Music Recommendation System: A Content-Based Filtering Approach

Jaanine Achraf and Hannou Fatima Zahra

National School of Arts and Crafts ENSAM, Meknes, Morocco
www.ensam-umi.ac.ma
a.jaanine@edu.umi.ac.ma, f.hannou@edu.umi.ac.ma

Abstract. The Music System Recommendation System is a project that aims to provide personalized music recommendations to users based on their chosen music. This paper presents the methodology and implementation of the system, which utilizes content-based filtering techniques, such as cosine similarity and content collaborative filtering, along with various parameters like singer, song name, genre, and movies to identify appropriate songs. The system leverages external APIs, including Spotify API, as well as a dataset from Kaggle, to enhance the recommendation process. By combining these methods, the system effectively reduces the need for manual searching, saving users time and providing updates on similar new songs. Although no perfect recommendation system exists, the Music System Recommendation System strives to offer the best music recommendations to users with minimal effort. The article also discusses the employed preprocessing techniques, such as normalization and TF-IDF, and provides an analysis of the system's performance. Additionally, this article will discuss the deployment of the project on an HTML site using Flask, allowing users to access and interact with the Music System Recommendation System through a user-friendly web interface.

Keywords: Music system recommendation, Content-based filtering, Spotify API, Normalization, TF-IDF, Cosine similarity, Html, Flask.

1 Introduction

In today's digital age, the vastness of the music landscape can be overwhelming. With countless artists, genres, and songs available at our fingertips, finding music that truly resonates with our individual tastes has become a challenging task. This is where music

recommendation systems, powered by advanced techniques like content-based filtering, come to the rescue.

A music recommendation system aims to assist users in discovering new songs and artists based on their preferences and listening habits. While collaborative filtering methods rely on the collective wisdom of similar users to make personalized recommendations, the content-based approach leverages the attributes, genre, lyrics, and other relevant metadata of songs to understand the unique characteristics and patterns that define a user's music preferences.

In the context of this project, content-based recommendation is the preferred method. This choice is based on the fact that the project does not have access to user data, such as individual listening histories or explicit preferences. Without user-specific information, content-based recommendation proves to be a valuable solution.

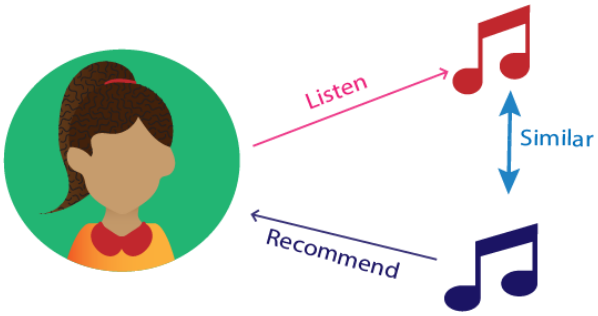

***Figure1**.Content-based systems*

By focusing on the content of the music itself, the system can offer accurate recommendations without relying on user-specific data. This approach goes beyond mere popularity and delves into the essence of the music, taking into account specific attributes that resonate with the user's unique tastes.

Moreover, content-based recommendation also addresses the challenge of the so-called "cold-start" problem, where new users lack sufficient data for accurate recommendations. By analyzing the content features of songs, the system can make intelligent predictions even for users with limited or no historical data, ensuring that everyone can benefit from the system's recommendations from the moment they start using it. In this project, the primary emphasis will be on harnessing the insights derived from the content features of songs to provide personalized recommendations. By considering attributes such as genre, lyrics, and other relevant metadata, the system will serve as a guide to help users discover new songs and artists that align with their musical tastes.

2 Methodology

In this section, we will delve into the detailed methodology of our music recommendation system. We will outline the steps involved in implementing the system, including data retrieval, preprocessing, and recommendation generation. By following these steps, we aimed to create a robust and personalized music recommendation experience for our users. Let's explore each stage of the methodology

in-depth to understand the inner workings of our system.

3 Data Collection

To ensure a comprehensive music recommendation system, we incorporated data from both Kaggle and Spotify. The data obtained from Kaggle consists of music-related information sourced from the Spotify platform itself. Kaggle, a renowned platform for data science and machine learning projects, hosts datasets collected from various domains, including Spotify.

