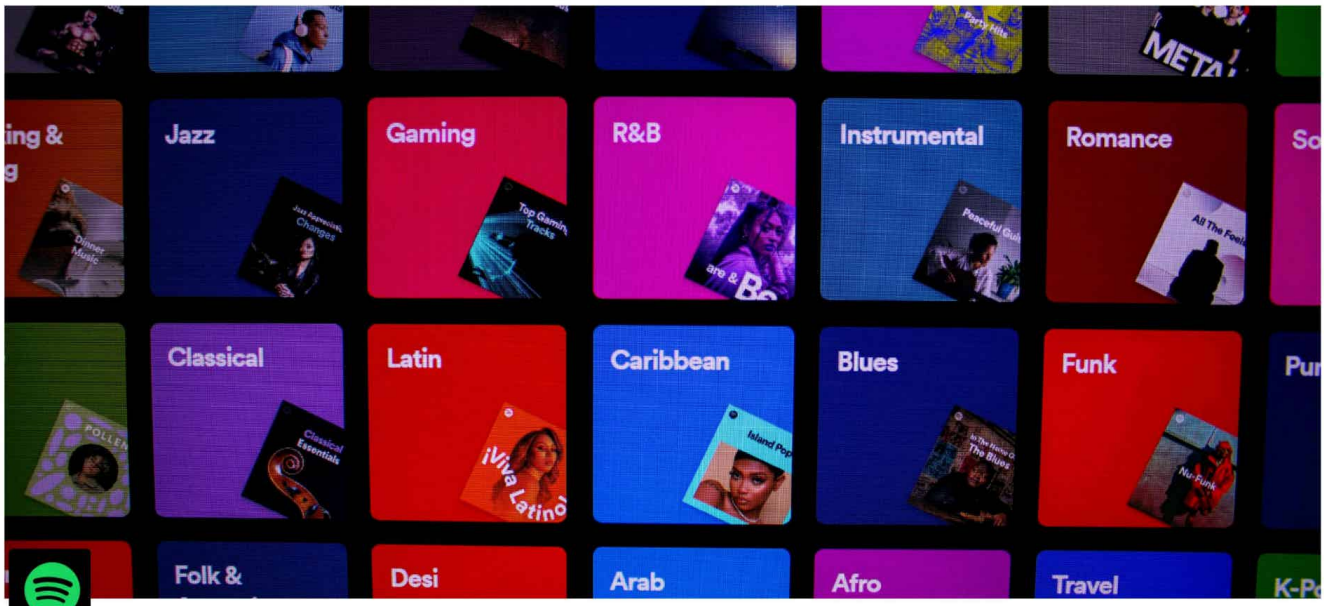


Report

Group Name : Foo

Spotify Data Analysis

1. Introduction

1.1 Overview

Music has always held a special place in human culture, serving as a medium for expression, emotion, connection, and a universal language. With the arrival of digital streaming platforms like Spotify, accessing a vast music library has become easier than ever before. Spotify, being one of the largest platforms globally, provides a unique opportunity to explore the world of music and its impact on our lives.

One of the reasons for choosing Spotify is because of the following facts :

> *"Today, more listeners than ever can discover, manage, and enjoy over 100 million tracks, 5 million podcasts titles, and 350,000 audiobooks a la carte on Spotify. We are the world's most popular audio streaming subscription service with more than 602 million users, including 236 million subscribers in more than 180 markets."*
>
> -
>
> **Spotify Newsroom**

The following are the objectives of this project :

1. Song-Recommender System

The desire to listen to a song with a similar mood and vibe while enjoying music is common among listeners. As a result, recommending songs based on the one currently being played has become a highly valued feature for major song-streaming platforms. Given the significant market demand, we are inspired to develop such a system using the available dataset.

2. Regression Models for Stream Prediction

Understanding the factors influencing number of streams through predictive modeling enables us to decode audience preferences, assist artists in content creation, and optimize music streaming platforms' recommendations. Hence it would be great to understand what all factors are there affecting the number of streams of a particular song.

3. Predicting Billboard Year-End Hot 100 Inclusion

The goal is to determine if we can utilize a machine learning model to predict whether a song will make it onto Billboard's Year-End Hot 100 list solely based on the song's characteristics.

2. About our dataset

2.1 Sources of our data collection

2.1.1 Most Streamed Spotify Songs 2023

This dataset contains a comprehensive list of the most famous songs of 2023 as listed on Spotify. The dataset offers a wealth of features which provides insights into each song's attributes, popularity, and presence on various music platforms.

2.1.2 19,000 Spotify Songs

This is a curated compilation of 19,000 Spotify tracks and their characteristics spanning the years 1964 to 2018.

2.1.3 Wikipedia of Billboard Year-End Top 100 List

For this source, we utilized Python scripting to scrape data from Wikipedia, explicitly targeting the Billboard Year-End Top 100 singles list. This comprehensive approach enabled us to gather information on the top songs for each year from 2000 to 2018. This data plays a vital role in our analysis, aiding us in distinguishing between songs that made it to the top 100 and those that did not.

2.2 Dataset Summary

2.2.1 Most Streamed Spotify Songs 2023

This dataset offers a comprehensive look at the most streamed songs of 2023 as listed on Spotify. Here's a breakdown of its components:

1. **Basic Information**:
   - **track_name**: Name of the song.
   - **artist(s)_name**: Name of the artist(s) associated with the song.
   - **artist_count**: Number of artists contributing to the song.
   - **released_year/month/day**: Date of release of the song.
2. **Platform Presence**:
   - **in_spotify_playlists**: Number of Spotify playlists the song is included in.
   - **in_spotify_charts**: Presence and rank of the song on Spotify charts.
   - **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.
3. **Streaming Statistics**:
   - **streams**: Total number of streams on Spotify.
4. **Musical Attributes**:
   - **bpm**: Beats per minute, indicating the tempo of the song.
   - **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.

```
dat.describe().T
```
✓ 0.1s

	count	mean	std	min	25%	50%	75%	max
artist_count	953.0	1.556139	0.893044	1.0	1.0	1.0	2.0	8.0
released_year	953.0	2018.238195	11.116218	1930.0	2020.0	2022.0	2022.0	2023.0
released_month	953.0	6.033578	3.566435	1.0	3.0	6.0	9.0	12.0
released_day	953.0	13.930745	9.201949	1.0	6.0	13.0	22.0	31.0
in_spotify_playlists	953.0	5200.124869	7897.608990	31.0	875.0	2224.0	5542.0	52898.0
in_spotify_charts	953.0	12.009444	19.575992	0.0	0.0	3.0	16.0	147.0
in_apple_playlists	953.0	67.812172	86.441493	0.0	13.0	34.0	88.0	672.0
in_apple_charts	953.0	51.908709	50.630241	0.0	7.0	38.0	87.0	275.0
in_deezer_charts	953.0	2.666317	6.035599	0.0	0.0	0.0	2.0	58.0
bpm	953.0	122.540399	28.057802	65.0	100.0	121.0	140.0	206.0
danceability_%	953.0	66.969570	14.630610	23.0	57.0	69.0	78.0	96.0
valence_%	953.0	51.431270	23.480632	4.0	32.0	51.0	70.0	97.0
energy_%	953.0	64.279119	16.550526	9.0	53.0	66.0	77.0	97.0
acousticness_%	953.0	27.057712	25.996077	0.0	6.0	18.0	43.0	97.0
instrumentalness_%	953.0	1.581322	8.409800	0.0	0.0	0.0	0.0	91.0
liveness_%	953.0	18.213012	13.711223	3.0	10.0	12.0	24.0	97.0
speechiness_%	953.0	10.131165	9.912888	2.0	4.0	6.0	11.0	64.0

Summary of numerical features of Most-Streamed-Songs-2023 Dataset

2.2.2 19,000 Spotify Songs

This dataset from Kaggle has 18,835 entries of spotify songs with all the same characteristics of song as that of the above dataset

```
1  dat.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
Index: 18835 entries, Boulevard of Broken Dreams to Up to Me
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   song_popularity   18835 non-null  int64
 1   song_duration_ms  18835 non-null  int64
 2   acousticness      18835 non-null  float64
 3   danceability      18835 non-null  float64
 4   energy            18835 non-null  float64
 5   instrumentalness  18835 non-null  float64
 6   key               18835 non-null  int64
 7   liveness          18835 non-null  float64
 8   loudness          18835 non-null  float64
 9   audio_mode        18835 non-null  int64
 10  speechiness       18835 non-null  float64
 11  tempo             18835 non-null  float64
 12  time_signature    18835 non-null  int64
 13  audio_valence     18835 non-null  float64
dtypes: float64(9), int64(5)
memory usage: 2.2+ MB
```

