**MUSIC RECOMMENDATION SYSTEM THROUGH**

**MACHINE LEARNING ALGORITHMS**

**Acknowledgment**

I want to express my sincere gratitude to everyone who contributed to the successful completion of this research. Firstly, I am deeply thankful to my supervisor for her invaluable guidance, insightful feedback, and unwavering support throughout this study. I also wish to thank my family and friends for their continuous encouragement and understanding during this process. Lastly, I appreciate all my peers and experts who contributed their time and knowledge, making this research possible.

Thank you

**Declaration**

I at this moment declare that this research is my work, conducted independently and without unauthorized assistance. All sources and references have been appropriately cited, and any direct contributions from other individuals or sources have been duly acknowledged. The findings and conclusions presented are based on rigorous analysis and reflect my original contributions to the field. This work adheres to academic integrity standards and ethical guidelines in research and reporting.

**Abstract**

The main theme of this dissertation is to develop a novel machine learning approach for recommending musical pieces to enhance user interactions. The fundamental goal is to use several different machine learning approaches to identify user preferences and afterwards recommend songs successfully. The classification models compared throughout the study are Logistic Regression, Random Forest, SVM, KNN, Gradient Boosting, and XGBoost classifiers that aim at identifying user preferences regarding certain tracks. Its problems like class imbalance in the given dataset were identified during the work and are reflected as the inability to achieve high accuracy in the classification of freely available, but less popular songs. That is why, approaches such as SMOTE and changing the weight of classes were discussed as possible solutions to this problem. The dissertation is divided into five chapters, which describe the main components of the system. Chapter One gives an overview of the study and entails aim and objectives, research questions and the rationale for the study. Chapter Two presents and discusses the related work in using music recommendation systems and recommendation algorithms, in addition to recommendation approaches. Chapter Three is about the method, where this study talks about how data is collected, preprocessed, how models are selected, and how Models are evaluated. Chapter Four gives the results and discussion in more detail, and discusses the analysis in light of the accuracy assessment and precision, as well as the flexibility in which users can interact with the applications. Last, Chapter Five focuses on the system design and deployment, as well as algorithm comparison, system efficiency, and further steps toward refining recommendation reliability and increasing the usability of the system. This work presents some recommendations on how the model will be further enhanced in addressing issues such as the imbalance of data and the further flexibility of the system as the user completes more transactions.

**Table of Contents**

[Chapter 1: Introduction 8](#_Toc178696324)

[1.1 Background 8](#_Toc178696325)

[1.2 Aim and Objectives 8](#_Toc178696326)

[1.3 Research Questions 9](#_Toc178696327)

[1.4 Significance of the Study 9](#_Toc178696328)

[1.4.1 Contribution to the Field of Artificial Intelligence and Machine Learning 9](#_Toc178696329)

[1.4.2 Impact on the Music Industry 10](#_Toc178696330)

[1.4.3 Broader Societal Implications 10](#_Toc178696331)

[1.5 Structure of the Dissertation 11](#_Toc178696332)

[1.6 Current Issues 11](#_Toc178696333)

[1.7 Summary 12](#_Toc178696334)

[Chapter 2: Literature Review 13](#_Toc178696335)

[2.1 Introduction 13](#_Toc178696336)

[2.2 This Attic Analysis 13](#_Toc178696337)

[2.2.1 Collaborative Filtering Theory 13](#_Toc178696338)

[2.2.2 Content-Based Filtering Theory 14](#_Toc178696339)

[2.2.3 Hybrid Systems Theory 14](#_Toc178696340)

[2.2.4 Theory of machine learning in recommendation 14](#_Toc178696341)

[2. 3 Key Concepts 15](#_Toc178696342)

[2.3.1 Collaborative Filtering 15](#_Toc178696343)

[2.3.2 Content-Based Filtering 16](#_Toc178696344)

[2.3.3 Hybrid Systems 17](#_Toc178696345)

[2.3.4 Machine Learning in Music Recommendation 17](#_Toc178696346)

[2.3 Challenges in Deep Learning for Music Recommendation System 18](#_Toc178696347)

[2.4 Future Directions 18](#_Toc178696348)

[2.5 Summary 19](#_Toc178696349)

[Chapter 3: Methodology 20](#_Toc178696350)

[3.1 Introduction 20](#_Toc178696351)

[3.2 Research Design 20](#_Toc178696352)

[3.3 Data Collection 21](#_Toc178696353)

[3.4 Dataset Description 22](#_Toc178696354)

[3.5 Machine Learning Algorithms in Music Recommendation 22](#_Toc178696355)

[3.6 Feature Extraction Techniques 23](#_Toc178696356)

[3.7 Evaluation Metrics 23](#_Toc178696357)

[3.8 Existing Solutions and Their Limitations 23](#_Toc178696358)

[3.9 Problem Statement 24](#_Toc178696359)

[3.10 Discussion of Issues 24](#_Toc178696360)

[3.11 Summary 25](#_Toc178696361)

[Chapter 4: Result and Discussion 26](#_Toc178696362)

[4.1 Introduction 26](#_Toc178696363)

[4.2 Critical analysis 26](#_Toc178696364)

[4.3 Findings 27](#_Toc178696365)

[Importing Libraries and Reading Data 27](#_Toc178696366)

[Data Exploration and Feature Correlation 28](#_Toc178696367)

[Exploratory Data Analysis (EDA) 29](#_Toc178696368)

[Sound features over time 30](#_Toc178696369)

[Correlation Analysis 31](#_Toc178696370)

[Clustering Analysis 32](#_Toc178696371)

[Model Evaluation 33](#_Toc178696372)

[Recommender System Development 38](#_Toc178696373)

[4.4 Discussion 39](#_Toc178696374)

[4.5 Summary 40](#_Toc178696375)

[Chapter 5: Conclusion and Recommendations 41](#_Toc178696376)

[5.1 Conclusion 41](#_Toc178696377)

[5.2 Recommendations 41](#_Toc178696378)

[5.3 Linking with Objectives 43](#_Toc178696379)

[5.4 Limitations 44](#_Toc178696380)

[5.5 Future Work 45](#_Toc178696381)

[References 46](#_Toc178696382)

# Chapter 1: Introduction

## 1.1 Background

The topic of this research area drains into the field of Artificial Intelligence (AI) where the study will strive to establish how AI has built deep recommendation models of music with complicated structures. AI being an innovation is incorporated in all categories of the economy particularly in the entertainment industry to enhance the user experience. Of all the subfields of AI, machine learning is one of the most effective ones, in as much as it boosts the systems’ ability to learn, adapt to the user’s specific preferences, and offer suggestions.

