Music Recommender System

Made by: Kanishk Jain, Vishrut Grover, Hemang Jain (Team KVH)

Objective

To generate similarity scores between various songs and develop a recommendation model that leverages content-based filtering techniques to suggest personalised song recommendations to users on the basis of previous songs they have heard, enhancing their music discovery experience.

Introduction

In the realm of music streaming and digital entertainment, providing accurate and personalised song recommendations is essential for improving user engagement and satisfaction. A song recommender system combines content-based filtering, which analyses the features and lyrics of songs (metadata) as well as artist features to deliver highly relevant music suggestions.

By integrating these advanced recommendation techniques, the system aims to enhance users' music discovery journey, ensuring they find new songs and artists that align with their tastes. The development of such a model not only enriches the user experience but also supports the music industry's efforts in promoting diverse and emerging artists, fostering a more dynamic and engaging music ecosystem.

Thought Process

The song recommender system leverages various features such as genre, audio features, artist features and lyrical content for content-based filtering.

By combining these approaches, the system can address the limitations of each method when used in isolation, providing more accurate and personalised recommendations.

Implementation

Data Collection

The complete metadata of 25k songs was scraped and collected from music streaming services like Spotify Web API.

Song name, Artist name, Genre, Album Details, Release Date, Track Popularity, and numerous other audio attributes are included in the metadata. Artist details were also obtained, including the artist's popularity, number of followers, and genres of work produced. These details will be useful for the model to compare the artist not just as an individual but also based on the artist's output.

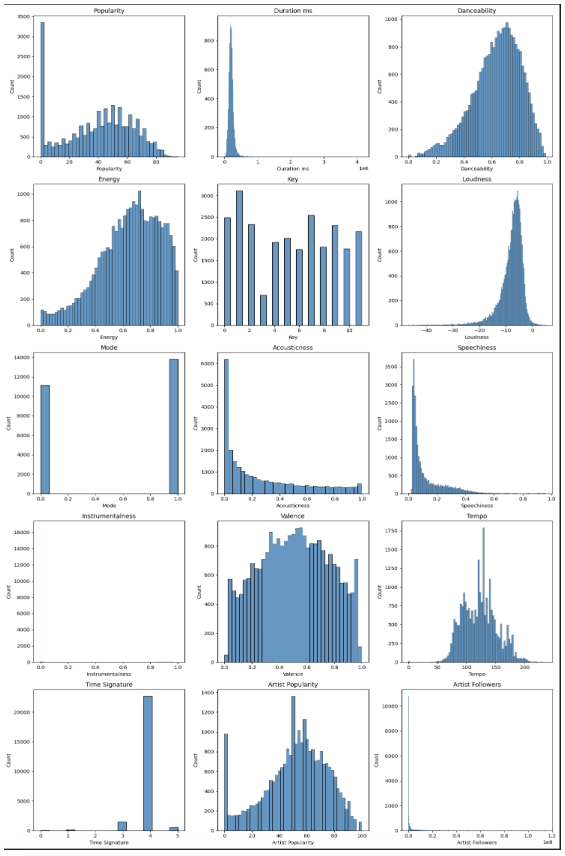
These music lyrics were used to scrape the song's lyrics using the YTmusic as the primary scraper and keeping Musixmatch APIs and scraping the lyrics directly from tekstowo website as a fallback.

Data Visualization and Preprocessing

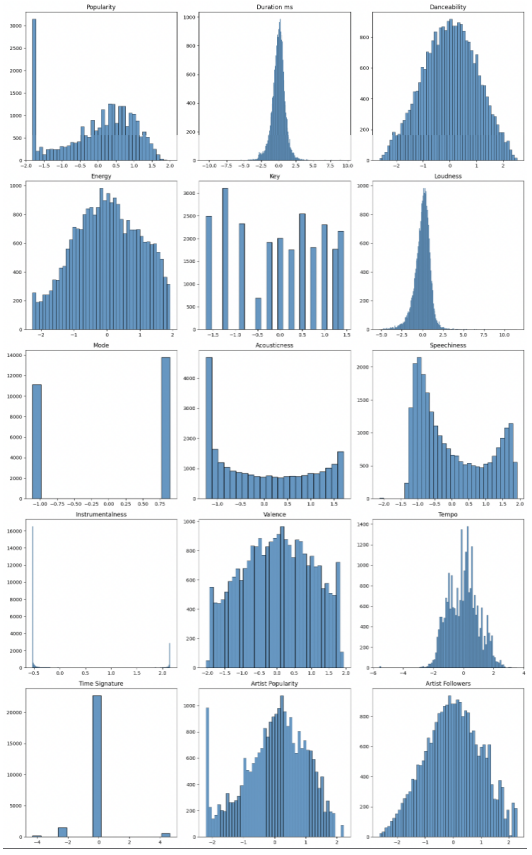
Embeddings for Spotify Features (Audio Features)

Numerous features, that is, the various numerical attributes or metrics associated with music tracks extracted from Spotify are visualised using the histplots and correlation matrix. Most of the graphs in histplots were skewed, thus we utilised Yeo-Johnson transformations to normally distribute the skewed data.

