Indira Gandhi Delhi Technical University for Women

(Established by Govt. of Delhi vide Act 09 of 2012)

Kashmere Gate, Delhi - 110006

REPORT FILE

For

IT WORKSHOP (BAI-108)

Department of Information Technology

MUSIC RECOMMENDATION SYSTEM IN R PROJECT

SUBMITTED BY:

1) PALAK MANGLA (043)

2) PARUL DHARIWAL (044)

3) PALAK JETHWANI (042)

B.Tech / AI & ML

Sem: 2

SUBMITTED TO:

Mr. SANTANOO PATTNAIK

IT WORKSHOP LAB

INDEX

ACKNOWLEDGEMENT

We want to convey our appreciation to Mr. Santanoo sir for his guidance and support in assisting us throughout the project titled "*MUSIC RECOMMENDATION SYSTEM IN R PROJECT*" enabling us to successfully complete it. Additionally, we would like to express our gratitude to our fellow batchmates for their assistance and continuous support during the project's duration. Lastly, we are thankful to Indira Gandhi Delhi Technical University for providing us with the opportunity to work on this project.

MUSIC RECOMMENDATION SYSTEM IN R PROJECT

OBJECTIVE

Many new music platforms are emerging as a result of the rapid growth of online and mobile platforms. These platforms provide songs lists from all across the world. Every person has a unique preference for music. The majority of individuals use online music streaming services such as Spotify, Apple Music, Google Play, or Pandora.

However, it can be challenging to choose between millions of tracks. Therefore, music service providers require an efficient approach to organize songs and assist their customers in choosing music by providing quality recommendations. As a result, an effective recommendation system is essential.

Many music streaming services, such as Gaana and Spotify, are currently focused on developing high-precision commercial music recommendation systems. These businesses make money by assisting their consumers in choosing suitable music and charging them for the quality of their suggestion service. As a result, there is a strong market for good music recommendation system.

The purpose of this data analysis project is to explore the music dataset and perform various analyses to gain insights into the data. The dataset contains information about different songs, including their titles, artists, genres, and various attributes such as BPM, danceability, duration, and popularity.

The objective of this project is to develop a music recommendation system that provides personalized song recommendations to users based on their preferences and song features. The system aims to enhance the user experience by suggesting songs that align with their musical tastes.

METHODOLOGY

DATA ACQUISITION

Dataset taken from the site:

https://www.kaggle.com/datasets/iamsumat/spotify-top-2000s-mega-dataset

Content

A LITTLE ABOUT THE DATASET:

- **Index**: ID
- **Title**: Name of the Track
- **Artist**: Name of the Artist
- **Top Genre**: Genre of the track
- **Year**: Release Year of the track
- **Beats per Minute (BPM)**: The tempo of the song
- **Energy**: The energy of a song **-** the higher the value, the more energetic. song
- **Danceability**: The higher the value, the easier it is to dance to this song.
- **Loudness**: The higher the value, the louder the song.
- **Length**: The duration of the song.
- **Liveness**: The higher the value the more spoken words the song contains
- **Popularity**: The higher the value the more popular the song is.

The dataset was loaded into R using the **read.csv** function, to ensure data quality missing values in the dataset were examined using the sum(is.na(music_data)) command. The preprocessing phase involved several steps, including data cleaning and feature engineering, to prepare the dataset for the recommendation system.

```
File_path<-file.path(getwd(),"musicdataset.csv")
File_path
music_data<-read.csv("musicdataset.csv")
music_data
sum(is.na(music_data))
```

DATA EXPLORATION

- The **head** function was used to display the first few rows of the dataset, giving an overview of the data.

```
> print(head(music_data))
  Index                           Title            Artist            Top.Genre Year Beats.Per.Minute..BPM. Energy Danceability Loudness..dB. Liveness Length..Duration. Popularity
1     1                         Sunrise       Norah Jones      adult standards 2004                    157     30           53           -14       11               201         71
2     2                     Black Night       Deep Purple           album rock 2000                    135     79           50           -11       17               207         39
3     3                  Clint Eastwood          Gorillaz  alternative hip hop 2001                    168     69           66            -9        7               341         69
4     4                   The Pretender      Foo Fighters    alternative metal 2007                    173     96           43            -4        3               269         76
5     5          Waitin' On A Sunny Day Bruce Springsteen          classic rock 2002                    106     82           58            -5       10               256         59
6     6 The Road Ahead (Miles Of The Unknown)      City To City alternative pop rock 2004                     99     46           54            -9       14               247         45
> 
```

- The **tail** function was used to display the last few rows of the dataset.

```
> print(tail(music_data))
    Index                                          Title         Artist       Top.Genre Year Beats.Per.Minute..BPM. Energy Danceability Loudness..dB. Liveness Length..Duration. Popularity
794   794                                    Dance Monkey    Tones and I australian pop 2019                     98     59           82            -6       15               209        100
795   795                                      Blauwe Dag Suzan & Freek       dutch pop 2019                     98     56           70            -7       28               183         68
796   796 Homburg - Single Version - 2009 Remaster - Mono   Procol Harum      album rock 2019                    142     66           36            -8        6               237         32
797   797                                       Uncharted     Kensington       dutch pop 2019                    139     53           59            -7       29               241         65
798   798                                       Despacito     Luis Fonsi           latin 2019                    178     80           66            -5        7               229         80
799   799                                 Dancing On My Own    Calum Scott australian pop 2019                    113     17           68            -9       10               260         28
> 
```

- The **summary** function provided summary statistics such as minimum, maximum, mean, and quartiles for each column.

```
Max.   :100.00
> print(summary(music_data))
     Index          Title              Artist            Top.Genre              Year      Beats.Per.Minute..BPM.     Energy        Danceability   Loudness..dB.        Liveness     Length..Duration.   Popularity    
 Min.   :  1.0   Length:799         Length:799         Length:799         Min.   :2000   Min.   : 49.0          Min.   : 5.00   Min.   :12.00   Min.   :-21.000   Min.   : 2.00   Min.   :122.0     Min.   : 12.00  
 1st Qu.:200.5   Class :character   Class :character   Class :character   1st Qu.:2005   1st Qu.:100.0          1st Qu.:47.00   1st Qu.:44.00   1st Qu.: -9.000   1st Qu.:10.00   1st Qu.:213.0     1st Qu.: 45.00  
 Median :400.0   Mode  :character   Mode  :character   Mode  :character   Median :2009   Median :120.0          Median :66.00   Median :54.00   Median : -7.000   Median :12.00   Median :239.0     Median : 59.00  
 Mean   :400.0                                                            Mean   :2010   Mean   :121.2          Mean   :62.74   Mean   :54.09   Mean   : -7.516   Mean   :19.46   Mean   :250.5     Mean   : 57.44  
 3rd Qu.:599.5                                                            3rd Qu.:2014   3rd Qu.:138.0          3rd Qu.:79.00   3rd Qu.:64.50   3rd Qu.: -5.000   3rd Qu.:22.00   3rd Qu.:272.0     3rd Qu.: 71.00  
 Max.   :799.0                                                            Max.   :2019   Max.   :205.0          Max.   :99.00   Max.   :95.00   Max.   : -2.000   Max.   :99.00   Max.   :809.0     Max.   :100.00  
> 
```

- The **str** function provided the structure of the dataset, including the data types of each column.

```
> print(str(music_data))
'data.frame':	799 obs. of  12 variables:
 $ Index                 : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Title                 : chr  "Sunrise" "Black Night" "Clint Eastwood" "The Pretender" ...
 $ Artist                : chr  "Norah Jones" "Deep Purple" "Gorillaz" "Foo Fighters" ...
 $ Top.Genre             : chr  "adult standards" "album rock" "alternative hip hop" "alternative metal" ...
 $ Year                  : int  2004 2000 2001 2007 2002 2004 2002 2006 2004 2002 ...
 $ Beats.Per.Minute..BPM.: int  157 135 168 173 106 99 102 137 148 112 ...
 $ Energy                : int  30 79 69 96 82 46 71 96 92 67 ...
 $ Danceability          : int  53 50 66 43 58 54 71 37 36 91 ...
 $ Loudness..dB.         : int  -14 -11 -9 -4 -5 -9 -6 -5 -4 -3 ...
 $ Liveness              : int  11 17 7 3 10 14 13 12 10 24 ...
 $ Length..Duration.     : int  201 207 341 269 256 247 257 366 223 290 ...
 $ Popularity            : int  71 39 69 76 59 45 74 69 77 82 ...
NULL
> 
```

