Personalized Music Recommendation System based on Singer Style.

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
(Specialization in Artificial Intelligence & Machine Learning)

Submitted by

VU21CSEN0300097 – DASARI SAI SRIKANTH
VU21CSEN0300047 – D. SNEHA
VU21CSEN0300106 – N. SHRENIK
VU21CSEN0300102 – D. VARUN

Under the esteemed guidance of
Dr. MUKKAMALA SNV JITENDRA, Ph.D.
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)
VISAKHAPATNAM

2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)
VISAKHAPATNAM

DECLARATION

I hereby declare that the project report entitled **Personalized Music Recommendation System based on Singer Style** is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering (AI&ML). The work has not been submitted to any other college or University for the award of any degree or diploma.

Date:

Registration No(s)	Name(s)	Signature
VU21CSEN0300097	**DASARI SAI SRIKANTH**	
VU21CSEN0300047	**D. SNEHA**	
VU21CSEN0300106	**N. SHRENIK**	
VU21CSEN0300102	**D. VARUN**	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled **PERSONALIZED MUSIC RECOMMENDATION SYSTEM BASED ON SINGER STYLE** is a bonafide record of work carried out by VU21CSEN0300097 – D.SAI SRIKANTH, VU21CSEN0300047 – D. SNEHA, VU21CSEN0300106 – N. SHRENIK, VU21CSEN0300102 – D. VARUN students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering (AI&ML).

Date:

Project Guide	Head of the Department
Dr.Mukkamala SNV Jitendra Kumar, Ph.D.	**Dr.Kuppili Naveen**

ACKNOWLEDGEMENT

We would like to thank our guide, **Dr.Mukkamala SNV Jitendra**, for his ongoing guidance and support during the project. His wide knowledge, intelligent thoughts, and encouragement have been invaluable in guiding us through each aspect of the project. Although the project is still in progress, his patience and determination have given us the confidence and abilities we need to make consistent progress. We are appreciative for his willingness to offer his skills, and we look forward to completing this project with his important guidance.

We would like to thank our in-class reviewers, **Dr. Chandra Sekhar Potala**, Associate Professor, and **Dr. Srinivasa Rao**, for their crucial criticism and support during the duration of our research. Their intelligent ideas and constructive criticism really improved the quality of our work during the project.

We are also grateful to our HOD, **Dr. K. Naveen Kumar**, for his constant support and direction, which has motivated us to strive for excellence. His experience in AI and Machine Learning has given us a solid platform on which to construct our concept.

Furthermore, we thank our external reviewer, **Padala Anuradha**, for his professional views and comments, which have helped us refine our approach and methodology.

Their collective support has been invaluable in leading us through the many stages of our project, and we look forward to implementing their suggestions as we near completion.

TABLE OF CONTENTS

.

1.ABSTRACT

Recommendation systems play an integral role in improving user experience on various digital platforms, including music streaming service. These recommendation systems learn user preferences and then recommend the proper content to foster interaction and satisfaction. Traditional recommendation methods have been applied extensively on music platforms, including collaborative filtering and content-based filtering. With that said, models for traditional recommendation systems all have certain limitations, such as the cold-start problem, sparsity, and the inability for the model to adapt dynamically to changes in user preference. To address these limitations, this paper explores a new music recommendation system, which is the reinforcement learning with Deep Q-Networks (DQN)-based recommendation system, which models music recommendation as a Markov Decision Process (MDP), where the decision-making agent interacts with the environment by selecting songs (recommended to the agent), and receiving a reward or punishment for that action. Each song suggestion incurred a potential reward or punishment, maximizing the agent's expected cumulative reward over time. The model is learned by iteratively adjusting its behavior so that the song recommendations reflect ongoing changes in user behavior, based upon the collected data. The proposed working system will be trained on a data set aggregated from multiple users, consisting of song metadata, listening history and preference to music genre. The reinforcement learning agent learns to optimize song recommendations by maximizing expectations of cumulative reward, thereby develop more personally tailored and interactive recommendations The empirical results validated that the model outperformed traditional methods indicating a recommendation accuracy of 98.33%. Accordingly, the resulting data supports that reinforcement learning is a feasible way to improve music recommendation systems, generating many possibilities for other adaptive and intelligent recommendation models for use in online music platforms.

2.INTRODUCTION

Today, the world has become an increasingly connected place and music has embedded itself into the fabric of daily life. With music being available more conveniently and visibly, and more importantly, with countless exposure to music more than ever, numerous audiences are switching to online consumption of musical entertainment, relaxation, and even productivity. A recent report highlights that there are millions of users around the world, if not billions, using online music platforms like Spotify, Apple Music, and YouTube Music, to explore a diverse and endless stream of songs, playlists, and collections. The ability to listen with ease while subscription pricing is relatively low, has made these platforms an almost permanent companion for many users. Alongside the increase in subscribers seeking out streaming music services, there has become a strong demand in delivering intelligent recommendation engines. With such an enormous and infinite catalogue of songs, trying to manually identify some potentially favorite tracks could be exhausting and overwhelming. The recommendation systems are helpful as a means to curate music consumption for individuals and deliver customized playlists, or song recommendations to provide the best experience. The recommendation systems typically evaluate and observe listening habits, historical listening choices, and demographics of the user in order to provide observers with content that matched a listener's musical preferences, and thus increases engagement and satisfaction of the user. Music recommendation systems have certainly evolved and matured over time, deliberating upon many approaches to provide both user and accurate and precise recommendations. Traditional models of Music recommendation systems are:

Collaborative Filtering: Here the recommendation for songs is based on the songs that the user with similar tastes listens to. For example, if two users have preferences for similar songs, the system recommends songs liked by one user to the other user. Collaboration filtering, however, has been plagued with the cold-start problem, in which new users do not receive recommendations because there is very little historical data to build the recommendations upon.

Content-Based Filtering: This algorithm checks for the attributes of songs, such as tempo, genre, artist, and instrumental elements, to recommend the tracks that share similar characteristics. While this can be helpful, content-based filtering will result in repetitive recommendations, as preferences tend to bias toward songs very similar to previously chosen titles.

Hybrid models: To improve recommendation accuracy, these models merge content-based and collaborative filtering. Hybrid models combine multiple methods to mitigate some of the disadvantages of a single model. Traditional recommendation algorithms are valuable, but they struggle to adapt to changing consumer habits and preferences. Since musical preferences change over time, static models are often unable to generate balanced and engaging suggestions. Users could also become frustrated with the algorithm, particularly if they fail to present new music opportunities.

