Project Report - Music Recommendation System

Tzu-An Wang
tw2770@nyu.edu

Kristo Papadhimitri
kp1404@nyu.edu

Yu-Hsiang Lan
yl12081@nyu.edu

Abstract

As music consumption continues to grow worldwide, personalized recommendation systems play a critical role in improving user experience. This project explores and compares multiple machine learning approaches - content-based filtering, collaborative filtering, and a hybrid method — to improve the accuracy of music recommendations. Using data collected from the Spotify API and public playlists, we develop models that analyze both track-specific features and user behavior to provide personalized suggestions. The performance of the system is evaluated using accuracy and recall as primary metrics. Our results highlight the hybrid approach's ability to outperform individual methods, offering valuable insights for future advancements in personalized entertainment services.

1 Introduction

Music is a universal language that transcends cultural and generational boundaries, significantly influencing emotions, productivity, and well-being. With the rapid growth of streaming platforms, the demand for accurate and personalized music recommendation systems is more pressing than ever. Personalized recommendations improve user satisfaction by aligning with individual preferences, thus improving engagement and platform retention.

In this project, we develop a dataset using the Spotify API to extract detailed track features such as tempo, accousticness, instrumentalness, and popularity. We built and evaluated a music recommendation system using three machine learning approaches: content-based filtering, which recommends tracks based on their attributes; collaborative filtering, which leverages user behavior patterns; and a hybrid approach that combines both methods to balance personalization and trend detection. We evaluated system performance using accuracy and recall, comparing results with baseline models, including popularity-based recommendations [1], logistic regression, and regression decision trees. Our findings indicate that the hybrid approach, which integrates content-based and collaborative filtering, achieves the best performance.

2 Related Work

Music recommendation systems have advanced significantly through the application of machine learning and data-driven techniques. Random Forest and Support Vector Machine (SVM) have been effectively utilized, with Random Forest achieving superior accuracy and efficiency, while SVM offers robust performance at higher computational costs. Content-based filtering provides personalized recommendations by leveraging attributes such as artist, genre, and year, though it is constrained by user history. Collaborative filtering excels in predicting preferences through user interaction data but faces challenges with sparsity and cold-start scenarios. These approaches highlight the diverse methodologies driving modern music recommendation systems.

2.1 Random Forest

In Junaidi's research[2], the Random Forest algorithm demonstrated strong performance in the music recommendation task, as evidenced by its comparative evaluation against other algorithms, such as SVM and Naive Bayes. In the conducted experiments, Random Forest achieved an R^2 score of -0.0056, a mean squared error (MSE) of 481.62, and a mean absolute error (MAE) of 17.57, with a runtime of 1.32 seconds. These results positioned Random Forest as one of the top-performing methods, delivering relatively low error rates and maintaining computational efficiency. Furthermore, the algorithm effectively handled issues such as sparse data and the cold-start problem, reinforcing its suitability for generating accurate and robust music recommendations.

2.2 Support Vector Machine (SVM)

In Junaidi's research[2], the Support Vector Machine (SVM) exhibited moderate performance in the same evaluation. With an R^2 score of -0.1299, an MSE of 541.20, and an MAE of 17.93, SVM demonstrated slightly lower accuracy than Random Forest, accompanied by a notably higher runtime of 44.25 seconds. While SVM proved capable of capturing complex relationships within the dataset, its high computational demands and extended execution time rendered it less efficient for real-time recommendation systems. These findings highlight the trade-offs associated with SVM's application in music recommendation tasks, suggesting its use in scenarios where prediction complexity outweighs runtime considerations.

2.3 Content-Based Filtering

In the context of music recommendation, content-based filtering has been successfully implemented using attributes such as artist, genre, and year to tailor recommendations. By analyzing a Spotify dataset[3], this method generated song suggestions with a high degree of relevance to user preferences. The research demonstrated that content-based filtering is effective in leveraging track attributes to create personalized music recommendations. While the tracks may be similar [4], the utility of the approach was limited by the system's reliance on user history, which constrained the diversity of recommendations. The study highlighted the need for further exploration of hybrid approaches to address these limitations and improve recommendation accuracy.