This project intends to develop some high-level algorithms, which would have pre-emptively deployed intelligent solutions, in the management and analysis of a user’s music profile. It also makes use of its machine learning ability to be trained by the users’ listening behavior, to recognize the pattern for a more detailed description of tastes and preferences, that could be otherwise difficult. This process of learning makes the recommendation system have a high level of music recommendations since it recommends music that the user is interested in listening to hence creating a positive impression on the users.

From the results obtained, it has been identified that there is a great need to incorporate the aspect of listening behaviour since it will assist the system in offering the user the content that is most appealing and interesting to her. Again, because of the application of machine learning, the recommendation system might discover how to improve the system’s capability to recommend the sort of music that the users would like in the following days.

This approach is useful in the fine-tuning of large music recommendation systems. A more personal and entertaining listening approach is another teaching point as to how AI and machine learning are very useful in the manufacturing of music.

## 1.2 Aim and Objectives

***Aim***

The research aim is to design and implement a novel music recommendation platform-based

on rich machine-learning models to generate customized playlists for users based on their past listening profiles.

***Research objectives***

* To compare and decide about various applied machine learning algorithms for the effectiveness of the music recommendation system.
* To develop a recommendation system that could promptly augment with the user’s changing preferences.
* To measure the effectiveness of the generated recommendations based on numerical results.
* To check the effectiveness of the proposed recommendation system in terms of accuracy and computational cost with existing similar systems.

## 1.3 Research Questions

1. What machine-learning approaches are most suitable for this task in terms of precision and speed?
2. What has helped to model user listening history and preferences to enhance the efficiency of offers on the music in question?
3. What are the potential performance measures for music recommender systems and how can here be quantified?
4. What are the advantages and disadvantages of the proposed music recommendation system as compared to existing recommendation systems and how does it perform in terms of recommendation accuracy?

## 1.4 Significance of the Study

The value is derived from the potential of increasing the enhancement of the musical recommendation systems using machine learning algorithms. Since music users are slowly moving to the internet for their content, customization has as a vital factor of competitiveness for streaming services. Two research questions correspond to the current systems’ deficiencies in terms of flexibility and effectiveness of the recommendations and the recommendation system.

### 1.4.1 Contribution to the Field of Artificial Intelligence and Machine Learning

This work is to be placed in the field that is about Artificial Intelligence and machine learning and presents a new perspective on the application of these technologies in the context of music recommendation. Thus, even though AI has been implemented in almost all industries, the role of Artificial Intelligence in enhancing entertainment, and particularly music, is still not very clearly defined. Applying a variety of multiple-level machine learning approaches and algorithms that are concerned with certain content characteristics, this paper provides a contribution towards the comprehension of AI as one of the strong tools that help to introduce and respond to multiple-level user preferences in the content item. Moreover, this study's focus on the dynamic nature of user preferences addresses a critical challenge in the field: the suggestion of a system that would allow the character to grow dynamically in terms of ability levels. Another downside of the basic recommendation models is that here are rigid and cannot be customized, hence low user participation. Given that the presented approach relies on the learning process from the feedback of the users which might shift with time, this work has been considered to provide a new course of building a new degree of adaptiveness in the context of recommendation systems for becoming generally acceptable for many other domains other than music.

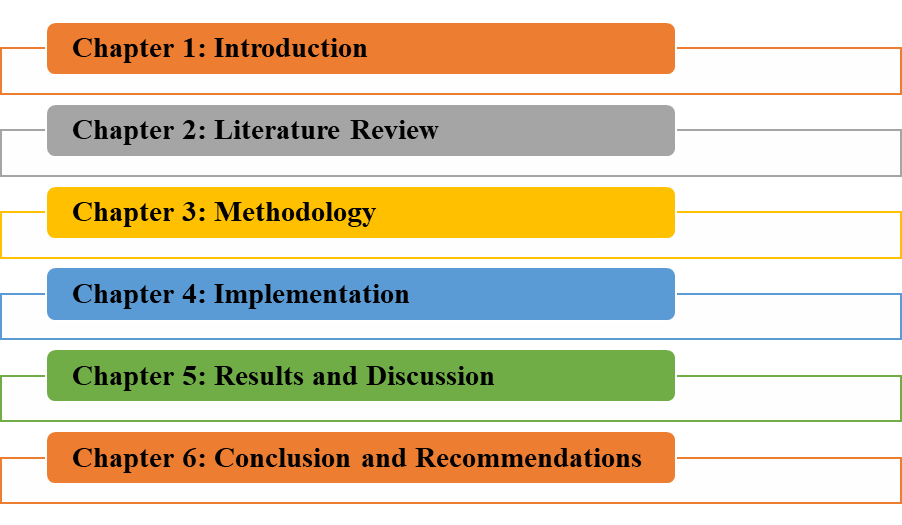
### 1.4.2 Impact on the Music Industry

The actual value of this study is not only in the academic circles, but it holds numerous deep potentials in the aspects of the music industry. As streaming has now taken over downloads as the means through which fans listen to their music, the battle for the soul, and the Evangelism Metric, begins. A recommendation system with a greater capacity to predict and track the preferences of the users in music not only increases the degree of satisfaction but also the permanence of the users, on which the expansion of the platforms that broadcast music fundamentally relies. It has thus helped streaming services offer a higher degree of precision and self-sufficiency in reaching out to total populations or the sub-populations of listeners according to their musical literacy and passion. It not only improves the quality of the experience the user has with the music but on the other hand it opens opportunities for artists to gain popularity since the system has suggested to the user a wide range of music, the new artists that would hardly be included into the ‘list of hits’.

### 1.4.3 Broader Societal Implications

On a broader scale, the study implications have perhaps generalized how other recommendation systems based on content are formulated in other industry fields. Similarly, as has been pointed out in the section, the approach of adaptive learning and user preference model has been used to enhance the personalization approach in the other domains: movies, books, and others. It makes the research round cross-industry applicable which is especially important given the growing trend in personalization which is considered in the research. Also, the study contributes to the problem of ethics in recommendation systems which involves the defense of users from specific content as well as protection from bias in the system. In that sense, the research is arranged to enable working on the construction of such systems that do not require users’ frailty to permanently occupy this, as many other products require today.