2.2.3 Billboard Top-100

This is the scraped dataset which has 1900 entries of Top-100 songs across the year 2000-2018. This dataset is merged with 19,000 spotify songs to obtain the Top100 Predictor Dataset stated below. In the file named *billboard_top100_scrape.ipynb,* we scraped data on the top 100 songs from the Billboard charts from Wikipedia spanning from 2000 to 2018.

```
dat.head()
```

✓ 0.0s

	Title	Artist	Year
0	Breathe	Faith Hill	2000
1	Smooth	Santana	2000
2	Maria Maria	Santana	2000
3	I Wanna Know	Joe	2000
4	Everything You Want	Vertical Horizon	2000

First 5 rows of Billboard-Top-100 Dataset

2.2.4 Top100 Predictor Dataset (2000-2018)

Data Preparation

Subsequently, we merged the billboard data with a Kaggle dataset featuring information on 19,000 songs , resulting in a new dataset named *"songs.csv."* This dataset contains three columns: “Artist”, “Title”, and a binary indicator called “Top100”, where 1 signifies inclusion in the Billboard list and 0 denotes otherwise. Now, for each song entry, we utilized the Spotify API to retrieve their URIs and then further scraping their characteristics such as Danceability, Tempo, Energy, Key, etc. This final dataset is called *“songs_complete_data.csv”.* This dataset has the following features :

```
df.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9226 entries, 0 to 9225
Data columns (total 19 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Artist            9226 non-null   object
 1   Title             9226 non-null   object
 2   Top100            9226 non-null   int64
 3   URI               9226 non-null   object
 4   Danceability      9226 non-null   float64
 5   Energy            9226 non-null   float64
 6   Key               9226 non-null   int64
 7   Loudness          9226 non-null   float64
 8   Mode              9226 non-null   int64
 9   Speechiness       9226 non-null   float64
 10  Acousticness      9226 non-null   float64
 11  Instrumentalness  9226 non-null   float64
 12  Liveness          9226 non-null   float64
 13  Valence           9226 non-null   float64
 14  Tempo             9226 non-null   float64
 15  Duration          9226 non-null   int64
 16  Time_Signature    9226 non-null   int64
 17  Release_Year      9226 non-null   int64
 18  Genre             9226 non-null   object
dtypes: float64(9), int64(6), object(4)
memory usage: 1.3+ MB
```

```
df.describe().T
```

0.0s

	count	mean	std	min	25%	50%	75%	max
Top100	9226.0	0.162692	0.369105	0.000000	0.000000	0.000000	0.000000	1.000
Danceability	9226.0	0.624711	0.154501	0.061700	0.526000	0.632000	0.736000	0.981
Energy	9226.0	0.662369	0.205162	0.002890	0.525000	0.694000	0.827000	0.997
Key	9226.0	5.334815	3.582050	0.000000	2.000000	5.000000	8.000000	11.000
Loudness	9226.0	-7.097625	3.367559	-36.729000	-8.674000	-6.316500	-4.782250	1.342
Mode	9226.0	0.644917	0.478565	0.000000	0.000000	1.000000	1.000000	1.000
Speechiness	9226.0	0.095369	0.095961	0.022800	0.036400	0.052700	0.108750	0.869
Acousticness	9226.0	0.234841	0.274592	0.000001	0.019225	0.111000	0.370750	0.996
strumentalness	9226.0	0.040521	0.149607	0.000000	0.000000	0.000005	0.000888	0.982
Liveness	9226.0	0.181757	0.145542	0.011900	0.092700	0.123000	0.229000	0.986
Valence	9226.0	0.536754	0.240879	0.024600	0.349000	0.535000	0.733000	0.990
Tempo	9226.0	121.393626	28.629974	46.591000	98.807000	120.032500	139.846500	216.115
Duration	9226.0	224414.087362	57226.121742	55213.000000	191373.000000	217614.500000	249142.000000	1799347.000
Time_Signature	9226.0	3.959462	0.289816	1.000000	4.000000	4.000000	4.000000	5.000
Release_Year	9226.0	2007.549426	13.697807	1945.000000	2002.000000	2014.000000	2018.000000	2020.000

3. Song Recommender System

The features that will be of use to build such a system will be the musical features such as acousticness, speechiness, valence etc. as well as the artists of the song. An artist play as important role here since they tend to have a major fan-following and people prefer to listen to their songs more than others despite their similarity.

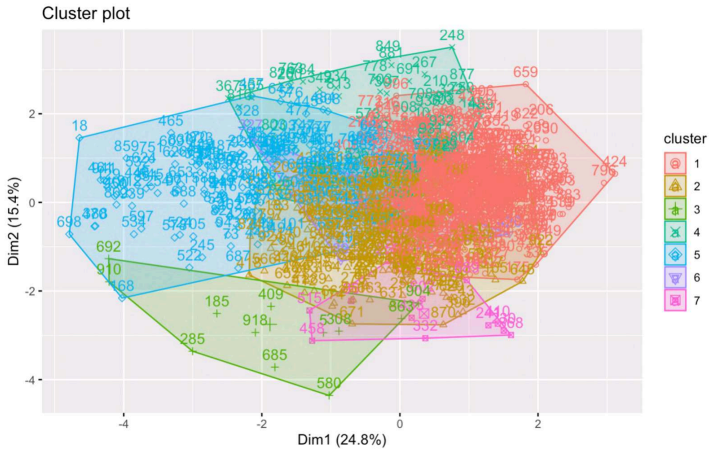
We will use unsupervised learning methods such as k-means to form clusters and observe the optimal number of clusters formed.

3.1 Preparing the data for the model:

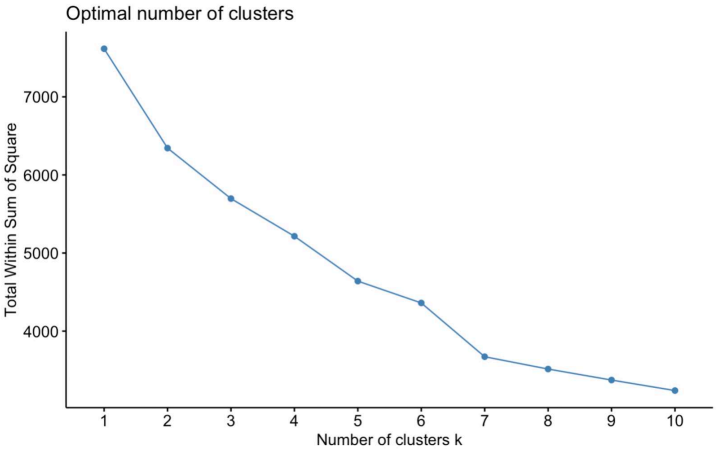
We will be using **Most Streamed Songs of 2023**.

3.2 EDA

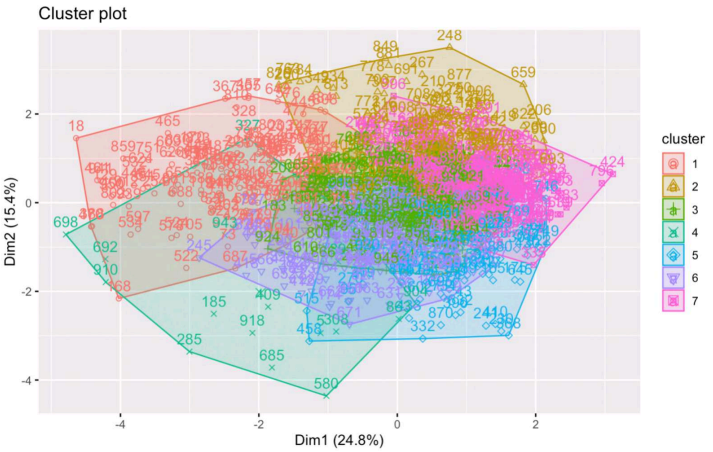
Using Hierarchical clustering with k = 7



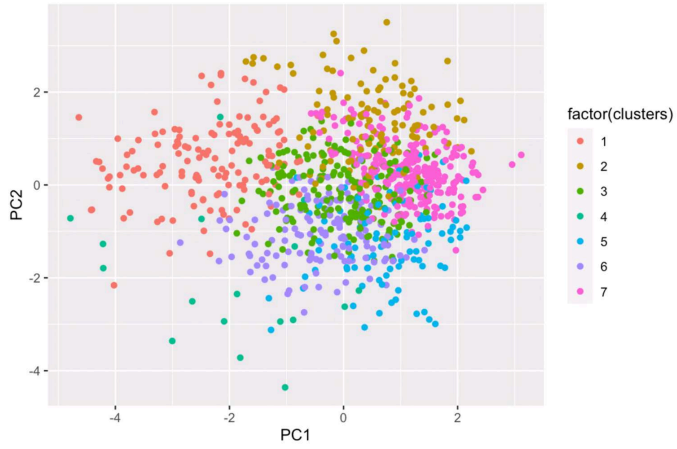
Finding the optimal number of clusters for k-means clustering. We can see that the “elbow” is formed at k = 7. Hence the optimal number of clusters is 7.



