd

```

Max.      :20717                                Max.      :0.9750    Max.
:1.0000000    Max.      :11.000
    Loudness      Speechiness      Acousticness      Instrumentalness
Liveness
    Min.      :-46.251    Min.      :0.02200    Min.      :0.0000011    Min.      :0.0000000    Min.
:0.0145
    1st Qu.: -8.765    1st Qu.:0.03570    1st Qu.:0.0444000    1st Qu.:0.0000000    1st
Qu.:0.0940
    Median : -6.514    Median :0.05070    Median :0.1900000    Median :0.0000023
Median :0.1250
    Mean   : -7.621    Mean   :0.09548    Mean   :0.2887929    Mean   :0.0546007    Mean
:0.1911
    3rd Qu.: -4.927    3rd Qu.:0.10400    3rd Qu.:0.4700000    3rd Qu.:0.0004253    3rd
Qu.:0.2340
    Max.    : 0.920    Max.    :0.96400    Max.    :0.9960000    Max.    :0.9950000    Max.
:1.0000
    Valence      Tempo      Duration_ms      Views
Likes
    Min.      :0.0000    Min.      : 37.11    Min.      : 30985    Min.      :2.600e+01    Min.      :
0
    1st Qu.:0.3390    1st Qu.: 97.00    1st Qu.: 180333    1st Qu.:1.920e+06    1st Qu.:
22432
    Median :0.5360    Median :119.97    Median : 213291    Median :1.495e+07    Median :
128174
    Mean   :0.5294    Mean   :120.71    Mean   : 224713    Mean   :9.553e+07    Mean   :
670591
    3rd Qu.:0.7250    3rd Qu.:139.95    3rd Qu.: 251973    3rd Qu.:7.156e+07    3rd Qu.:
527366
    Max.    :0.9930    Max.    :243.37    Max.    :4676058    Max.    :8.080e+09    Max.
:50788652
    Comments      Licensed      official_video      Stream
log_Acousticness
    Min.      : 0    Length:19532    Mode :logical    Min.      :6.574e+03    Min.
:-13.7111
    1st Qu.: 533    Class :character    FALSE:4287    1st Qu.:1.781e+07    1st
Qu.: -3.0691
    Median : 3352    Mode :character    TRUE :15245    Median :4.980e+07    Median
: -1.4500
    Mean   : 27886                                Mean   :1.371e+08    Mean
: -1.7159
    3rd Qu.: 14541                                3rd Qu.:1.391e+08    3rd
Qu.: -0.1201
    Max.    :16083138                                Max.    :3.387e+09    Max.
: 5.5175
    log_Speechiness    log_Views      log_Likes      log_Comments      Duration_s
log_Duration_s
    Min.      :-3.794    Min.      : 3.296    Min.      : 0.00    Min.      : 0.000    Min.      :
30.98    Min.      :3.434
    1st Qu.: -3.296    1st Qu.:14.468    1st Qu.:10.02    1st Qu.: 6.280    1st Qu.:
180.33    1st Qu.:5.195
    Median : -2.930    Median :16.520    Median :11.76    Median : 8.117    Median :
213.29    Median :5.363
    Mean   : -2.606    Mean   :16.088    Mean   :11.42    Mean   : 7.743    Mean   :
224.71    Mean   :5.360
    3rd Qu.: -2.154    3rd Qu.:18.086    3rd Qu.:13.18    3rd Qu.: 9.585    3rd Qu.:
251.97    3rd Qu.:5.529
    Max.    : 3.288    Max.    :22.813    Max.    :17.74    Max.    :16.593    Max.
:4676.06    Max.    :8.450
    log_Stream
    Min.      : 8.791
    1st Qu.:16.695
    Median :17.723
    Mean   :17.645
    3rd Qu.:18.751