## 1.5 Structure of the Dissertation



**Figure 1.1: Dissertation Structure**

(Source: Self-created)

## 1.6 Current issues

Some challenges currently affect the efficiency of music recommendation systems. Data scarcity is another issue, given the fact that there are hundreds of millions of songs, although the number of interactions per song is relatively small, recommendation systems face the problem of data sparsity, which hinders the accurate representation of users’ interests. Another problem is a cold start, this means the service faces the challenge of recommending songs to new users as well as new songs for which it does not have any information based on which the songs have been recommended to the users.

The music preference of users strongly depends on the context and, therefore, the method that lessens the informative content of users’ preferences has been quite insufficient. This also involves the necessity of real-time flexibility about recommendations since users want recommendations to always be modern and up-to-date and this requires systems with minimum delays in their update. The question of privacy also comes into play when compiling and analyzing data for targeting and recommending certain products, which also has to be considered to maintain users’ trust. These problems raise awareness of the challenges in building efficient recommendation solutions for music and stress the importance of further research in this area.

## 1.7 Summary

This chapter begins with defining the contemporary significance of the concept of personalized music recommendation during the existence of the digital environment and presenting the role of AI in enhancing the users’ experience. Then, it moves to the background of the study and pinpoints that current systems are incompetent to learn the changing nature of preferences of the end-users. There is a research aim and objectives in defining the differences between different machine learning algorithms, establishment of the proposed adaptive recommendation system, and its efficiency. Given below are the research questions which are to serve as the foundation of the study concerning the suitability of the different types of machine learning techniques, the strategy for planning the user preferences model as well as the measures of performance. Here the paper’s relevance is described in a more precise manner, regarding how it might in any way enrich the theoretical understanding of AIs and how it might be implemented in the musical business.

# 

# Chapter 2: Literature Review

## 2.1 Introduction

The Literature Review acknowledges the literature on the existing studies on recommendation systems with a special focus on the ones that have used Machine Learning approaches within the sphere of music recommendation. This section deals with the justification of the study concerning AI and the objectives of the study present an example of an existing trend, approaches, and gaps of existing knowledge in the field. In the first part of the chapter, the authors present the trends to progress from simple rule-based systems for recommendations to the new generation of intelligent systems for recommendations. It then progresses to analyze the various subcategories of machine learning that have been applied in music recommendation and these are collaborative filtering, content-based filtering, and hybrid systems. Furthermore, there would be a review of the improvements made on the user preference modeling, paying attention to the dynamic and contextual data employed about the users to improve on the recommendation. The discussions made in the present chapter concerning the previous research analysis and the portrayal of the current systems’ inadequacy provide the research study in the field with the necessary base.

## 2.2 This Attic Analysis

The framework for music recommendation systems lies in the areas of information seeking, data mining, and modeling of consumers’ behavior. The main purpose of such systems is to make predictions regarding user preferences to improve the satisfaction obtained by the user of a given service (HIEN et al., 2024). This section examines the theoretical foundation of several of the identified concepts earlier namely; collaborative filtering, content-based filtering, hybrid systems, and machine learning techniques.

### 2.2.1 Collaborative Filtering Theory

The collaborative filtering theory that is employed presupposes that if two users have agreed in the past, here will agree in the future. It uses the total of the users’ inclination to suggest items. User-based collaborative filtering uses distances or similarities like correlation coefficients, essentially Pearson’s or cosine similarity to find users with similar ratings (DEGE, J. and SANG, 2024). Item-based, collaborative filtering on the other hand determines similar items according to the users. The main theoretical problem of collaborative filtering is the difficulty of processing sparse data and ensuring the applicability of big data.

### 2.2.2 Content-Based Filtering Theory

Content-based filtering theory is hypothesized based on principles of information retrieval and pattern recognition. It presupposes that products and/or services that share certain characteristics have to be relevant to the user (ANGAMUTHU and TROJOVSKÝ, 2023). In music recommendation, it encompasses the identification and extraction of parameters common with the audio signals such as type of music, the rate of beats per minute, and key. The system then builds a facilitating profile by analysing the previously liked products and then targets other similar products. The theoretical difficulty, in this case, is to capture the user’s preferences and update the profile as the preferences change.

### 2.2.3 Hybrid Systems Theory

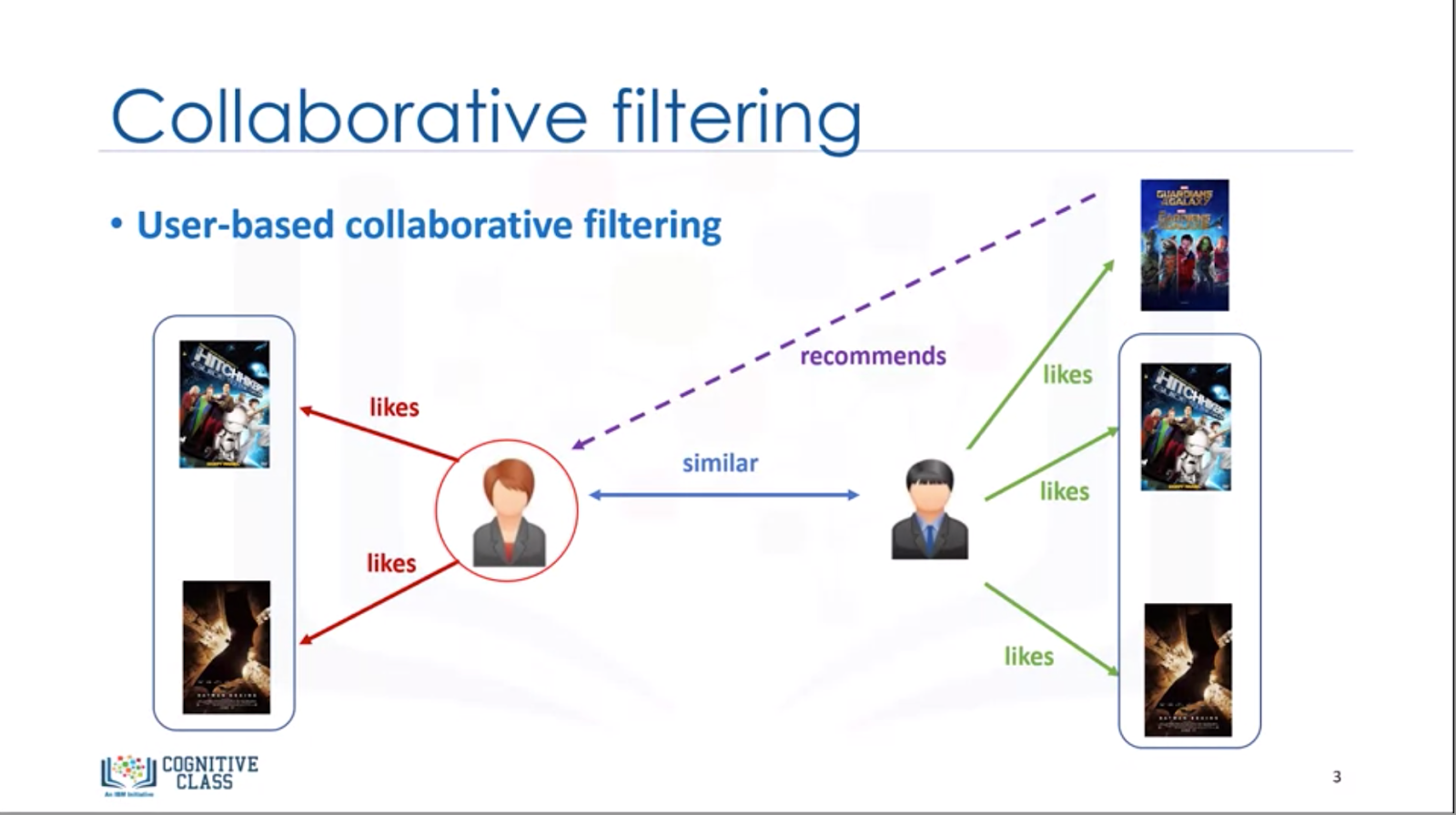
Hybrid recommendation systems theory incorporates several recommendation methods to improve performance. Thus, hybrid systems that include collaborative and content-based filtering have overcome the drawbacks of each of these approaches (BHASKARAN and MARAPPAN, 2023). The theoretical framework in the context of hybrid systems includes the development of algorithms that enable the combination of multiple sources of data as well as the combination of other recommendation methods. This has been done in various techniques including weighted hybrid, switching hybrid, and mixed hybrid techniques. Here lies the dilemma of trying to strike the right mix of the two techniques to optimize the recommendation quality and user-friendly nature of the recommendations.