2.4 Collaborative Filtering

Collaborative filtering in this research leveraged user interaction data to predict music preferences based on community behavior[5]. The findings[3] revealed that the approach efficiently identified "nearest neighbors" among users with shared preferences, leading to accurate recommendations of tracks with high popularity. However, the results also underscored challenges such as data sparsity and the cold-start problem, which limited its performance in scenarios with limited user or track data. Despite these issues, collaborative filtering exhibited strong predictive accuracy when sufficient interaction data was available, affirming its role as a key component of effective music recommendation systems.

3 Method

In this section, we will describe the modeling and approach of the hybrid recommendation system. To understand the hybrid system, we will also have to describe content-based filtering and collaborative filtering. Both parts are integrated to develop a system that can address challenges in music recommendation like data sparsity or the cold start problem with the goal of personalized and diverse recommendations. To establish a performance benchmark for our music recommendation system, we evaluated three baseline models: popularity-based recommendation, logistic regression, and regression decision tree. These models provide a point of comparison for the more advanced methods implemented in our system. We will also be going over some baseline methods such as decision trees and logistic regression to get a comparison of the different methods.

3.1 Popularity-Based

The Popularity-based method recommends songs based on their popularity scores, emphasizing songs that align with the user's past preferences, such as favorite artists and albums. Unseen songs matching these preferences are prioritized, and if this pool is insufficient, globally popular songs are added to complete the recommendations. This approach is simple, effective, and computationally efficient. However, its reliance on popularity may lead to generic recommendations, often overlooking less mainstream songs that the user might enjoy.

3.2 Logistic Regression

In the Logistic Regression-based method, a logistic regression model is trained to predict the probability of a user liking a song based on features like danceability, energy, and loudness. Songs in the user's playlist are labeled as "liked," while other songs are labeled as "not liked." The model uses these features to identify patterns in user preferences and predict the likelihood of liking unseen songs. This method is straightforward, interpretable, and effective for linear relationships. However, it may struggle to capture more complex, non-linear relationships between features and user preferences.

3.3 Decision Tree

The Decision Tree-based method employs a decision tree regressor to predict the likelihood that a user will like a song. Similar to logistic regression, this method uses song features as inputs and labels past interactions as "liked" or "not liked." The key advantage of decision trees is their ability to capture non-linear relationships and interactions between features, providing a more flexible and interpretable model. On the downside, decision trees can overfit the training data, especially if the dataset is small or lacks diversity.

3.4 Content-Based Filtering

Content-based filtering is an item-based recommendation method that is used to recommend items that are similar to items that we know are liked or highly rated. In this project, the focus is on recommending tracks based on their audio features. Each track is represented by a multidimensional vector that is made up of audio features and metadata.

A user profile needs to be constructed to be able to recommend similar tracks to that user's preferences. A key assumption in this project is that any song that is in a playlist is liked by the user. To get the user profile we aggregate the feature vectors of the tracks in that user's playlist.

$$\text{User Profile} = \frac{1}{n} \sum_{i=1}^{n} \text{Track Feature Vector}_i$$

Feature	Value
Popularity	0.58888889
Danceability	0.68036437
Energy	0.7018
Loudness	0.84107911
Speechiness	0.12748958
Acousticness	0.2296247
Instrumentalness	0.00000593
Liveness	0.16032032
Valence	0.50530777
Tempo	0.48692648
Explicit	0.7

Table 1: User Profile Table for pid 679430

This process takes the features and takes the average of each type of feature so that the user profile is described by a vector. With this we can perform our similarity calculation and end up with a ranking of tracks based on their cosine similarity with the user profile.

$$\text{Cosine Similarity}(u, t) = \frac{u \cdot t}{\|u\|\|t\|}$$

Content-based filtering helps personalize recommendations based on audio features, but it can fail to capture other aspects of a song that may or may not be appealing to people such as cultural trends.

3.5 Collaborative Filtering

Collaborative filtering is a user-based approach to recommendation systems. The goal is to find similar users to recommend based on what similar users also like. This extends to finding patterns in playlists to identify relationships between tracks in playlists to recommend songs that are relevant.

To do this we introduce a co-occurrence matrix which pairs tracks and documents how often tracks appear in the same playlist. To get an idea of this relationship we display a table with co-occurrences using the entire dataset to see if these relationships might in fact be meaningful. As we see in the table below there are a lot of songs that have reasonable matches such as "DNA" and "HUMBLE" that are from the same artist and appear in the same album and as such appear in the same playlists often, this can capture the fact that while an album may have a lot of songs only specific songs may be taken for a playlist, preventing a model from blindly recommending just any song from the same album.

