ARTIFICAL INTELLIGENCE AND MACHINE LEARNING FUNDAMENTALS WITH CLOUD COMPUTING AND GEN AI BY MICROSOFT

SPOTIFY MUSIC RECOMMENDATION SYSTEM

BY

Batch - IV

HEMANTH S

hemanthsubramaniyan@gmail.com

Naan Mudhalvan ID : au8100810021102015

Under the Guidance of

Name of Guide (P.Raja, Master Trainer)

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to everyone who contributed to the successful completion of this project on **Spotify Music Recommendation System using Python**.

Firstly, I extend my heartfelt thanks to my project guide, [P.RAJA], for their invaluable guidance, support, and encouragement throughout this project. Their insights and feedbackwere crucial in refining my understanding of the project, particularly in the field of music recommendation system.

I am grateful to my institution, [ANNA UNIVERSITY – BIT CAMPUS TRICHY], and the Department of [AUTOMOBILE DEPARTMENT], for providing the resources and a conducive environment to explore this project. Special thanks to the faculty members fortheir constant support and motivation, which greatly influenced my approach to problem-solving in this project.

I also wish to acknowledge the open-source community and online platforms for providing excellent resources on Python programming, image processing, and machine learning. The tutorials, libraries, and forums played an essential role in helping me implement effective algorithms and understand complex coding concepts, contributing significantly to the successful realization of this project.

Finally, I am thankful to my family and friends for their continuous encouragement, patience, and understanding during this time.

Thank you.

ABSTRACT of the Project

This research explores the augmentation of user-artist interaction on Spotify by developing and evaluating a content-based recommendation system. Departing from traditional reliance on song plays, our methodology enables users to input specific song preferences or manually adjust Spotify audio features, integrating advanced machine learning techniques with the platform's extensive audio feature metrics. By computing song similarities grounded in numerical audio feature data, the algorithm generates a curated list of tracks within the artist's playlist, aligning with the user's specified preferences. The primary aim is to catalyze an enriched exploration of an artist's catalog, fostering heightened traffic to both the artist's playlist and Spotify profile. The anticipated contributions extend to the broader discourse on personalized music discovery within the Spotify ecosystem, with potential implications for the evolving landscape of digital music platforms. This research seeks to bridge the gap between users and artists, offering a more tailored and engaging music exploration experience within the Spotify paradigm.

This research explores the augmentation of user-artist interaction on Spotify by developing and evaluating a Content-based recommendation system. Departing from traditional reliance on song plays, our methodology Enables users to input specific song preferences or manually adjust Spotify audio features, integrating advanced Machine learning techniques with the platform’s extensive audio feature metrics. By computing song Similarities grounded in numerical audio feature data, the algorithm generates a curated listof tracks within The artist’s playlist, aligning with the user’s specified preferences. The primary aim is to catalyze an enriched Exploration of an artist’s catalog, fostering heightenedtraffic to both the artist’s playlist and Spotify profile. The Anticipated contributions extend to the broader discourse on personalized music discovery within the Spotify Ecosystem, withpotential implications for the evolving landscape of digital music platforms. This research Seeks to bridge the gap between users and artists, offering a more tailored and engaging music exploration Experience within the Spotify paradigm.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1Problem Statement:

The primary goal of Spotify's recommendation system is to enhance user experience and engagement by providing personalized music recommendations.

This involves:

- Increasing user satisfaction: Delivering relevant and enjoyable music suggestions that match individual tastes.
- Boosting user retention: Keeping users engaged with the platform by continuously offering new and relevant content.
- Facilitating music discovery: Helping users discover new artists, genres, and songs that they might not have encountered otherwise.
- Optimizing user journey: Providing a seamless and intuitive recommendation experience.

By achieving these goals, Spotify aims to strengthen its position as a leading music streaming platform and drive user growth and revenue.

1.2 Motivation :

- Personalized Recommendations: Tailoring music suggestions to individual preferences to increase satisfaction and engagement.
- Seamless User Journey: Providing a smooth and intuitive experience by delivering relevant recommendations at the right time.
- Continuous Engagement: Keeping users actively engaged with the platform by offering a constant stream of new and relevant music.

- Reduced Churn: Minimizing user attrition by providing a valuable andpersonalized listening experience.

- Expanding Musical Horizons: Introducing users to new artists, genres, and songs that align with their tastes.

- Fostering a Deeper Connection: Helping users discover music that resonates with them on an emotional level.

- Increased User Acquisition: Attracting new users by offering a superior music discovery experience.

- Premium Subscription Growth: Encouraging users to upgrade to premium subscriptions to access exclusive features and ad-free listening.

- Revenue Generation: Driving revenue through ad impressions, premium subscriptions, and merchandise sales.

By effectively addressing these motivations, Spotify's recommendation system aims to solidify its position as the leading music streaming platform and provide a unique and valuable experience for its users.

1.3 Objective:

The primary objective of Spotify's recommendation system is to enhance user experience and engagement by delivering highly personalized music recommendations.

This involves:

- Increasing User Satisfaction: Providing relevant and enjoyable music suggestions that match individual tastes.

- Boosting User Retention: Keeping users actively engaged with the platform by offering a continuous stream of new and relevant music.

- Facilitating Music Discovery: Helping users discover new artists, genres, and songs that they might not have encountered otherwise.

- Optimizing User Journey: Providing a seamless and intuitive recommendation experience.

1.4 Scope of The Project:

- Personalized Recommendations: Developing algorithms to tailor music suggestions to individual user preferences based on listening history, genre preferences, and social interactions.
- Playlist Generation: Creating personalized playlists based on user preferences, mood, activity, and social context.
- Real-time Recommendations: Delivering timely recommendations as users interact with the platform.

CHAPTER 2

LITERATURE SURVEY

2.1 Review relevant literature or previous work in this domain.

Reviewing relevant literature and previous work in Spotify music recommendation system provides valuable insights into the advancements, methodologies, and challenges in the field.Here's an overview of significant developments in the domain

Personalized Recommender Systems :

Personalization issues adapting to the individual desires, interests, and preferences of every user. They're tools for suggesting things to users.

Content-based Recommender Systems :

Pasquale Lops, Marco American state Gemmis, and Giovanni Semeraro, 2010 [1] in their paper Content-based Recommender Systems: State of the Art and Trends discusses the most problems associated with the illustration of things, ranging from easy techniques for representing structured information to a lot ofcomplicated techniques returning from {the information|the knowledge|the information} Retrieval analysis space for unstructured data. This work is split into three components.