### 2.2.4 Theory of machine learning in recommendation

Modern recommendation methods have been built on the advanced theory of machine learning. Structural decomposition techniques like Singular Value Decomposition (SVD) transform the user-item matrix into latent factors that depict the user’s and item’s preferences. Recent advancements in the field include the use of deep learning models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) which have been used to look into audio signals and user behavior to provide better recommendations (LIANG, 2023). These models have captured a lot of detail about the data which enables them to provide better and detailed recommendations. The main theoretical question in the context of applying machine learning to recommendation systems concerns what models have captured users’ preferences with reasonable accuracy given very limited information.

## 2. 3 Key Concepts

### 2.3.1 Collaborative Filtering

****

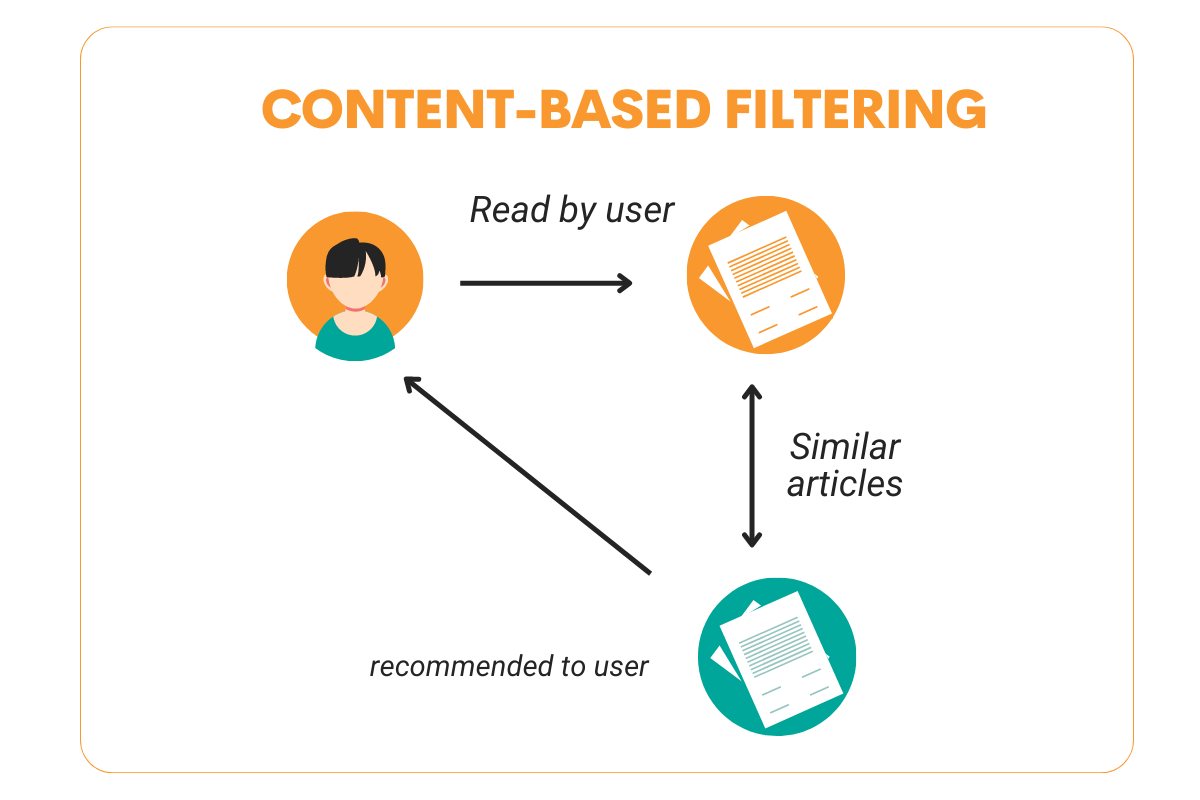
**Figure 2.1: Collaborative filtering**

(Source: LIU et al., 2024)

Recommendation based on the nearest neighbor, also considered as the most frequently used method in the recommendation systems is known as the technique called collaborative filtering. So, it works under the logic that if two users are similar in their tendencies, here have rated or enjoyed the same items. There are two main types of collaborative filtering, there are two models that are used namely the user-based and item-based. User-based collaborative filtering works based on the similarity of users’ preferences and has items, which had been liked by similar users as recommendations. The main problems with CF are, first, data sparseness where there could be a shortage of user-item interactions so that the recommendation for users and items could be off including the problems like the cold start problem where there is not enough information about a new user or an item to recommend to

him/her or the item.