Looking for unique genres present in the dataset:

```
> music_genre<-unique(music_data$Top.Genre)
> print(music_genre)
  [1] "adult standards"            "album rock"                 "alternative hip hop"        "alternative metal"
  [5] "classic rock"               "alternative pop rock"       "pop"                        "modern rock"
  [9] "detroit hip hop"            "alternative rock"           "dutch indie"                "garage rock"
 [13] "dutch cabaret"              "permanent wave"             "classic uk pop"             "dance pop"
 [17] "modern folk rock"           "dutch pop"                  "dutch americana"            "alternative dance"
 [21] "german pop"                 "afropop"                    "british soul"               "irish rock"
 [25] "disco"                      "big room"                   "art rock"                   "danish pop rock"
 [29] "neo mellow"                 "britpop"                    "boy band"                   "carnaval limburg"
 [33] "arkansas country"           "latin alternative"          "british folk"               "celtic"
 [37] "chanson"                    "celtic rock"                "hip pop"                    "east coast hip hop"
 [41] "dutch rock"                 "blues rock"                 "electro"                    "australian pop"
 [45] "belgian rock"               "downtempo"                  "reggae fusion"              "british invasion"
 [49] "finnish metal"              "canadian pop"               "bow pop"                    "dutch hip hop"
 [53] "dutch metal"                "soft rock"                  "acoustic pop"               "acid jazz"
 [57] "dutch prog"                 "candy pop"                  "operatic pop"               "trance"
 [61] "scottish singer-songwriter" "mellow gold"                "alternative pop"            "dance rock"
 [65] "atl hip hop"                "eurodance"                  "blues"                      "canadian folk"
 [69] "big beat"                   "art pop"                    "uk pop"                     "glam metal"
 [73] "brill building pop"         "g funk"                     "happy hardcore"             "belgian pop"
 [77] "classic schlager"           "contemporary country"       "barbadian pop"              "gabba"
 [81] "chamber pop"                "british singer-songwriter"  "indie pop"                  "australian rock"
 [85] "nederpop"                   "australian indie folk"      "folk-pop"                   "electropop"
 [89] "edm"                        "metropopolis"               "irish pop"                  "electronica"
 [93] "alaska indie"               "irish singer-songwriter"    "stomp and holler"           "australian dance"
 [97] "australian psych"           "laboratorio"                "contemporary vocal jazz"    "rock-and-roll"
[101] "glam rock"                  "classic soundtrack"         "icelandic indie"            "danish pop"
[105] "compositional ambient"      "neo soul"                   "streektaal"                 "italian pop"
[109] "indie anthem-folk"          "la pop"                     "baroque pop"                "ccm"
[113] "electro house"              "austropop"                  "australian americana"       "latin"
>
```

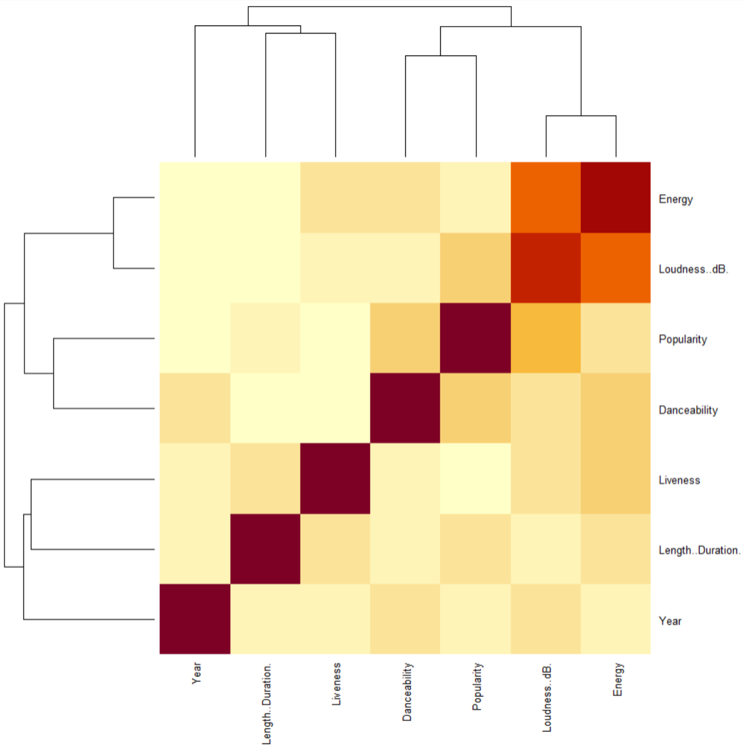
```
> print(genre_count)
                  Top.Genre  n
1                 dutch pop 70
2                 dance pop 61
3               dutch indie 52
4               modern rock 45
5                album rock 43
6         alternative metal 35
7            permanent wave 29
8             dutch cabaret 27
9          alternative rock 25
10             british soul 25
11                      pop 25
12          adult standards 21
13               neo mellow 17
14        alternative dance 14
15               irish rock 14
16            dutch hip hop 13
17          dutch americana 10
18         carnaval limburg  9
19               dutch rock  9
20             classic rock  8
21                 art rock  7
22                 big room  7
23              celtic rock  7
24              chamber pop  7
25         modern folk rock  7
26             belgian rock  6
27               blues rock  6
28                  britpop  6
29           classic uk pop  6
30               dance rock  6
31          detroit hip hop  6
32         arkansas country  5
33                  art pop  5
34                 boy band  5
35         british invasion  5
36                    disco  5
37                  electro  5
38               german pop  5
39             acoustic pop  4
40     alternative pop rock  4
```

VISUALIZATION OF THE DATA

1) Correlation Matrix: The correlation matrix quantifies the relationships between different attributes of songs. The matrix provides correlation coefficients between variables such as year, energy, danceability, loudness, liveness, length/duration, and popularity. It helps identify the strength and direction of these relationships.

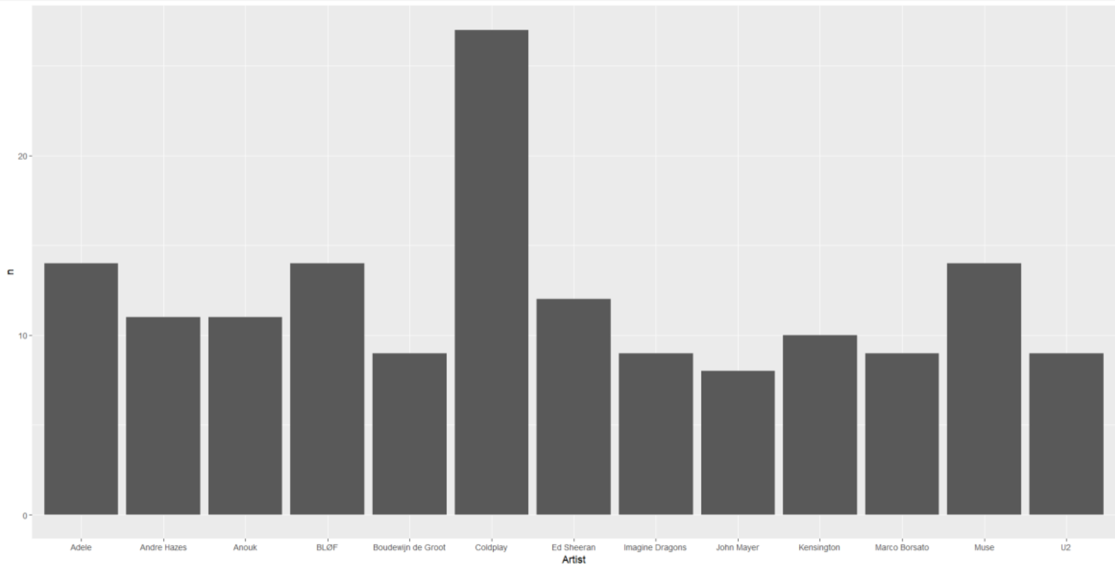
Heatmap of Correlation Matrix: The heatmap visually represents the correlation matrix using colors. It allows for a quick assessment of the strength and patterns of correlations between variables.

```
correlation_matrix<-cor(music_data[,c("Year","Energy","Danceability","Loudness..dB.","Liveness","Length..Duration.","Popularity")])
print(correlation_matrix)
heatmap(correlation_matrix,
        cmap = ggplot2::scale_fill_viridis_c(),
        cexRow = 0.8,
        cexCol = 0.8,
        margins = c(10, 10))
```