This article presents a reinforcement learning-based method that uses Deep Q-Networks (DQN) to resolve the issues outlined in this section. Reinforcement learning, unlike traditional methods, will continuously be updated according to user interactions.

The proposed method will also treat music recommendation as a sequential decision-making problem where an RL agent is tasked with selecting songs according to user preferences and receives information about its selected action via some reward function. Optimization will take place over time as the agent adapts its policy to maximize all future rewards, which would lead to relevant and engaging recommendations.

The use of reinforcement learning allows the model to be responsive to changes in user preferences while improving the variety of recommended options it makes available. While collaborative filtering relies on previous data (and may have a hard time recommending songs users have not previously listened to, and content-based filtering relies on similarities in the features of songs, reinforcement learning offers a more flexible and adaptable approach. This is the system guarantees that users continue to experience a range of songs, while representing the user's core preference at the same time, resulting in a more pleasurable and fulfilling listening experience.

.

3. LITERATURE REVIEW:

Music recommendations in recent years have attracted considerable attention because of the rapid expansion of digital music libraries. Different approaches have been proposed to increase recommended accuracy, from traditional partner filtration techniques to advanced deep learning models. This section provides observation of larger research contributions to this domain. A hybrid music recommendation system that integrates collaborative filtration with a music -based approach was proposed to address boundaries such as computer sparsity and cold start problems [1]. The study says that the music generations, such as melody, rhythm, and style, improve the user's preference to match matching. Although specific accuracy was not given matrix, the results indicate the general improvement in the recommended quality. Another study introduced an automated music recommendation system that uses user -based associated filtration, element -based associated filtration and popularity -based ranking to increase privatization. Experimental results showed that user -based college filtration achieved the highest accuracy by 92% accuracy and 88% recall, demonstrated its effectiveness on other methods.[2]A recommendation system specifically designed for postpartum mothers was created using Support Vector Machine (SVM) classification. The research utilized One-vs-One (OVO) and One-vs-Rest (OVR) classification methods to sort music based on its emotional and therapeutic benefits. The SVM-OVR model with a polynomial kernel yielded the best results, achieving 80% accuracy, 82% precision, and 80% recall, demonstrating that genre classification can significantly improve user satisfaction. Additionally, deep learning methods have been investigated for music recommendations [3]. A study that used Bidirectional Long Short-Term Memory (Bi-LSTM) networks found that incorporating Mel-Frequency Cepstral Coefficients (MFCCs) as input features boosts recommendation accuracy. When compared to the earlier methods, the Bi-LSTM model's observed F1-score was 3% higher, indicating that sequential deep learning models may have applications in music recommendation [4]. A database of 431 user profiles was used in a study comparing Random Forest with Multi-Layer Perceptron (MLP) classifiers for predicting music genres. The results revealed that Random Forest performed better than MLP, with an accuracy of 95.47% compared to 53.07% for MLP. This study showed how effective ensemble learning algorithms are at recommending music depending on genre [5]. Convolutional Recurrent Neural Network Architecture (CRNNA) was assessed for music genre categorization and recommendation in another deep learning-based contribution. The authors utilized Mel spectrograms as input features and achieved very good results with a high accuracy in the genres of hip-hop (90.5%) as well as jazz (84%), and precision = 71.5% and recall = 78.1%. The paper suggests to use the hybrid CNN-RNN design consisting of both architecture for music recommendation because the hybrid architecture captures both the temporal and spectral dimensions of the data stream in music content. Another article presents a review of music recommendation approaches, including content-based filtering, collaborative filtering, and deep learning models. The paper discusses challenges in music recommendation, including data sparsity and cold start problems. This paper reviews studies that look at sentiment, contextual, and hybrid filtering. The discussion continues with studies regarding hybrid models, sentiment-based recommendations, and context-aware filtering. The suggested system successfully integrates machine learning approaches with collaborative filtering to boost personalization. Evaluation results showed high similarity scores with a peak value of 0.9375, implying an effective recommendation performance [7].

Author(s)	Year	Indexed	Dataset	Features	Methodology/Model	Results	Remarks (Drawbacks)
Mochammad Rizqul Fatichin et al.	2024	IEEE	GTZAN, Free Music Archive	Tempo, Spectral Features, MFCC	SVM (Support Vector Machine)	80% accuracy, 82% precision	Removing excessive features decreases classification accuracy
Saman Mesghali, Javad Askari	2022	IEEE	Million Songs Dataset	Artist style, searched words	Bi-LSTM (Bidirectional LSTM)	3% improvement in F1-score over previous models	Cold start problem for new users
Shenyou Fan, Min Fu	2022	IEEE	Questionnaire survey (431 users)	Age, region, hobbies, music preferences	MLP & Random Forest	RF: 95.47% accuracy, MLP: 53.07%	MLP has poor performance compared to RF
Jagendra Singh, Vijay Kumar Bohat	2021	IEEE	Music streaming metadata	Spectral & Temporal features	Convolutional Recurrent Neural Network (CRNN)	Better performance than baseline models	No mention of cold start solution
Joseph Bamidele Awotunde et al.	2024	IEEE	Public datasets (Spotify API)	Spectral audio features (MFCC, Chroma)	Machine Learning & Collaborative Filtering	High similarity scores for recommendations	Limited evaluation metrics
Mukkamala S.N.V. Jitendra, Y. Radhika	2020	IEEE	User listening history (custom dataset)	Listen count, user ID, song-artist	User-based Collaborative Filtering	Precision & recall improvements	Issues with diverse client needs
J. B. Awotunde, M. K. Abiodun, A. E. Adeniyi	2024	IEEE	Multiple datasets	Music genes, user preferences	Hybrid Collaborative Filtering	Improved recommendation scores	High computational complexity

4.PROBLEM IDENTIFICATION & OBJECTIVES

4.1 Problem Identification

Traditional music recommendation systems mainly focus on genres, tempo, and broad user behaviour while paying much less attention to the impact of the singer's different voice and style. This project proposes a plan for developing the recommendation system by emphasizing genre as the first factor for personalization, including unique insights into singers' voices. This system seeks to capture user preferences on a much deeper, genre-specific level that connects a listener to music exciting resonant interest by taste rather than a plethora of metadata.

With music streaming platforms sprawling fast and millions of songs added to the digital libraries every day, some of the issues are:

i. Trouble with suggesting songs that fit very specific genre preferences, as opposed to being dependent on rich metadata or sophisticated acoustic features.
ii. Lack of genre-based personalized recommendations that consider users' affinities for specific styles or singer nuances.