### 2.3.2 Content-Based Filtering

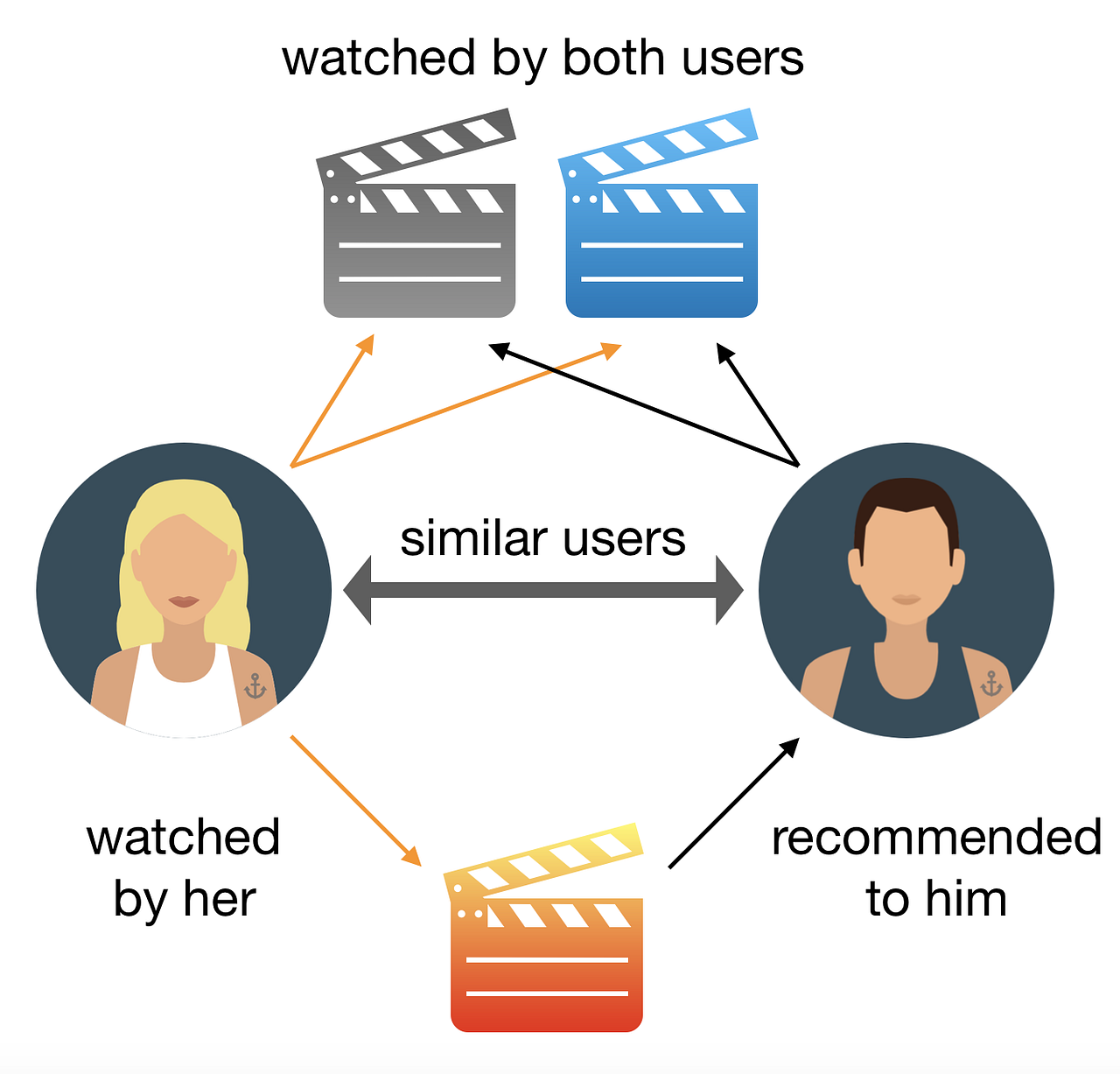
****

**Figure 2.2: Content-based filtering**

(Source: JAISWAL et al., 2024)

Content-based filtering delivers recommendations driven by the items’ qualitative traits. In the case of music recommendation, this has been applied to the current and past songs, tempo, key, and instrumentation of the song, and even other songs of the same artist (LIU *et al*., 2024). The system proceeds to suggest other items with comparable attributes to the ones that the user has favorably endorsed in the past. Another advantage of content-based filtering is independence from other users’ data, which enables the approach to solve the cold start problem. However, it has the disadvantage that it is not very flexible for capturing more subtle details of how users’ preferences develop over time because it is heavily dependent on known item characteristics.

### 2.3.3 Hybrid Systems

****

**Figure 2.3: Hybrid filtering**

(Source: JING, 2024)

The hybrid recommendation systems consist of the integration of two or more recommendation approaches to take advantage of the superior features of each while avoiding the disadvantages of others. Combining both the collaborative filtering and content filtering techniques, hybrid systems are capable of giving better and more reliable recommendations. For example, a combined system could apply Cf to find similar users and then make use of Cb to filter the results based on the characteristics of the items. This approach has effectively solved the issues of data sparsity and the cold start at the same time while recommending the personal user’s items based on the users’ behavior and node characteristics.

### 2.3.4 Machine Learning in Music Recommendation

It must be noted that machine learning has greatly enhanced the field of music recommendation. To improve the recommendation’s accuracy and specificity, matrix factorization, Neural networks, and deep learning techniques have been adopted. Collaborative filtering methods like Singular Value Decomposition (SVD) perform the matrix factorization of the user-item interaction matrix into low-ranking factors that characterize the user preferences and items (JAISWAL *et al*., 2024). There have been recommendations based on the analysis of audible signals and other user activities through the employment of neural networks, deep learning varieties being CNNs and RNNs. Compared to previously used models, these models have found a much more detailed pattern in the offered data and thus give much more individual recommendations.

## 2.3 Challenges in Deep Learning for music recommendation system

Machine learning has been enhanced by deep learning in the area of music recommendation systems and they pose several more issues. A major disadvantage is the requirement for big sample sizes and high-quality data for effective use of big data methods. As with many other deep learning models, deep learning models for recommending songs need substantial amounts of training data to capture complex patterns in users’ preferences and musical features well. However, the accumulation of such datasets and their cleaning could be costly and are often associated with privacy issues related to users. In cases where a user has a low listening history, it becomes difficult for the system to make recommendations since there is inadequate information upon which to base a hypothesis of the user’s likes (Fayyaz *et al.* 2020). Another important issue is the interpretability of the model since the resulting system is expected to have a transparent explanation of its decision-making process. The models trained through deep learning are black-box models that do not allow for easy identification of why a given recommendation was made. This lack of transparency may create concern from the public hence reducing their level of trust in the system. However, equal representation is also important when it comes to recommendations as well. This could make users disengaged if the system is always recommending the same type of music which causes no diversification. However, the issue of providing at the same time individualized suggestions and a fairly broad choice remains a challenge.