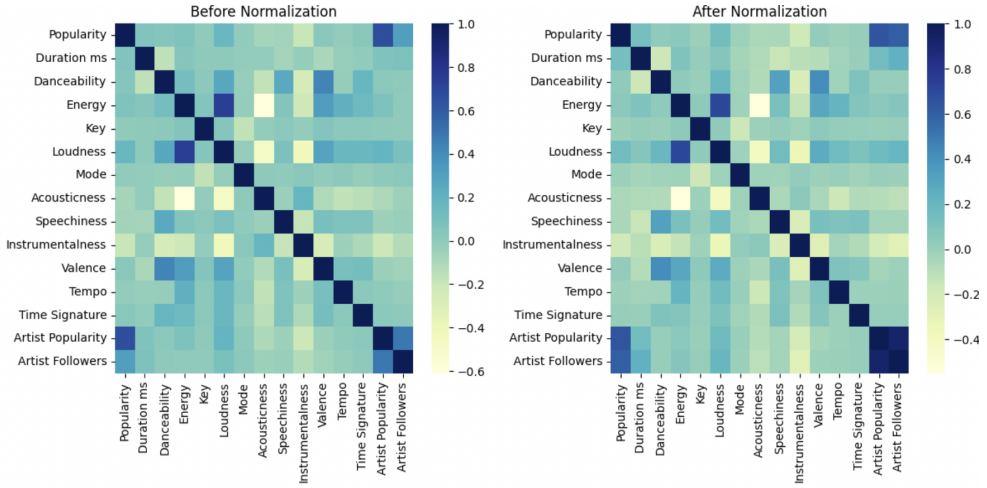
Yeo-Johnson Transformations were favoured over Box-Cox Transformations, which only apply to precisely positive values. These were then scaled so that they lie within a specific range



Histplots Before Transformation



Histplots After Transformation



Correlation Matrix Before and After Transformation

Embeddings for Artist Features
Separate embeddings are crafted for artist information to differentiate each artist not just as an individual but also based on their body of work.
Initially, we utilized Sentence Transformer, which converts sentences or texts into numerical vectors (embeddings) using various pre-trained models trained on extensive text data. However, we transitioned to BERT-based models, that is, BERT uncased, primarily because genres are predominantly in English.
This switch ensures that the embeddings accurately capture the nuances and stylistic elements inherent in English-language genres, thereby enhancing the model's ability to categorize and profile artists effectively based on their musical genres and other numerical features.

Embeddings for Song Lyrics
We transitioned from Word2Vec to XLM-RoBERTa for embedding lyrics because XLM-RoBERTa offers significant advantages, especially for multilingual data like song lyrics.
Unlike Word2Vec, which requires language-specific models and embeddings, XLM-RoBERTa is pretrained on a diverse range of languages, making it versatile for lyrics in different languages without needing separate models. Also Word2vec has a smaller input window
This model leverages a transformer architecture, allowing it to capture more complex linguistic relationships and nuances in the lyrics, which can improve the quality and relevance of the embeddings.
By using XLM-RoBERTa, we ensure that our embeddings are robust across various languages, enhancing the accuracy and utility of our lyric analysis and processing tasks.

One Hot Encoded Languages as an Embedding
We were able to get languages from our multilingual lyrics and decided to one hot encode them and put them in a list and then use them as an embedding.
This helped us generate language-specific songs, which resulted in a better accuracy

Features Description

Feature Embeddings (Song Metadata Features):

- **Name:** Name of the song

- **Release Date:** Date of when the song was released. We extracted the year of the release date as it was more suitable to work on. (Listeners of the same generation are likely to like songs from the same era)

- **Popularity:** A track's popularity is between 0 and 100, with 100 being the most popular. The popularity is calculated by algorithm and is based mostly on the total number of plays the track has had and how recent those plays are.