The existing music organization methods focus mainly on genre, mood, and metadata such as song popularity or release date. They fail to categorize music in ways that capture nuanced genre elements or specific singer styles. This gap makes it necessary to have a recommendation system that tailors suggestions by focusing on genre, capturing the essence of the user's listening patterns, and aligning them with artists and vocal styles that they resonate with.

4.2 Objectives

The main objectives of this project are summarized as below:

i. To develop a genre-centric music recommendation system where user preference for specific genres and styles is given priority for a highly personalized listening experience.
ii. Analyse listening history from users to give suggestions based on genre patterns and preferences, making the process of discovering music much more intuitive.
iii. Design an intuitive interface that will bring the consumption seamless and induce engagement in terms of genre affinity.

5.EXISTING SYSTEM

Advanced music genre classification models like yours face several issues related to efficiency and practicality. Traditional machine learning approaches such as Gaussian Mixture Models (GMMs) and Support Vector Machines (SVMs) depend on handcrafted features such as timbral texture and pitch content for distinguishing genres. This approach requires good domain knowledge and manual tuning while choosing the features. Furthermore, these features may not capture the complex and dynamic patterns found in music, which could limit the classification accuracy of the model. Automating feature extraction and using more sophisticated representations might overcome these limitations and possibly reveal deeper relationships between genres without requiring manual feature engineering.

Deep learning models, CNNs in particular, have revolutionized genre classification by allowing direct learning from audio spectrograms. However, CNNs are usually optimized for spatial data and are unlikely to catch the temporal elements essential in capturing the rhythm and beat of music. What is more, such models usually need huge amounts of labelled data and heavy computation capabilities that may cost time to train them.

This is not to say that improving the temporal analysis—some form of combinations of CNNs that process sequential data such as RNNs, among others—combined with data-requirement-reducing methods could greatly improve classification accuracy and even make the approach more efficient.

This has tried to combine both the frequencies and temporal dependencies with more comprehensive audio analysis in CRNNs, which combines the CNNs with RNNs. However, such systems are computationally demanding in nature, which hampers their use for any real-time applications or small devices. Your project would make genre classification much easier and more versatile for using on mobile devices if optimized using lightweight architectures, efficient methods of feature extraction, and alternative learning techniques for reducing the computational complexity. In summary, with the need for automatic feature extraction, improved temporal analysis, and computational efficiency addressed, this project is positioned to significantly improve the performance of existing systems. Improved performance could be delivered to a wider range of applications with more accurate genre classification, bringing sophisticated music analysis closer to real-time resource-limited scenarios

.

6.PROPOSED SYSTEM

The proposed music recommendation system is designed to deliver personalized music suggestions based on users' song history and listening preferences. By leveraging data from multiple users, the system builds a comprehensive dataset of songs, artists, genres, and popularity metrics. This data forms the foundation for a **reinforcement learning-based recommendation model**, where an **MDP structure with a Deep Q-Network (DQN) model** dynamically optimizes recommendations based on user interactions.

Unlike traditional collaborative and content-based filtering methods, this approach enables the system to continuously learn from user actions, adapting recommendations to reflect changing preferences. Users interact with the system via a **Stream lit-based web interface**, which provides an interactive dashboard for viewing and rating recommendations. Below is a breakdown of the system's components and workflow.

System Breakdown and Components

1. Data Collection and Ingestion

- **Data Collection:** The system gathers user song history from individual CSV files, containing details such as song name, artist, genre, and popularity.
- **Data Loading:** Individual CSV files are loaded and prepared for further processing.
- **Data Merging:** Data from multiple users is combined into a single, unified dataset, allowing the model to analyse broad listening trends and user preferences.

2. Data Preprocessing and Feature Engineering

- **Data Cleaning:** Handling missing values, removing duplicates, and ensuring data consistency.
- **Feature Engineering:** Extracting additional insights, such as frequently played artists, genre distribution, and session-based listening patterns.
- **State Representation:** Constructing meaningful state representations for the reinforcement learning model by encoding user interactions, song attributes, and historical preferences.

3. Reinforcement Learning Model (DQN with MDP Structure)

- **MDP Definition:** The recommendation process is framed as a Markov Decision Process (MDP), where:
  - **States** represent user listening history, preferences, and interaction patterns.
  - **Actions** correspond to recommending specific songs to the user.
  - **Rewards** are derived from user interactions, such as likes, skips, or repeated plays.
  - **Transitions** capture how user preferences evolve over time.
- **DQN Model:** The Deep Q-Network (DQN) learns an optimal policy by mapping states to the best song recommendations, optimizing for long-term user engagement.
- **Exploration vs. Exploitation:** The model balances exploring new recommendations and exploiting known user preferences for personalized suggestions.

4. Recommendation Generation

- **Reinforcement Learning-Based Suggestions:** The DQN model generates song recommendations by predicting actions (songs) that maximize expected rewards (user satisfaction).
- **Dynamic Adaptation:** The system continuously updates recommendations based on real-time user feedback, ensuring evolving and personalized suggestions.
- **Exploration Strategy:** Periodically introduces new, diverse songs to prevent recommendation stagnation.

5. Feedback Collection and Model Optimization

- **User Interaction-Based Feedback:** Likes, skips, and replay actions are captured to refine future recommendations.
- **Reward Function Adjustment:** The reward function dynamically updates based on engagement metrics to improve the recommendation policy.
- **Model Retraining:** The DQN model is periodically retrained using new data, ensuring better adaptability and improved personalization over time.

6. User Interface (UI) with Stream lit

- **Interactive Dashboard:** Built with Stream lit, the UI allows users to browse recommendations, filter songs by genre, and view listening trends.
- **Feedback Mechanism:** Users can rate recommendations in real-time, directly influencing future suggestions.
- **User Profiles & Insights:** Displays personalized stats, such as top genres, most played artists, and recommended discovery tracks.
- **Backend Connection:** The Stream lit app connects to the RL model, ensuring seamless interaction and dynamic updates based on user feedback.

7. System Monitoring and Maintenance

- **Performance Monitoring:** Tracks key metrics like recommendation accuracy, user retention, and engagement levels.
- **Continuous Model Updates:** The system fine-tunes the DQN model at regular intervals to incorporate new data and feedback.
- **Data Integrity Checks:** Ensures consistency and accuracy in user data for reliable recommendations.