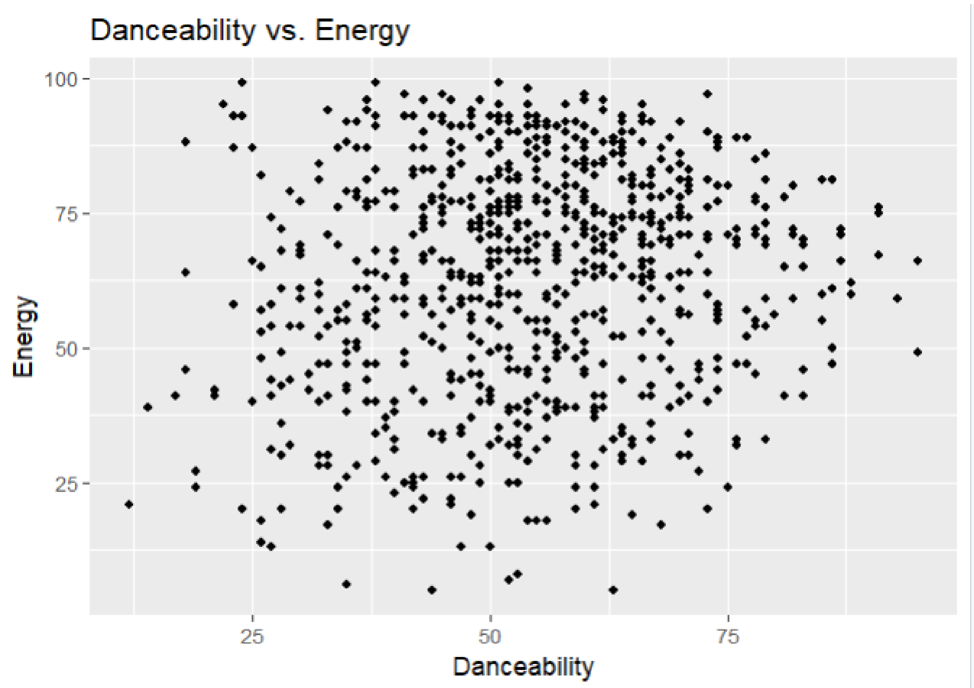
2) Artist Count Plot: The plot represents the count of songs for each artist. Artists with more than 6 songs are included. It gives an insight into the number of songs contributed by different artists.

```
artist_count_plot <- music_data %>%
  group_by(Artist) %>%
  summarize(n = n()) %>%
  filter(n > 7) %>%
  ggplot(aes(x = Artist, y = n)) +
  geom_col()
print(artist_count_plot)
```



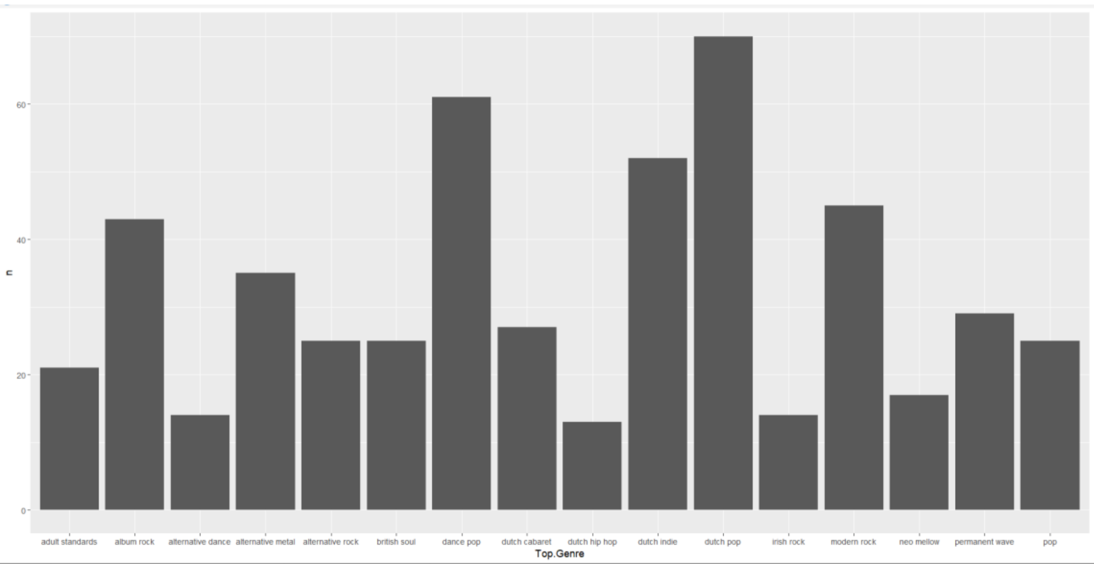
3) Danceability vs. Energy: This scatter plot displays the relationship between danceability and energy. It helps visualize if there is any correlation between these attributes and whether songs with higher danceability tend to have higher energy levels.

```
ggplot(data = music_data, aes(x = Danceability, y = Energy)) +
  geom_point() +
  xlab("Danceability") +
  ylab("Energy") +
  ggtitle("Danceability vs. Energy")
```



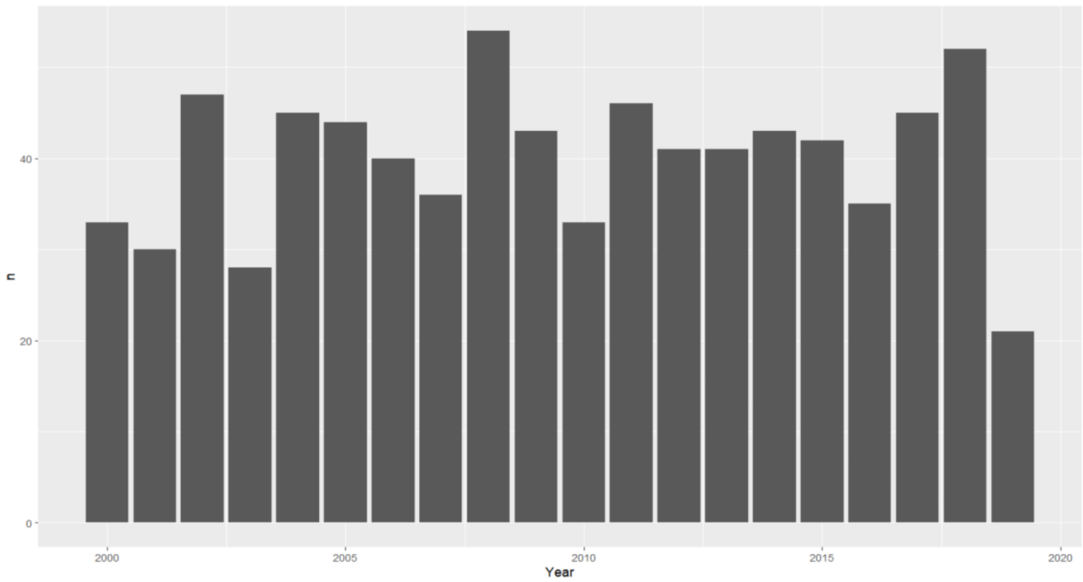
4) Genre Count Plot: The plot shows the count of songs for each top genre. Genres with more than 10 songs are included. It provides an overview of the distribution of songs across different genres.

```r
genre_count_plot <- music_data %>%
  group_by(Top.Genre) %>%
  summarize(n = n()) %>%
  filter(n > 10) %>%
  ggplot(aes(x = Top.Genre, y = n)) +
  geom_col()
print(genre_count_plot)
```