- **Duration (ms):** The duration of the track in milliseconds. (Some people prefer to listen to long songs while some prefer shorter songs)

- **Explicit:** Explicit lyrics are song lyrics that contain language and themes that are considered inappropriate for children or sensitive audiences. This feature returns a boolean value that indicates whether the track has explicit lyrics.

- **Genre:** List of genres corresponding to songs, initially we scraped them, later we got the artist genres from the 25k songs database.
  We dropped this features and it highly resembles with the artist genre, making it not very useful

- **Danceability:** Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity.

- **Energy:** represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale.

- **Key:** The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on.

- **Loudness:** The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude).

- **Mode:** Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. A mode is a group of eight consecutive pitches with no skipped pitches, and the first and last tones are repeated.

- **Speechiness:** Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audiobook, poetry), the closer to 1.0 the attribute value

- **Acousticness:** Acousticness in music refers to a song's likelihood of being dominated by acoustic instruments. Acoustic instruments include guitars, pianos, drums (without heavy electronic manipulation), violins, etc.

- **Instrumentalness:** Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater the likelihood the track contains no vocal content

- **Valence:** Describes the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

- **Tempo:** The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the

average beat duration.

- **Time Signature:** The time signature (metre) is a notational convention to specify how many beats are in each bar (or measure). (Bar is the vertical lines in music notes)

Lyrics Embedding:

- **Lyrics :** This feature shows the scrapped lyrics of the song, for this we used YTMsuic API, Musixmatch and also directly scraped lyrics from Tekstowo.
  Running Genius on local initially gave us the complete songs but for just a few songs and BeautifulSoup wasn't working on the major websites, so we didn't use it in the end.
  So, we majorly used YTMusic API which gave us the complete lyrics most of the time and as a fallback option used Tekstowo where we directly scraped the lyrics from the website using BeautifulSoup and if still the lyrics weren't found we used Musixmatch which got us just 30% of the lyrics but something is better than nothing!
  Then, we got our embeddings using the XLM-RoBERTa model.

Artist Embedding Features:

- **Artist:** Label encoded the name of the artist of the song (Listeners like to listen to song of the same artists)

- **Album:** Label encoded the album name of which the song is part of. (Songs in the same album are similar)

- **Artist Genres:** This features tells about the all the genres of work produced by the artist, we used BERT uncased base to generate embeddings for genre.

- **Artist Popularity:** A rating of close to 100 indicates that an artist is popular. Artist popularity is a number between 0 and 100. Essentially, artist-composed tracks are used to compute this.

- **Artist Followers:** It tells the information about the total followers of an artist on spotify.

Similarity Function:

Weights for different sorts of music characteristics, such as lyrics, artist information, and other numerical attributes like popularity, energy, and danceability, are defined independently depending on the intuition of the value of metrics and hit and trial basis. These weights are then applied to the features of each song in the dataset, which is flattened such that each row corresponds to a song with suitably weighted features.

The weighted feature vector for a given song name from the flattened dataset is retrieved to calculate the average feature vector for a list of songs. This combined vector represents the overall characteristics of the given songs. Now this combined vector is used to find the similarity between the given song and all the vectors of the song in a flattened dataset using Cosine Similarity, Inverse Manhattan distance, and Equality Check.

The overall similarity score for two songs is computed by weighted averaging these feature-specific similarity scores, weighted according to the importance of each feature. This similarity score can then be used to sort and select the top N most similar songs, returning a DataFrame with the song names and their similarity scores. This approach enables recommendations based on the similarity of song features, facilitating personalised music suggestions.

Results:

1) **Average Similarity Score:** This criterion evaluates the average similarity score of the last song in the playlist compared to the rest of the playlist

```
[176]: print("Average Similarity Score over complete Users Data: ", total / I, "\n Number of playlis

Average Similarity Score over complete Users Data:  0.7309872068419996
 Number of playlists:  2913
```

2) **Accuracy Based on a Threshold:** This criterion measures the model's accuracy when the average similarity score is thresholded at 0.7. This means based on the first n-1 songs, what is the similarity score of the nth song now if the similarity score is above 0.7 consider it positive, else negative. So, we'll be reporting the percent of positives.

```
[177]: avg_pos_rate = (pos / num) * 100

       print("Accuracy based on threshold at 0.7 is: ", avg_pos_rate)

Accuracy based on threshold at 0.7 is:  76.1071060762101
```