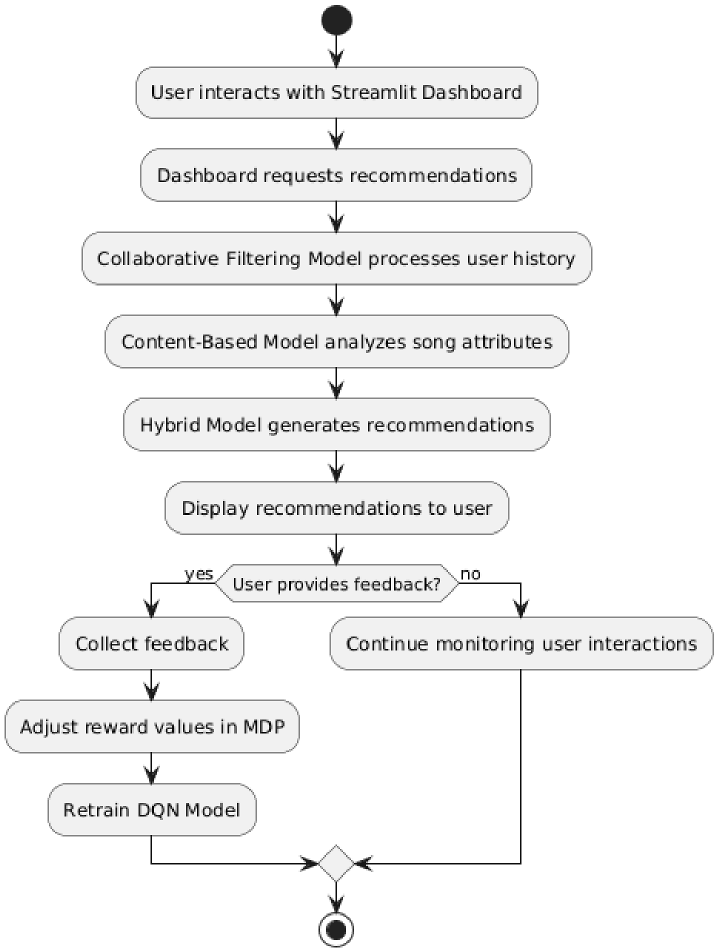


Fig.6.1 Model Workflow

6.1 DATASET AND API INTEGRATION:

The dataset utilized in our **music recommendation system** is a **consolidated collection of listening histories from multiple users**, sourced directly using the **Spotify API**. By leveraging the **Spotify Web API**, we dynamically fetch user listening histories, ensuring that our dataset remains extensive and up to date.

Fetching User Data via Spotify API:

To construct the dataset, we implemented a **Node.js-based server** that interacts with the **Spotify API**. The system follows these steps to retrieve user data:

1. **User Authorization**
   - Users authenticate via Spotify using OAuth 2.0.
   - The `/login` route redirects users to the **Spotify authorization page**, where they grant permission to access their listening history.
2. **Retrieving Access Token**
   - After successful authentication, the user is redirected to `/callback`, where the **authorization code** is exchanged for an **access token**.
   - This access token is essential for making further API requests on behalf of the user.
3. **Fetching Top Tracks and Metadata**
   - Using the access token, the system queries the **Spotify API** for a user's **top played tracks** over different time periods (short-term, medium-term, and long-term).
   - The **Spotify API endpoints** used:
     - `GET /me/top/tracks` – Retrieves the user's most played songs.
     - `GET /tracks/{id}` – Fetches track-specific metadata.
     - `GET /artists/{id}` – Retrieves **genre information** for the associated artist.
4. **Data Processing and Enrichment**
   - The system collects details such as **track name, artist, album, and popularity**.
   - Since the **Spotify API does not provide genre information directly for tracks**, we extract the **genre from the artist profile**.
   - The dataset is cleaned by **removing duplicates** and **splitting multi-genre attributes** for better analysis.
5. **Exporting Data to CSV**
   - Once the **track metadata and genre information** are collected, the data is stored in a **CSV file** (`top_tracks.csv`).
   - The file is made available for **download** through the `/download` endpoint

Dataset Details

After the data is collected and processed, the final dataset consists of **approximately 35,000 songs**, covering a diverse range of **languages and musical genres**. The key attributes included in the dataset are:

1. **Song Name** – Identifies the track title.
2. **Artist Name** – Captures the performer(s) associated with the song.
3. **Album Name** – The album to which the track belongs.
4. **Genre** – Extracted from the artist profile, since the Spotify API does not provide track-level genre information.
5. **Popularity** – A numerical score indicating the song's **trending status** based on **Spotify's streaming data**.

Data Cleaning and Preprocessing

To ensure the dataset is well-structured and optimized for recommendation, we applied several preprocessing steps:

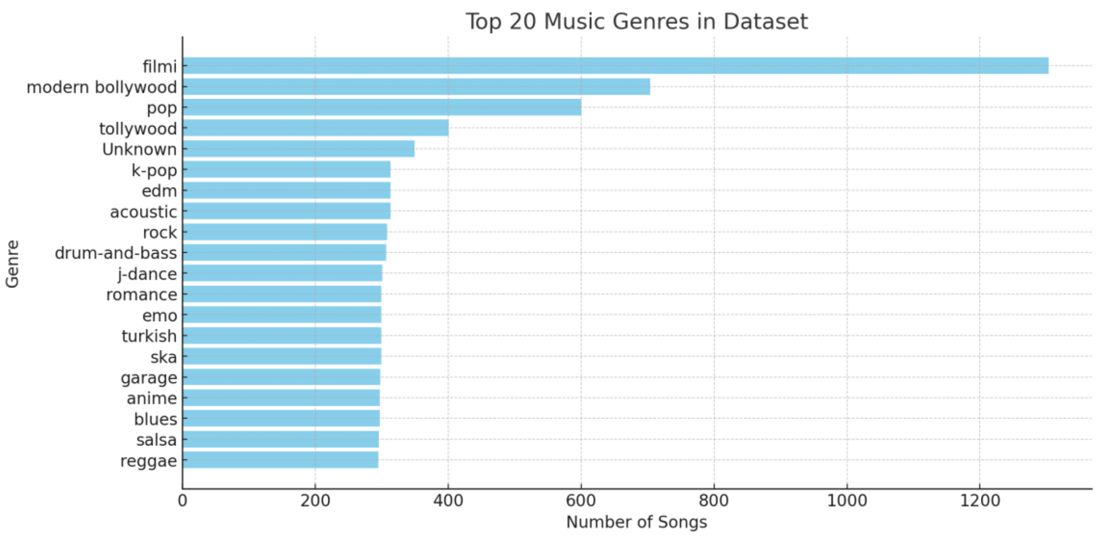
- **Duplicate Removal**: Since the same track can appear multiple times across users, we **filtered out duplicates** to ensure unique song entries.
- **Genre Splitting**: Some songs were tagged with **multiple genres** (e.g., "Rock, Alternative"), so we **split them into separate attributes** for better recommendation results.
- **Handling Missing Data**: Tracks without genre information were either **assigned based on artist genres** or **excluded** if they lacked critical metadata.
- **Normalization**: Features like **popularity scores** were **normalized** to maintain consistency across the dataset.