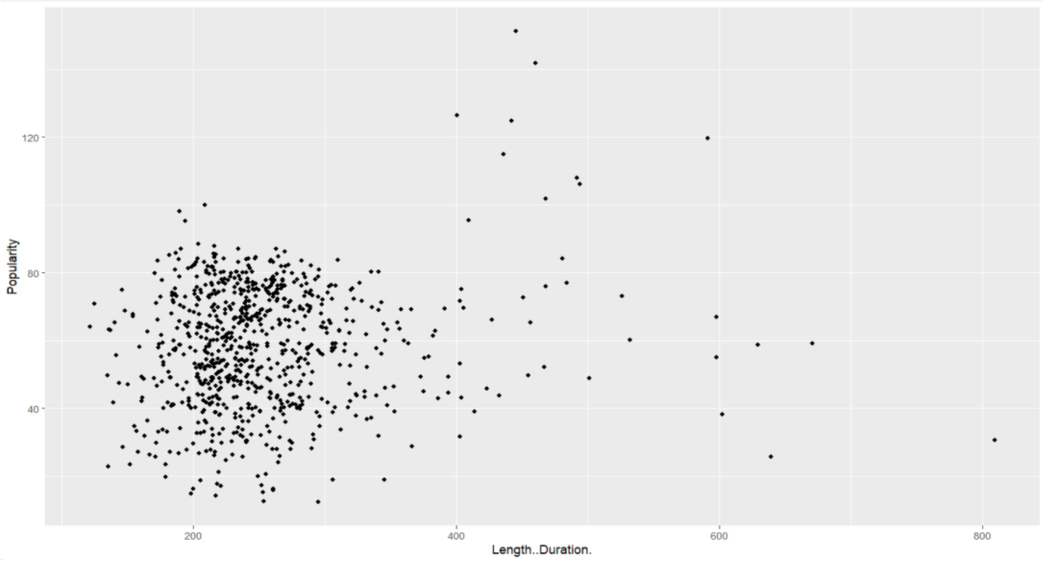
5) <u>Year Count Plot</u>: This plot displays the count of songs for each year. It helps visualize the distribution of songs over time, indicating which years have a higher or lower number of releases.

```r
> year_count_plot<-music_data %>%
+   count(Year,sort=TRUE) %>%
+   ggplot(aes(x=Year,y=n))+geom_col()
> print(year_count_plot)
>
```



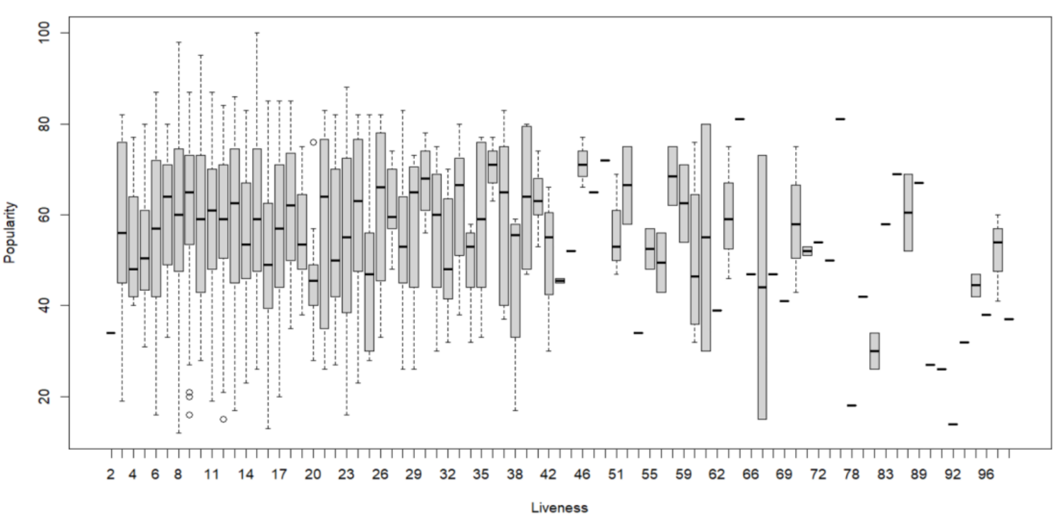
6) <u>Length vs. Popularity</u>: The jitter plot shows the relationship between song length and popularity. It gives a scattered representation of how the duration of a song relates to its popularity.

```
> music_data %>%
+   select(Title,Beats.Per.Minute..BPM.,Danceability,Length..Duration.,Popularity) %>%
+   group_by(Title) %>%
+   summarise_all(sum) %>%
+   ggplot(aes(x=Length..Duration.,y=Popularity))+geom_jitter()
> 
```



7) <u>Popularity vs. Liveness (Boxplot)</u>: The boxplot shows the distribution of popularity across different levels of liveness. It helps compare the median, quartiles, and potential outliers in popularity for different liveness categories.

```
> plot(music_data$Liveness,music_data$Popularity)
> boxplot(Popularity~Liveness,data=music_data)
> 
```



Overall, these graphs provide insights into the distribution, relationships, and patterns within the music dataset. They assist in understanding variables, identifying trends, and exploring potential correlations between attributes.

CORRELATION ANALYSIS

A correlation matrix was calculated to explore the relationships between various numerical variables in the dataset. The cor() function was used to compute the correlation coefficients, and the results were stored in the correlation_matrix variable.

```
> correlation_matrix<-cor(music_data[,c("Year","Energy","Danceability","Loudness..dB.","Liveness","Length..Duration.","Popularity")])
> print(correlation_matrix)
                         Year       Energy Danceability Loudness..dB.    Liveness Length..Duration.   Popularity
Year               1.00000000 -0.065945349   0.05240345   -0.02153571 -0.02926872      -0.043848059 -0.039250432
Energy            -0.06594535  1.000000000   0.14734541    0.72238675  0.16652483       0.005369493  0.119782470
Danceability       0.05240345  0.147345415   1.00000000    0.04883080 -0.09015558      -0.098747655  0.216114542
Loudness..dB.     -0.02153571  0.722386750   0.04883080    1.00000000  0.05688718      -0.045013724  0.299709702
Liveness          -0.02926872  0.166524826  -0.09015558    0.05688718  1.00000000       0.026290429 -0.113455882
Length..Duration. -0.04384806  0.005369493  -0.09874766   -0.04501372  0.02629043       1.000000000 -0.002756057
Popularity        -0.03925043  0.119782470   0.21611454    0.29970970 -0.11345588      -0.002756057  1.000000000
> 
```

VARIABLE NAME	DEPENDENT / INDEPENDENT	DEPENDENT VARIABLES
Year	INDEPENDENT	NIL
Energy	DEPENDENT	Loudness(0.72), Popularity(0.12)
Danceability	DEPENDENT	Popularity(0.22)
Loudness..dB.	DEPENDENT	Energy(0.72)
Liveness	INDEPENDENT	NIL
Length..Duration.	INDEPENDENT	NIL
Popularity	DEPENDENT	Energy(0.12), Danceability(0.22), Loudness(0.30)

By looking at the correlation matrix, we can make the following observations about the dependency of variables on each other:

Year: There is a very weak negative correlation between the year and the other variables. This suggests that the year of the music release has little to no impact on the other attributes.