Using k-means(k = 7) to form clusters.



We use PCA to reduce the dimensionality of the musical features and take the first two principle components which explain the most variabiltiy. Plot a scatter graph of all the songs with the colours denoting which cluster each song belongs to.



3.3 Models

Our goal was to recommend a song based on the current song. We made two such models using different approaches.

1. We create a new column with the euclidean norm of the musical features for each song. Then calcuate the euclidean distance between the current song and all other songs, and take the top 5 songs closest to it. The final output will re-order the top 5 recommendations to put a song with a common singer at the top, if there exist a common singer between the recommneded song and the current song. The idea behind this is that people like to listen to the same singer.
2. As shown above, we can divide the songs into 7 clusters. The second model identifies which cluster the current song belongs to and then randomly selects a song from the same cluster. The aim behind this method is to promote exploration for the singer. To try out new songs that may be far from the current song but remain fairly similar.

We repeatedly specify the importance of aritists in recommending a song and how an aritist plays an important role in the taste of a person. Since our models use distance and clustering, we naturally ask whether an artist makes similar songs.

3.4 Hypothesis Testing

Hypothesis: Songs with common artists fall in the same cluster.

Null Hypothesis: There is no association between song's with common artist and clusters.

Alternate Hypothesis: There is association between song's with common artist and clusters.

To test this, we create a contingency table between artists and clusters, of some artists selected at random.

	1	2	3	4	5	6	7
070 Shake	0	0	2	0	0	0	0
A Boogie Wit da Hoodie	1	0	0	0	0	0	0
AgroPlay	0	1	0	0	0	0	0
Altamash Faridi	0	0	0	0	0	0	1
Amitabh Bhattacharya	1	0	0	0	0	0	1
Amitabha Bhattacharya	1	0	0	0	0	0	0
Anuel Aa	0	0	1	0	0	1	1
Bad B	0	2	1	0	5	3	4
Beam	0	0	0	1	0	0	0
Bvga Beatz	0	0	0	0	0	0	1
Calvin Harris	0	0	0	0	1	0	1
Chris Brown	0	1	0	0	0	0	0
DJ 900	0	1	0	0	0	0	0
DJ Luian	0	0	1	0	0	1	0
Do��u	0	0	1	0	0	0	0
Ellie Goulding	0	0	0	0	0	1	0
Gabito Ballesteros	0	0	0	0	0	0	1
Jay Rock	0	0	1	0	0	0	0
JID	0	0	1	0	1	0	0
Jimin	1	0	1	0	0	0	1
JVKE	0	0	1	0	0	0	0
Kaliii	0	1	0	0	0	0	0
KayBlack	0	1	0	0	0	0	0
Ke personajes	0	0	0	0	0	1	0
Khalid	1	0	0	0	0	0	1
Kyla	0	0	1	0	0	0	0
La Kelly	0	1	0	0	0	0	0
Lana Del Rey	1	0	1	0	0	0	0
LE SSERAFIM	0	0	1	0	0	0	0
LiL CaKe	0	0	0	0	0	0	1
Lil Uzi Vert	0	0	2	0	0	0	0
Lil Wayne	0	0	0	0	0	1	1
Lucenzo	0	0	0	0	0	0	1
Macklemore	0	0	0	0	0	0	1
Mari Fernandez	0	0	0	0	2	0	0
Milo j	0	0	0	0	0	0	1
Mora	0	1	1	0	0	2	0
Morgan Wallen	0	0	0	0	0	0	1
Muni Long	0	0	1	0	0	0	0
Natanael Cano	0	0	0	0	0	0	1
NAV	1	0	0	0	0	0	0
Nico Valdi	0	0	0	0	0	0	1
NLE Choppa	0	0	1	0	0	0	0
Offset	0	0	0	0	0	1	0
Peso P	0	0	0	0	1	0	12
Prod Malax	0	0	0	0	0	0	1
Riar Saab	0	0	0	0	0	0	1
Roisee	0	0	1	0	0	0	0

	1	2	3	4	5	6	7
21 Savage	0	6	4	0	1	2	0
Anne-Marie	0	0	1	0	0	0	0
Aqua	0	1	0	0	0	0	0
Bad Bunny	0	2	1	0	4	3	4
Bizarrap	0	2	1	0	1	0	1
BTS	0	0	1	0	0	0	3
Central Cee	0	1	0	0	0	0	0
Coi Leray	0	0	2	0	0	0	0
Daft Punk	0	1	0	0	0	0	1
David Guetta	0	0	0	0	1	0	0
DJ Matt D	1	0	0	0	0	0	0
Don Toliver	0	0	1	1	0	0	2
Duki	0	0	1	0	0	0	1
Eladio Carrion	0	0	0	0	0	1	0
Elton John	0	0	0	0	0	0	2
Emilia	0	0	0	0	0	0	1
Feid	0	2	2	0	0	1	1
FMK	0	0	0	0	0	1	1
Grupo Frontera	0	0	1	0	0	5	0
Ice Spice	0	1	0	0	1	0	2
Jay Rock	0	0	1	0	0	0	0
Jung Kook	0	0	0	0	1	0	2
Junior H	0	0	1	0	1	0	2
Kali Uchis	0	0	1	0	0	0	0
Kim Petras	0	0	1	0	0	0	0
Lil Durk	0	2	0	0	0	0	0
Lit Killah	0	0	0	0	0	0	1
Madonna	0	0	0	0	0	0	1
Manuel Turizo	0	0	0	0	0	0	1
Maria Becerra	0	0	0	0	0	1	3
MC Kevin o Chris	0	0	0	0	0	0	1
Metro Boomin	2	3	6	1	0	1	0
Myke Towers	0	0	1	0	0	0	1
Peso Pluma	0	0	0	0	1	0	10
Playboi Carti	0	0	0	0	0	0	1
Pnau	0	0	0	0	0	0	1
Quevedo	0	1	1	0	0	2	2
ROSAL🌹	2	0	0	0	0	0	1
Rusherking	0	0	0	0	0	0	1
Selena G	0	0	0	0	0	0	1
Shakira	0	1	0	0	0	0	2
Sky Rompiendo	0	0	1	0	0	0	0
Stephen Sanchez	1	0	0	0	0	0	0
Swae Lee	0	0	0	0	0	0	1
The Creator	0	1	2	0	3	0	0
The Kid Laroi	0	0	0	0	0	1	0
The Weeknd	0	0	1	0	1	0	1
Tiago pzk	0	1	0	0	0	1	3

The tables shown in the figure have names of artists as rows and the clusters(1,2,3,4,5,6,7) as columns. They shows how many songs of the corresponding artist belong to a particular cluster.

We use Pearson's Chi-square test to test the independence between the artists & clusters.

```
        Pearson's Chi-squared test

data:  new_table
X-squared = 179.81, df = 144, p-value = 0.023
```

```
        Pearson's Chi-squared test

data:  new_table
X-squared = 197.61, df = 144, p-value = 0.002031
```

As we can see, the p-value is very low (<0.05). Hence, we can reject the null-hypothesis and conclude that song's of an artist tend to fall in the same cluster.

4. Regression Models for Stream Prediction

We will use linear regression , ridge regression to model the target variable (Number of Streams)

4.1 Data Description

We will be using **Most Streamed Songs of 2023**.

4.2 Data Cleaning and Preparation

- We can see ***Key*** column has null values and on careful observation we see that out of the 12 musical Keys only one ***"C"* key** was mssing everywhere hence we filled it with that.
- We found out that ***streams*** column was of object type hence we converted it to integer and removed the row which was coerced to become NA since it contained string in place of number.
- columns ***in_deezer_playlist*** and ***in_shazam_charts*** were object type because of comma separated representation of numbers so we removed the commas and then converted both of them into numbers . Moreover ***in_shazam_charts*** there were missing entries hence we removed those rows from the dataframe.
- Lastly we converted the categorical data of ***keys*** to ordinal data from 0 to 11 for keys in an increasing orderr of their frequency and for ***modes*** column we converted Major and Minor to 1 and 0 respectively.