The primary half presents the essential ideas of content-based recommender systems, a high- level design, and their main blessings and disadvantages. The second half a review ofthe state of the art of systems adopted in many application domains by describing eachclassical and advanced technique for representing things and user profiles. The foremost wide adopted techniques for learning user profiles also are conferred. The last half discusses trends and future analysis which could lead towards ensuing generation of systems, by describing the role of User Generated Content as how for taking under consideration evolving vocabularies, and also the challenge of feeding users with lucky recommendations, that's to mention amazingly fascinating things that they could not have otherwise discovered.

Hybrid Recommender Systems :

Survey and Experiments, explains numerous recommendation techniques. These techniques show the complementary benefits and downsides. It compares the assorted techniques and shows that techniques area unit higher supported the analysis metrics. This reality has provided an incentive for analysis in hybrid recommender systems that mix techniques for improved performance. It proposes numerous hybrid approaches which may be accustomed recommendation systems supported the appliance for higher accuracy and results.

Recommendation System Using Association Rules Mining :

Luo Zhenghua, 2012 [3] in the realization of individualized recommendation system on book sale applies the association rules in data processing to e-commerce business systems of book sales, styles AN individualized recommendation system of book sales, and introduces the flow of the advice system and therefore the specific realization procedures of information input, knowledge preprocessing, association rules existence and individualized recommendation. Results show that the net website supported this has shown nice performance.

Hybrid Approach for Collaborative Filtering :

Gilbert Badaro, Hazem Hajj, Wassim El-Hajj, and Lama Nachman, 2013 [4] in hybrid approach for cooperative filtering for recommender systems talks a couple of new hybrid approach for determining the matter of finding the ratingsof unrated things in the user-item ranking matrix by a weighted combination of user primarily based} and item-based cooperative filtering. The projected technique provides enhancements in addressing 2 major challenges of recommender systems: accuracy of recommender systems and scantness of information. The analysis of the system shows the superiority of the answer compared to complete user-based cooperative filtering or item- based cooperative filtering.

The literature survey shows that a hybrid model is projected which mixes user-based cooperative filtering and item-based cooperative filtering by adding the anticipated ratings from every technique and multiplying them with a weight that comes with the accuracy of every technique alone. The approach advantages from the correlation between not solely users alone or things alone however from each at the same time. The analysis was conducted on movielens dataset. the selection of weights was thought of by victimization and adjusting mean absolute error. therefore the survey shows that the hybrid approach improves the information scantness drawback and therefore the accuracy of the system effectively and with efficiency.

Content and collaborative based filtering and association rule mining :

Anand Shanker Tewari, Abhay Kumar, and Asim Gopal bartender, [5]proposes a replacement approach to book recommendation system by combining options of content primarily based filtering, cooperative filtering, and association rule mining. The literature survey shows that numerous parameters like content and quality of the book by doing cooperative filtering of ratings by alternative consumers. the aim of this technique is to advocate books to the client that suits their interest. this technique works offline and stores recommendations within the buyer's internet profile. It finds out the class of the book that the client has bought earlier, like a novel, science, engineering, etc. from the consumer's internet profile. It finds out the subcategory of the book.

It performs content-primarily based filtering in class /subcategory, to search out the books that are unit abundant just like the books that the client has bought earlier from the consumer's past history record. On the results of the on top of the step, item primarily based cooperative filtering is performed. This step truly evaluates the standard of the recommending books supported by the rating given to those books by the opposite consumers. From the book dealing info, realize all transactions whose class and subclass are the same as found in step1 and step2.

Non-Personalized Recommender Systems :

Non-personalized recommender systems are the only form of recommender systems. They are doing not take into consideration the non-public preferences of the users. The recommendations made by these systems are identical for every client.

Non-Personalized and User-based Collaborative Filtering :

Anil Poriya, Neev Patel, Tanvi Bhagat, and Rekha Sharma, Ph. D, [6], in their paper Non-Personalized Recommender Systems and User-based cooperative Recommender Systems describes however websites these days extremely rely on recommender systems. It provides United States insight into 2 common techniques: non customized recommendation and cooperative filtering. Non Customized recommendations use 2 sorts of algorithms: collective opinion recommender and Basic product association recommender.

The literature review describes, collective opinion recommender that essentially recommends restaurants supported the typical score given to that by different customers. The typical is calculated victimization spherical meanratings. But these averages lack context throughout recommendations. Thus basic product association recommender is employed. It provides helpful non- personalized recommendations in an exceeding context. Recommendations might not be essentially specific to the user however specific to what the user is presently doing (viewing/buying). The recommendations during this system are similar to all or any users and lack personalization and therefore won't attractive to everybody. Thus cooperative filtering is employed. The cooperative recommender systems overcome the dearth of the personalization involved non-personalized recommender systems. Conjointly no item knowledge is required for this approach and its domain freelance. The machine time is low for model primarily based approach.

2.2 Mention any existing models, techniques, or methodologies related to theproblem.

Here's an overview of prominent models, techniques, and methodologies commonly used inSpotify music recommendation system :

Data Collection and Retrieval :

The foundation of this research relies on the Spotify API, facilitatedthrough the Spotify library, to access and retrieve playlist tracks. Playlist data, particularly track information, including unique track identifiers (track ID) and relevant audio features, is obtained through the use of Spotify playlist URIs and creator information.

Data Wrangling and Scaling :

Upon retrieving the playlist tracks, a comprehensive datawrangling process ensues to isolate numeric features Crucial for subsequent analyses.

Employing the KMeans algorithm, numeric features are scaled to ensure Uniformity and comparability. This meticulous preprocessing aims to create a robust foundation for subsequent Modeling and clustering.

Modeling and Clustering :

Utilizing the KMeans algorithm, the scaled numeric features undergo clustering to discern inherent patterns Within the dataset. The ensuing clusters provide valuable insights into the distribution and grouping of tracks Based on their audio features. To visualize these clusters, the Plotly library is employed, offering an interactive And informative representation.

Binary Classification Model :

In parallel, a binary classification model, specifically a Logistic Regression model, is implemented to predict Certain attributes within the dataset. This model is rigorously evaluated through the creation of a confusion Matrix and further elucidated through LIME visualization, enhancing transparency and interpretability.

Analysis :

Clustering Analysis :

The results of the KMeans clustering analysis are presented, elucidating the identified clusters and their Significance in the context of audio features. The visual representation of these clusters using Plotly serves as a Powerful tool for comprehending the intricate relationships and patterns inherent in the dataset.