Energy: There is a positive correlation between energy and loudness (0.72), indicating that songs with higher energy tend to be louder. There is also a positive correlation between energy and popularity (0.12), suggesting that more energetic songs may be more popular.

Danceability: There is a positive correlation between danceability and popularity (0.22), implying that songs that are more danceable have a higher likelihood of being popular.

Loudness: There is a strong positive correlation between loudness and energy (0.72), suggesting that louder songs tend to have higher energy levels.

Liveness: There is a weak negative correlation between liveness and popularity (-0.11), implying that live recordings or songs with higher liveness may be less popular.

Length/Duration: There is no significant correlation between the length/duration of a song and the other variables (-0.003 to 0.026), indicating that song duration does not strongly influence the other attributes.

Popularity: Popularity shows a weak positive correlation with energy (0.12), danceability (0.22), and loudness (0.30), suggesting that these attributes may have some influence on a song's popularity.

It's important to note that correlation does not imply causation, and the strength and significance of these correlations may vary. Further statistical analysis or modeling would be required to determine the precise relationships and their significance.

RECOMMENDATION MODEL

The recommendation model used in this project is based on content-based recommender. Content-based recommender is a commonly used technique in recommendation systems that uses the commonly used method cosine similarity method.

Feature Extraction: Identify the relevant features or attributes that you want to use for similarity calculation. These features will be used to create a profile for each item (in this case, songs) in the dataset. For example, you may choose to use attributes like genre, artist, and song duration.

Vectorization: Convert the extracted features into a numerical representation for each song. This can be done using various methods such as one-hot encoding or numerical scaling. The goal is to represent each song as a vector in a multi-dimensional feature space.

Before building the recommendation model, the music dataset underwent preprocessing steps to transform it into a suitable format for collaborative filtering. This included extracting keywords from song titles and creating a binary matrix to indicate the presence of each music genre. The resulting feature matrix represented the features of each song in terms of keywords and genre presence. The data was transformed into a suitable format for modeling or algorithm implementation. Genre and artist data was converted to genre matrix and artist matrix. Further a similarity matrix was created.

User Profile: To generate personalized recommendations, the model required information about the user's preferences. In this case, the user's favorite music genres were defined and used to create a user profile matrix. The user profile matrix represented the user's preferences in terms of genre presence, aligning with the feature matrix.

Similarity Calculations:

The next step involved calculating the similarity between the feature matrix and the user profile matrix. Similarity scores were computed using the cosine similarity measure, which measures the cosine of the angle between two vectors. It provides a measure of similarity between two vectors, with values ranging from -1 to 1, where 1 indicates perfect similarity.

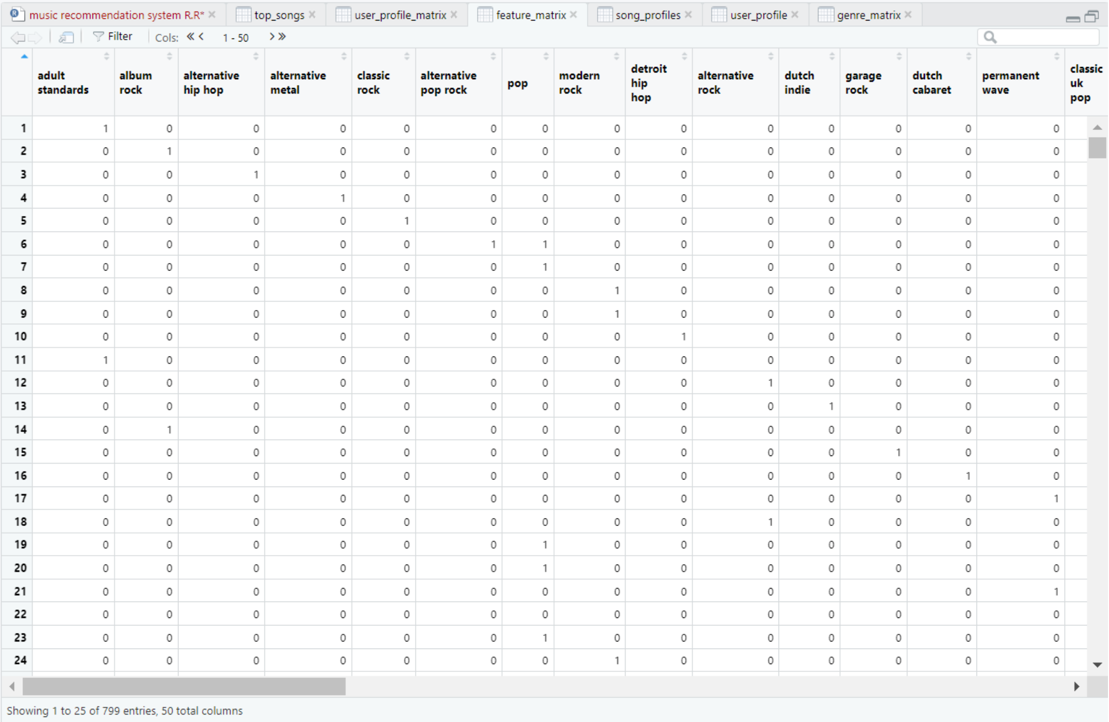
Recommendation Generation:

Once the similarity scores were calculated, they were merged with the original dataset to associate each song with its corresponding similarity score. The dataset was then sorted based on the similarity scores in descending order. The top N recommended songs were selected from the sorted dataset and presented to the user as personalized recommendations.

In addition to genre-based recommendations, artist-based recommendations were also provided. The process for artist-based recommendations was similar to genre-based recommendations. The presence of each artist in the dataset was identified, and a user profile matrix representing the user's favorite artists was created. Similarity scores were computed using the cosine similarity measure, and the dataset was sorted based on these scores to generate artist-based recommendations.

<u>**BASED ON GENRE:**</u>

```r
music_data$keywords <- str_split_fixed(music_data$Title, "\\s+", n = 12) %>%
  + apply(., 1, function(x) paste(unique(x), collapse = " "))
genres <- unique(unlist(strsplit(as.character(music_data$Top.Genre), ",")))
genre_matrix <- sapply(genres, function(Top.Genre) grepl(Top.Genre, music_data$Top.Genre,
                                                          ignore.case = TRUE))
feature_matrix <- cbind(music_data$keywords, genre_matrix)
View(genre_matrix)
View(feature_matrix)
song_profiles <- cbind(music_data[, c("Index", "Title")], genre_matrix)
```