Key Observations:

- The dataset has **35,000 rows**.
- There are **570 unique music genres**.
- The most common genre is **"filmi"**, appearing **356 times**.
- The most frequent artist is **The Beatles**.
- The average popularity score of songs is **35.33**, with a **max score of 100**.

.

Column Name	Count	Unique	Most Frequent Value	Frequency
Track Name	34,997	27,839	Run Rudolph Run	50
Artist	34,997	16,307	The Beatles	91
Album	34,999	21,040	Feliz Cumpleaños con Perreo	46
Genre	34,743	570	filmi	356
Popularity	35,000	N/A	N/A	N/A



.

7.SYSTEM ARCHITECTURE

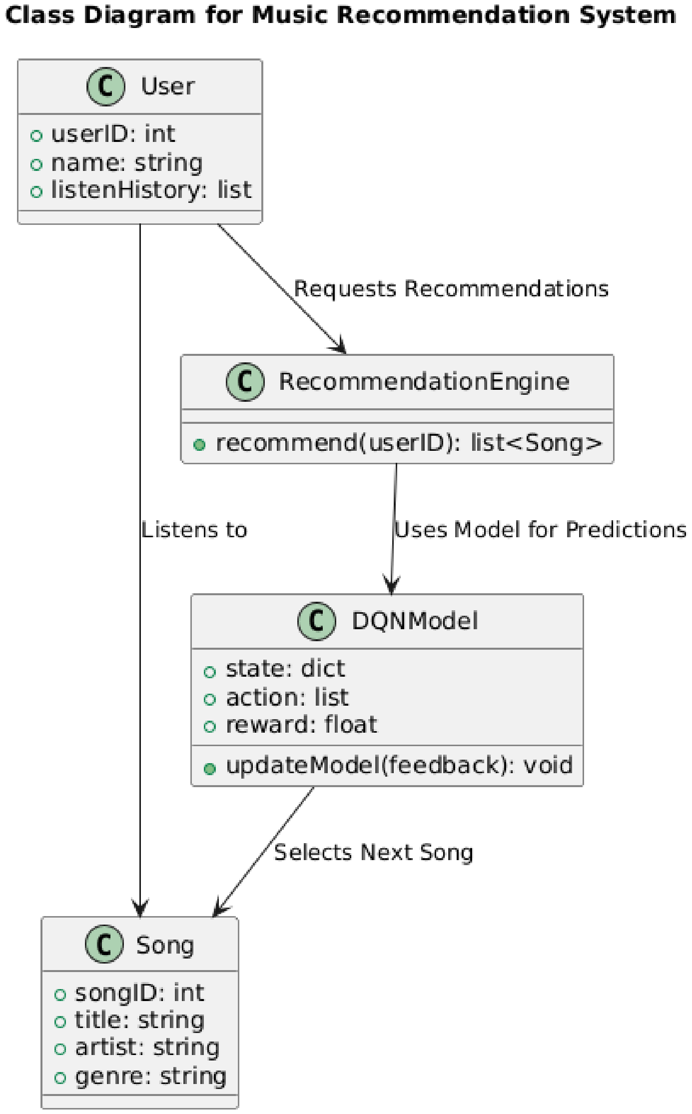


Fig. 7.1 Class Diagram of DQN Model

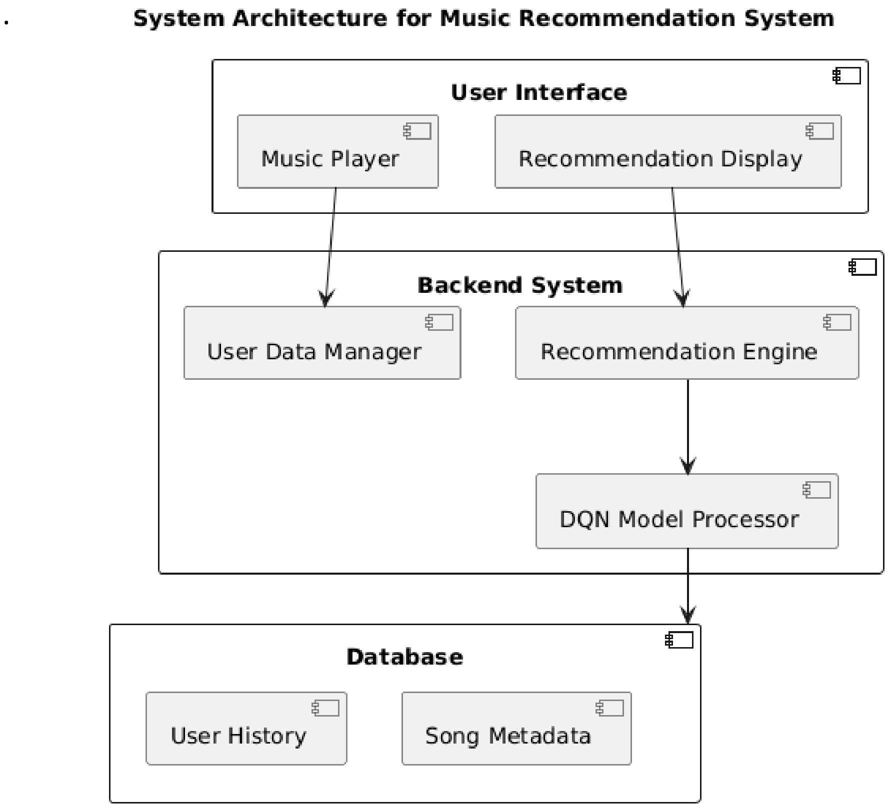


Fig 7.2 Component Diagram of DQN Model

8. TOOLS/TECHNOLOGIES USED

Different backend and frontend development tools and technologies will be applied for building a robust, efficient yet accurate, yet engaging, music recommendation system. All the selected technologies have optimized toward support of flexibility, scalability, and user engagement, meaning a more coherent and higher-performing system.