Binary Classification Model Evaluation :

The binary classification model's performance is thoroughly evaluated through the construction and Interpretation of a confusion matrix. This matrix offers a detailed breakdown of the model's predictive Capabilities, facilitating a nuanced understanding of its strengths and limitations. LIME visualization further Enhances model interpretability by providing insights into individual predictions.

Recommendation System Implementation :

Building upon the insights derived from clustering and binary classification, the recommendation system is

Implemented. Leveraging the KMeans clusters and Spotify API through Spotify, the system computes vectors And cosine distances to generate personalized recommendations. This process is detailed to underscore the Systematic approach employed in transforming analytical findings into actionable user-centric Recommendations.

Stream lit Web App Development :

The methodology concludes with the development of a user-friendly Stream lit web application. This application Incorporates text boxes and a sidebar for user input, allowing users to actively engage with the Recommendation system. Additionally, the app provides descriptive information regarding audio features, Fostering user understanding and participation.

Through the intricate amalgamation of Spotify API interactions, data wrangling, clustering, binary classification, And recommendation system implementation, this comprehensive methodology seeks to unravel the intricacies Of user-artist interaction and playlist engagement within the Spotify ecosystem. The subsequent analysis Endeavors to distill actionable insights from the amassed data, contributing to the broader discourse on personalized music discovery.

2.3 Highlight the gaps or limitations in existing solutions and how your project will address them.

Challenges and Future Improvements :

- **Handling the "Cold Start" problem:** Continuously improve the ability to recommend to new users or new songs by incorporating more data and leveraging hybrid approaches.
- **Serendipity and Novelty:** Striving for not just relevant but also diverse and unexpected recommendations to keep users engaged.
- **Improving Real-Time Processing:** Fine-tuning algorithms to make moreinstant, personalized recommendations as users listen to music.

CHAPTER 3

Proposed Methodology

3.1System Design:

Designing the recommendation system for Spotify involves building a system that can suggest personalized music tracks to users based on their preferences, behavior, and various other factors. Here's a step-by-step breakdown of how to approach thesystem design for Spotify's recommendation engine.

❖ Key Components of the Recommendation System:

- **Data Collection:** Gather data on user interactions, such as listens, skips, likes, playlists, and search queries.
- **Feature Engineering:** Process and transform raw data into meaningful features,like song characteristics (e.g., genre, tempo), user behavior, and contextual information (e.g., time of day, location).
- **Modeling:** Use machine learning and/or collaborative filtering to generate recommendations.
- **Ranking and Personalization:** Rank the generated recommendations based on relevance for each user.
- **Real-time Processing:** Update recommendations based on real-time interactions (e.g., likes, skips).
- **Offline and Online Learning:** Use offline learning (batch processing) for model training and online learning for real-time adaptation.

❖ Type of Recommendations :

I. Collaborative Filtering:

User-User Collaborative Filtering: Recommend songs liked by similar users.

Item-Item Collaborative Filtering: Recommend songs that are similar to the ones the user has already interacted with.

II. Content-Based Filtering:

Recommend songs based on their characteristics (e.g., genre, artist, tempo).

Use features extracted from the audio content, metadata, or user profiles.

III. Hybrid Model:

Combine collaborative filtering, content-based filtering, and deep learning models to increase recommendation quality.

For example, Spotify combines collaborative filtering with content-based approaches and personalization techniques.

IV. Deep Learning (Neural Networks):

Autoencoders for collaborative filtering: learn user and song embeddings.

Recurrent Neural Networks (RNNs): for sequential recommendation based on listening history.

Neural collaborative filtering: combining embeddings of users and items in a neural network architecture.

V. Context-Aware Recommendations:

Personalize recommendations based on context, such as location, time of day, or mood (e.g., relaxing music in the evening).

Spotify uses contextual bandit algorithms to serve contextual recommendations.

❖ Data Sources:

The recommendation system relies on several types of data:

User Behavior Data: Songs listened to, skips, time spent listening, user interactions, searches, playlists created, etc.

Content Data: Song metadata (artist, genre, tempo, acoustic features), audio features(e.g., loudness, key, beats per minute).

Social Data: Following users, shared playlists, likes, and social interactions.

Contextual Data: Device, time of day, location, etc.

❖ Data Ingestion and Processing Pipeline:

Real-time Streaming: Use platforms like Kafka, AWS Kinesis, or Google Pub/Sub for real-time user interactions (listens, skips).

Batch Processing: Periodic updates of user data, music features, and model training using frameworks like Apache Spark or Google Dataflow.

❖ Data Storage:

Use distributed databases like Cassandra, DynamoDB**, or** HBase **for storing user data and song metadata.**

Use HDFS **or cloud-based storage for big data processing.**

❖ Recommendation Engine (Backend):

Collaborative Filtering Model: Implement with matrix factorization or deep learning techniques (e.g., using TensorFlow or PyTorch).

Content-Based Filtering: Use song embeddings, often generated using algorithms like Word2Vec or TF-IDF, to recommend songs with similar characteristics.

Ranking System: After generating a set of recommendations, rank them based on factors like relevance, freshness, and diversity. Techniques like learning to rank (e.g., XGBoost) can be used here.

❖ Serving Layer:

The recommendations are served via an API (e.g., GraphQL or RESTful APIs) to the front-end apps.

Caching Layer: Use Redis or Memcached to cache frequently requested data to ensure low-latency responses.

❖ Real-Time Update & Personalization:

Continuous training of models on the fly using online learning or reinforcement learning (e.g., using Bandit algorithms to adapt recommendations in real-time based on feedback).

A/B Testing: Continuous experiments to test the effectiveness of different recommendation strategies.

❖ **Scalability & Fault Tolerance:**

Sharding: The recommendation data can be sharded across multiple servers based on user IDs or song IDs to ensure scalability.

Load Balancing: Distribute requests across servers to manage traffic and avoid bottlenecks.

Data Replication: Use replication to ensure data durability and availability across distributed systems.

Caching: Use caching mechanisms (e.g., Redis) to reduce load on backend systems and speed up recommendation retrieval.