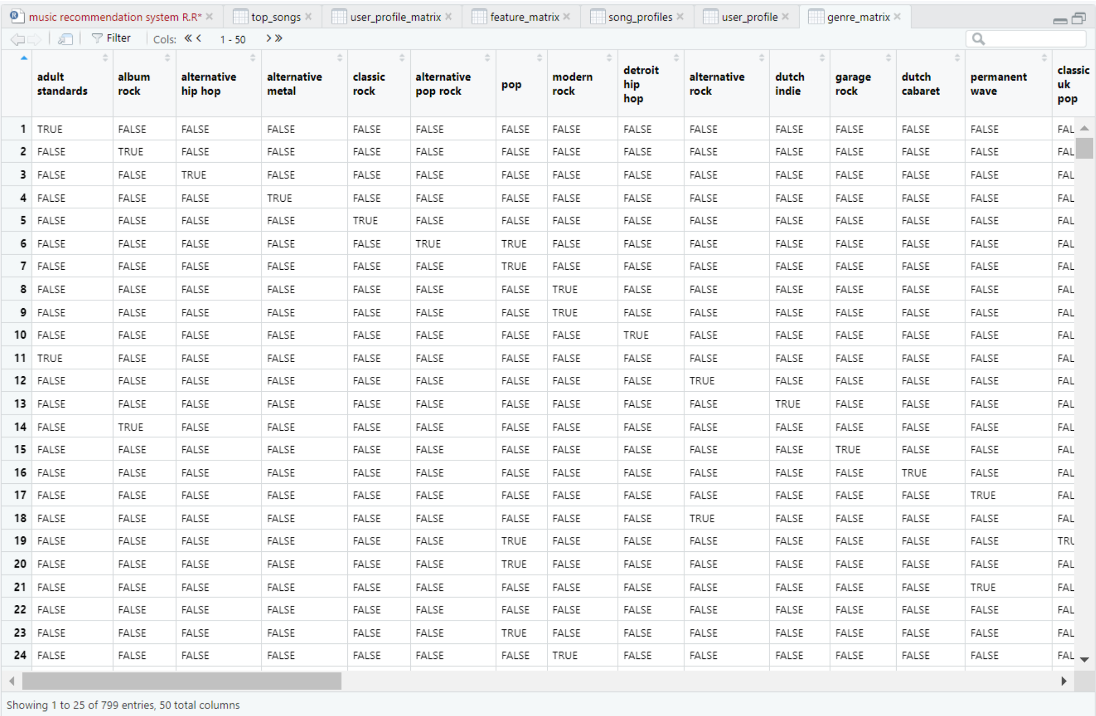


music recommendation system R.R* | top_songs | user_profile_matrix | feature_matrix | song_profiles | user_profile | genre_matrix

Filter Cols: 1 - 50

	adult standards	album rock	alternative hip hop	alternative metal	classic rock	alternative pop rock	pop	modern rock	detroit hip hop	alternative rock	dutch indie	garage rock	dutch cabaret	permanent wave	classic uk pop
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
9	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
10	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
13	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
14	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
16	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
17	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
19	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
20	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	1	0	0	0	0	0	0	

Showing 1 to 25 of 799 entries, 50 total columns



music recommendation system R.R* | top_songs | user_profile_matrix | feature_matrix | song_profiles | user_profile | genre_matrix

Filter Cols: 1 - 50

	adult standards	album rock	alternative hip hop	alternative metal	classic rock	alternative pop rock	pop	modern rock	detroit hip hop	alternative rock	dutch indie	garage rock	dutch cabaret	permanent wave	classic uk pop
1	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
2	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
3	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
4	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
5	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
6	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
8	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
9	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
10	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
11	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
12	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FAL
13	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FAL
14	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
15	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FAL
16	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FAL
17	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FAL
18	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FAL
19	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRU
20	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
21	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FAL
22	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
23	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL
24	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FAL

Showing 1 to 25 of 799 entries, 50 total columns

```r
favorite_genres <- c("pop","modern rock")
user_profile <- data.frame(title = "User Profile", genre_matrix = as.numeric(genres %in%
                                                                 favorite_genres))
similarity_scores <- proxy::simil(feature_matrix, user_profile$genre_matrix, method =
                                     "cosine")
user_genres <- c("pop","modern rock")
# Compute the user profile based on favorite genres
user_profile <- data.frame(genre_matrix = as.numeric(colnames(feature_matrix) %in%
                                                       favorite_genres))

# Create a matrix from the user_profile column to match the dimensions of feature_matrix
user_profile_matrix <- matrix(user_profile$genre_matrix, ncol = ncol(feature_matrix), byrow =
                                 TRUE)
similarity_scores <- proxy::simil(feature_matrix, user_profile_matrix, method = "cosine")
N <- 25
data_recommendations <- cbind(data, similarity = similarity_scores)
View(user_profile_matrix)
View(user_profile)
View(song_profiles)
# Convert feature_matrix to matrix type
feature_matrix <- as.matrix(feature_matrix)

# Convert user_profile_matrix to matrix type
user_profile_matrix <- as.matrix(user_profile_matrix)

# Convert missing values to 0
feature_matrix[is.na(feature_matrix)] <- 0
user_profile_matrix[is.na(user_profile_matrix)] <- 0
similarity_scores <- proxy::simil(feature_matrix, user_profile_matrix, method = "cosine")
similarity_matrix <- as.matrix(similarity_scores)
similarity_df <- data.frame(Index = 1:nrow(similarity_matrix), similarity_scores =
                              as.vector(similarity_matrix))
data_recommendations <- merge(data, similarity_df, by = "Index")
sorted_recommendations <-
  unique(data_recommendations[order(data_recommendations$similarity_scores, decreasing =
                                      TRUE), ])
top_songs <- sorted_recommendations[1:N, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
print(top_songs)
View(top_songs)
```

```
> top_songs <- sorted_recommendations[1:N, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
> print(top_songs)
    Index                           Title          Artist   Top.Genre Popularity
7       7                  She Will Be Loved        Maroon 5         pop         74
8       8                 Knights of Cydonia            Muse modern rock         69
9       9                    Mr. Brightside     The Killers modern rock         77
24     24                   Somebody Told Me     The Killers modern rock         69
44     44                    American Idiot       Green Day modern rock         78
76     76                       Dirty Diana Michael Jackson         pop         55
96     96                          21 Guns       Green Day modern rock         75
112   112               Everybody's Changing           Keane modern rock         54
113   113                  How to Save a Life        The Fray modern rock         80
117   117                       Feeling Good            Muse modern rock         51
119   119                      Use Somebody  Kings of Leon modern rock         76
165   165                           Uprising            Muse modern rock         76
189   189                       Plug in Baby            Muse modern rock         53
198   198                          Starlight            Muse modern rock         73
204   204           Supermassive Black Hole            Muse modern rock         73
213   213      Wake Me up When September Ends       Green Day modern rock         77
236   236                         Fireflies        Owl City         pop         79
247   247 Will You Be There - Single Version Michael Jackson         pop         53
255   255                  Time Is Running Out            Muse modern rock         69
290   290    Black or White - Single Version Michael Jackson         pop         64
307   307                              Human     The Killers modern rock         73
313   313                       Bend & Break           Keane modern rock         43
315   315                   Sing for Absolution            Muse modern rock         57
339   339                         Resistance            Muse modern rock         63
340   340                          Bedshaped           Keane modern rock         41
> View(top_songs)
>
```

	Index	Title	Artist	Top.Genre	Popularity
7	7	She Will Be Loved	Maroon 5	pop	74
8	8	Knights of Cydonia	Muse	modern rock	69
9	9	Mr. Brightside	The Killers	modern rock	77
24	24	Somebody Told Me	The Killers	modern rock	69
44	44	American Idiot	Green Day	modern rock	78
76	76	Dirty Diana	Michael Jackson	pop	55
96	96	21 Guns	Green Day	modern rock	75
112	112	Everybody's Changing	Keane	modern rock	54
113	113	How to Save a Life	The Fray	modern rock	80
117	117	Feeling Good	Muse	modern rock	51
119	119	Use Somebody	Kings of Leon	modern rock	76
165	165	Uprising	Muse	modern rock	76
189	189	Plug in Baby	Muse	modern rock	53
198	198	Starlight	Muse	modern rock	73
204	204	Supermassive Black Hole	Muse	modern rock	73
213	213	Wake Me up When September Ends	Green Day	modern rock	77
236	236	Fireflies	Owl City	pop	79
247	247	Will You Be There - Single Version	Michael Jackson	pop	53
255	255	Time Is Running Out	Muse	modern rock	69
290	290	Black or White - Single Version	Michael Jackson	pop	64
307	307	Human	The Killers	modern rock	73
313	313	Bend & Break	Keane	modern rock	43
315	315	Sing for Absolution	Muse	modern rock	57
339	339	Resistance	Muse	modern rock	63
340	340	Bedshaped	Keane	modern rock	41