After Doing all this we get the cleaned and prepared data as in ***Fig -2*** .

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 902 entries, 0 to 952
Data columns (total 24 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   track_name            902 non-null    object
 1   artist(s)_name        902 non-null    object
 2   artist_count          902 non-null    int64 
 3   released_year         902 non-null    int64 
 4   released_month        902 non-null    int64 
 5   released_day          902 non-null    int64 
 6   in_spotify_playlists  902 non-null    int64 
 7   in_spotify_charts     902 non-null    int64 
 8   streams               902 non-null    int64 
 9   in_apple_playlists    902 non-null    int64 
 10  in_apple_charts       902 non-null    int64 
 11  in_deezer_playlists   902 non-null    int64 
 12  in_deezer_charts      902 non-null    int64 
 13  in_shazam_charts      902 non-null    int64 
 14  bpm                   902 non-null    int64 
 15  key                   902 non-null    int64 
 16  mode                  902 non-null    int64 
 17  danceability_%        902 non-null    int64 
 18  valence_%             902 non-null    int64 
 19  energy_%              902 non-null    int64 
 20  acousticness_%        902 non-null    int64 
```

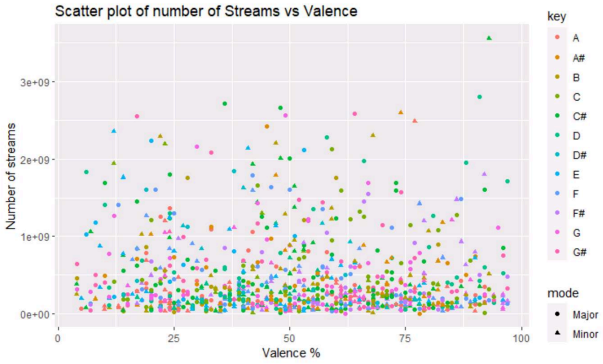
```
 21  instrumentalness_%    902 non-null    int64
 22  liveness_%            902 non-null    int64
 23  speechiness_%         902 non-null    int64
dtypes: int64(22), object(2)
memory usage: 176.2+ KB
```

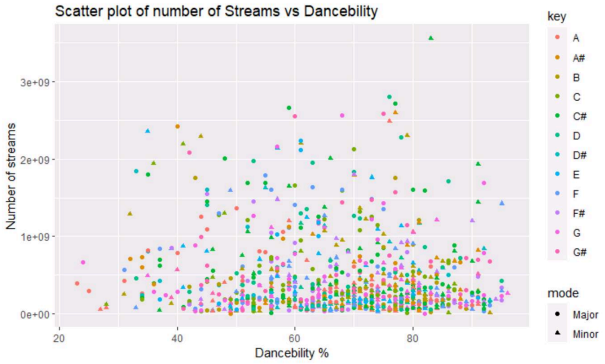
4.3 EDA

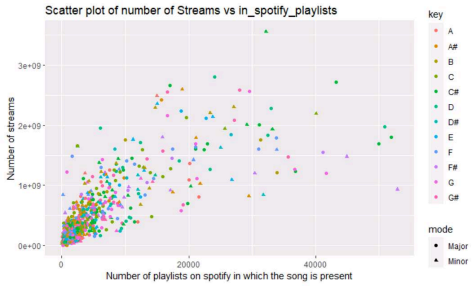
Here we start by analysing by observing if the values are correlated with the musical features or not

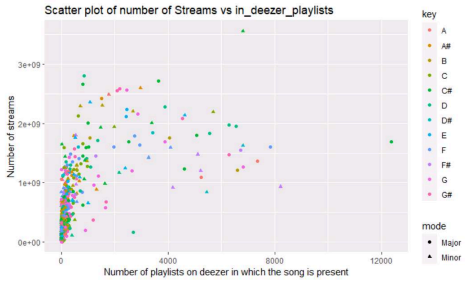
We start by making some graphs to observe this. Even before making graphs

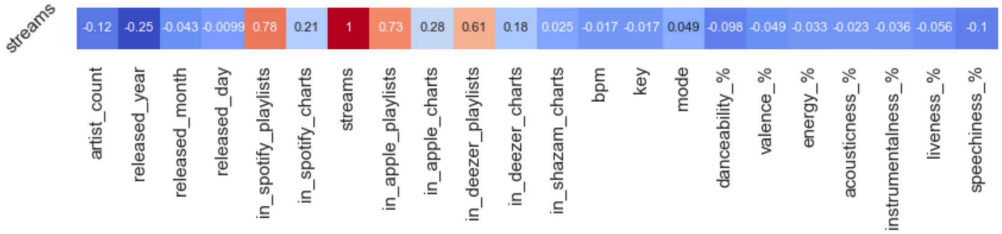
- We may think of songs having high danceability and Energy to be having more number of streams because they are played numerous times in parties ,concerts and in clubs.
- There could be a major factor among the songs characteristics features like valence, bpm that could play a significant role in increasing the number of streams.





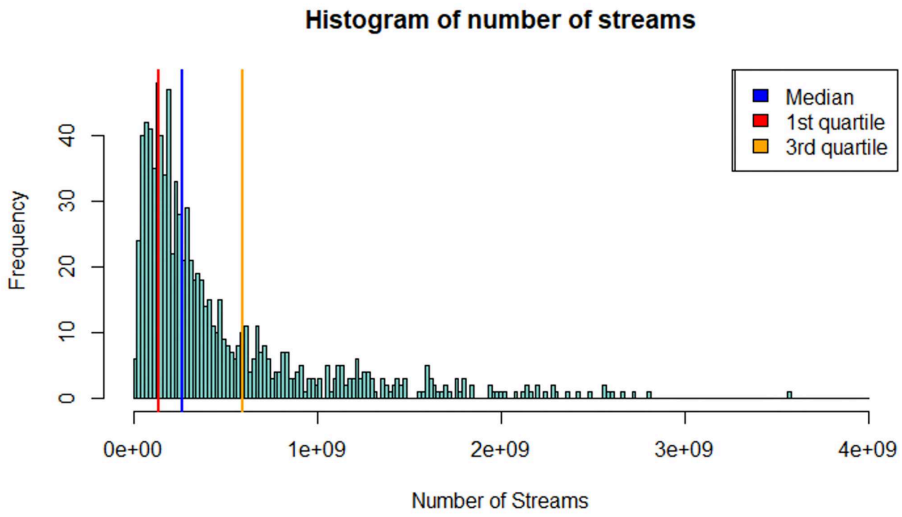






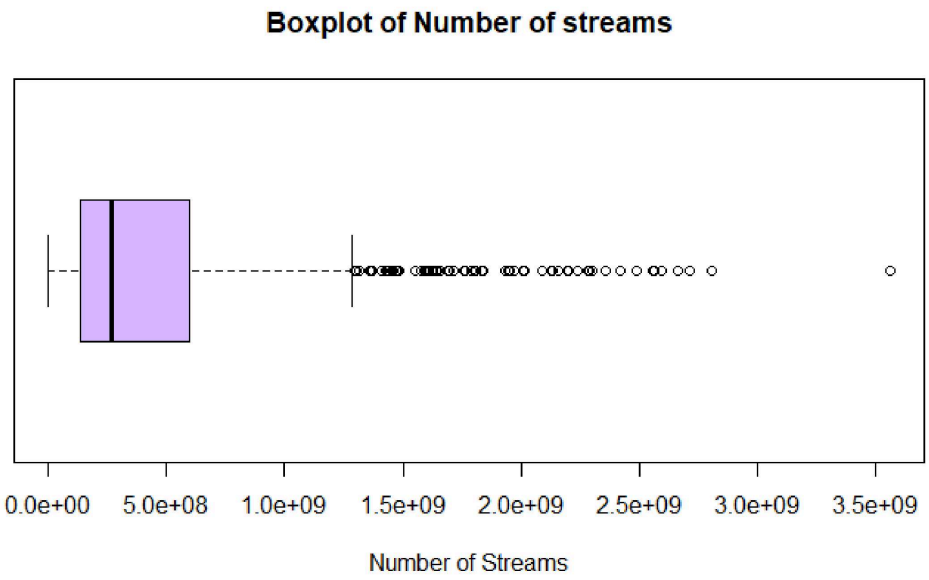
After observing the plots, it seems like the muscial features aren't linearly related to number of streams. There is almost no correlation between them.

Before going further, we take a look at histograms of number of streams as well as boxplot for getting insight into how the number of songs are distributed. We clearly observe that it is positively skewed. Moreover, most of the songs don't even cross ***1 Billion streams*** .