```

```

Max.      :21.943
> categorical = sapply(clean_data,is.character)
>
> # Convert categorical variables to numeric
> clean_data[,categorical] = lapply(clean_data[,categorical],function (x)
as.numeric(factor(x)))
> class(clean_data$Licensed)
[1] "numeric"
>
> # Independent Variables
> IV = c("Danceability", "Energy", "Loudness", "log_Speechiness",
"log_Acousticness", "Valence", "Tempo", "log_Duration_s", "log_Views",
"log_Likes", "log_Comments", "Album_type", "Key", "Licensed", "official_video")
> # Plot a Histogram distribution of mean Streams
> par(mfrow = c(1,2))
> hist(clean_data$Stream, xlab = "Streams", ylab = "Number of Songs", main =
"Histogram of Streams")
>
> # Plot a Histogram distribution of ln(Streams)
> par(mfrow = c(1,2))
> hist(clean_data$log_Stream, xlab = "ln(Streams)", ylab = "Number of Songs",
main = "Histogram of ln(Streams)")
>
> # Plot a Boxplot of ln(Streams)
> par(mfrow = c(1,2))
> boxplot(clean_data$log_Stream, xlab = "ln(Streams)", ylab = "Number of Songs",
main = "Boxplot of ln(Streams)")
> # plot histogram for Danceability in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Danceability, main = "Histogram of Danceability", xlab =
"Danceability")
>
> # plot boxplot for Danceability in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Danceability, main = "Boxplot of Danceability")
> # plot histogram for Energy in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Energy, main = "Histogram of Energy", xlab = "Energy")
>
> # plot boxplot for Energy in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Energy, main = "Boxplot of Energy")
> # plot histogram for Loudness in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Loudness, main = "Histogram of Loudness", xlab = "Loudness")
>
> # plot boxplot for Loudness in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Loudness, main = "Boxplot of Loudness")
> # plot histogram for Speechiness in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Speechiness, main = "Histogram of Speechiness", xlab =
"Speechiness")
>
> # plot histogram for log_Speechiness in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$log_Speechiness, main = "Histogram of ln(Speechiness)", xlab =
"ln(Speechiness)")
>
> # plot boxplot for Speechiness in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Speechiness, main = "Boxplot of Speechiness")
> # plot histogram for Acousticness in clean_data
> par(mfrow = c(1,2))

```

```
> hist(clean_data$Acousticness, main = "Histogram of Acousticness", xlab =
"Acousticness")
>
> # plot histogram for logit_Acousticness in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$log_Acousticness, main = "Histogram of logit(Acousticness)",
xlab = "ln(Acousticness)")
>
> # plot boxplot for logit_Acousticness in clean_data
> boxplot(clean_data$log_Acousticness, main = "Boxplot of logit(Acousticness)")
> # plot histogram for Valence in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Valence, main = "Histogram of Valence", xlab = "Valence")
>
> # plot boxplot for Valence in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Valence, main = "Boxplot of Valence")
> # plot histogram for Tempo in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Tempo, main = "Histogram of Tempo", xlab = "Tempo")
>
> # plot boxplot for Tempo in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$Tempo, main = "Boxplot of Tempo")
> # plot histogram for Duration_s in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Duration_s, main = "Histogram of Duration_s", xlab =
"Duration_s")
>
> # plot histogram for log_Duration_s in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$log_Duration_s, main = "Histogram of ln(Duration_s)", xlab =
"ln(Duration_s)")
>
> # plot boxplot for log_Duration_s in clean_data
> par(mfrow = c(1,2))
> boxplot(clean_data$log_Duration_s, main = "Boxplot of ln(Duration_s)")
> # plot histogram for Views in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$Views, main = "Histogram of Views", xlab = "Views")
>
> # plot histogram for log_Views in clean_data
> par(mfrow = c(1,2))
> hist(clean_data$log_Views, main = "Histogram of ln(Views)", xlab = "ln(Views)")
>
> par(mfrow = c(1,2))
> boxplot(clean_data$log_Views, main = "Boxplot of ln(Views)")
> par(mfrow = c(1,2))
> hist(clean_data$Likes, main = "Histogram of Likes", xlab = "Likes")
>
> par(mfrow = c(1,2))
> hist(clean_data$log_Likes, main = "Histogram of ln(Likes)", xlab = "ln(Likes)")
>
> par(mfrow = c(1,2))
> boxplot(clean_data$log_Likes, main = "Boxplot of ln(Likes)")
> par(mfrow = c(1,2))
> hist(clean_data$Comments, main = "Histogram of Comments", xlab = "Comments")
>
> par(mfrow = c(1,2))
> hist(clean_data$log_Comments, main = "Histogram of ln(Comments)", xlab =
"ln(Comments)")
>
> par(mfrow = c(1,2))
> boxplot(clean_data$log_Comments, main = "Boxplot of ln(Comments)")
```

```
> # plot bar plot for Album_type in clean_data
> barplot(table(clean_data$Album_type), main = "Barplot of Album_type", xlab =
"Album Type", ylab = "Frequency")
> # plot bar plot for Key in clean_data
> barplot(table(clean_data$Key), main = "Barplot of Distinct Key Integers", xlab
= "Key", ylab = "Count")
> # plot bar plot for Licensed in clean_data
> barplot(table(clean_data$Licensed), main = "Barplot of Licensed", xlab =
"Licensed", ylab = "Frequency")
> # plot bar plot for official_video in clean_data
> barplot(table(clean_data$official_video), main = "Barplot of official_video",
xlab = "Does Official Video Exist?", ylab = "Frequency")
> # First half of the multivariate ggpair plot
> # correlation of log(Stream) with log(Views), log(Likes), log(Comments),
log(Duration(second)), log(Speechiness)
> first_half = clean_data[c("log_Stream", "log_Views", "log_Likes",
"log_Comments", "log_Duration_s", "log_Speechiness")]
> ggpairs(first_half)
>
>
> # Second half of the multivariate ggpair plot
> # correlation of log(Stream) with, Danceability, Energy, Loudness,
log(Acousticness), Valence, Tempo
> second_half = clean_data[c("log_Stream", "Danceability", "Energy", "Loudness",
"log_Acousticness", "Valence", "Tempo")]
> ggpairs(second_half)
> # Perform paired t-test to check if the mean of Stream and Views are equal
> # H0: u1 = u2
> # H1: u1 > u2
> t.test(clean_data$log_Stream, clean_data$log_Views, alternative = "greater",
paired = TRUE)
```