3) **Human Testing:** Here, five human testing samples are added where we have provided 3 songs on the basis of which our model gives the fourth prediction.

```
[211]: recommended_song4 = recommend_songs_based_on_three("Road to Nowhere" ,
                                                         "Bones" ,
                                                         "Stereo Hearts (feat. Adam Levine)", data,
                                                         similarity_functions, numerical_weights)

       print(recommended_song4)

                  Name  Average Similarity
2500  Higher Ground           0.871436
```

"Higher Ground" is rooted in funk and soul. "Road to Nowhere" is New Wave and Alternative Rock. "Bones" is Pop Rock and Alternative Rock. "Stereo Hearts" is Pop Rap and Alternative Hip Hop. **All songs have genres related to pop-rock, instrumental and all the songs have high upbeat tempo** above 100BPM "Higher Ground" focuses on social change and spiritual elevation. "Road to Nowhere" is philosophical, exploring the journey of life. "Bones" discusses resilience and overcoming challenges. **They all have a deeper, reflective theme.**

```
[212]: ended_song5 = recommend_songs_based_on_three('Genie In a Bottle', 'Despacito',
                                                    'Como Yo Le Doy - Spanglish Version',
                                                    data, similarity_functions, numerical_weights)
       recommended_song5)
```

	Name	Average Similarity
16865	I Know You Want Me (Calle Ocho)	0.857222

"I Know You Want Me (Calle Ocho)" shares influences with "Despacito" and "Como Yo Le Doy." It also **has dance and Latin hip hop elements**, similar to "Como Yo Le Doy."
"I Know You Want Me (Calle Ocho)" focuses on party, flirtation, and nightlife, similar to "Como Yo Le Doy," which also has **themes of party and sensuality.**
"Despacito" has a romantic and sensual theme, aligning with the flirtatious aspect of "I Know You Want Me (Calle Ocho)."

```
[210]: recommended_song3 = recommend_songs_based_on_three("Body",
                                                          "There Is a Light That Never Goes Out - 20
                                                          "Millennium", data,
                                                          similarity_functions, numerical_weights)
       print(recommended_song3)
```

	Name	Average Similarity
5259	This Charming Man - 2011 Remaster	0.851315

"This Charming Man " and "There Is a Light That Never Goes Out," both by The Smiths, share the same **high artist popularity** (0.77) and **follower count** (0.051350), reflecting their significant influence and fan base. "This Charming Man," "There Is a Light That Never Goes Out," and "Millennium" share **high energy**, indicating a blend of acoustic elements and lively sound.
All songs have strong uplifting beats , fitting them into **the modern pop/indie pop genre** .

```
▷ recommended_song1 = recommend_songs_based_on_three("Silence" , "Roses" , "Animal", data,
                                                     similarity_functions, numerical_weights)
  print(recommended_song1)
```