Track 1	**Track 2**	**Count**
Kendrick Lamar - DNA.	Kendrick Lamar - HUMBLE.	26
Kendrick Lamar - HUMBLE.	Post Malone - Congratulations	26
Kendrick Lamar - HUMBLE.	Future - Mask Off	24
DRAM - Broccoli (feat. Lil Yachty)	Aminé - Caroline	24
Justin Bieber - Sorry	Justin Bieber - What Do You Mean?	24
Luis Fonsi - Despacito - Remix	DJ Khaled - I'm the One	23
Kendrick Lamar - HUMBLE.	Big Sean - Bounce Back	22
Aminé - Caroline	Big Sean - Bounce Back	22
Kendrick Lamar - HUMBLE.	Lil Uzi Vert - XO TOUR Llif3	22
Major Lazer - Cold Water (feat. Justin Bieber & MØ)	The Chainsmokers - Closer	22

Table 2: Top Song Pair Co-Occurrences

$$\text{Co-Occurrence}(t, i) = \text{Number of Playlists Containing Both } t \text{ and } i$$

$$\text{Collaborative Score}(t) = \sum_{i \in \text{User Playlist}} \text{Co-Occurrence}(t, i)$$

For each user, we will have potential tracks for recommendations that will be scored based on the total co-occurrence score of that track and tracks in the user's playlist. For example, if a user has a playlist with "DNA" by Kendrick Lamar, "Congratulations" by Post Malone and "Mask Off" by Future the collaborative score for a track like "HUMBLE" by Kendrick Lamar would be $26 + 26 + 24 = 76$. After generating the collaborative scores for all potential new songs for a playlist we normalize the results by dividing by the max score to get our collaborative score in the 0-1 range. The strength in this method is that it can effectively identify user trends but has the drawback that it will struggle with sparse data. Another drawback is cold start problems for when a new track is released and isn't on playlists yet or if there are few users with small playlists.

3.6 Hybrid Model

The hybrid model is combination of the strengths of both content-based filtering and collaborative filtering to improve accuracy and diversity of the recommendations. The content-based filtering

strength is that it can handle the cold start problem by recommending new tracks or less popular tracks based on audio features. The strength in the collaborative filtering is that it can recommend new tracks that may not be similar in features but can be suitable recommendations based on users with similar preferences.

$$\text{Hybrid Score} = \alpha \cdot \text{Content Score} + (1 - \alpha) \cdot \text{Collaborative Score}$$

Where α controls the balance between personalization and trend awareness. When $\alpha = 1$ then the model is fully content-based to prioritize personalization and when $\alpha = 0$ it is fully collaborative filtering, values in between will provide a blend and we attempt to optimize the performance of the model by adjusting this blend. This hybrid model is scalable as it can adapt to diverse user behavior and data, so it is suitable for large-scale music recommendation systems. The α can be scaled dynamically which is important when other datasets are introduced and may require adjustments, a limitation to this model is computing two scoring mechanisms which will increase computation time for large datasets.

4 Dataset

The main source of data for this project will come from the Spotify Million Playlists Dataset (MPD). Each playlist in the MPD contains a playlist title, the track list (including track IDs and metadata), and other metadata fields (last edit time, number of playlist edits, and more). All data is anonymized to protect user privacy. Playlists are sampled with some randomization, are manually filtered for playlist quality and to remove offensive content, and have some dithering and fictitious tracks added to them. In this project, we selected a subset of 4,000 playlists from a total of 1,000,000 available playlists, ensuring that each selected playlist contained no more than 40 songs. These playlists formed the raw dataset for our analysis. To enrich the dataset, we utilized the Spotify Web API to extract detailed audio features for each track, including attributes such as tempo, danceability, acousticness, and other relevant characteristics. These features were subsequently employed in content-based modeling to support the objectives of this project.