Paired t-test

```
data: clean_data$log_Stream and clean_data$log_Views
t = 98.315, df = 19531, p-value < 2.2e-16
alternative hypothesis: true mean difference is greater than 0
95 percent confidence interval:
 1.53182      Inf
sample estimates:
mean difference
 1.557885
```

```
> # Extract Stream and official_video column from clean_data
> data <- clean_data[,c("log_Stream", "official_video")]
>
> # separate the data into two dataframes group by official_video
> official_video <- data[data$official_video == TRUE,]
> non_official_video <- data[data$official_video == FALSE,]
>
> # Plot boxplot of Stream for official_video and non_official_video
> boxplot(official_video$log_Stream, non_official_video$log_Stream, names =
c("With Music Video", "Without Music Video"), ylab = "log(stream)")
>
> # Perform F-test to check if the variance of Stream for official_video and
non_official_video are equal
> var.test(official_video$log_Stream, non_official_video$log_Stream)
```

F test to compare two variances

```
data: official_video$log_Stream and non_official_video$log_Stream
F = 1.1215, num df = 15244, denom df = 4286, p-value = 3.573e-06
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
```

```

1.068678 1.176187
sample estimates:
ratio of variances
1.121511

>
> # Perform t-test to check if the mean of Stream for official_video and
non_official_video are equal, with variance assumed to be not equal
> # H0: u1 = u2
> # H1: u1 > u2
> t.test(official_video$log_Stream, non_official_video$log_Stream, alternative =
"greater", var.equal = FALSE)

Welch Two Sample t-test

data: official_video$log_Stream and non_official_video$log_Stream
t = 14.785, df = 7214, p-value < 2.2e-16
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
0.3605478      Inf
sample estimates:
mean of x mean of y
17.73451 17.32882

> # Perform Pearson correlation test to check the correlation between log_Stream
and Tempo
> # H0: There is no correlation between log_Stream and Tempo (rho = 0)
> # H1: There is a correlation between log_Stream and Tempo (rho != 0)
> cor.test(clean_data$log_Stream, clean_data$Tempo)

Pearson's product-moment correlation

data: clean_data$log_Stream and clean_data$Tempo
t = 4.1966, df = 19530, p-value = 2.722e-05
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
0.01599804 0.04402128
sample estimates:
cor
0.03001556

> # 4.2.4 The single most important song characteristic that is affecting
streams
>
> # Fit the initial model
> model1 = lm(log_Stream ~ Danceability, data = clean_data)
> model2 = lm(log_Stream ~ Energy, data = clean_data)
> model3 = lm(log_Stream ~ Loudness, data = clean_data)
> model4 = lm(log_Stream ~ log_Acousticness, data = clean_data)
> model5 = lm(log_Stream ~ Valence, data = clean_data)
> model6 = lm(log_Stream ~ Tempo, data = clean_data)
>
> # Extract information for each model (replace "model_name" with actual names)
> models <- list(model1=model1, model2=model2, model3=model3, model4=model4,
model5=model5, model6=model6)
> for (model_name in names(models)) {
+   current_model <- models[[model_name]]
+
+ # Extract coefficients
+ coefficients <- summary(current_model)$coefficients
+
+ # Calculate p-values for coefficients
+ p_values <- summary(current_model)$coefficients[, 4]
+