	Name	Average Similarity
408	Happier	0.865042

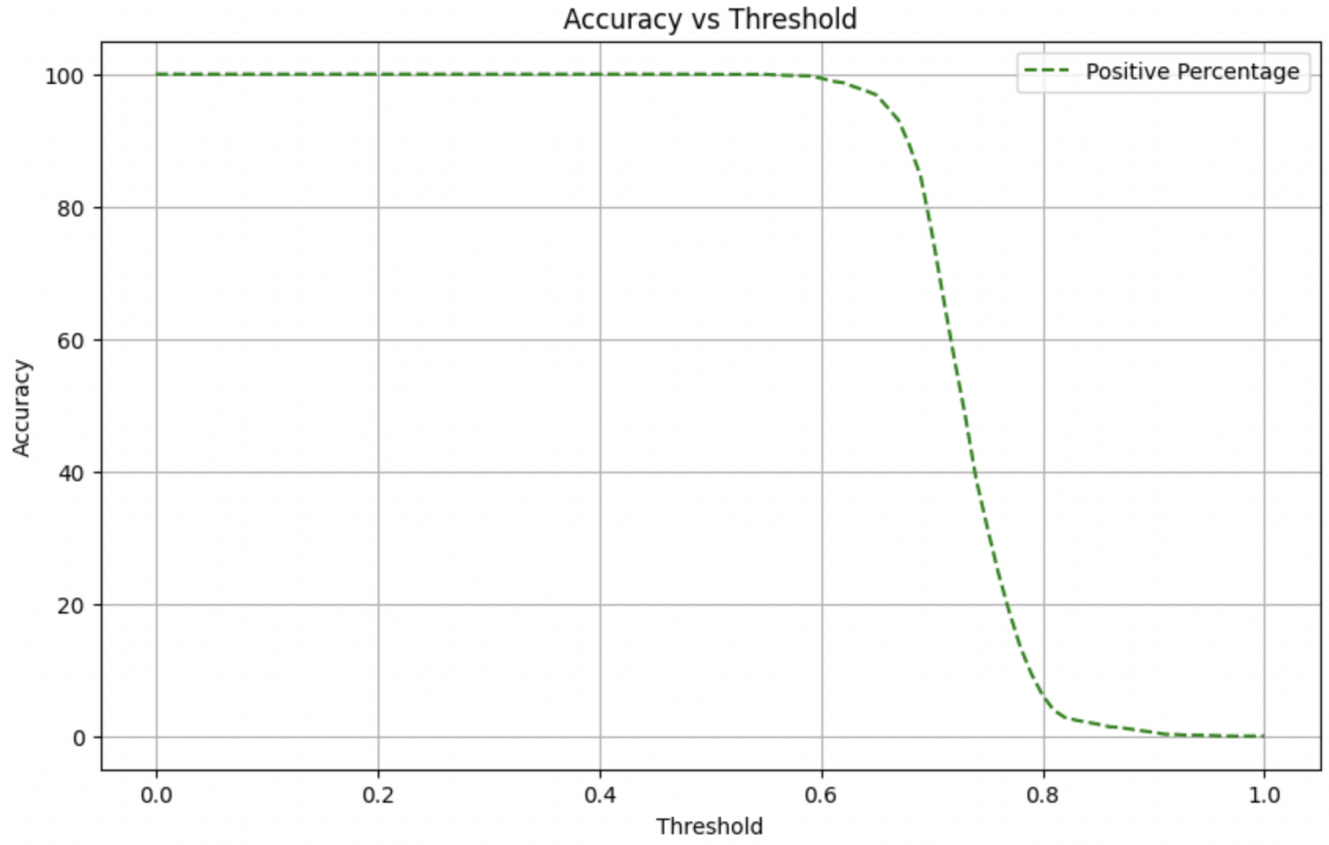
"Silence" and "Roses" have shown varying levels of danceability, but "Animal" is moderately danceable. "Happier" have a moderate to **high danceability score** (0.695344). "Silence," "Animal," and "Roses" all exhibit high tempo values. "Happier" has a **tempo** value of 0.423528.
“Happier” fall into **electronic, pop, or dance-pop categories**, similar to "Silence," "Animal," or "Roses."

```
[209]: recommended_song2 = recommend_songs_based_on_three('Khalibali',
                                                          'Main Aai Hoon Up Bihar Lootne',
                                                          'Tinku Jiya', data, similarity_functions,
                                                          numerical_weights)
       print(recommended_song2)
```

	Name	Average Similarity
11281	Bam Lahiri	0.827027

All three songs exhibit **high energy levels**, suggesting "Bam Lahiri" also have high energy, (0.762995). "Tinku Jiya" and "Main Aai Hoon Up Bihar Lootne" show **high danceability**, while "Khalibali" is moderate. "Bam Lahiri" have high danceability (0.835020). All songs have **moderate popularity scores**. "Bam Lahiri" could have a popularity score of 0.589474. "Bam Lahiri" fall into the **Bollywood or Indian Pop genre**, similar to "Khalibali," "Tinku Jiya," and "Main Aai Hoon Up Bihar Lootne."

Accuracy Vs Threshold Curve



Blockers

- Musixmatch API only gives a portion of the lyrics in the free edition, and there are certain limits on scraping data from Kaggle using BeautifulSoup, thus lyrics data is scraped locally.
- There were numerous constraints when scrapping songs from Genius using the API and even Selenium so had to switch to other methods..
- We also examined and processed the audio data initially, however the majority of the audio features that we were able to infer were provided by the Spotify API. We did not appear to find it very useful, so we removed it.
- Numpy array (embeddings) getting imported as string when imported from csv file, which in turn gave very much problem in using operations on embeddings and numerical features.
- Multiple occurrences of the same song but only difference was that they were from different Albums or were published by multiple artists in their albums.

Neglected Approaches

We were also able to get the song audio and generated their mel spectrograms for audio analysis but later felt that features like tempo, instrumentalness, acousticness, etc which we got from the Spotify API represented the audio analysis quite well.

Conclusion And Future Prospects

- Create a model to train the weights to eliminate manual weight changes, so that it can offer us the best similarity scores.
- To incorporate data about movie names and stars in the song, which might also be a significant element to compute similarity between songs.
- To incorporate a collaborative filtering model with content-based similarity scores to enhance recommendations, leveraging both user preferences and song features for personalised suggestions.