3.2 Data Flow Diagram:

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary stepto create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

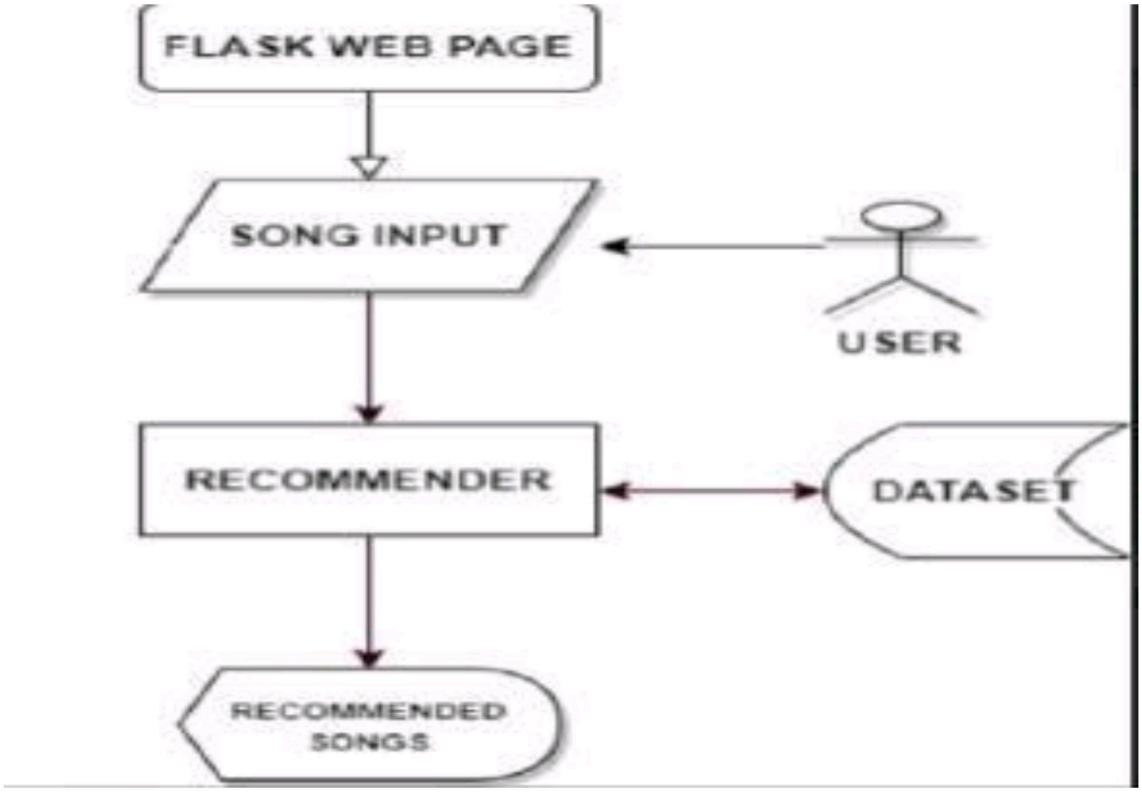


FIGURE I

The KMeans clustering algorithm is deployed to identify inherent patterns within the scaled numeric features of playlist tracks. This algorithm partitions the dataset into distinct clusters based on the similarity of audio features, providing a nuanced understanding of the underlying structure.

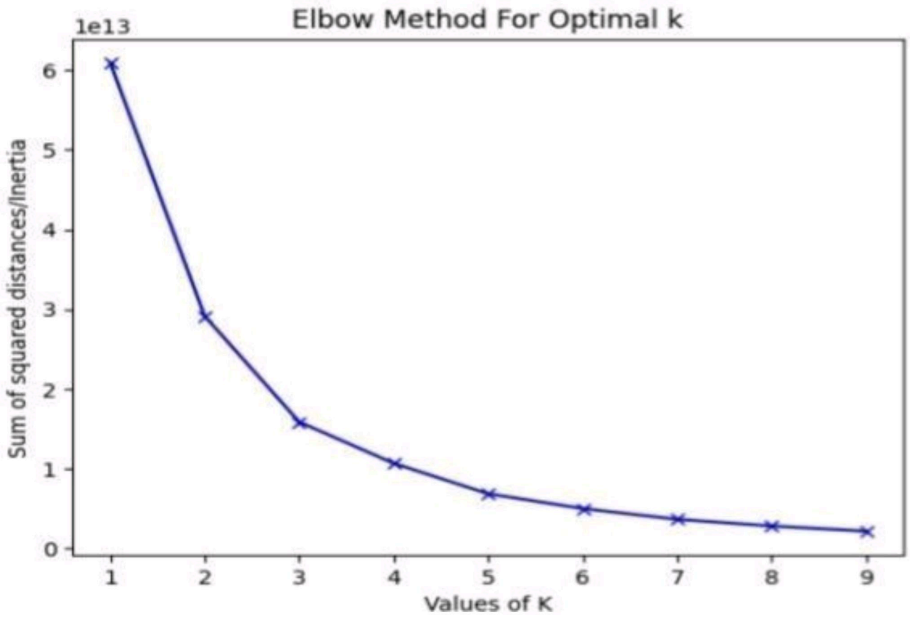


FIGURE II

3.3 Advantages :

- **Click-through Rate (CTR):** How often recommended songs are clicked orplayed.
- **Engagement Rate:** How long users engage with recommended tracks.
- **Diversity:** The variety of songs recommended.
- **Novelty:** How many new or undiscovered songs are recommended.
- **User Satisfaction:** Metrics like user retention and satisfaction surveys can beused to measure how well recommendations align with user preferences.

3.4 Requirement Specification :

Hardware Requirements:

The main memory required is 8 GB & above so that the whole program can reside on the same memory at once. This will avoid the requirement to swap the memory contents of the system. The hard disk drive is requiredto store the program permanently on the storage. The processor is required to process the data quickly on the system. A Computer/Laptop is require t o enable the user to interact with the system while on the go.

Software Requirements:

The operating systems used will be windows 7& above.

Programming languages usedare Python, HTML5, CSS3, Bootstrap .