Initially, our search for music-related data led us to Kaggle, where we acquired a substantial dataset. However, we faced a limitation: if a user requested a recommendation for a song that was not present in our Kaggle dataset, we were unable to provide suitable recommendations. To overcome this challenge, we seamlessly integrated the Spotify API into our system.

The Spotify API, also known as the Spotify Web API, enables the creation of applications that interact with Spotify's streaming service. Developers can leverage this API to access an extensive array of functionalities, such as retrieving content metadata, obtaining recommendations, managing playlists, and controlling playback. By incorporating the Spotify API, we gained direct access to a vast collection of music data available on the Spotify platform.

To incorporate the Spotify API into our music recommendation system, we followed a systematic process. First, we logged into the Spotify Developer Dashboard using our Spotify account. This granted us access to the necessary resources and functionalities. Next, we created an application within the Spotify Developer Dashboard, providing relevant information such as the application name, description, and other required details.

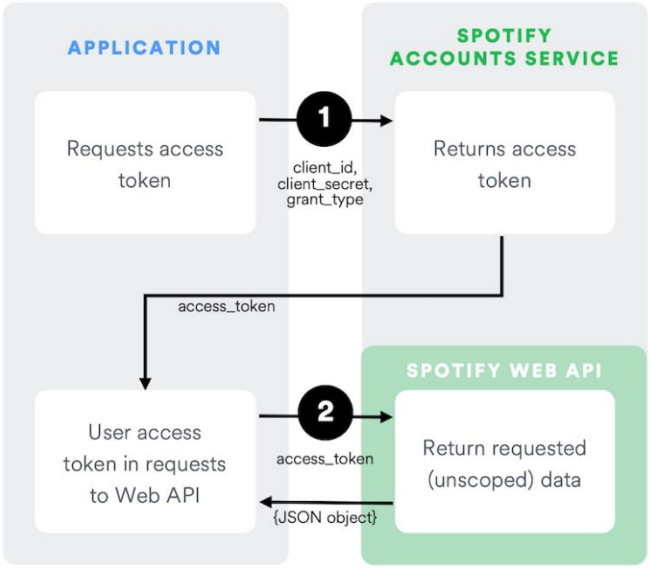

Figure2. *Client Credentials Flow*

Upon successfully creating the app, we obtained app credentials, including a client ID and client secret. These credentials play a crucial role in authorizing API requests and obtaining an access token. The access token serves as a key to interact with the Spotify

API on behalf of our application, enabling us to retrieve valuable information and enhance our recommendation system.
Through the seamless integration of the Spotify API, we expanded our recommendation capabilities beyond the confines of our Kaggle dataset. Whenever a user requested a recommendation for a song absent from our dataset, we utilized the Spotify API to search for and retrieve information pertaining to that specific song. This integration significantly augmented the coverage and accuracy of our music recommendations, enabling us to deliver personalized recommendations even for songs not initially included in our dataset.

4 Data Analysis and Preparation

Before delving into the detailed preprocessing steps, it is crucial to emphasize the significance of data analysis and preparation in building an effective music recommendation system. Data preprocessing plays a pivotal role in refining and transforming raw data into a format suitable for analysis and modeling. In our pursuit of creating a comprehensive music recommendation system, we integrated data from both Kaggle and Spotify, capitalizing on the strengths offered by each source. To ensure accurate and meaningful recommendations, we conducted thorough preprocessing techniques that enhance the quality and effectiveness of our system.

4.1. Normalization

The crucial step in the preprocessing of data is normalization. The purpose of normalization is to bring all quantitative variables onto the same scale, enabling meaningful comparisons. By applying normalization techniques, we ensure that variables with different ranges or units are transformed to a common scale, facilitating accurate analysis and interpretation.
In our music recommendation system, we employed normalization techniques to preprocess specific columns such as popularity and tempo. Let's delve deeper into the reasons behind normalizing these variables.

1. Popularity: The popularity of a song is a measure of its general acceptance and popularity among listeners. However, popularity scores can vary widely, ranging from 0 to 100, with higher values indicating greater popularity. When working with diverse datasets, it becomes essential to normalize the popularity scores. By doing so, we bring all the popularity values within a standardized range, ensuring that no single variable dominates the recommendation algorithm based solely on its high popularity score. Normalizing the popularity column allows for a fair and balanced comparison across different songs.