```



```
+ # Extract p-values from summary
+ # Calculate R-squared
+ r_squared <- summary(current_model)$r.squared
+
+ # Print results
+ cat("Model:", model_name, "\n")
+ cat("Coefficients:\n", coefficients, "\n")
+ cat("P-values:\n", p_values, "\n")
+ cat("R-squared:", r_squared, "\n")
+ cat("\n") }
Model: model1
Coefficients:
 17.29214 0.5684041 0.04599693 0.07153399 375.9413 7.945931 0 2.031313e-15
P-values:
 0 2.031313e-15
R-squared: 0.003222445

Model: model2
Coefficients:
 17.34213 0.4772467 0.03702492 0.0552319 468.3906 8.64078 0 6.006299e-18
P-values:
 0 6.006299e-18
R-squared: 0.003808435

Model: model3
Coefficients:
 18.08952 0.05826348 0.02251264 0.002529009 803.527 23.03807 0 6.780636e-116
P-values:
 0 6.780636e-116
R-squared: 0.02645726

Model: model4
Coefficients:
 17.53788 -0.06270114 0.01411463 0.004572763 1242.532 -13.71187 0 1.357897e-42
P-values:
 0 1.357897e-42
R-squared: 0.009535213

Model: model5
Coefficients:
 17.65065 -0.009794083 0.02808617 0.04815195 628.4463 -0.2033995 0 0.8388249
P-values:
 0 0.8388249
R-squared: 2.118345e-06

Model: model6
Coefficients:
 17.44254 0.001681056 0.04976937 0.0004005796 350.4674 4.196558 0 2.721994e-05
P-values:
 0 2.721994e-05
R-squared: 0.0009009337

>
> # Generate qq-plot of residuals
> qqnorm(model1$residuals) # model1, Danceability
> qqline(model1$residuals)
>
> qqnorm(model2$residuals) # model2, Energy
> qqline(model2$residuals)
>
> qqnorm(model3$residuals) # model3, Loudness
> qqline(model3$residuals)
>
> qqnorm(model4$residuals) # model4, log_Acousticness
```

```

> qqline(model4$residuals)
>
> qqnorm(model5$residuals) # model5, Valence
> qqline(model5$residuals)
>
> qqnorm(model6$residuals) # model6, Tempo
> qqline(model6$residuals)
>
> # Compile into a DF for Regression
> regression_df = clean_data[c(IV,"log_Stream")]
> regression_df
>
> # Splitting the dataset into 80% train, 20% test
> regression_split = sample.split(regression_df$log_Stream, SplitRatio = 0.8)
> train = subset(regression_df, regression_split == TRUE)
> test = subset(regression_df, regression_split == FALSE)
> test
> names(train)
[1] "Danceability"      "Energy"            "Loudness"          "log_Speechiness"
"log_Acousticness"
[6] "Valence"           "Tempo"             "log_Duration_s"    "log_Views"
"log_Likes"
[11] "log_Comments"      "Album_type"        "Key"               "Licensed"
"official_video"
[16] "log_Stream"
> # Fit the linear regression model with the test set
> linreg =
lm(log_Stream~Danceability+Energy+Loudness+log_Speechiness+log_Acousticness+Valen
ce+Tempo+log_Duration_s+log_Views+log_Likes+log_Comments+Album_type+Key+Licensed+
official_video, data = train)
> summary(linreg)

Call:
lm(formula = log_Stream ~ Danceability + Energy + Loudness +
    log_Speechiness + log_Acousticness + Valence + Tempo + log_Duration_s +
    log_Views + log_Likes + log_Comments + Album_type + Key +
    Licensed + official_video, data = train)

Residuals:
    Min       1Q   Median       3Q      Max
-8.6205 -0.7858  0.0416  0.8157  7.5030

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   14.1377315  0.2296235  61.569 < 2e-16 ***
Danceability    0.0320972  0.0780128   0.411 0.680759
Energy        -0.5127604  0.0848483  -6.043 1.54e-09 ***
Loudness        0.0135515  0.0035905   3.774 0.000161 ***
log_Speechiness -0.1707137  0.0117487 -14.530 < 2e-16 ***
log_Acousticness -0.0397520  0.0051340  -7.743 1.03e-14 ***
Valence        -0.2059979  0.0519950  -3.962 7.47e-05 ***
Tempo          0.0008183  0.0003543   2.310 0.020907 *
log_Duration_s  -0.1902902  0.0341334  -5.575 2.52e-08 ***
log_Views       0.0910284  0.0163811   5.557 2.79e-08 ***
log_Likes       0.3796117  0.0213736  17.761 < 2e-16 ***
log_Comments   -0.0403999  0.0095581  -4.227 2.38e-05 ***
Album_type     -0.3014763  0.0125360 -24.049 < 2e-16 ***
Key            -0.0063351  0.0028064  -2.257 0.024000 *
Licensed       -0.0751573  0.0381679  -1.969 0.048956 *
official_videoTRUE -0.5581247  0.0431743 -12.927 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.258 on 15609 degrees of freedom