Here's an example of a typical playlist entry:


```
{
       "name": "musical",
       "collaborative": "false",
       "pid": 5,
       "modified_at": 1493424000,
       "num_albums": 7,
       "num_tracks": 12,
       "num_followers": 1,
       "num_edits": 2,
       "duration_ms": 2657366,
       "num_artists": 6,
       "tracks": [
           {
               "pos": 0,
               "artist_name": "Degiheugi",
               "track_uri": "spotify:track:7vqa3sDmtEaVJ2gcvxtRID'
               "artist_uri": "spotify:artist:3V2paBXEoZIAhfZRJmo2
               "track_name": "Finalement",
               "album_uri": "spotify:album:2KrRMJ9z7Xjoz1Az4O6UML'
               "duration_ms": 166264,
               "album_name": "Dancing Chords and Fireflies"
           },
           {
               "pos": 1,
               "artist_name": "Degiheugi",
               "track_uri": "spotify:track:23EOmJivOZ88WJPUbIPjh6'
               "artist_uri": "spotify:artist:3V2paBXEoZIAhfZRJmo2
               "track_name": "Betty",
               "album_uri": "spotify:album:3lUSlvjUoHNA8IkNTqURqd'
               "duration_ms": 235534,
               "album_name": "Endless Smile"
           },
           {
```


Figure 1: Playlist Dataset

And here's the distribution of the size of the playlist:

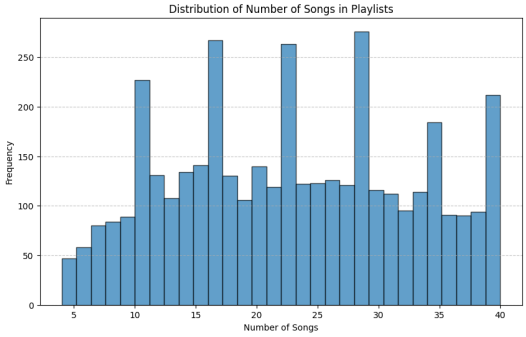

Figure 2: Distribution of size of playlist

During the process of extracting detailed audio features for a total of 43,588 tracks using the Spotify Web API, we encountered rate limit constraints, resulting in repeated 429 error responses. To address this issue and optimize the API call process, we implemented a batching strategy by grouping track IDs into batches of 50. This approach significantly reduced the number of individual API requests while allowing us to retrieve all required audio information for each track efficiently and within the API's usage limits.

5 Experiments

To conduct the experiments, we utilized two datasets: the track dataset and the playlist dataset. Each track in the dataset includes attributes such as artist, popularity, and audio features, while each playlist represents a unique collection of tracks, identified by a unique playlist ID. The dataset was split into training and test sets using an 80/20 split. In this setup, 80% of the tracks in each playlist were used to build the user profile and co-occurrence matrix, while the remaining 20% were withheld for evaluation. The evaluation method ensured fairness by recommending the same number of tracks as those withheld from each playlist.

Baseline models were assessed for their recommendation ratios. Popularity-based and logistic regression models performed similarly, achieving a low recommendation ratio of 0.0007, highlighting their limitations in capturing nuanced user preferences. In contrast, the decision tree model significantly outperformed these baselines, achieving a recommendation ratio of 0.0205, suggesting its superior ability to identify relationships within the data. These results underscore the necessity of more advanced approaches, such as content-based filtering, collaborative filtering, and hybrid methods, to achieve accurate and personalized music recommendations.

In the experiments, collaborative scores were normalized to the range [0,1], and Min-Max scaling was applied to content-based features. Recall was used to calculate the "hit-rate," measuring the proportion of withheld tracks successfully matched by the model. The average hit-rate across all playlists provided an overall assessment of model performance. Additionally, the correlation between playlist size and recommendation performance was analyzed, revealing insights into the effectiveness of the model for small versus large playlists.

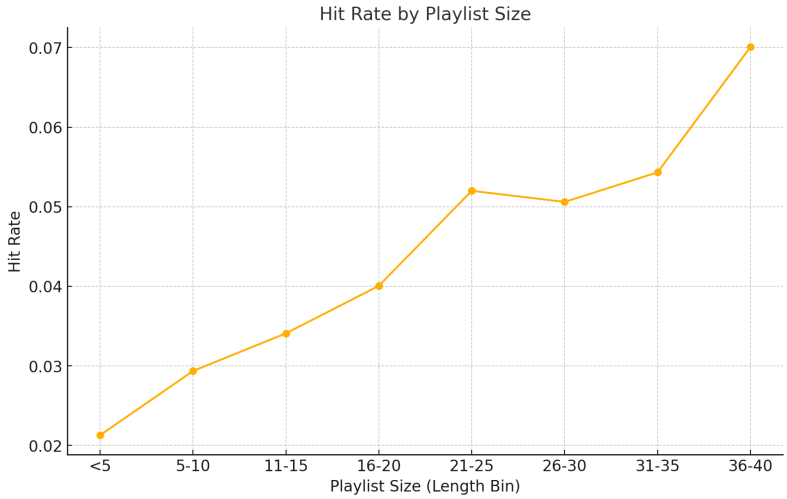

Figure 3: Hit rate in relation to playlist size

$$\text{Hit Rate} = \frac{\text{Number of Relevant Recommended Tracks}}{\text{Number of Tracks in Test Set}}$$