Showing 1 to 25 of 25 entries, 5 total columns

BASED ON ARTISTS:

```
artists <- unique(unlist(strsplit(as.character(music_data$Artist), ",")))
artist_matrix <- sapply(artists, function(Artist) grepl(Artist, music_data$Artist,ignore.case = TRUE))
real_matrix <- cbind(music_data$keywords, artist_matrix)
print(artist_matrix)
View(artist_matrix)
View(real_matrix)
songs <- cbind(music_data[, c("Index", "Title")], artist_matrix)
```

	Norah Jones	Deep Purple	Gorillaz	Foo Fighters	Bruce Springsteen	City To City	Maroon 5	Muse	The Killers	Eminem	Elvis Presley	The White Stripes	De Dijk	Ten Years After	Arctic Monkeys	Paul de Leeuw
1	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
2	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
3	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
4	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
5	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
6	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
8	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
9	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
10	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
11	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
12	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
13	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE
14	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
15	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
16	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE
17	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
18	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
19	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
20	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
21	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
22	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
23	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
24	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Showing 1 to 25 of 799 entries, 50 total columns music recommendation system R.R* | top_tracks | user_matrix | real_matrix | songs | user | artist_matrix | genre_matrix | user_profile

Filter Cols: « ‹ 1 - 50 › »

	Norah Jones	Deep Purple	Gorillaz	Foo Fighters	Bruce Springsteen	City To City	Maroon 5	Muse	The Killers	Eminem	Elvis Presley	The White Stripes	De Dijk	Ten Years After	Arctic Monkeys	Paul de Leeuw
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
10	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	

Showing 1 to 25 of 799 entries, 50 total columns

```r
songs <- cbind(music_data[, c("Index", "Title")], artist_matrix)
favorite_artists<- c("Coldplay", "Adele")
user <- data.frame(title = "User Profile", artist_matrix = as.numeric(artists %in%favorite_artists))
similarity_score_1 <- proxy::simil(real_matrix,user$artist_matrix, method ="cosine")
user_artists <- c("Coldplay", "Adele")
user<- data.frame(artist_matrix = as.numeric(colnames(real_matrix) %in%favorite_artists))
user_matrix <- matrix(user$artist_matrix, ncol = ncol(real_matrix), byrow = TRUE)
similarity_score_1 <- proxy::simil(real_matrix, user_matrix, method = "cosine")
N1 <- 15
data_recommendation_1 <- cbind(data, similarity = similarity_score_1)
View(user_matrix)
View(user)
View(songs)
real_matrix <- as.matrix(real_matrix)
user_matrix <- as.matrix(user_matrix)
real_matrix[is.na(real_matrix)] <- 0
user_matrix[is.na(user_matrix)] <- 0
similarity_score_1 <- proxy::simil(real_matrix, user_matrix, method = "cosine")
similarities_matrix_1 <- as.matrix(similarity_score_1)
similarities_df_1 <- data.frame(Index = 1:nrow(similarities_matrix_1), similarity_score_1 =as.vector(similarities_matrix_1))
data_recommendation_1 <- merge(music_data, similarities_df_1, by = "Index")
sorted_tracks <-unique(data_recommendation_1[order(data_recommendation_1$similarity_score_1, decreasing =TRUE), ])
top_tracks <- sorted_tracks [1:N1, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
print(top_tracks)
View(top_tracks)
```

```
> top_tracks <- sorted_tracks [1:N1, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
> print(top_tracks)
    Index                          Title   Artist      Top.Genre Popularity
17     17                 Speed of Sound Coldplay permanent wave         69
21     21                        Fix You Coldplay permanent wave         81
31     31                  The Scientist Coldplay permanent wave         84
33     33               Chasing Pavements    Adele   british soul         63
71     71              Make You Feel My Love    Adele   british soul         73
137   137                           Talk Coldplay permanent wave         63
156   156 God Put a Smile upon Your Face Coldplay permanent wave         64
201   201                     Green Eyes Coldplay permanent wave         63
240   240                         Yellow Coldplay permanent wave         82
254   254                      Amsterdam Coldplay permanent wave         57
258   258                 Hometown Glory    Adele   british soul         59
262   262                    In My Place Coldplay permanent wave         70
311   311                   Viva La Vida Coldplay permanent wave         78
324   324                    Violet Hill Coldplay permanent wave         61
329   329                          Lost! Coldplay permanent wave         53
> View(top_tracks)
>
```

	Index	Title	Artist	Top.Genre	Popularity
17	17	Speed of Sound	Coldplay	permanent wave	69
21	21	Fix You	Coldplay	permanent wave	81
31	31	The Scientist	Coldplay	permanent wave	84
33	33	Chasing Pavements	Adele	british soul	63
71	71	Make You Feel My Love	Adele	british soul	73
137	137	Talk	Coldplay	permanent wave	63
156	156	God Put a Smile upon Your Face	Coldplay	permanent wave	64
201	201	Green Eyes	Coldplay	permanent wave	63
240	240	Yellow	Coldplay	permanent wave	82
254	254	Amsterdam	Coldplay	permanent wave	57
258	258	Hometown Glory	Adele	british soul	59
262	262	In My Place	Coldplay	permanent wave	70
311	311	Viva La Vida	Coldplay	permanent wave	78
324	324	Violet Hill	Coldplay	permanent wave	61
329	329	Lost!	Coldplay	permanent wave	53

Overall, the recommendation model leveraged collaborative filtering techniques, specifically user-based collaborative filtering, to generate personalized music recommendations. By calculating similarity scores and sorting the dataset based on these scores, the model identified songs that were similar to the user's preferences in terms of genre or artist presence. This approach allows users to discover new songs that align with their musical tastes and enhances their overall music listening experience.