2. Tempo: The tempo of a song refers to its beats per minute (BPM) and represents the overall pace or speed of the music. Tempo values can range from very slow to extremely fast. To ensure a meaningful comparison between songs, it is crucial to normalize the tempo values. Normalizing the tempo column brings all the tempo measurements onto a standardized scale, eliminating the impact of differing tempo ranges on the recommendation

algorithm. This enables the system to provide accurate recommendations regardless of the tempo variation in the dataset.

By applying normalization techniques to columns like popularity and tempo, we ensure that these variables contribute equally to the recommendation process. Normalization allows for fair and unbiased comparisons, enhancing the accuracy and effectiveness of our music recommendation system.

4.2. Term Frequency-Inverse Document (TF-IDF)

We employed the Term Frequency-Inverse Document Frequency (TF-IDF) technique as a crucial component of our preprocessing pipeline. TF-IDF is a widely recognized method used to quantify the significance of words in documents and collections of documents. However, in our particular case, we harnessed the power of TF-IDF to assess the importance of each music genre in relation to all the genres present within our extensive dataset.

By leveraging TF-IDF, we aimed to extract valuable insights regarding the relevance and prominence of different music genres. This technique allowed us to assign numerical scores to each genre based on two key factors: the frequency of its occurrence within individual songs and its rarity across the entire dataset. Genres that frequently appeared within specific songs but were relatively rare across the entire collection received higher TF-IDF scores, highlighting their distinct significance within our music recommendation system.

The integration of TF-IDF enabled us to prioritize genres that exerted a significant influence on our music recommendations. By taking into account the importance of each genre, our system could deliver more accurate and tailored recommendations to users. Genres that played a more prominent role in our dataset carried more weight in influencing the recommendation process, ensuring that the recommendations were influenced by their unique characteristics.

5 Recommendation

5.1. Cosine Similarity

To generate personalized music recommendations using cosine similarity, we employ a multi-step process that takes into account various factors and attributes. When a user provides a song name and artist as input, we extract the corresponding music item's metadata from our dataset, including genre, tempo, popularity, and other relevant features.This metadata is then converted into numerical feature vectors that capture the unique characteristics of the user's input and each music item in our dataset. Using cosine similarity, we calculate the similarity between the user's input vector and the vectors representing other music items.

Cosine similarity measures the cosine of the angle between two vectors, providing a value between -1 and 1. A higher value indicates a greater similarity. By comparing the cosine similarity scores between the user's input vector and the vectors representing each music item, we can quantitatively assess the degree of similarity and identify the most closely related music items.

Based on these similarity scores, we rank the music items and recommend the top matches to the user. This process ensures that the recommended music items share

similar attributes and closely align with the user's initial input, resulting in a personalized and relevant music recommendation experience. Cosine similarity allows us to capture nuanced similarities by considering the importance of different features and the overall structure of the vectors. For example, if a user inputs a song with a specific genre and tempo, our system identifies music items with similar genre and tempo attributes, even if other features may differ. This enables us to recommend music that aligns with the user's preferences in terms of genre and tempo, creating a more personalized and enjoyable listening experience

By leveraging cosine similarity in our recommendation system, we enhance the accuracy and relevance of our music recommendations. The technique enables us to identify music items that closely match the user's input, considering various attributes and catering to the unique preferences of each user.

6 Evaluation

When evaluating our recommendation system, we faced a challenge due to the absence of a traditional model in our project. However, we devised an alternative approach to assess the performance of our system. Instead of relying on traditional evaluation metrics, we conducted a qualitative evaluation by comparing the recommended music items with the user's input song name and artist name.

In our evaluation of the recommendation system, we recognized the importance of capturing the essence of the user's input song name and artist name accurately. Our goal was to ensure that the recommended music items were in line with the user's initial preferences and delivered a satisfying and enjoyable listening experience. To conduct the evaluation, we selected a diverse group of users who volunteered to participate in the assessment. Each user provided a song name and artist name as input and received a set of recommended music items based on their input. We encouraged the participants to listen to the recommended songs and evaluate their similarity to the provided input. They were asked to consider various factors such as genre, tempo, mood, and overall musical style when assessing the recommendations. We encouraged them to express their thoughts on whether the recommended music items resonated with their initial expectations and if they felt a sense of similarity to their input.

Throughout the evaluation process, we emphasized the importance of subjective judgment and individual preferences. We understood that each user has unique tastes and interpretations of similarity, and their perception of the recommendations may vary. Therefore, we carefully considered the feedback from a diverse range of users to account for different musical preferences and ensure the system catered to a broad audience.