CHAPTER 4

Implementation and Result

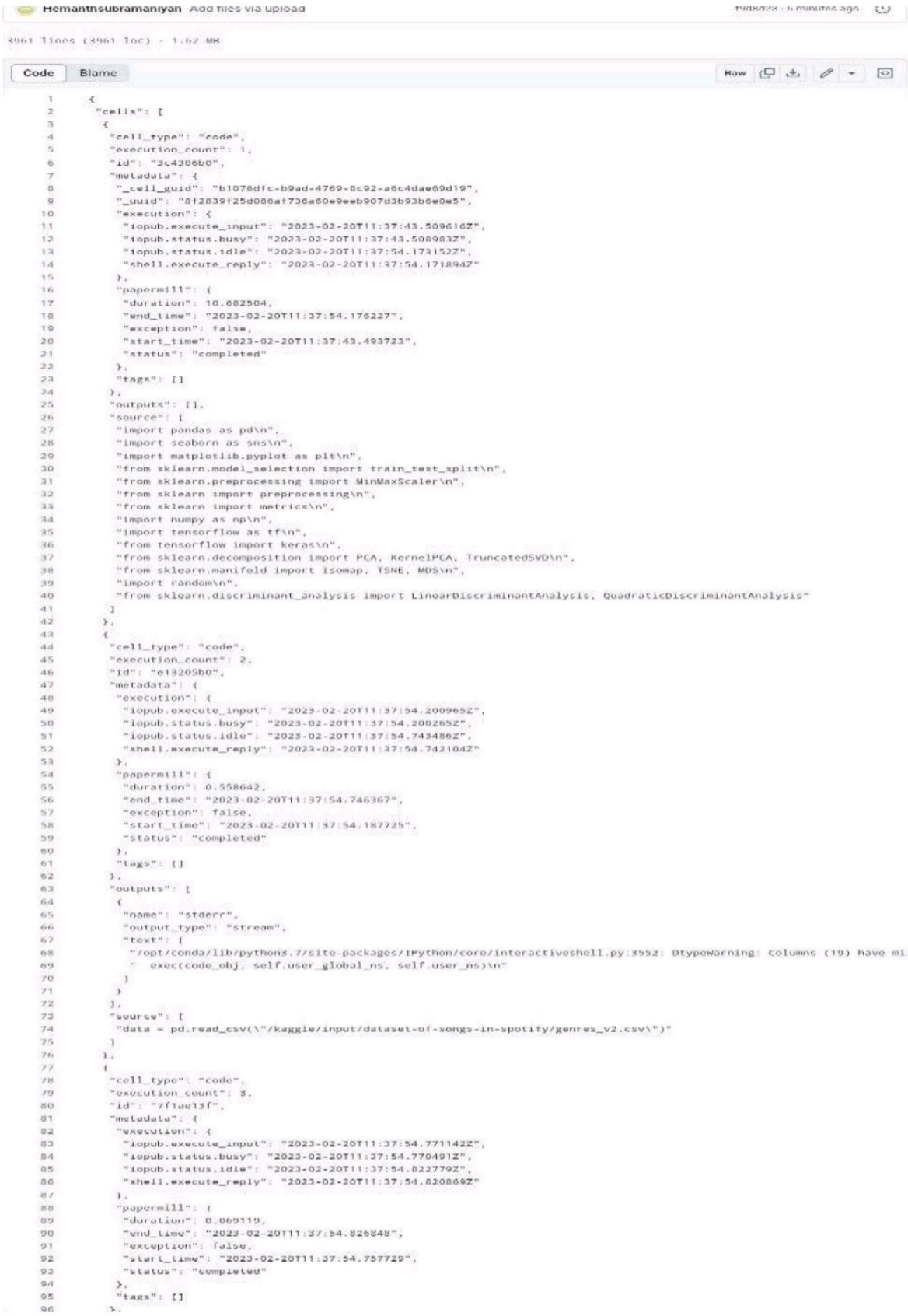
In the system, First user inputs the song which he/she wants; once the required song is inputted by the user, that ten similar songs are recommended to him. Initially, the process takes into consideration by taking three main features, that is Title, Artist, and Top Genre, which is done by taking Angular distance and Euclidean distance. For this, we have taken the class Count Vectorizer and method cosine similarity. Count vectorizer is stored in an object which is used to count the number of terms that appeared in a particular feature; after that, structured data is used by cosine similarity to find the similarity score. Before the data is processed by the count vectorizer class, since we are using multiple parameters/ features to find the similarity score, a function is created to merge the contents of all the rows of the specified features. In case any NaN values are found, they are replaced with an empty string.

In the system, First user inputs the song which he/she wants; once the required song is inputted by the user, that ten similar songs are recommended to him. Initially, the process takes into consideration by taking three main features, that is Title, Artist, and Top Genre, which is done by taking Angular distance and Euclidean distance. For this, we have taken the class Count Vectorizer and method cosine similarity. Count vectorizer is stored in an object which is used to count the number of terms that appeared in a particular feature; after that, structured data is used by cosine similarity to find the similarity score. Before the data is processed by the count vectorizer class, since we are using multiple parameters/ features to find the similarity score, a function is created to merge the contents of all the rows of the specified features. In case any NaN values are found, they are replaced with an empty string.