```
> summary(songs_data$streams)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
2.762e+03 1.368e+08 2.697e+08 4.692e+08 5.989e+08 3.563e+09
```

$$\text{outlier lower bound} = 3^{rd}\text{ quartile} + 1.5 * \text{IQR} \approx 1.29 * 10^9$$



From the plots above we can see that there are many outliers in the data. Hence they may affect our model and cause errors.

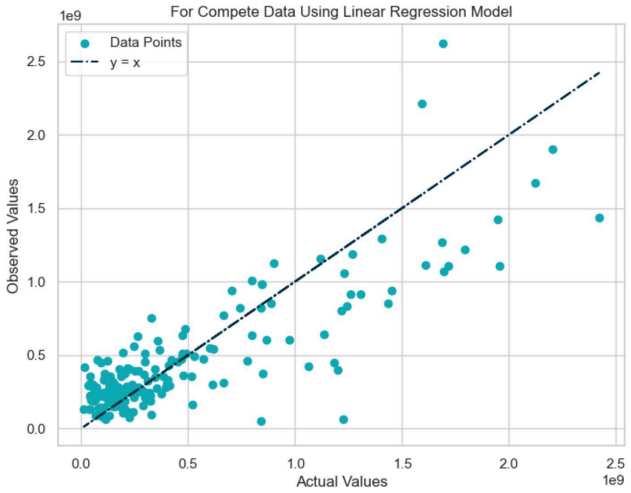
4.4 Trial and Error with Models

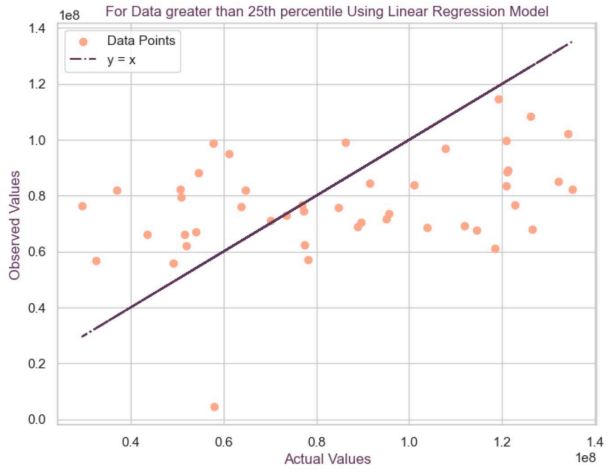
We Are going to train the Dataset for predicting the Number of streams for different songs using their various features by Linear Regression and Ridge regression. Hereunder is given a summary of those results.

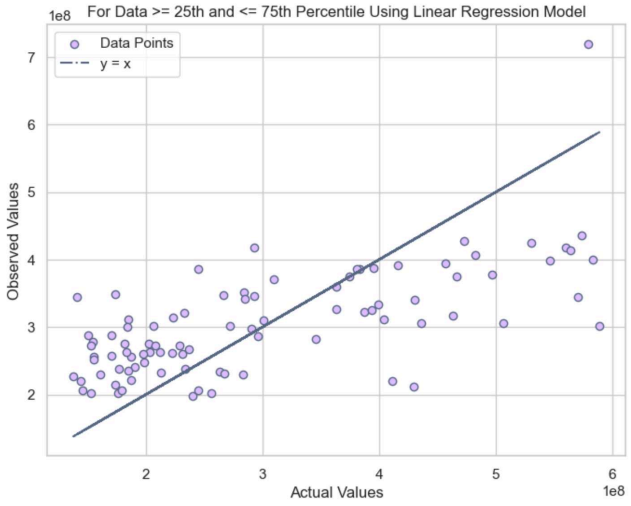
	Linear Regression		Ridge Regression	
	MSE (Mean Squared Error)	MAPE (in %) (Mean Absolute Percentage Error)	MSE (Mean Squared Error)	MAPE(in %)(Mean Absolute Percentage Error)
Complete Data	$7.9268 * 10^{16}$	107.866	$7.9265 * 10^{16}$	107.863

	Linear Regression		Ridge Regression	
Data Less than 25 percentile	$9.232 * 10^{15}$	34.827	$9.226 * 10^{15}$	34.831
Between 25 and 75 percentile	$9.811 * 10^{15}$	30.867	$9.810 * 10^{15}$	30.869
Data Above 75 percentile	$2.244 * 10^{17}$	26.280	$2.243 * 10^{17}$	26.288

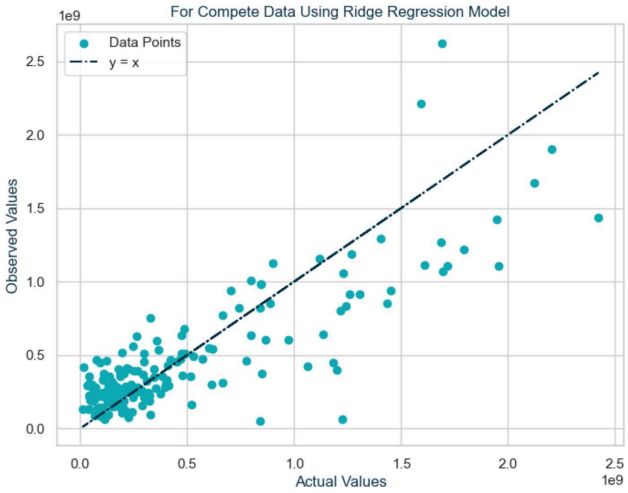
4.4.1 Linear Regression Plots

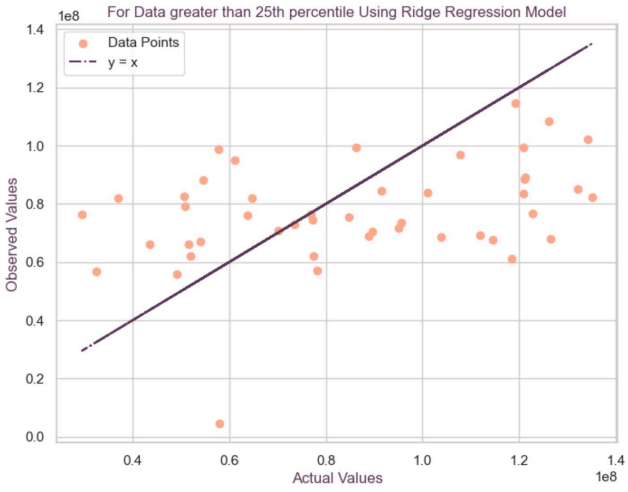


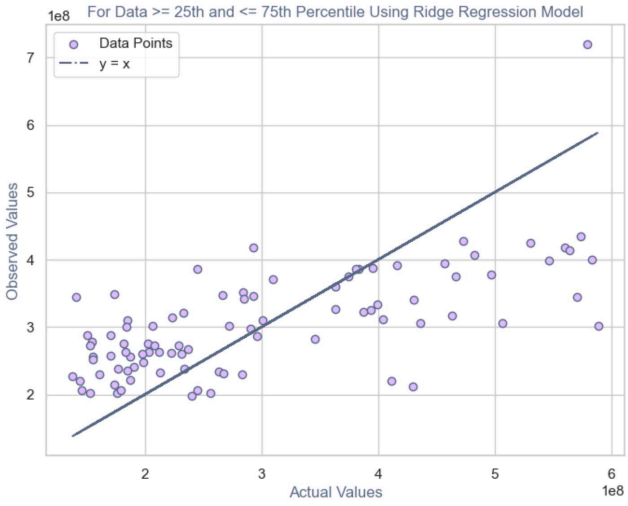


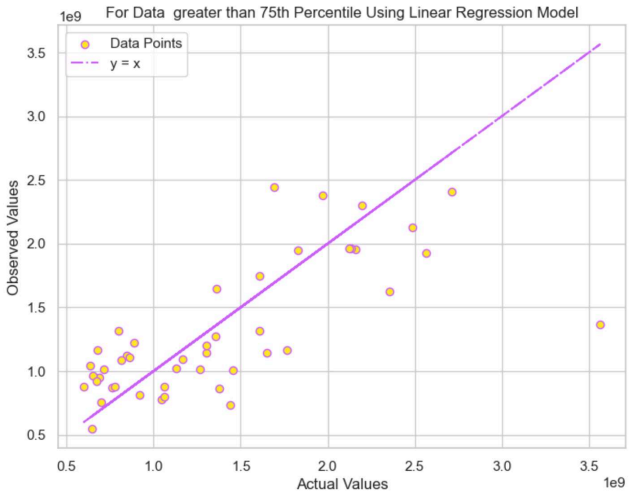


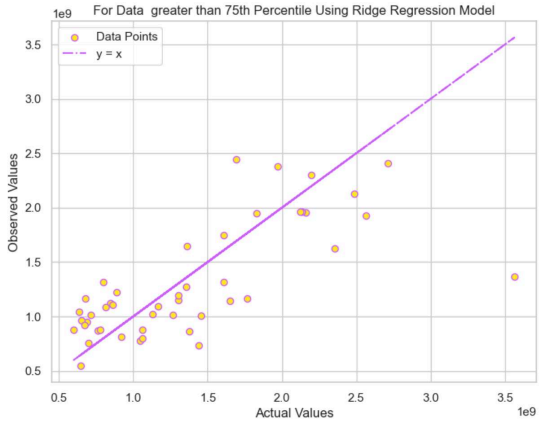
4.4.2 Ridge Regression Plots











4.5 Conclusion

From the metrics table, the linear model performs poorly on the entire dataset. This is due to the non-linearity between the number of streams and the features.

We then tried fitting the model on data lying in the inter-quartile range and above the 3rd quartile. The data above the 3rd quartile is quite important because it represents the songs which are very popular. Since data below the 1st quartile represents songs that are unpopular they are of not much importance to us, we can safely disregard it. From the metric tables we see that the linear model fits relatively well on data above the 3rd quartile.