## 2.4 Future Directions

Recommendation systems of music will be dominated by the extensive use of more sophisticated machine learning approaches, especially deep learning and reinforcement learning. Given the complexity and multidimensionality of considered user preferences, recommendation systems have to be developed further and more successfully. An active area of further research is to take a multi-dimensional approach and implement systems that simultaneously use techniques such as collaborative filtering, content-based filtering, or deep learning to generate even more accurate and specific recommendations.

Also, with the help of reinforcement learning, it is possible to predict the current and instant user interactions and adapt to this to constantly improve the corresponding recommendations. This approach could a great extent improve user engagement, as the suggestions made will be more relevant and sensitive to the specific context the user is in.

Another is the lack of ethical understanding as a future direction concerning how algorithms that generate recommendations secure against bias or aspect of diversity in recommended pieces of music. Appropriate incentives should be made for researchers and developers to design algorithms that not only improve the functionality of the platform based on users’ satisfaction but also enrich the users with more selection of genres and artists.

## 2.5 Summary

The literature on music recommendation systems is vast and contains a plethora of methods and strategies that are used in an attempt to improve the quality of recommendations. Collaborative Filtering, Content-Based Filtering, and hybrid system all have their respective pros and cons, or, in other words, have different opportunities and threats. Machine learning has taken this one step further, giving the ability to model intricate patterns that involve the data of the users. However, several research limitations have been addressed saying there is a lack of adaptive recommender systems, strategies for dealing with the cold start problem are inadequate, evaluation metrics that capture the users’ preferences are missing, and there is a

concern regarding privacy and security.

In this literature review, the common terminology, theoretical frameworks, and existing research areas of the subject, music recommendation systems, have been discussed. The subsequent chapters of this research could intend to extend from these findings concentrating on the creation of a new recommendation system that has implemented advanced artificial intelligent models to tackle the highlighted deficiencies &amp; improve the recommendation experience. The techniques and aims of this study are based on three filtering methods: collaborative, content-based, and machine learning, to propose an accurate, secure, and adaptable music recommendation system.

# 

# Chapter 3: Methodology

## 3.1 Introduction

Recommendation systems, in connection with music in particular, have also shifted significantly in the last couple of decades. In the past, such systems that were known of incorporated a manual documented catalog and some simple rules into recommending songs. As soon as collections of digital music started to appear, there was a need to use more advanced methods of profiling (KLEĆ et al. 2023). Incorporated techniques facilitated the early systems of hypertext and included the collaborative filtering technique based on users' behaviors and tendencies. Subsequently, over time there have been content filtering exercises, done on the properties of the music that include genre, tempo, and hence mood among others. Contemporary approaches to perform smart recommendations for music include matrix, content-based, as well as extra factors.

## 3.2 Research Design

Despite the study goals and research questions being clear and specific, the research design for this study is as follows: A structured method is used where both experimental and comparative research paradigms are utilized to investigate the performance of different algorithms as a way of providing customized music streaming.

First, the paper starts with an analysis of the literature in an attempt to outline the most suitable machine learning algorithms to be used in the recommendation systems. Based on this review, preprogrammed algorithms for testing are inclusive of collaborative filtering, content-based filtering, and hybrid filtering. However, the recent outstanding performance of deep learning techniques, such as neural networks, in several high-dimensional data makes this an option of consideration.

The idea of the research design lies in the integration of these algorithms into a recommendation system developed from scratch. Modularity allows the integration and testing of this different algorithm The system is intended to be modular. This modularity also serves well in the case of comparing the accuracy and computational complexity of each of the algorithms.

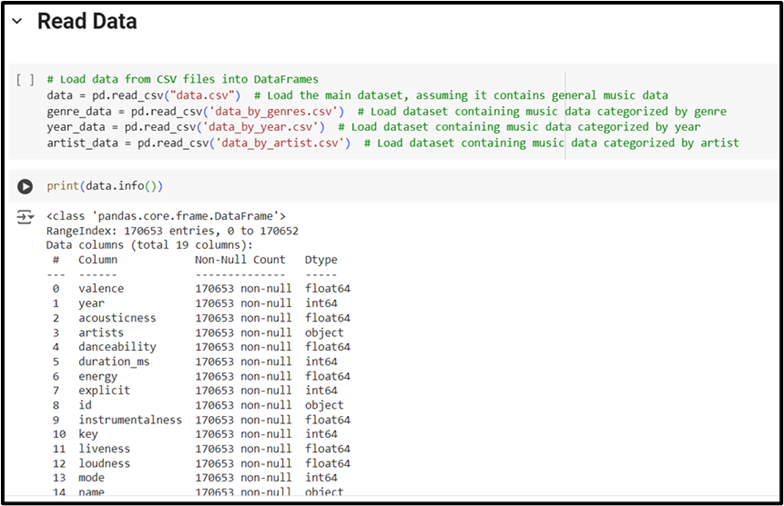
As part of the enhanced recommendation system testing, the research design conceived entails a feedback loop to track user preferences to incorporate this properly within the system. This real-time adaptation is necessary for assessing the effectiveness of the real-time adaptation to users’ shifting preferences which is important for attaining the objective of high users’ satisfaction.

In the initial process, the research has planned to consider both the numerical and user feedback results to generate recommendations for the users. However, the implementation of interviews and survey for the collection of user feedback has required the ethical approval and it has created complexity in the successful project management. Due to this reason, this research design has only considered numerical results for the generation of recommendations.

Another feature of the study is the mixed evaluation approach that is based on the use of quantitative coefficients. The evaluation criteria include accuracy, precision, recall, as well and computational cost to give a clear evaluation of each algorithm.

Last, but not the least, a contrast in the proposed system with existing music recommendation systems is also a part of the research design. The purpose here is to make a comparison to demonstrate the strengths and weaknesses of the proposed system so that the overall performance of the proposed system will be understood in terms of a benchmark that is relevant to the industry.