main Hemanth-au810021102015 / Hemanth S.ipynb.txt ↑ Top

Code Blame Raw

```
          "  </tbody>\n",
          "</table>\n",
          "<p>42305 rows × 20 columns</p>\n",
          "</div>"
         ],
         "text/plain": [
          "       danceability  energy  key  loudness  mode  speechiness  acousticness  \\\n",
          "0             0.831   0.814    2    -7.364     1       0.4200       0.059800   \n",
          "1             0.719   0.493    8    -7.230     1       0.0794       0.401000   \n",
          "2             0.850   0.893    5    -4.783     1       0.0623       0.013800   \n",
          "3             0.476   0.781    0    -4.710     1       0.1030       0.023700   \n",
          "4             0.798   0.624    2    -7.668     1       0.2930       0.217000   \n",
          "...             ...     ...  ...       ...   ...          ...            ...   \n",
          "42300         0.528   0.693    4    -5.148     1       0.0304       0.031500   \n",
          "42301         0.517   0.768    0    -7.922     0       0.0479       0.022500   \n",
          "42302         0.361   0.821    8    -3.102     1       0.0505       0.026000   \n",
          "42303         0.477   0.921    6    -4.777     0       0.0392       0.000551   \n",
          "42304         0.529   0.945    9    -5.862     1       0.0615       0.001890   \n",
          "\n",
          "       instrumentalness  liveness  valence    tempo            type  \\\n",
          "0              0.013400    0.0556   0.3890  156.985  audio_features   \n",
          "1              0.000000    0.1180   0.1240  115.080  audio_features   \n",
          "2              0.000004    0.3720   0.0391  218.050  audio_features   \n",
          "3              0.000000    0.1140   0.1750  186.948  audio_features   \n",
          "4              0.000000    0.1660   0.5910  147.988  audio_features   \n",
          "...                 ...       ...      ...      ...             ...   \n",
          "42300          0.000345    0.1210   0.3940  150.013  audio_features   \n",
          "42301          0.000018    0.2050   0.3830  149.928  audio_features   \n",
          "42302          0.000242    0.3850   0.1240  154.935  audio_features   \n",
          "42303          0.029600    0.0575   0.4880  150.042  audio_features   \n",
          "42304          0.000055    0.4140   0.1340  155.047  audio_features   \n",
          "\n",
          "                          id                                  uri  \\\n",
          "0      2Vc6NJ9PW9gD9q343XFRKx  spotify:track:2Vc6NJ9PW9gD9q343XFRKx   \n",
          "1      7pgJBLVz5VmnL7uGHmRj6p  spotify:track:7pgJBLVz5VmnL7uGHmRj6p   \n",
          "2      0vSWgAlfpye0WCGeNmuNhy  spotify:track:0vSWgAlfpye0WCGeNmuNhy   \n",
          "3      0VSXnJqQkwuH2ei1nOQ1nu  spotify:track:0VSXnJqQkwuH2ei1nOQ1nu   \n",
          "4      4jCeguq9rMTlbMmPHuO7S3  spotify:track:4jCeguq9rMTlbMmPHuO7S3   \n",
          "...                       ...                                 ...   \n",
          "42300  46bXU7Sgj7104ZoXxzz9tM  spotify:track:46bXU7Sgj7104ZoXxzz9tM   \n",
          "42301  0he2ViGMUO3ajKTxLOfWVT  spotify:track:0he2ViGMUO3ajKTxLOfWVT   \n",
          "42302  72DAt9Lbpy9EUS29OzQLob  spotify:track:72DAt9Lbpy9EUS29OzQLob   \n",
          "42303  6HXgExFVuE1c3cq9QjFCcU  spotify:track:6HXgExFVuE1c3cq9QjFCcU   \n",
          "42304  6MAAMZImxcvYhRnxDLTufD  spotify:track:6MAAMZImxcvYhRnxDLTufD   \n",
          "\n",
          "                                                track_href  \\\n",
          "0      https://api.spotify.com/v1/tracks/2Vc6NJ9PW9gD...   \n",
          "1      https://api.spotify.com/v1/tracks/7pgJBLVz5Vmn...   \n",
          "2      https://api.spotify.com/v1/tracks/0vSWgAlfpye0...   \n",
          "3      https://api.spotify.com/v1/tracks/0VSXnJqQkwuH...   \n",
          "4      https://api.spotify.com/v1/tracks/4jCeguq9rMTl...   \n",
          "...                                                  ...   \n",
          "42300  https://api.spotify.com/v1/tracks/46bXU7Sgj710...   \n",
          "42301  https://api.spotify.com/v1/tracks/0he2ViGMUO3a...   \n",
          "42302  https://api.spotify.com/v1/tracks/72DAt9Lbpy9E...   \n",
          "42303  https://api.spotify.com/v1/tracks/6HXgExFVuE1c...   \n",
          "42304  https://api.spotify.com/v1/tracks/6MAAMZImxcvY...   \n",
          "\n",
          "                                                analysis_url  duration_ms  \\\n",
          "0      https://api.spotify.com/v1/audio-analysis/2Vc6...       124539   \n",
          "1      https://api.spotify.com/v1/audio-analysis/7pgJ...       224427   \n",
          "2      https://api.spotify.com/v1/audio-analysis/0vSW...        98821   \n",
          "3      https://api.spotify.com/v1/audio-analysis/0VSX...       123661   \n",
          "4      https://api.spotify.com/v1/audio-analysis/4jCe...       123298   \n",
          "...                                                  ...          ...   \n",
          "42300  https://api.spotify.com/v1/audio-analysis/46bX...       269208   \n",
          "42301  https://api.spotify.com/v1/audio-analysis/0he2...       210112   \n",
          "42302  https://api.spotify.com/v1/audio-analysis/72DA...       234823   \n",
          "42303  https://api.spotify.com/v1/audio-analysis/6HXg...       323200   \n",
          "42304  https://api.spotify.com/v1/audio-analysis/6MAA...       162161   \n",
          "\n",
          "       time_signature      genre  \\\n",
          "0                   4  Dark Trap   \n",
          "1                   4  Dark Trap   \n",
          "2                   4  Dark Trap   \n",
          "3                   3  Dark Trap   \n",
          "4                   4  Dark Trap   \n",
          "...               ...        ...   \n",
          "42300               4  hardstyle   \n",
          "42301               4  hardstyle   \n",
          "42302               4  hardstyle   \n",
          "42303               4  hardstyle   \n",
          "42304               4  hardstyle   \n",
          "\n",
          "                                                song_name  \n",
          "0                                     Mercury: Retrograde  \n",
          "1                                               Pathology  \n",
          "2                                                Symbiote  \n",
          "3      ProductOfDrugs (Prod. The Virus and Antidote)  \n",
          "4                                                   Venom  \n",
          "...                                                   ...
```