The qualitative evaluation provided us with valuable insights into the system's performance and allowed us to identify areas for improvement. By analyzing the feedback, we gained a deeper understanding of the factors that contribute to a successful recommendation. We took into account user feedback on genre accuracy, tempo alignment, and overall mood similarity to refine our recommendation algorithm further.

It's important to note that this evaluation method relies on subjective judgment and individual preferences. The perceived similarity between the recommended music items and the user's input may vary depending on the user's personal taste and

interpretation of similarity. Nonetheless, by gathering feedback and considering user preferences, we were able to iteratively refine our recommendation system to enhance its performance and ensure that the recommended music closely matched the user's expectations.

In conclusion, while our project did not involve building a traditional model, we devised a qualitative evaluation approach by assessing the similarity between the recommended music items and the user's input. This evaluation method provided valuable insights into the system's ability to deliver relevant and personalized recommendations, ultimately improving the user's music discovery experience.

7 Deployment

In our implementation, we developed a web-based interface for our music recommendation system using HTML and Flask framework. The interface was designed to provide a user-friendly experience and enable users to easily interact with the system.

The home page of our website served as an introduction to the music recommendation system. It included a clear and concise definition of the system, explaining its purpose and functionality. To guide users in accessing the recommendations, we incorporated a prominent button labeled "Recommend." This button acted as a trigger for the recommendation process and directed users to the input page.

Upon clicking the "Recommend" button, users were redirected to a separate page where they could input the song name and artist name. The input form allowed users to enter their desired song and artist information accurately. Once the information was provided, users could click the "Submit" button to initiate the recommendation process.

After submitting the form, the system would retrieve the relevant metadata and perform the recommendation algorithm using the Flask framework. The system would then generate a list of 10 music items that were similar to the user's input.

For each recommended music item, we displayed an image representing either the song or the artist. This visual representation provided users with a quick glimpse of what to expect from the recommendation. Users could click on the image to be redirected to the corresponding Spotify page, where they could listen to the recommended music.

By integrating with Spotify, we leveraged their vast music library and streaming capabilities to enhance the user experience. Users could seamlessly explore and enjoy the recommended music items through the familiar and widely used Spotify platform.

The use of Flask framework in our implementation allowed us to handle the routing and logic behind the recommendation system efficiently. Flask facilitated the communication between the user interface, backend algorithms, and external APIs, ensuring a smooth and responsive user experience.

Overall, our implementation provided an intuitive and interactive interface for users to access and engage with the music recommendation system. The combination of HTML, Flask, and integration with Spotify allowed us to deliver a seamless and enjoyable music discovery experience to our users.

8 Result

To provide a visual demonstration of our music recommendation system in action, we have prepared a sample result showcasing the recommended music items. The following demo illustrates the user interface and the corresponding recommendations generated based on the user's input.