Also, there isn't any significant difference between a linear regression model and ridge regression model.

---

5. Predicting Billboard Year-End Hot 100 Inclusion

5.1 Motivation

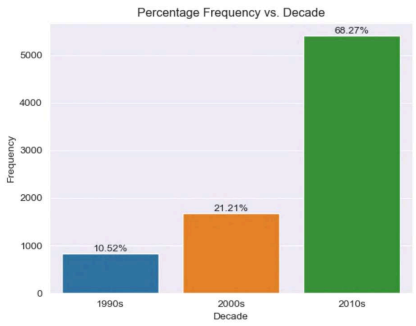
The aim of this project was to determine if a machine learning classifier could forecast whether a song would achieve hit status, known as Hit Song Science, by analyzing its inherent audio characteristics.

As a result, music streaming apps can potentially save money by reducing the costs associated with licensing or acquiring unpopular music while increasing profitability through higher user retention and more effective targeted advertising.

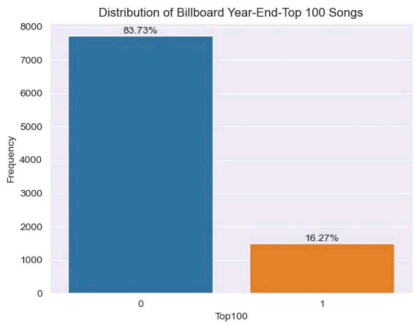
5.2 EDA

After Cleaning the Top-100-Predictor Dataset, around 9300 songs remained.

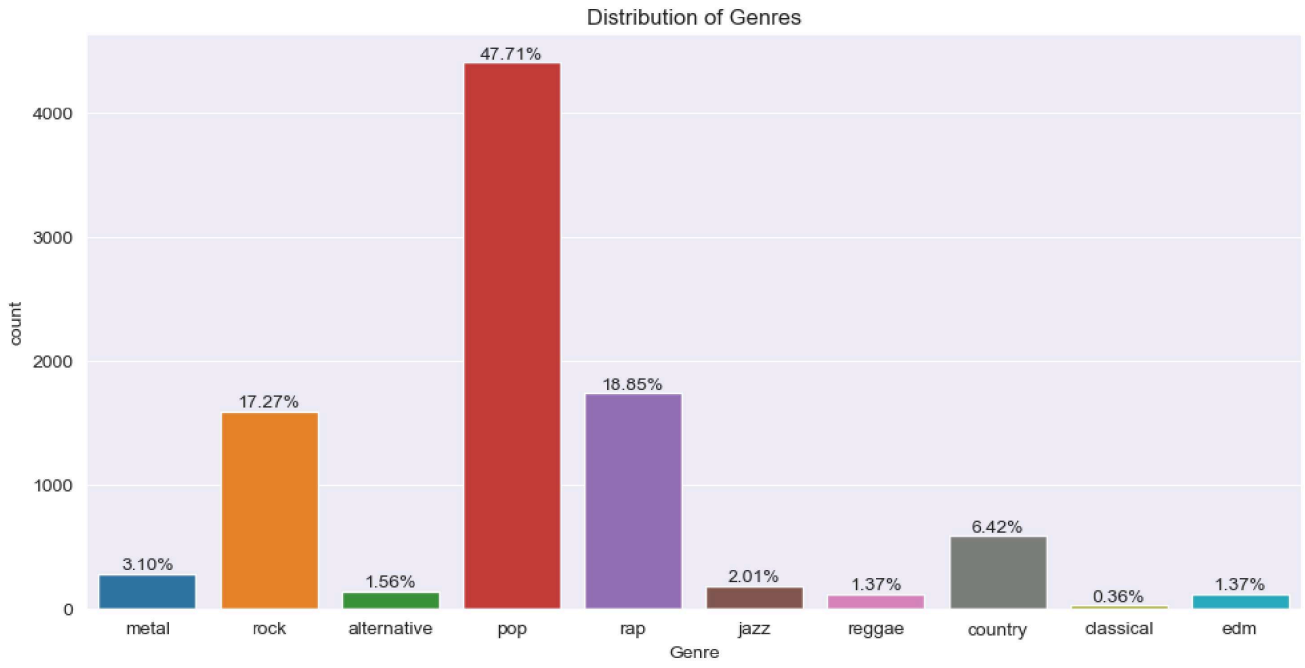
5.2.1 Data Distribution



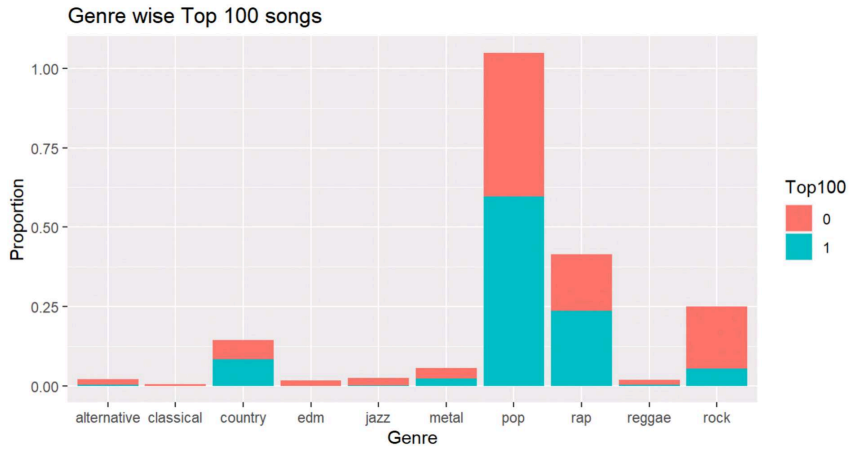
Decade-wise Distribution of Songs



frequency distribution of our sample

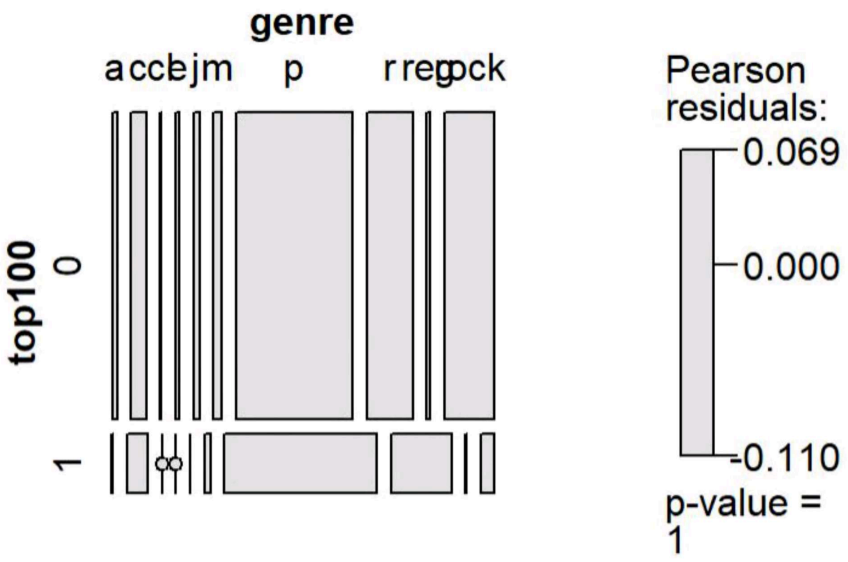


Distribution of Genre



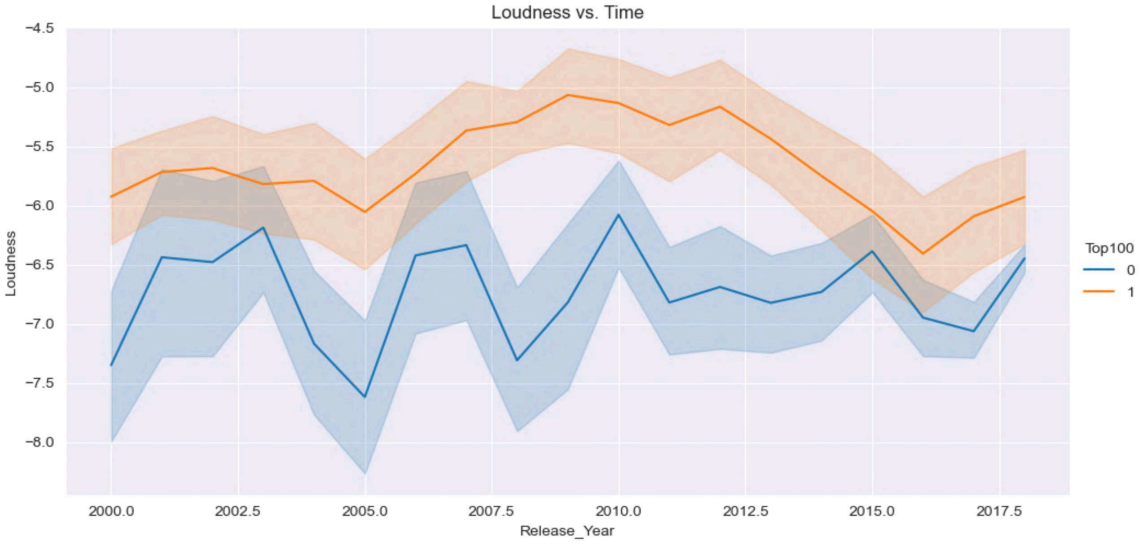
Genre Distribution of Top-100 and Non-Top-100 songs

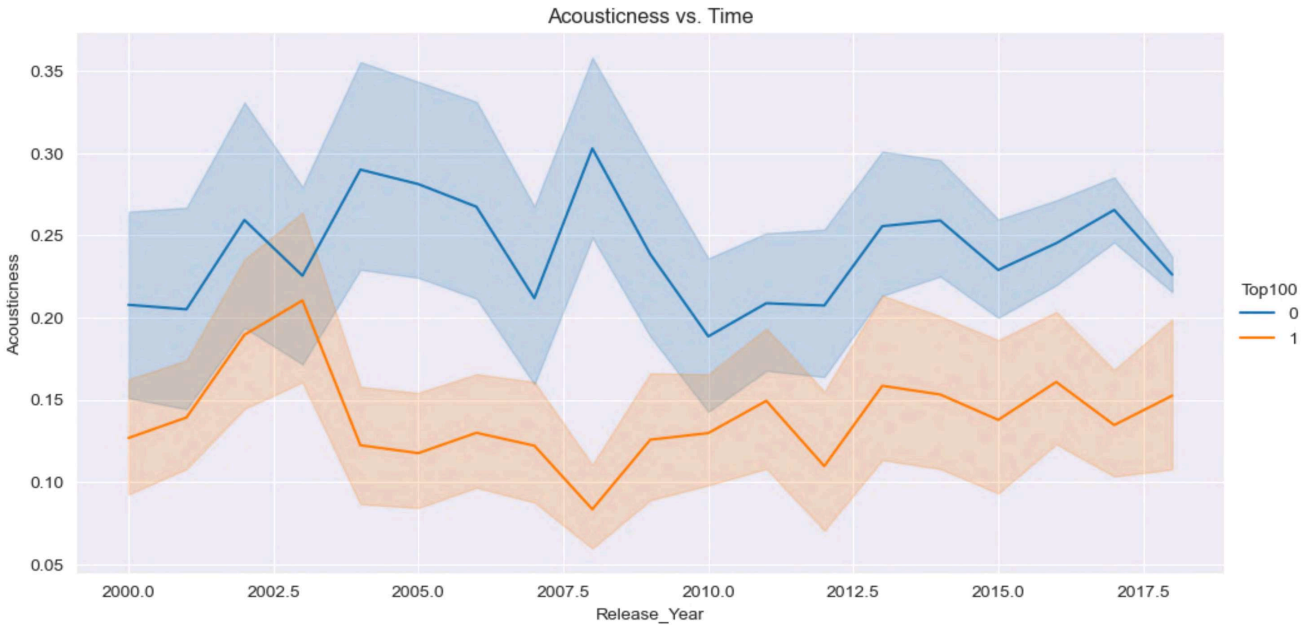
Here we can see that, most of the top 100 songs are from “pop” genre followed by “rap”, which can also be inferred from the following mosaic plot.



Here, the genre's are coded by their initials.

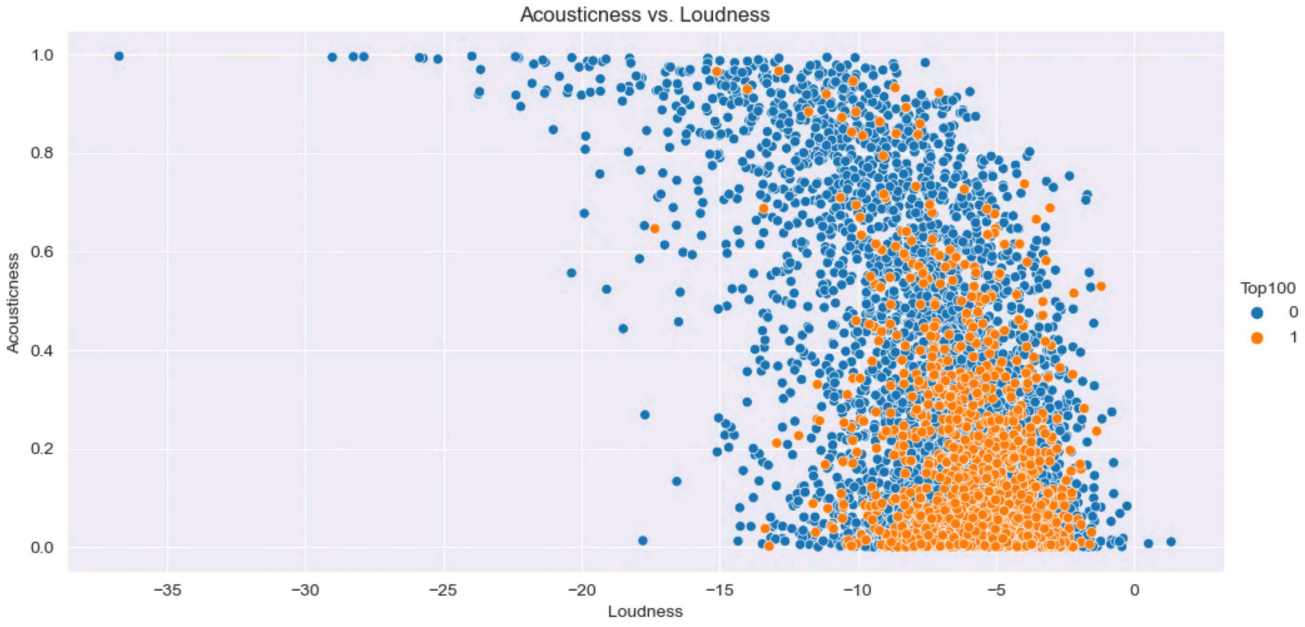
5.2.1 Song Features over time

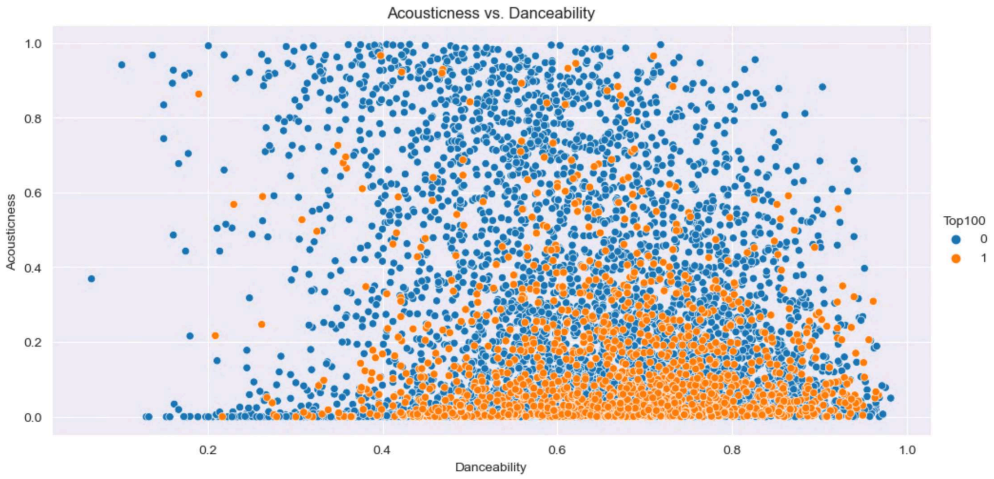




The separability of data in certain features such as Acousticness and Loudness, indicates potentialy significant features that can help distinguish between the two classes.

5.2.3 Feature Comparisons





The above plots shows the separability of data accross two song-features, this suggests that data may separate accross an *n-dimensional* feature space. Thus, an unsupervised clustering model may help us classify the data.

5.3 Model and Results