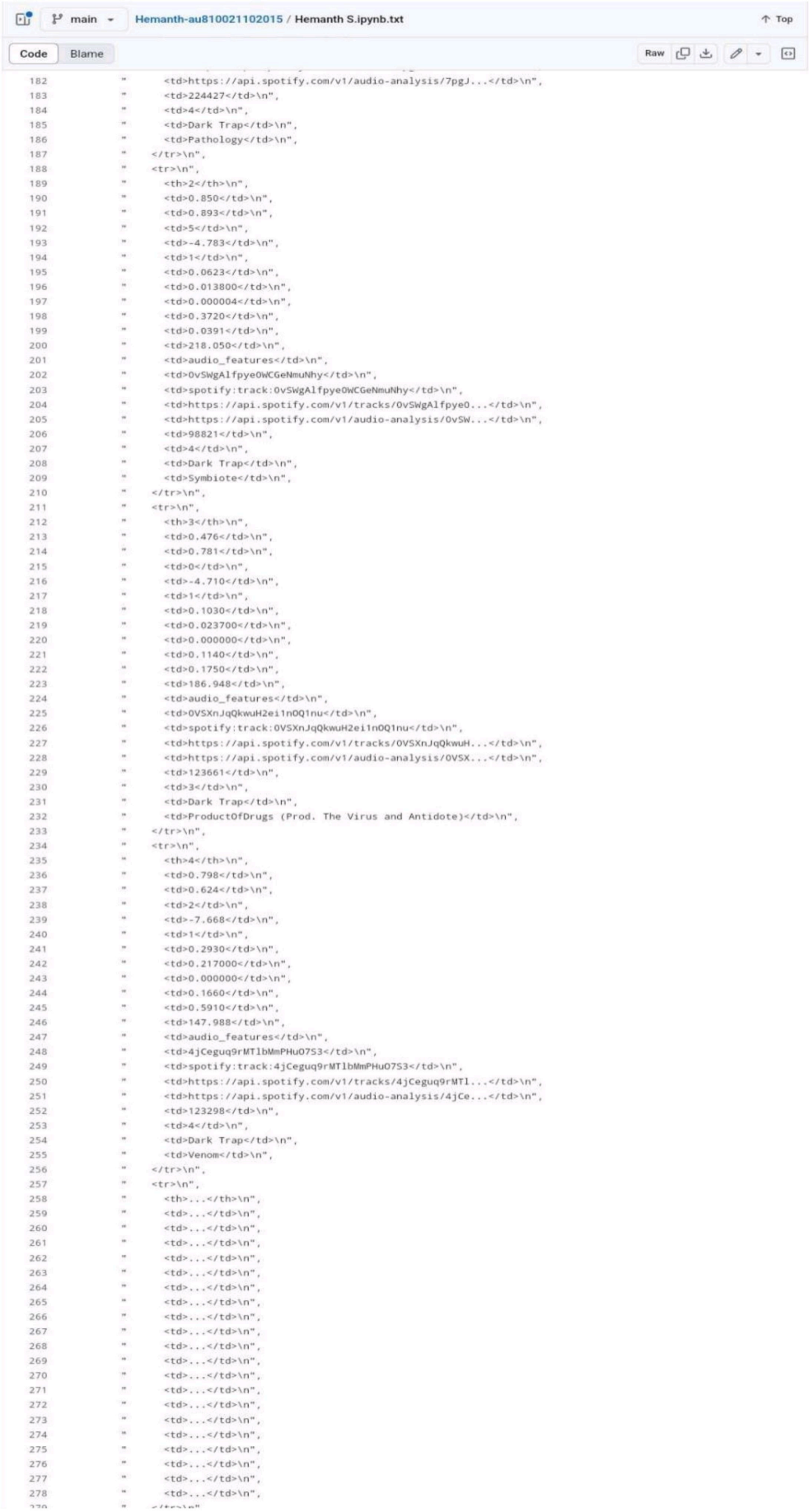


Hemanthsubramaniyan Add files via upload

4961 lines (4961 loc) · 1.62 MB

Code Blame Raw


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "3c4306b0",
   "metadata": {
    "_cell_guid": "b1076dfc-b9ad-4769-8c92-a6c4dae69d19",
    "_uuid": "8f2839f25d086af736a60e9eeb907d3b93b6e0e5",
    "execution": {
     "iopub.execute_input": "2023-02-20T11:37:43.509616Z",
     "iopub.status.busy": "2023-02-20T11:37:43.508983Z",
     "iopub.status.idle": "2023-02-20T11:37:54.173152Z",
     "shell.execute_reply": "2023-02-20T11:37:54.171894Z"
    },
    "papermill": {
     "duration": 10.682504,
     "end_time": "2023-02-20T11:37:54.176227",
     "exception": false,
     "start_time": "2023-02-20T11:37:43.493723",
     "status": "completed"
    },
    "tags": []
   },
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import seaborn as sns\n",
    "import matplotlib.pyplot as plt\n",
    "from sklearn.model_selection import train_test_split\n",
    "from sklearn.preprocessing import MinMaxScaler\n",
    "from sklearn import preprocessing\n",
    "from sklearn import metrics\n",
    "import numpy as np\n",
    "import tensorflow as tf\n",
    "from tensorflow import keras\n",
    "from sklearn.decomposition import PCA, KernelPCA, TruncatedSVD\n",
    "from sklearn.manifold import Isomap, TSNE, MDS\n",
    "import random\n",
    "from sklearn.discriminant_analysis import LinearDiscriminantAnalysis, QuadraticDiscriminantAnalysis"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "e1320bb0",
   "metadata": {
    "execution": {
     "iopub.execute_input": "2023-02-20T11:37:54.200965Z",
     "iopub.status.busy": "2023-02-20T11:37:54.200265Z",
     "iopub.status.idle": "2023-02-20T11:37:54.743486Z",
     "shell.execute_reply": "2023-02-20T11:37:54.742104Z"
    },
    "papermill": {
     "duration": 0.558642,
     "end_time": "2023-02-20T11:37:54.746367",
     "exception": false,
     "start_time": "2023-02-20T11:37:54.187725",
     "status": "completed"
    },
    "tags": []
   },
   "outputs": [
    {
     "name": "stderr",
     "output_type": "stream",
     "text": [
      "/opt/conda/lib/python3.7/site-packages/IPython/core/interactiveshell.py:3552: DtypeWarning: Columns (19) have mi
      "  exec(code_obj, self.user_global_ns, self.user_ns)\n"
     ]
    }
   ],
   "source": [
    "data = pd.read_csv(\"/kaggle/input/dataset-of-songs-in-spotify/genres_v2.csv\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "id": "7f1ee13f",
   "metadata": {
    "execution": {
     "iopub.execute_input": "2023-02-20T11:37:54.771142Z",
     "iopub.status.busy": "2023-02-20T11:37:54.770491Z",
     "iopub.status.idle": "2023-02-20T11:37:54.822779Z",
     "shell.execute_reply": "2023-02-20T11:37:54.820869Z"
    },
    "papermill": {
     "duration": 0.069119,
     "end_time": "2023-02-20T11:37:54.826848",
     "exception": false,
     "start_time": "2023-02-20T11:37:54.757729",
     "status": "completed"
    },
    "tags": []
   },
```