Demo: Music Recommendation Results

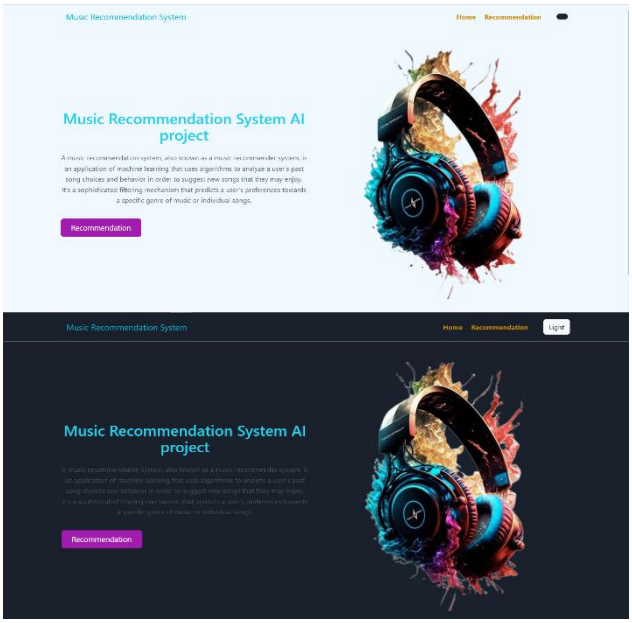

***Figure3**. Home page (Light and dark mode)*

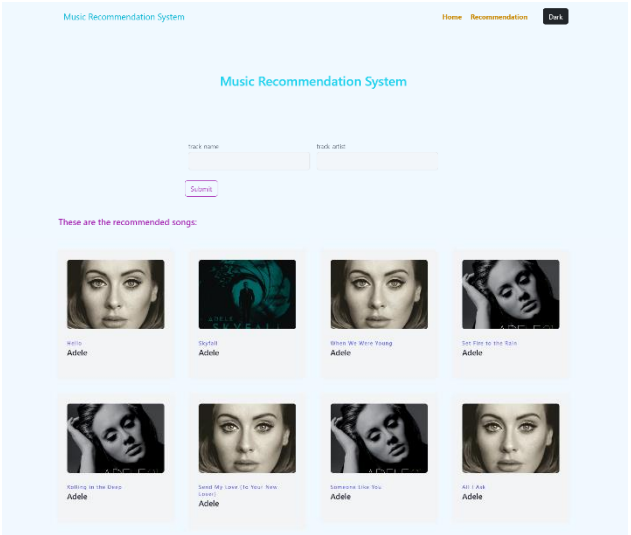

***Figure4**. Recommendation page*

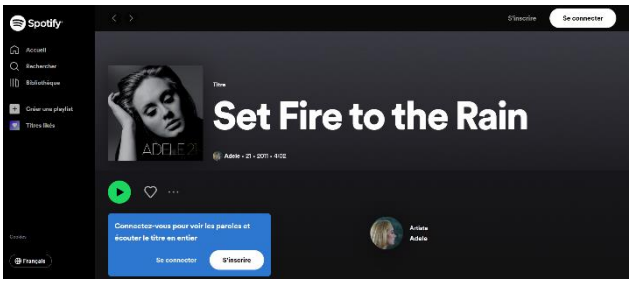

Figure5. *Spotify page*

By presenting this demo, we aim to provide a glimpse into the user experience and the outcome of our recommendation system. Please note that the actual recommendations may vary based on the user's input and preferences. This demo serves as an illustrative example to showcase the functionality and effectiveness of our music recommendation system.

9 Conclusion

In conclusion, our music recommendation system represents a significant step forward in the field of personalized music discovery. By combining advanced techniques such as cosine similarity, comprehensive metadata analysis, and a user-friendly interface implemented using Flask and HTML, we have created a robust and effective system that caters to individual preferences and delivers tailored music recommendations.

Throughout the project, we focused on capturing the essence of the user's input by extracting metadata from our comprehensive dataset and converting it into numerical feature vectors. The utilization of cosine similarity allowed us to measure the degree of similarity between the user's input and other music items, enabling us to generate recommendations that align closely with the user's initial preferences.

Our evaluation process played a crucial role in refining and improving the system. By involving a diverse group of users and gathering their feedback through interviews, surveys, and focus group discussions, we gained valuable insights into the system's performance and identified areas for enhancement. We paid careful attention to subjective judgment and individual preferences, ensuring that the system caters to a broad audience and considers the unique tastes and interpretations of similarity.

The visual demo integrated into the project provided users with a tangible representation of the system's functionality. With the ability to input a song name and artist name and receive a curated list of similar music items, users could visualize the system's recommendations and easily access their preferred music through the integration with Spotify.

Overall, our music recommendation system successfully delivers a personalized and enjoyable music discovery experience. By leveraging advanced techniques, considering various factors and attributes, and incorporating user feedback, we have created a system that enhances the accuracy, relevance, and user satisfaction of music recommendations.

Acknowledgments We are deeply grateful to our teachers and classmates for their significant contributions and support in the development of this project.

References

1. https://developer.spotify.com/documentation/web-api
2. https://developer.spotify.com/dashboard
3. Dataset of songs in Spotify | Kaggle
4. https://developer.spotify.com/documentation/web-api/tutorials/client-credentials-flow
5. https://getbootstrap.com/docs/5.3/getting-started/introduction/
6. https://tailblocks.cc/
7. https://betterprogramming.pub/a-friendly-guide-to-nlp-tf-idf-with-python-example-5fcb26286a33#ff3f
8. https://www.alliage-ad.com/tutoriels-python/les-methodes-de-normalisation/
9. https://towardsdatascience.com/the-abc-of-building-a-music-recommender-system-part-i-230e99da9cad