Given the unbalanced nature of the dataset, any model chosen would automatically yield high accuracy. Apart from trying to be very accurate, we also want our model to have a high AUC (so that TPR is maximized and FPR is minimized). This helps us see how well the model can tell the difference between the two types of data. A higher AUC means the model is better at distinguishing between them.

After looking at the data, we decided to only utilise the songs released between 2000 and 2018. This is because music changes over time, and the characteristics of songs from the '90s might not be the same as those from the 2000s and 2010s. Spotify songs data after 2018 is not available on any other trusted source.

Below is the list of models we tested :

5.3.1 Logistic Regression

Logistic Regression (LR) is a commonly used classification algorithm when the target variable is categorical. This method aims to create a correlation between features and the desirable output. Generally, exist two types of logistic regression problems, the binary and the multi-class ones. Logistic regression model uses a logistic function to determine a binary dependent variable in its main form. This function re-frames the log-odds to meaningful probabilities and finally the labeled values are ”0” and ”1”. The logistic function is described by the equation

$$p(X\,;b\,,w) = \frac{e^{w.X+b}}{1+e^{w.X+b}} = \frac{1}{1+e^{-w.X+b}}$$

5.3.2 Improved Logistic Regression

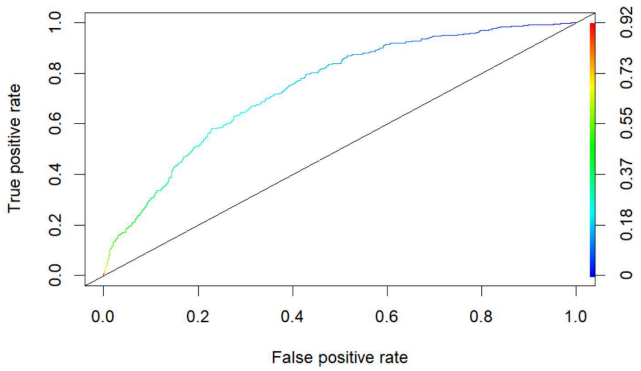
This model is similar to the previous logistic regression model where the less significant features are excluded like Danceability , key and some less popular genre.

5.3.3 Penalised Regression:

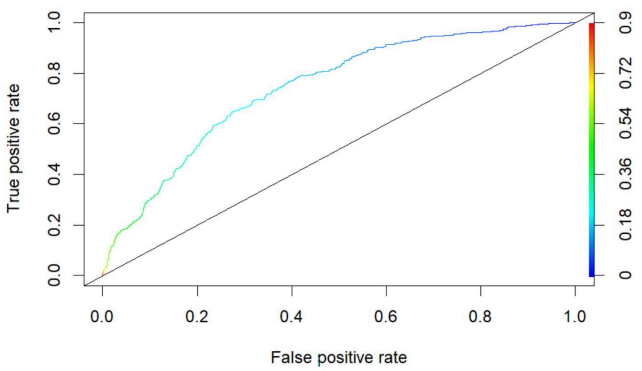
Due to large number of spotify features, we chose to use a penalized logistic regression model. This kind of model penalizes itself if it has too many features to work with. It does this by shrinking the importance of less relevant features, which helps simplify the model and makes it more manageable. Specifically, we used the following techniques that apply this penalty to the regression process.

