A Synopsis

ON

Harmonizing Tunes: Building a Content-Based Song Recommendation System

Submitted by-

Anubhav Gupta – 2010990096

Arushika Goel - 2010993522

Dakshay Ahuja - 2010990178

Keshav Garg - 2010991593

Supervised By-

Ajay Kumar

Department of Computer Science and Engineering

CHITKARA UNIVERSITY
RAJPURA (PATIALA) PUNJAB-140401 (INDIA)

October 2023

CONTENTS

Title **Page No.**

1. Abstract:

In an era where the world's musical repertoire is at our fingertips, the quest to discover the perfect tune remains a challenge. This project centres on the development of a content-based song recommendation system with the primary objective of suggesting songs like a given reference track. Leveraging an extensive dataset comprising music tracks and their associated attributes, such as danceability, energy, key, loudness, and more, the project harnesses the power of machine learning to create an intelligent recommendation model.

The key innovation of this system is its ability to analyse the inherent characteristics of songs and establish meaningful connections between them. By dissecting the audio features of each track and understanding their musical nuances, the model can determine the stylistic and thematic similarities among songs.

The recommendation process is initiated by a user-provided reference song, from which the system identifies and suggests tracks that share common musical traits. This approach transcends traditional genre-based recommendations, ensuring that the recommendations are finely tuned to the user's specific preferences.

The project's success hinges on the comprehensive evaluation of the recommendation model. Through rigorous testing and validation, we assess the accuracy and relevance of the recommended songs, ensuring that they align closely with the reference track's characteristics. For practical use, we deploy the system on Heroku, making it accessible to users through a user-friendly web interface. This deployment allows music enthusiasts to explore and discover songs that align with their musical preferences easily.

In conclusion, this project aims to provide a valuable and user-centric solution for music enthusiasts seeking songs that harmonize with their chosen reference track. By exploring the realm of content-based song recommendations, we aim to enrich the music listening experience, unlocking a world of musical exploration based on the unique attributes of each song.

2. Methodology:

2.1. Data Collection and Preprocessing-

This section is pivotal in preparing the data for analysis and recommendation.

Data Source: The data originates from a substantial dataset that contains audio features for an extensive collection of songs, exceeding 1.2 million tracks. This dataset was sourced from a music data repository on Kaggle, offering a rich diversity of songs, genres, and audio attributes.

Data Preprocessing:

Data Cleaning: Data cleaning is an indispensable step to ensure that the dataset is free from inconsistencies, inaccuracies, and data quality issues. This includes identifying and handling missing values, addressing duplicates, and mitigating any outlier data points. By rectifying these issues, the dataset's quality and reliability are greatly enhanced.

Normalization: To facilitate meaningful comparisons and calculations, numerical attribute values are normalized to create a consistent scale. This normalization process is important as it ensures that various audio features contribute uniformly to the recommendation process.

Feature Extraction: Audio features are extracted from the raw dataset, encompassing essential attributes for song recommendation. These features include danceability, energy, key, loudness, and other factors that are vital for understanding the musical characteristics of each song.

2.2. Feature Engineering-

Feature engineering is focused on transforming raw data into meaningful attributes for analysis and recommendation.

TF-IDF Calculation: TF-IDF, or Term Frequency-Inverse Document Frequency, is applied to calculate the importance of each audio feature within the vast dataset. This process assigns numerical values to the audio attributes, reflecting their significance in describing songs.

Attribute Vector Creation: Attribute vectors are crafted for each song by employing the TF-IDF values calculated for the audio features. These vectors serve as numerical representations of song attributes, allowing the system to grasp the importance of each attribute in characterizing a song's unique characteristics.

2.3. Recommendation Model-

The recommendation model, powered by TF-IDF and cosine similarity, is the heart of the system's ability to suggest songs based on user input.

Cosine Similarity: Cosine similarity, a mathematical metric, is used to quantify the similarity between the attribute vectors of songs. It calculates the cosine of the angle between two vectors, providing a numerical measure of how alike songs are in terms of their audio features. Higher cosine similarity values indicate a greater degree of feature similarity.

User Input: A user-friendly web interface is developed to empower users to provide a reference song. This reference song serves as the foundation for generating song recommendations. User choice plays a central role in shaping the recommendation process and ensuring that the suggestions align with their preferences.

Song Ranking: Songs in the dataset are ranked based on their cosine similarity with the attribute vector of the user-provided reference song. Those songs with the highest cosine similarity values are considered the most similar and are recommended to the user. This ranking ensures that the most relevant songs are presented first.

2.4. Evaluation-

The evaluation section is a critical checkpoint for assessing the recommendation system's performance.

Evaluation Metrics: A set of industry-standard evaluation metrics, including Mean Average Precision (MAP) and others, is applied to gauge the quality and accuracy of the song

recommendations. These metrics yield quantitative measures of how effectively the recommendations align with user preferences and expectations.

Hyperparameter Tuning: To fine-tune the recommendation model, hyperparameter tuning is employed. Model parameters, such as TF-IDF weighting schemes, are adjusted, and experimented with to optimize the system's recommendation performance. This iterative process ensures that the system delivers the most accurate and relevant recommendations.

2.5. Deployment-

Practical deployment ensures that the recommendation system is accessible to users.

User Interface: A user-friendly web interface is thoughtfully designed, making it convenient for users to input their reference track. The interface is intuitive and user-centric, enhancing the overall user experience and making it effortless to interact with the system.