```
182 "     <td>https://api.spotify.com/v1/audio-analysis/7pgJ...</td>\n",
183 "     <td>224427</td>\n",
184 "     <td>4</td>\n",
185 "     <td>Dark Trap</td>\n",
186 "     <td>Pathology</td>\n",
187 "   </tr>\n",
188 "   <tr>\n",
189 "     <th>2</th>\n",
190 "     <td>0.850</td>\n",
191 "     <td>0.893</td>\n",
192 "     <td>5</td>\n",
193 "     <td>-4.783</td>\n",
194 "     <td>1</td>\n",
195 "     <td>0.0623</td>\n",
196 "     <td>0.013800</td>\n",
197 "     <td>0.000004</td>\n",
198 "     <td>0.3720</td>\n",
199 "     <td>0.0391</td>\n",
200 "     <td>218.050</td>\n",
201 "     <td>audio_features</td>\n",
202 "     <td>0vSWgAlfpyeOWCGeNmuNhy</td>\n",
203 "     <td>spotify:track:0vSWgAlfpyeOWCGeNmuNhy</td>\n",
204 "     <td>https://api.spotify.com/v1/tracks/0vSWgAlfpyeO...</td>\n",
205 "     <td>https://api.spotify.com/v1/audio-analysis/0vSW...</td>\n",
206 "     <td>98821</td>\n",
207 "     <td>4</td>\n",
208 "     <td>Dark Trap</td>\n",
209 "     <td>Symbiote</td>\n",
210 "   </tr>\n",
211 "   <tr>\n",
212 "     <th>3</th>\n",
213 "     <td>0.476</td>\n",
214 "     <td>0.781</td>\n",
215 "     <td>0</td>\n",
216 "     <td>-4.710</td>\n",
217 "     <td>1</td>\n",
218 "     <td>0.1030</td>\n",
219 "     <td>0.023700</td>\n",
220 "     <td>0.000000</td>\n",
221 "     <td>0.1140</td>\n",
222 "     <td>0.1750</td>\n",
223 "     <td>186.948</td>\n",
224 "     <td>audio_features</td>\n",
225 "     <td>0VSXnJqQkwuH2ei1nOQ1nu</td>\n",
226 "     <td>spotify:track:0VSXnJqQkwuH2ei1nOQ1nu</td>\n",
227 "     <td>https://api.spotify.com/v1/tracks/0VSXnJqQkwuH...</td>\n",
228 "     <td>https://api.spotify.com/v1/audio-analysis/0VSX...</td>\n",
229 "     <td>123661</td>\n",
230 "     <td>3</td>\n",
231 "     <td>Dark Trap</td>\n",
232 "     <td>ProductOfDrugs (Prod. The Virus and Antidote)</td>\n",
233 "   </tr>\n",
234 "   <tr>\n",
235 "     <th>4</th>\n",
236 "     <td>0.798</td>\n",
237 "     <td>0.624</td>\n",
238 "     <td>2</td>\n",
239 "     <td>-7.668</td>\n",
240 "     <td>1</td>\n",
241 "     <td>0.2930</td>\n",
242 "     <td>0.217000</td>\n",
243 "     <td>0.000000</td>\n",
244 "     <td>0.1660</td>\n",
245 "     <td>0.5910</td>\n",
246 "     <td>147.988</td>\n",
247 "     <td>audio_features</td>\n",
248 "     <td>4jCeguq9rMTlbMmPHuO7S3</td>\n",
249 "     <td>spotify:track:4jCeguq9rMTlbMmPHuO7S3</td>\n",
250 "     <td>https://api.spotify.com/v1/tracks/4jCeguq9rMTl...</td>\n",
251 "     <td>https://api.spotify.com/v1/audio-analysis/4jCe...</td>\n",
252 "     <td>123298</td>\n",
253 "     <td>4</td>\n",
254 "     <td>Dark Trap</td>\n",
255 "     <td>Venom</td>\n",
256 "   </tr>\n",
257 "   <tr>\n",
258 "     <th>...</th>\n",
259 "     <td>...</td>\n",
260 "     <td>...</td>\n",
261 "     <td>...</td>\n",
262 "     <td>...</td>\n",
263 "     <td>...</td>\n",
264 "     <td>...</td>\n",
265 "     <td>...</td>\n",
266 "     <td>...</td>\n",
267 "     <td>...</td>\n",
268 "     <td>...</td>\n",
269 "     <td>...</td>\n",
270 "     <td>...</td>\n",
271 "     <td>...</td>\n",
272 "     <td>...</td>\n",
273 "     <td>...</td>\n",
274 "     <td>...</td>\n",
275 "     <td>...</td>\n",
276 "     <td>...</td>\n",
277 "     <td>...</td>\n",
278 "     <td>...</td>
```

CHAPTER 5

Discussion and Conclusion

5.1 Git Hub Link of the Project:

https://github.com/Hemanthsubramaniyan/Hemanth-au810021102015.git

5.2 Limitations:

- **Cold Start Problem:** Difficulty in recommending songs for new users or new songs with little data. This can be mitigated by using hybrid models and content-based filtering.
- **Scalability:** Serving recommendations to millions of users in real time.
- **Data Privacy:** Ensure user data privacy by using techniques like differential privacy or anonymization.

5.3 Future Scope:

In the future, we would like to try the following things:

1. Using audio signal (e.g. audio frequency) to recommend songs
2. Trying content-based algorithm
3. Trying Convolutional Neural Network
4. Making the recommender system a real-timesystem
5. Trying clustering techniques to recommend music.

5.4 Conclusion:

Designing Spotify's recommendation system requires a robust architecture that can scale to billions of users, continuously adapt to user preferences, and deliver personalized, relevant music suggestions. Leveraging a hybrid approach that combines collaborative filtering, content-based techniques, and deep learning will help in creating a dynamic, highly personalized experience. The system must also be capable of handling real-time data and updates, ensuring that the recommendations stay fresh and relevant.

REFERENCES

I. Cheng, P. F., & Chiang, M. F. (2019). A Novel Music Recommendation System Combining Collaborative Filtering and Social Media Analysis: A Case Study of Spotify. IEEE Access, 7, 79309-79320.

II. Wang, Y., Wang, C., & Zhang, Y. (2014). Collaborative filtering recommendation algorithm based on user trust and item ratings..

III. Bonnin, G., Jannach, D., Roy, R., & Marques, H. M. (**2014**). Automated generation of music playlists: Survey and experiments. ACM Computing Surveys (CSUR)

IV. Abdollahpouri, H., & Burke, R. (2018). Users' biases in the evaluation of recommendation algorithms.

V. Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-n Recommendation tasks. In Proceedings of the fourth ACM conferenceon Recommender systems.

VI. Wang SL, Wu CY. Application of context-aware and personalized recommendation to implement an Adaptive ubiquitous learning system.

VII. Galant, S., 2020. Data Spotlight: How Spotify Wrapped Makes Music Data Feel Personable. Retrieved December 3, 2020, from Springboard website.

VIII. Bateira, J.L.: Spotify-ed-music recommendation and discovery in spotify (2014).

IX. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation Algorithms. In: Proceedings of the 10th international conference on World Wide Web. Pp. 285–295. ACM (2001)