1. **Ridge Regression**
    - All the features are included in the model, but variables with minor contribution have their coefficients close to zero.
    - Performs L2 regularization, i.e., adds penalty equivalent to the **square of the magnitude** of coefficients
2. **Lasso Regression**
    - The coefficients of less contributive features are forced to zero and only the most significant features are kept

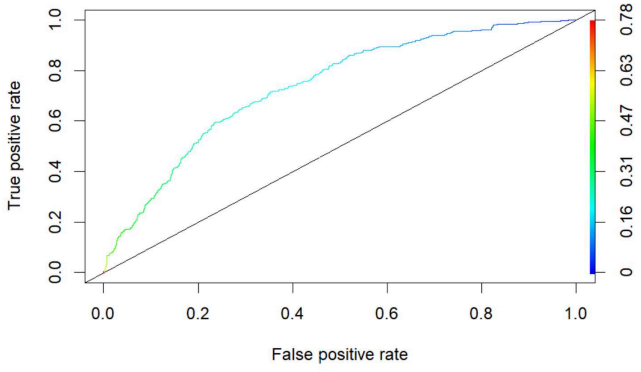
- Performs L1 regularization, i.e., adds penalty equivalent to the **absolute value of the magnitude** of coefficients



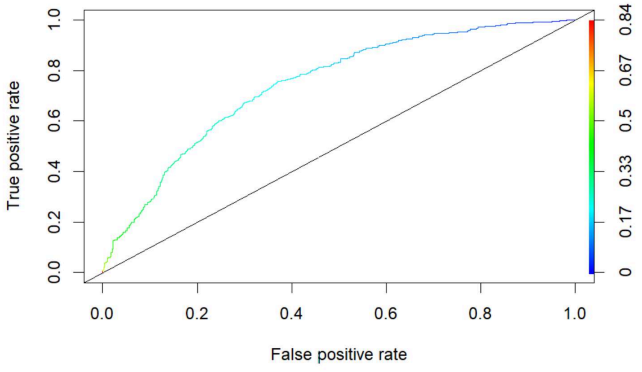
Logistic Regression



Improved Logistic Regression



Ridge Regression



Lasso Regression

5.4 ROC Curve for Test Dataset

Model Summary

Model	Accuracy	TPR	AUC
Baseline	0.7984	na	na
Logistic Regression	0.8057	0.1372	0.7422
Improved Logistic Regression	**0.8068**	**0.1400**	**0.7425**
Ridge	0.8023	0.0476	0.7357
Lasso	0.8018	0.0560	0.7429

6. Conclusion

1. **Association between Songs and Artists:** Songs by the same artist often share similarities and tend to be grouped together.
2. **Recommendation Systems:** Two recommendation systems were developed to suggest the next song for listeners. One system utilizes a method called k-means clustering, while the other employs hierarchical clustering.
3. **Regression Model for Song Streams Prediction:** A regression model was created to estimate the number of streams a song might receive based on various features such as tempo, lyrics, and popularity. The model showed a Mean

Absolute Percentage Error (MAPE) of approximately 30.867% for songs within the 25th to 75th percentile of popularity, and about 26.268% for songs above the 75th percentile.

4. **Top 100 Classification Models:** Machine learning models were constructed to predict whether a song would reach the top 100 charts. Both a logistic regression model and an enhanced logistic model achieved an accuracy of around 80% and an Area Under the Curve (AUC) of about 74%, indicating their effectiveness in predicting song success.