## 3.3 Data Collection

The first process that should be carried out in creating the music recommendation system is data gathering. The main data source for this project will be Kaggle, which contains multiple CSV files filled with song data. Such datasets are comprised of data on audio attributes, music type, year of creation as well as users’ preferences. 

**Figure 3.1: Data Collection and Preprocessing**

(Source: Acquired from Google Colab)

Data gathering should also be accurate as it provides the solution to build reliable models in machine learning. After that, comes the preprocessing phase of the data that have been collected. This entails data cleaning to deal with various factors such as missing data and other forms of data inconsistencies.

## 3.4 Dataset Description

The data source of this study is several CSV files containing different attributes and characteristics of the music tracks. The main dataset, data. Some of the basic attributes of the dataset include acousticness, danceability, energy, instrumental Nѕ, liveness, loudness, speсkiness and tempo, valence or happiness, duration in milliseconds, explicit content or ‘lyrics indicating sexual content’, the key and mode, and the year of release of the track as well as the popularity score of each track. Additional datasets provide more specific insights: finally, the data shall be analyzed based on the genre that the data falls in. csv splits the track by genre and there is one more column – data\_by\_year. csv distinguishes data by Release year and data by artist. The tracks can be split into sub-directories by artist using csv. Such datasets open up opportunities to investigate the tendencies of musical preferences, the contours, specific peculiarities, and overall and particular characteristics of musical pieces belonging to certain genres and the distribution of the latter from the viewpoint of their popularity. It helps in cluster and classification to the investigation of musical characteristics and prediction of the track preference depending on characteristics and tags.

## 3.5 Machine Learning Algorithms in Music Recommendation

Artificial intelligence methods are relevant in modern procedures used in music recommendatory systems. The methods based on matrix factorization and neighboring are used quite actively including collaborative filtering. SVD and other types of methods for matrix factorization predict the user preference for the items the user has never visited by decomposing the user-item matrix into latent factors. The related approaches considering the concept of neighbors are, for example, k-nearest neighbors (k-NN), which make recommendations according to the analogy of similar users or similar items (BUTKOWSKI et al. 2024). Algorithms of content-based filtering operate with the analysis of a song’s features in its audio to give suggestions. The buoyant decision trees, the SVMs, and neural networks have also been employed to recognize the relationship between the sounds as well as the features preferred by the users.

## 3.6 Feature Extraction Techniques

Feature extraction is still one of the most important stages in the creation of various technology of recommending in the musical sphere. Among the features, tempo, danceability, energy, and valence are crucial for providing a listener with the informativeness of the given song. Such attributes have been obtained with the use of tools that have worked with audio data such as the Librosa library in Python which offers various functions of signal processing. The most commonly used types of feature vectors and recognized parameterization techniques include the MFCCs, which express the characteristics of short-term power spectrums of a sound and define the timbre of music. Such other attributes include chroma vectors which speak of uniformity or dissimilarity in the grouping of notes, spectral contrast which provides probability information on a song’s harmonic content, and tones which depict the orientation of harmonic structures. First, there is descriptive metadata that contributes to the search by an artist, the genre, the year of release, and relevant lyrics which are also good for content-based filtering.

## 3.7 Evaluation Metrics

The evaluation of the given music recommendation systems has the following indices of effectiveness. As performance parameters to evaluate the accuracy of the recommendations given by the system, we have the following; Precision Recall and F1 Score. Precision is the percent of recommended records, that are relevant, while recall is the percent of relevant records out of all records recommended (LIU et al. 2024). The F1-score is the average of the precision and the recall, which means that the mean of the two key accuracy measurements is the most favorable result. Other valuable measures are mean reciprocal rank (MRR) where the first relevant item falls in the list of the recommendations; and normalized DCG where the position of the relevant items in the list of the ranks is of great importance.

## 3.8 Existing Solutions and Their Limitations

In the present scenario, all the potential functioning music streaming active services like Spotify, Apple Music, and Pandora have incorporated sophisticated recommendations. Spotify products include App Recommendations, Discover Weekly, and Daily Mix are examples of collaborative filtering and deep learning mobile traffic. Apple Music for the first time has party curators and recommendations based on the users’ listening habits while Pandora was based upon the Music Genome Project in which many songs were tagged. Still, there could be a point in analyzing the treatments that are in place in cases when such solutions are not necessarily devoid of their flaws. All the systems are nearly blanketed with a cold start problem where new users or items have little data or information to handle.

## 3.9 Problem Statement

Digital music has extended infinite potentialities for its users and encircles this with an infinite number of songs that are not allowed an opportunity to look for songs here would probably like. The currently available music recommendation systems though useful in some occasions become a disappointment because of the poor performance of the system in taking aspects of the users’ tastes and preferences into consideration. Thus, the present study aims to fill this gap by providing a recent intelligent music recommendation model that applies algorithms. The proposed system will be capable of learning from the recent data obtained from users’ preferences concerning music streaming and combining the learned data into user-specific playlists to improve the applicability of the system.

## 3.10 Discussion of Issues

|  |  |
| --- | --- |
| **Issue Category** | **Description** |
| **Ethical Issues** | * **User Privacy:** Safeguarding personal data and listening habits is crucial to maintain user trust. * **Data Usage Transparency:** Users should be informed about how their data will be used and have the option to opt-out. * **Bias in Algorithms:** Ensuring that recommendation algorithms do not inadvertently favor certain genres or demographics over others. |
| **Legal Issues** | * **Data Protection Regulations:** Compliance with laws such as GDPR, which governs the collection and processing of personal data in the EU. * **Intellectual Property Rights:** Ensuring that music data used for training does not violate copyright laws. * **Licensing Agreements:** Proper licensing for the use of music content in the recommendation system. |
| **Professional Issues** | * **Accountability:** Developers must ensure the reliability and accuracy of the recommendation system. * **Continuous Improvement:** Ongoing updates and evaluations of the system to meet evolving user needs and technological advancements. * **Collaboration:** Working with diverse teams to incorporate multiple perspectives in the algorithm development process. |
| **Social Issues** | * **Access to Music:** Addressing disparities in access to music streaming services among different socioeconomic groups. * **Cultural Sensitivity:** Recommendations should respect and represent diverse musical traditions and cultures. * **Impact on Music Discovery:** Balancing personalized recommendations with opportunities for users to discover new and less mainstream music. |

**Table 3.1: Overview of ethical, legal, professional, and social issues**

(Source: Self-created)

## 3.11 Summary

The process of designing and assessing the music recommendation system based on the machine learning approach is described. It starts with a clear description of the general research approach; this work opted for the experimental and comparison method in evaluating the performance of various algorithms. The chapter then moves into giving details on the nature of the dataset employed, and its applicability to the intended research objectives. The choice of machine learning algorithms is then talked about under the justification which states the type of machine learning algorithm chosen is relevant to the research questions and objective of the study. The recommendation system design and how the recommendation system is built are discussed by explaining the steps of data cleaning, preprocessing, training the recommendation system models, and integrating this into the existing recommendation system.