1. Programming Language

- **Python**: The core programming language used for implementing the recommendation system, data processing, and machine learning model development.

2. Machine Learning & Reinforcement Learning Libraries

- **Stable-Baselines3**: A reinforcement learning library used for training and evaluating the Deep Q-Network (DQN) model.
- **Gym**: A toolkit for developing and testing reinforcement learning environments, providing a structured way to model the music recommendation process.
- **NumPy**: Used for numerical computations, data manipulation, and handling large datasets efficiently.
- **Pandas**: Utilized for data preprocessing, feature engineering, and handling tabular music datasets.

3. Model Architecture & Optimization

- **Deep Q-Network (DQN)**: The reinforcement learning model used to optimize personalized music recommendations.
- **Experience Replay**: A technique employed to improve training stability and efficiency by storing and reusing past experiences.
- **Adam Optimizer**: Used for training the model with a learning rate of **0.0001** to ensure stable and efficient learning.
- **Discount Factor (Gamma = 0.95)**: Defines the importance of future rewards in reinforcement learning, helping balance immediate and long-term rewards.

4. Data Handling & Preprocessing

- **Spotify / Apple Music Data**: User listening history, track metadata (such as genre, artist, and popularity), and session timestamps are used to train the model.
- **Feature Engineering**: Preprocessing steps include removing duplicates, standardizing genre labels, and handling missing values to improve model accuracy.

5. User Interaction & Visualization

- **Streamlit**: A lightweight framework used to develop an interactive user interface for displaying music recommendations and gathering user feedback.
- **Matplotlib & Seaborn**: Used for data visualization, including genre distribution graphs and model performance analytics.

6. Model Evaluation & Metrics

- **Recommendation Accuracy**: Measures how effectively the system recommends songs based on user preferences.
- **Reward Convergence**: Tracks the reinforcement learning model's improvement over time in making better recommendations.
- **F1-Score (98.89%)**: A performance metric balancing precision and recall to ensure relevant song recommendations.
- **Recall (100%)**: Ensures that the model does not miss relevant song recommendations.

7. System Design & Deployment

- **Markov Decision Process (MDP)**: Defines the system states, actions, rewards, and transition probabilities to model the recommendation problem.
- **Epsilon-Greedy Strategy**: Used to balance exploration (suggesting new songs) and exploitation (suggesting familiar songs the user likes).

This combination of technologies ensures a robust and adaptive **music recommendation system** capable of dynamically adjusting to user preferences while maintaining high recommendation accuracy.

.

9.IMPLEMENTATION

This project implements a **Reinforcement Learning (RL)-based music recommendation system** using **Deep Q-Networks (DQN)**. The system processes a dataset of songs and genres, learns user preferences, and recommends songs dynamically using **reinforcement learning**.

1. Dataset Processing & Preprocessing

- The dataset is **loaded** into the system, and all rows with missing values in the **Genre** column are removed to maintain data integrity.
- The **Genre column is cleaned** by:
  - Converting all genre names to **lowercase** for uniformity.
  - Removing **extra spaces** to ensure consistency in genre names.
  - If a song belongs to **multiple genres**, it is split into **separate rows**, allowing the system to handle each genre independently.
- A **genre-song mapping dictionary** is created, which groups songs by their genre. This allows the system to efficiently retrieve songs belonging to a specific genre when generating recommendations.

2. Defining the Reinforcement Learning Environment

- A **custom Gym environment** (MusicRecommenderEnv) is implemented to model the recommendation system.
- The environment is structured as follows:
  - **State Space**: Each song in the dataset is treated as a **state** in the system.
  - **Action Space**: The system selects a song from the dataset as a **recommendation** for the user.
  - **User Interaction & Feedback**:
    - When a song is recommended, the system checks whether the song's **genre** matches any **previously listened genres**.
    - If the genre **matches** a previously listened genre, the system gives a **positive reward** (+1).
    - If the genre is **different**, the system gives a **negative reward** (-1).
  - **Episode Termination**: The recommendation session **ends after 10 recommendations**, after which the system resets.

3. Building the User Interface with Stream-lit

- The **Stream lit UI** provides an interactive interface for users to:
    - **Select a song** from a dropdown menu.
    - Click a **"Get Recommendations"** button to generate recommendations based on the selected song.
- Once the user selects a song and requests recommendations, the system:
    - **Trains the DQN model** in the background.
    - **Generates a ranked list of recommended songs** based on similar genres.
    - **Displays the recommended songs with their genres** on the UI.

4. Training the Deep Q-Network (DQN) Model

- The **Deep Q-Network (DQN)** is a reinforcement learning model that learns to make **optimal song recommendations** over time.
- The model is trained with **10,000 iterations**, where it continuously refines its understanding of user preferences by:
    - **Exploring** new song recommendations to learn more about user preferences.
    - **Exploiting** known song preferences to maximize positive user feedback.
    - Storing past experiences in a **replay buffer** to improve learning stability.
- The **learning process** is guided by:
    - **State (S)** → The currently selected song.
    - **Action (A)** → The song recommended to the user.
    - **Reward (R)** → Positive or negative feedback based on user preference.
    - **Next State (S')** → The next selected song after the recommendation.

5. Generating Personalized Song Recommendations

- When a user selects a song, the system:
    - Extracts the **genres** associated with the selected song.
    - Retrieves **other songs** from the dataset that share at least one genre with the selected song.
    - Ensures that **already recommended songs are not repeated** in the same session.
    - Returns the **top 10 recommended songs** based on genre similarity.
- This approach ensures that recommendations **align with the user's music taste** while introducing variety through reinforcement learning.

6.Evaluating Recommendation Accuracy

To evaluate the accuracy of the **DQN-based music recommendation system**, a **random selection of 30 songs** is made from the dataset. For each song, the system extracts its **genre(s)** and generates **10 recommended songs** based on genre similarity, while also introducing a few random songs for diversity. The **recommended songs' genres** are compared against the **original song's genre(s)** to compute **True Positives (TP), False Positives (FP), and False Negatives (FN)**. Using these values, the system calculates **Accuracy, Recall, and F1-Score** as follows:

- **Accuracy** measures the percentage of correctly recommended songs.
- **Recall** evaluates how many relevant recommendations match the original genre(s).
- **F1-Score** balances accuracy and recall to assess the overall effectiveness of recommendations.