Heroku Deployment: The Heroku platform is harnessed for hosting and deploying the recommendation system. Heroku's scalability and accessibility make it an ideal choice for ensuring that the system is readily available to a broad user base and can adeptly handle increased user traffic and demand.

This comprehensive methodology provides a roadmap for the entire project, encompassing data collection and preprocessing, feature engineering, model development, evaluation, and user-friendly deployment. It ensures that the recommendation system is technically sound, produces high-quality recommendations, and is accessible to users. The use of a large dataset from Kaggle with diverse audio features adds depth and richness to the recommendation process.

3. Tools & Technologies:

The successful development and deployment of the content-based song recommendation system rely on a range of tools and technologies:

3.1. Data Collection and Preprocessing-

Python: Python is selected as the primary programming language for its versatility and a rich ecosystem of libraries for data science and machine learning. We utilize Python to perform a variety of tasks, including data collection, data preprocessing, and model development.

Pandas: Pandas is a Python library that plays a pivotal role in data preprocessing. It allows us to efficiently load, manipulate, and clean the large dataset of audio features. Data cleaning operations involve handling missing values, addressing duplicate records, and ensuring data consistency.

Jupyter Notebook: Jupyter Notebook provides an interactive environment for data exploration and documentation. It is employed for creating, executing, and documenting data preprocessing workflows. This helps maintain transparency and allows for collaborative data analysis.

Dataspell: Dataspell, a powerful spell-checking and text analysis tool for Python, is used to enhance the quality of textual data within the dataset. It ensures that text-based attributes, such as song titles, artist names, and album titles, are free from spelling errors and inconsistencies. Clean and accurate textual data contributes to more effective recommendations.

3.2. Feature Engineering and Recommendation Model

Scikit-Learn: Scikit-Learn is a powerful machine learning library in Python. It is utilized for feature engineering and the development of the recommendation model. Scikit-Learn provides tools for calculating TF-IDF (Term Frequency-Inverse Document Frequency) values, computing cosine similarity, and creating the recommendation model. The library's

wide range of algorithms and functionalities ensures that the recommendation system is technically robust and capable of generating high-quality song recommendations.

NumPy: NumPy is a fundamental library for numerical operations and efficient array processing in Python. It is essential in various mathematical computations, including the calculation of cosine similarity. NumPy accelerates numerical operations and optimizes the performance of the system, particularly when dealing with large matrices and arrays.

3.3. Web Development and Deployment

Flask: Flask is a lightweight and highly extensible web framework in Python. It is employed for developing the user interface and handling user interactions. Flask enables the creation of a user-friendly web interface where users can conveniently provide a reference song. It plays a crucial role in guiding the user through the recommendation process.

HTML/CSS: HTML and CSS are used for designing and styling the user interface. These technologies ensure that the web interface is not only user-friendly but also visually appealing. HTML structures the content, while CSS adds styling and enhances the user experience.

3.4. Development and IDE

PyCharm: PyCharm is the integrated development environment (IDE) used for software development in Python. It offers a wide range of features, including code editing, debugging, and version control integration. PyCharm enhances the development process by providing a productive and user-friendly coding environment.

3.5. Evaluation and Testing

Metrics Libraries: For evaluating the recommendation system, we rely on industry-standard evaluation metrics libraries, such as those provided by Scikit-Learn. These libraries offer the necessary tools to calculate and assess the quality of song recommendations. Metrics like Mean Average Precision (MAP) are used to measure how effectively the recommendations

align with user preferences and expectations. This rigorous evaluation process ensures that the system's recommendations are accurate and relevant.

3.6. Data Source

Kaggle: Kaggle is the source of the music data used in the project. The dataset, containing audio features for over 1.2 million songs, is obtained from Kaggle's music data repository. This dataset provides a diverse and extensive collection of songs, offering a wide variety of audio attributes for analysis and recommendation. It serves as the foundation for building the recommendation system.

These tools and technologies collectively form a cohesive and powerful toolkit for developing a content-based song recommendation system. They are chosen for their capabilities in data handling, preprocessing, model development, user interface creation, system deployment, and evaluation. The combination of these tools ensures that the recommendation system is technically robust and user-friendly, delivering high-quality recommendations to users.

4. References:

1. **Kaggle Music Data Repository**:
https://www.kaggle.com/datasets/rodolfofigueroa/spotify-12m-songs/data: This dataset forms the basis of our project, containing audio features for over 1.2 million songs. It serves as a valuable resource for analysing and recommending songs.

2. Scikit-Learn Documentation: https://scikit-learn.org/: The official documentation of Scikit-Learn, a fundamental machine learning library, provided guidance and reference for our feature engineering and recommendation model development.

3. Python Documentation: https://python.org/doc/: The Python documentation was referenced for information on Python programming, Pandas, and NumPy, which played a central role in our data preprocessing and analysis.

4. Heroku: https://devcenter.heroku.com/categories/reference: Heroku's platform documentation and resources were consulted for the deployment of our recommendation system. Heroku provided the hosting infrastructure for making our system accessible to users.

5. Flask: https://flask.palletsprojects.com/en/3.0.x/: Flask's official documentation and tutorials were instrumental in the development of the user interface for our system.

6. TF-IDF: https://en.wikipedia.org/wiki/Tf%E2%80%93idf: The Wikipedia page on TF-IDF provided background information on Term Frequency-Inverse Document Frequency, which was a critical component of our recommendation model.