```

```
Multiple R-squared:  0.4173,      Adjusted R-squared:  0.4167
F-statistic: 745.2 on 15 and 15609 DF,  p-value: < 2.2e-16

>
> # Only use significant variables in the refined MLR model
> linreg_significant =
lm(log_Stream~Energy+Loudness+log_Speechiness+log_Acousticness+Valence+log_Duration_s+log_VIEWS+log_Likes+log_Comments+Album_type+official_video, data = train)
> summary(linreg_significant)

Call:
lm(formula = log_Stream ~ Energy + Loudness + log_Speechiness +
    log_Acousticness + Valence + log_Duration_s + log_VIEWS +
    log_Likes + log_Comments + Album_type + official_video, data = train)

Residuals:
    Min       1Q   Median       3Q      Max
-8.6832 -0.7875  0.0421  0.8149  7.4651

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    14.205102   0.209781   67.714 < 2e-16 ***
Energy         -0.515495   0.083539   -6.171 6.96e-10 ***
Loudness         0.014411   0.003466    4.158 3.22e-05 ***
log_Speechiness -0.168379   0.011345  -14.842 < 2e-16 ***
log_Acousticness -0.040360   0.005116   -7.889 3.23e-15 ***
Valence        -0.197706   0.046681   -4.235 2.30e-05 ***
log_Duration_s  -0.196573   0.033988   -5.784 7.45e-09 ***
log_VIEWS        0.088169   0.016308    5.406 6.52e-08 ***
log_Likes        0.381878   0.021284   17.942 < 2e-16 ***
log_Comments    -0.040013   0.009547   -4.191 2.79e-05 ***
Album_type      -0.300045   0.012442  -24.115 < 2e-16 ***
official_videoTRUE -0.625146   0.026926  -23.218 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.258 on 15613 degrees of freedom
Multiple R-squared:  0.4167,      Adjusted R-squared:  0.4163
F-statistic: 1014 on 11 and 15613 DF,  p-value: < 2.2e-16

>
> # Predict results of the test set using our trained model
> test$predicted_log_Stream = predict(linreg_significant,test)
>
> # Plot Actual and Predicted log(Streams)
> plot(test$log_Stream,test$predicted_log_Stream,xlab = "Actual log(Stream)",
ylab = "Predicted log(Stream)", main = "Predicted Streams vs Actual Streams
(Log-Transformed)")
>
> # Plot a best fit line
> abline(lm(test$predicted_log_Stream~test$log_Stream))
>
> # Calculate residuals
> residuals <- test$log_Stream - test$predicted_log_Stream
> # Square the residuals
> squared_residuais <- residuals^2
> # Calculate mean squared residuals
> mean_squared_residuais <- mean(squared_residuais)
> # Calculate RMSE
> rmse <- sqrt(mean_squared_residuais)
>
> print(rmse)
[1] 1.264073
```

7. References

Link to the online dataset:

<https://www.kaggle.com/datasets/salvatorerastelli/spotify-and-youtube/data>