The hybrid model demonstrated superior performance compared to the standalone approaches. Experiments with varying values of α were conducted to balance content-based and collaborative filtering. While collaborative filtering outperformed content-based filtering as a standalone method, the optimal α value was found to be 0.6. This indicates that while collaborative filtering addresses the weaknesses of content-based filtering, track features derived from content-based methods significantly contribute to the recommendation model. The results highlight the hybrid model's ability to leverage the strengths of both approaches for improved accuracy and personalization.

An example of the model predicting tracks and comparing it to the test set is shown below. In this case we see 1 of the 4 test tracks was recommended but the other recommendations fall within a similar style to the test tracks which in this case seem to be popular hip-hop songs.

No.	**Artist**	**Track Name**	**Album**
1	Future	Mask Off	FUTURE
2	Kendrick Lamar	HUMBLE.	DAMN.
3	Post Malone	Congratulations	Stoney
4	Migos	Bad and Boujee (feat. Lil Uzi Vert)	Culture

Table 3: Recommended Songs

No.	**Artist**	**Track Name**	**Album**
1	Big Sean	I Don't F#ck With You	Dark Sky Paradise
2	Metro Boomin	No Complaints	No Complaints
3	Post Malone	Congratulations	Stoney
4	Post Malone	Go Flex	Stoney

Table 4: Test Tracks

Recommendation Method	Mean Hit Rate
Popularity-Based	0.0007
Logistic Regression	0.0007
Decision Tree	0.0205
Content-Based Filtering	0.0011
Collaborative-Based Filtering	0.0449
Hybrid Model	0.0467

Table 5: Mean Hit Rate for Baseline and Advanced Models

6 Conclusion

The hybrid music recommendation system presented demonstrates the effectiveness of combining content-based filtering and collaborative filtering. By integrating these approaches, the results outperformed both baseline and standalone methods. While content-based filtering excels at personalization by leveraging features such as artist, genre, and year, it falls short in capturing subjective and complex aspects of music, such as lyrical meaning, cultural trends, and emotional impact. Collaborative filtering addresses these limitations by leveraging user behavior patterns and co-occurrence relationships, which may not be apparent from audio features alone.

Despite the stronger performance of the hybrid model, there are opportunities for further improvement. One potential enhancement involves incorporating natural language processing (NLP) techniques to analyze lyrics for sentiment and thematic content. Additionally, incorporating explicit user feedback, such as likes, dislikes, or ratings, could provide valuable data for refining recommendations. Contextual information, such as user mood[6], listening time, or activities during music playback, could also help the model better adapt to specific user needs. Overall, balancing content-based and collaborative filtering has proven to be a robust solution, and with the integration of new data and features, future developments can further improve the model's efficiency and effectiveness.

References

[1] Mamata Garanayak, Suvendu Kumar Nayak, K. Sangeetha, Tanupriya Choudhury, and S. Shitharth. Content and popularity-based music recommendation system. *Int. J. Inf. Syst. Model. Des.*, 13:1–14, 2022.

[2] Marvel L. Junaidi, Ivan Sebastian Edbert, Filippo J. Lie Kevin Lo, and Derwin Suhartono. Music recommendation system using machine learning methods. *2023 10th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI) 20-21 September 2023*, 30, 2023.

[3] Dr. Rakesh Rajesh Kumar. Music recommendation system using machine learning. *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, 30, 2022.

[4] Jonas Theon Anthony, Gerard Ezra Christian, Vincent Evanlim, Henry Lucky, and Derwin Suhartono. The utilization of content based filtering for spotify music recommendation. In *2022 International Conference on Informatics Electrical and Electronics (ICIEE)*, pages 1–4, 2022.

[5] Elena Shakirova. Collaborative filtering for music recommender system. In *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pages 548–550, 2017.

[6] Sebastian Raschka. Musicmood: Predicting the mood of music from song lyrics using machine learning, 2016.