MUSIC RECOMMENDATION SYSTEM R PROJET CODE

```r
File_path<-file.path(getwd(),"musicdataset.csv")
File_path
music_data<-read.csv("musicdataset.csv")
music_data
sum(is.na(music_data))
library(dplyr)
library(tidyr)
library(stringr)
library(proxy)
library(recommenderlab)
library(ggplot2)
library(data.table)
library(reshape2)
library(tidyverse)
####
print(head(music_data))
print(tail(music_data))
print(summary(music_data))
print(str(music_data))
music_genre<-unique(music_data$Top.Genre)
print(music_genre)
genre_count<-music_data %>%
  count(Top.Genre,sort=TRUE)
print(genre_count)
genre_count_plot <- music_data %>%
  group_by(Top.Genre) %>%
  summarize(n = n()) %>%
  filter(n > 10) %>%
  ggplot(aes(x = Top.Genre, y = n)) +
  geom_col()
print(genre_count_plot)
year_count_plot<-music_data %>%
  count(Year,sort=TRUE) %>%
  ggplot(aes(x=Year,y=n))+geom_col()
print(year_count_plot)
artist_sort<-music_data %>%
  count(Artist,sort=TRUE) %>%
  ggplot(aes(x=n))+geom_density()
artist_sort
top_10_artist<-head(artist_sort,10)
top_10_artist %>%
```

```r
artist_sort
top_10_artist<-head(artist_sort,10)
top_10_artist %>%
  ggplot(aes(x=n))+geom_col()
music_data %>%
  select(Title,Beats.Per.Minute..BPM.,Danceability,Length..Duration.,Popularity) %>%
  group_by(Title) %>%
  summarise_all(sum) %>%
  ggplot(aes(x=Length..Duration.,y=Popularity))+geom_jitter()
correlation_matrix<-cor(music_data[,c("Year","Energy","Danceability","Loudness..dB.","Liveness","Length..Duration.","Popularity")])
print(correlation_matrix)
heatmap(correlation_matrix,
        cmap = ggplot2::scale_fill_viridis_c(),
        cexRow = 0.8,
        cexCol = 0.8,
        margins = c(10, 10))
plot(music_data$Liveness,music_data$Popularity)
ggplot(data = music_data, aes(x = Danceability, y = Energy)) +
  geom_point() +
  xlab("Danceability") +
  ylab("Energy") +
  ggtitle("Danceability vs. Energy")
boxplot(Popularity~Liveness,data=music_data)
artist_count_plot <- music_data %>%
  group_by(Artist) %>%
  summarize(n = n()) %>%
  filter(n > 6) %>%
  ggplot(aes(x = Artist, y = n)) +
  geom_col()
print(artist_count_plot)
any_missing <- any(is.na(music_data))
any_missing
recommendation_model <- recommenderRegistry$get_entries(dataType = "realRatingMatrix")
names(recommendation_model)
lapply(recommendation_model, "[[", "description")
####################################################################################
####################################################################################
music_data$keywords <- str_split_fixed(music_data$Title, "\\s+", n = 12) %>%
  + apply(., 1, function(x) paste(unique(x), collapse = " "))
genres <- unique(unlist(strsplit(as.character(music_data$Top.Genre), ",")))
genre_matrix <- sapply(genres, function(Top.Genre) grepl(Top.Genre, music_data$Top.Genre
```

```r
  + apply(., 1, function(x) paste(unique(x), collapse = " "))
genres <- unique(unlist(strsplit(as.character(music_data$Top.Genre), ",")))
genre_matrix <- sapply(genres, function(Top.Genre) grepl(Top.Genre, music_data$Top.Genre,
                                                          ignore.case = TRUE))
feature_matrix <- cbind(music_data$keywords, genre_matrix)
View(genre_matrix)
View(feature_matrix)
song_profiles <- cbind(music_data[, c("Index", "Title")], genre_matrix)
favorite_genres <- c("pop","modern rock")
user_profile <- data.frame(title = "User Profile", genre_matrix = as.numeric(genres %in%
                                                                              favorite_genres))
similarity_scores <- proxy::simil(feature_matrix, user_profile$genre_matrix, method =
                                    "cosine")
user_genres <- c("pop","modern rock")
# Compute the user profile based on favorite genres
user_profile <- data.frame(genre_matrix = as.numeric(colnames(feature_matrix) %in%
                                                          favorite_genres))

# Create a matrix from the user_profile column to match the dimensions of feature_matrix
user_profile_matrix <- matrix(user_profile$genre_matrix, ncol = ncol(feature_matrix), byrow =
                                    TRUE)
similarity_scores <- proxy::simil(feature_matrix, user_profile_matrix, method = "cosine")
N <- 25
data_recommendations <- cbind(data, similarity = similarity_scores)
View(user_profile_matrix)
View(user_profile)
View(song_profiles)
# Convert feature_matrix to matrix type
feature_matrix <- as.matrix(feature_matrix)

# Convert user_profile_matrix to matrix type
user_profile_matrix <- as.matrix(user_profile_matrix)

# Convert missing values to 0
feature_matrix[is.na(feature_matrix)] <- 0
user_profile_matrix[is.na(user_profile_matrix)] <- 0
similarity_scores <- proxy::simil(feature_matrix, user_profile_matrix, method = "cosine")
similarity_matrix <- as.matrix(similarity_scores)
similarity_df <- data.frame(Index = 1:nrow(similarity_matrix), similarity_scores =
                                as.vector(similarity_matrix))
data_recommendations <- merge(data, similarity_df, by = "Index")
```

```r
similarity_df <- data.frame(Index = 1:nrow(similarity_matrix), similarity_scores =
                                as.vector(similarity_matrix))
data_recommendations <- merge(data, similarity_df, by = "Index")
sorted_recommendations <-
  unique(data_recommendations[order(data_recommendations$similarity_scores, decreasing =
                                      TRUE), ])
top_songs <- sorted_recommendations[1:N, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
print(top_songs)
View(top_songs)
##########################################################################################
##########################################################################################
artists <- unique(unlist(strsplit(as.character(music_data$Artist), ",")))
artist_matrix <- sapply(artists, function(Artist) grepl(Artist, music_data$Artist,ignore.case = TRUE))
real_matrix <- cbind(music_data$keywords, artist_matrix)
print(artist_matrix)
View(artist_matrix)
View(real_matrix)
songs <- cbind(music_data[, c("Index", "Title")], artist_matrix)
favorite_artists<- c("Coldplay", "Adele")
user <- data.frame(title = "User Profile", artist_matrix = as.numeric(artists %in%favorite_artists))
similarity_score_1 <- proxy::simil(real_matrix,user$artist_matrix, method ="cosine")
user_artists <- c("Coldplay", "Adele")
user<- data.frame(artist_matrix = as.numeric(colnames(real_matrix) %in%favorite_artists))
user_matrix <- matrix(user$artist_matrix, ncol = ncol(real_matrix), byrow = TRUE)
similarity_score_1 <- proxy::simil(real_matrix, user_matrix, method = "cosine")
N1 <- 15
data_recommendation_1 <- cbind(data, similarity = similarity_score_1)
View(user_matrix)
View(user)
View(songs)
real_matrix <- as.matrix(real_matrix)
user_matrix <- as.matrix(user_matrix)
real_matrix[is.na(real_matrix)] <- 0
user_matrix[is.na(user_matrix)] <- 0
similarity_score_1 <- proxy::simil(real_matrix, user_matrix, method = "cosine")
similarities_matrix_1 <- as.matrix(similarity_score_1)
similarities_df_1 <- data.frame(Index = 1:nrow(similarities_matrix_1), similarity_score_1 =as.vector(similarities_matrix_1))
View(similarities_df_1)
data_recommendation_1 <- merge(music_data, similarities_df_1, by = "Index")
print(data_recommendation_1)
sorted_tracks <-unique(data_recommendation_1[order(data_recommendation_1$similarity_score_1, decreasing =TRUE), ])
```

138:82 (Untitled)

```r
similarity_score_1 <- proxy::simil(real_matrix, user_matrix, method = "cosine")
similarities_matrix_1 <- as.matrix(similarity_score_1)
similarities_df_1 <- data.frame(Index = 1:nrow(similarities_matrix_1), similarity_score_1 =as.vector(similarities_matrix_1))
View(similarities_df_1)
data_recommendation_1 <- merge(music_data, similarities_df_1, by = "Index")
print(data_recommendation_1)
sorted_tracks <-unique(data_recommendation_1[order(data_recommendation_1$similarity_score_1, decreasing =TRUE), ])
top_tracks <- sorted_tracks [1:N1, c("Index", "Title", "Artist", "Top.Genre","Popularity")]
print(top_tracks)
View(top_tracks)
```

CONCLUSION

In conclusion, this project successfully analyzed a music dataset and provided personalized music recommendations based on user preferences. By utilizing various data manipulation and visualization techniques, the project uncovered insights about the dataset, such as genre distribution, popularity of artists, and correlations between variables. The user-based and artist-based recommendation models enhanced the music listening experience by suggesting songs that aligned with the user's preferences. This project demonstrates the application of data analysis and recommendation systems in the music industry, providing valuable insights for music enthusiasts and industry professionals alike.

SCOPE OF IMPROVEMENT

Data Sparsity: The recommendation system may face challenges when dealing with sparse data, where users have rated or interacted with only a small portion of the available items. Sparse data can limit the accuracy and relevance of recommendations, particularly for niche or less popular items.

Limited Domain Coverage: The music recommendation system focuses on the analysis and recommendation of songs based on the provided dataset. It may not cover other aspects of the music domain, such as lyrics analysis, music production techniques, or cultural context, which could provide additional insights for a more comprehensive recommendation system.

Contextual Factors: The recommendation system may not consider contextual factors such as mood, location, or specific occasions, which can influence user preferences. Incorporating contextual information can enhance the relevance and personalization of recommendations.

Explainability and Transparency: Enhance the transparency and explainability of the recommendation system. Users may appreciate understanding why certain recommendations are being made, such as by providing explanations based on user preferences, similarity metrics, or music features.

Continuous Feedback and User Input: Collect feedback from users and incorporate it into the recommendation system's improvement cycle. Encourage users to provide explicit feedback, such as ratings or reviews, to further refine the system's understanding of their preferences.