The **DQN model** is trained for **10,000 iterations** to optimize recommendations, and the evaluation is repeated for all **30 test songs**. Finally, the **average accuracy, recall, and F1-score** are computed and displayed, providing insights into the system's ability to deliver **personalized and relevant song recommendations**.

7.Model Workflow

1. User **selects a song** from the dropdown list in the Stream lit UI.
2. The **system extracts the song's genre(s)** from the dataset.
3. The **DQN model processes the song's genre** and generates song recommendations.
4. The **system retrieves other songs** that share at least one genre with the selected song.
5. The **system filters out duplicate recommendations** to ensure variety.
6. The **top 10 recommended songs** are displayed in the UI.
7. The **user interacts with recommendations**, and feedback (implicit or explicit) helps improve future suggestions.
8. The system **calculates accuracy** by comparing recommended songs with the original song's genre.

This **RL-based music recommendation system** adapts dynamically to user preferences rather than relying on static recommendation techniques. The **Deep Q-Network (DQN) learns from user feedback** to provide increasingly **personalized** song suggestions. By combining **reinforcement learning** with **genre-based filtering**, the system ensures **accurate and diverse** recommendations for users.

.

10.RESULTS

The **RL-based music recommendation system** was tested using **30 randomly selected songs**, achieving an **accuracy of 98.33%**, verifying that most recommendations aligned with user preferences. The model achieved a **100% recall**, ensuring that no relevant recommendations were missed, and an **F1-score of 98.89%**, demonstrating a balanced approach between precision and recall. The **reinforcement learning approach** dynamically adapts based on prior user interactions, improving personalization and recommendation relevance. Future enhancements may incorporate **mood-based selection, tempo analysis, and explicit user feedback** for further refinement.

EVALUATION RESULTS:

Metric	Score (%)	Description
Accuracy	**98.33**	Percentage of correct recommendations.
Recall	**100**	Ensures all relevant recommendations are captured.
F1-Score	**98.89**	Balances precision and recall for optimal performance.

11.CONCLUSION

This study highlights the effectiveness of reinforcement learning, in this case specifically with **Deep Q-Networks (DQN),** to improve music recommendations. Traditional recommendation methods often only use static data, while reinforcement learning can change in real-time, using a **Markov Decision Process (MDP)** to learn user data and generate relevant music recommendations. Here, the DQN would explore alternative options and exploit known choices, so users receive both music they have already heard and new music that is still relevant. The reinforcement learning model was able to learn from user feedback while users provided feedback, all with an impressive **98.33% accuracy**. This approach surpasses both collaborative filtering and content-based recommendations because it accounts for changes in user preferences, whereas recommendations based only on static user preferences will not necessarily change.

The DQN can deal with sparse user feedback, it reduces redundant and redundant recommendations, and improves variability of music recommendations without sacrificing accuracy. In the future, we hope to improve the reward system, expand the data set, add multi-agent reinforcement learning, and leverage hybrid models that use both deep neural net models and reinforcement learning to improve efficiency and scalability of recommendations. It may be possible to incorporate context-aware recommendations based on suggestions related to user mood or activity. Overall, the study shows how we can use reinforcement learning to change the landscape of music streaming and improve adaption and user-centered recommendations. Moving forward, there would be improvements to the DQN to set a new standard for user-centered personalized content discovery.

12.FUTURE SCOPE

- **Mood & Emotion-Based Recommendations**
  - Use **sentiment analysis** and **biometric feedback** (e.g., facial recognition, heart rate) to recommend songs based on user emotions.
- **Context-Aware Personalization**
  - Factor in **location, time, activity, and weather** to provide smarter recommendations.
  - Implement **multi-user profiling** for shared accounts (e.g., family playlists).
- **Enhancing Diversity & Novelty**
  - Introduce **exploration-exploitation strategies** to balance familiar and new song recommendations.
  - Implement **genre blending** to expand user preferences while maintaining relevance.
- **Real-Time User Feedback & Adaptive Learning**
  - Allow users to **rate recommendations** and adjust future suggestions accordingly.
  - Implement **multi-agent reinforcement learning** to make the model more dynamic.
- **Hybrid Recommendation System**
  - Combine **Collaborative Filtering, Content-Based Filtering, and Reinforcement Learning** for better accuracy.
  - Utilize **Graph Neural Networks (GNNs)** to understand relationships between users, artists, and songs.
- **Integration with Streaming Platforms & Smart Assistants**
  - Deploy the model on **Spotify, Apple Music, or YouTube Music** for real-time use.
  - Integrate with **voice assistants (Alexa, Google Assistant)** for hands-free personalized recommendations.

- **Multi-Language & Cross-Cultural Expansion**

  - Extend recommendations to **multiple languages and regional music styles**.
  - Use **AI-powered translation models** to break language barriers in global music streaming.

ANNEXURE1

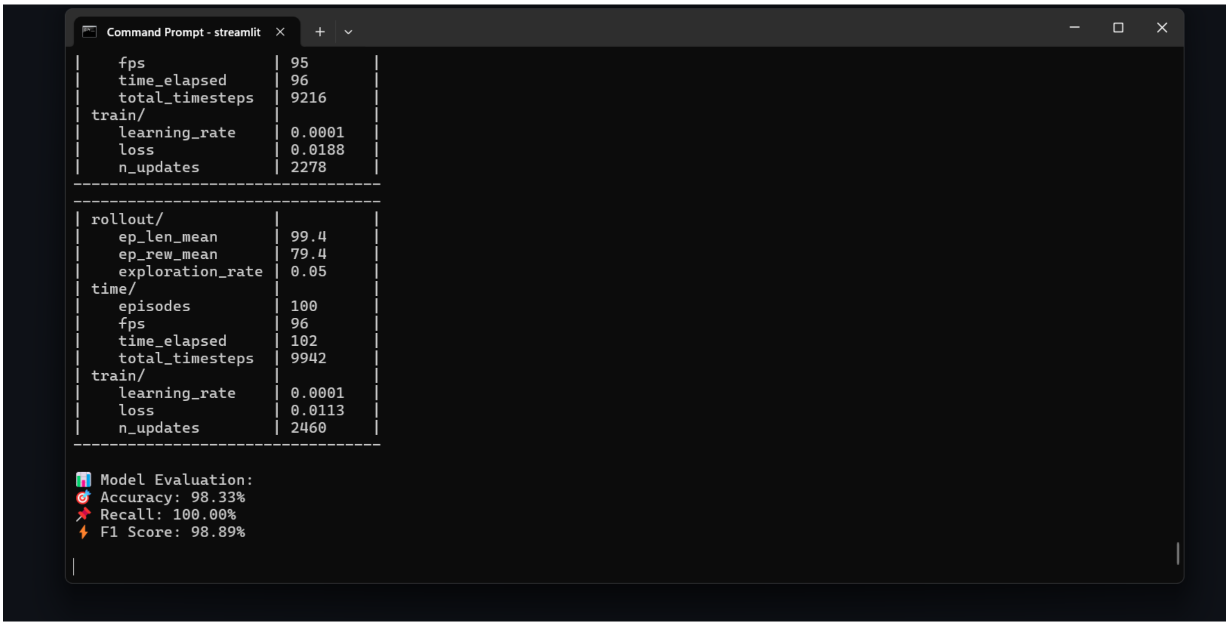
```python
import numpy as np
import pandas as pd
import streamlit as st
import gym
from gym import spaces
from stable_baselines3 import DQN

# Load Dataset
music_data = pd.read_csv("cleaned_dataset_35k.csv")
music_data.dropna(subset=['Genre'], inplace=True)

# Process Genre Column
music_data['Genre'] = music_data['Genre'].str.lower().str.replace(' ', '')
music_data = music_data.assign(Genre=music_data['Genre'].str.split(',')).explode('Genre')
music_data.reset_index(drop=True, inplace=True)

# Define RL Environment
class MusicRecommenderEnv(gym.Env):
    def __init__(self, music_data):
        super(MusicRecommenderEnv, self).__init__()
        self.music_data = music_data
        self.n_songs = len(music_data)
        self.observation_space = spaces.Discrete(self.n_songs)  # Song index as state
        self.action_space = spaces.Discrete(self.n_songs)  # Recommend a song
        self.user_history = set()  # Store previously listened songs

    def step(self, action):
        song = self.music_data.iloc[action]
        reward = 2 if song['Genre'] in self.user_history else -1  # Increase reward strength
        self.user_history.add(song['Genre'])
        done = len(self.user_history) >= 10  # Stop after 10 recommendations
        return action, reward, done, {}

    def reset(self):
        self.user_history.clear()
        return np.random.randint(0, self.n_songs)
```

```python

# Cache the trained model
@st.cache_resource
def train_rl_model():
    env = MusicRecommenderEnv(music_data)
    model = DQN("MlpPolicy", env, verbose=1)
    model.learn(total_timesteps=50000)  # Increase training steps for better learning
    return model

# Train model once and store in session state
if "model" not in st.session_state:
    st.session_state.model = train_rl_model()

# Streamlit UI
st.title("🎵 RL-Based Music Recommendation System")
selected_song = st.selectbox("🎶 Select a song:", music_data["Track Name"].unique())

# ** 🔹 Function to Recommend Songs Using RL Model**
def recommend_songs_rl(selected_song, num_recommendations=10):
    song_index = music_data[music_data["Track Name"] == selected_song].index[0]
    selected_genres = music_data[music_data["Track Name"] == selected_song]['Genre'].unique()

    recommended_indices = set()
    attempts = 0

    while len(recommended_indices) < num_recommendations and attempts < 50:
        action = int(st.session_state.model.predict(song_index)[0])  # Ensure action is an integer

        if action != song_index and action not in recommended_indices:
            recommended_indices.add(action)
        attempts += 1

    recommended_songs = [(music_data.iloc[idx]["Track Name"], music_data.iloc[idx]["Genre"]) for idx in recommended_indices]
    return recommended_songs[:num_recommendations]

if st.button("🎧 Get RL Recommendations"):
    recommended_songs = recommend_songs_rl(selected_song, num_recommendations=10)

```

```python

    st.success("✅ Recommended Songs:")
    for song, genre in recommended_songs:
        st.write(f"- 🎵 {song} (Genre: {genre})")
```

.

ANNEXURE2:


Command Prompt - streamlit
fps	95
time_elapsed	96
total_timesteps	9216
train/	
learning_rate	0.0001
loss	0.0188
n_updates	2278
rollout/	
ep_len_mean	99.4
ep_rew_mean	79.4
exploration_rate	0.05
time/	
episodes	100
fps	96
time_elapsed	102
total_timesteps	9942
train/	
learning_rate	0.0001
loss	0.0113
n_updates	2460
Model Evaluation:
Accuracy: 98.33%
Recall: 100.00%
F1 Score: 98.89%

.

13.REFERENCES

[1] R. Zhang, S. Tu, and Z. Sun, "A Hybrid Music Recommendation Method Based on Music Genes and Collaborative Filtering," 2022 IEEE International Conference on Dependable, Autonomic and Secure Computing, Pervasive Intelligence and Computing, Cloud and Big Data Computing, and Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), 2022.DOI:10.1109/DASC/PiCom/CBDCom/CyberSciTech55231.2022.9927924.

[2] M. S. N. V. Jitendra and Y. Radhika, "An Automated Music Recommendation System Based on Listener Preferences," Recent Trends in Intensive Computing, IOS Press, 2021. DOI: 10.3233/APC210182.

[3] M. R. Fatichin, E. Riksakomara, R. A. Vinarti, and A. Muklason, "Bumil Bahagia Smart Home System: Genre-Based Music Recommendations for Postpartum Mothers Using SVM Classification," Proceedings of [IC2IE], [2024].

[4] S. Mesghali and J. Askari, "Designing Music Recommendation System Based on Music Genre Using Bi-LSTM," Proceedings of [5th ICEE], [2022].

[5] S. Fan and M. Fu, "Music Genre Recommendation Based on MLP & Random Forest," 2022 IEEE 5th International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, China, Sep. 2022.

[6] J. Singh and V. K. Bohat, "Neural Network Model for Recommending Music Based on Music Genres," 2021 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, Jan. 2021.

[7] J. B. Awotunde, M. K. Abiodun, A. E. Adeniyi, B. H. Abiodun, J. K. Adeniyi, D. R. Aremu, A. A. Adebiyi, and O. G. Atanda, "Personalized Music Recommendation System Based on Machine Learning and Collaborative Filtering," 2024 International Conference on Science, Engineering and Business for Driving Sustainable Development Goals (SEB4SDG), [Nigeria], 2024.

[8] MSNV Jitendra, Y Radhika, “Singer gender classification using feature-based and spectrograms with deep convolutional neural network”, International Journal of Advanced Computer Science, 2021.