# Chapter 4: Result and Discussion

## 4.1 Introduction

This chapter is a detailed elaboration of data analysis, feature engineering, and modeling used in the construction of the music recommender system. It first imports other necessary libraries like Pandas, and NumPy and visualization tools like Matplotlib and Seaborn that create a good structure of the music data flow in the system. Parameter, including acoustic, danceability, and energy were undertaken to determine its impact on the popularity of songs. Descriptive statistics techniques were used to depict distributions of music characteristics over time, while feature entailment was done to examine the effects of these features on popularity.

## 4.2 Critical analysis

Several steps have been imperative in the model development of the music recommender system concerning data acquisition, processing, modeling, and assessment. Thus, the approach used is methodologically reasonable, although several significant topics require further examination.

Firstly, the identification of characteristics like acoustics, danceability, and energy for algorithm calculation of song popularity has been quite informative since they more or less relate to user interaction. However, it is noteworthy, that the decision concerning the target variable, popularity tends to be somewhat narrow, it can not precisely reflect the preferences of consumers, which are perhaps weakly deterministic and would be more adequately described by such complex variables as frequency of listening or individual feedback (Sulthana *et al.* 2020). Furthermore, the K-Means clustering for the genre-based, while useful in aggregating similar tracks, does not accurately consider the dynamicity of the generation of new sub-categories of genres, which would halve a lowering impact on the precision of highly specific recompositions.

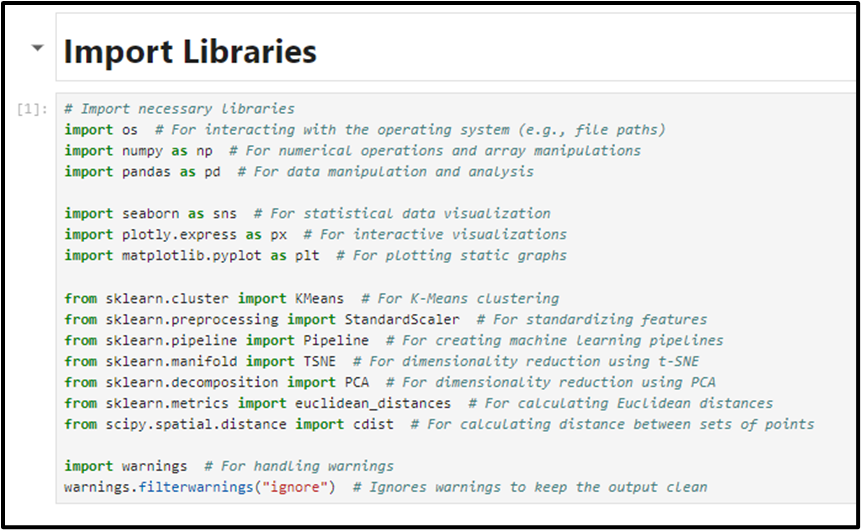
From a modeling perspective, the decision to augment our Logistic Regression model with Random Forest, SVM, KNN, Gradient Boosting and XGBoost provided a comfortable mean classification rate. Random forest, SVM and XGBoost performed reasonably well while Logistic Regression was worst affected by class imbalance primarily concerning instances of class 1. This created a skewed distribution in favor of the prediction of the popular songs because Logistic Regression did not consider the misclassification cost of such instances. In case of imbalances, Gradient Boosting demonstrated potential in addressing the problem through boosting but it was also not effective in modeling the minority class. KNN despite being basic and successful in known scenarios using distance-based voting and worst with imbalanced data was not as great on this data set. As such, it is suggested that further application of the strategies such as the oversampling techniques that include SMOTE on the minority class can significantly enhance the prediction accuracies. Such approaches would enable models to improve on the features and make a distinction between those tunes that are rarely played and the rest, improving the rating of classification.

Also, an enhancement of the rule-based recommendation system explains the possibility of customizing the recommendations based on the genre, artist, or mood selections (Xu *et al.* 2021). Although practical, this approach is not heavenly complex enough regarding utilizing such deep learning techniques as neural collaborative filtering that might offer even more precise recommendations. Hence, further enhancement should focus on enhanced recommendation technique, learning algorithms, and the problem of data splitting to build a more realistic solution that can respond effectively to the user’s activities and preferences.

## 4.3 Findings

### Importing Libraries and Reading Data

These fundamental libraries out of which the most used are numpy, pandas, seaborn, plotly, matplotlib, and many from the scikit-learn for machine learning and data arrangement.



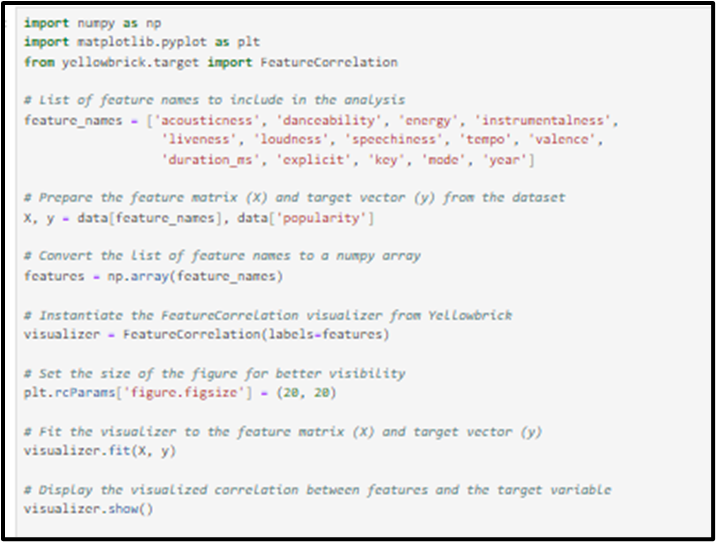
**Figure 4.1: Import Library**

(Source: Acquired from Jupyter Notebook)

These libraries are used to manage data, perform and visualize alterations as well as apply techniques of artificial intelligence. All the general information of music has been put into Pandas DataFrames for the overall treatment by categorizing it according to genre, year, and artists.

### Data Exploration and Feature Correlation

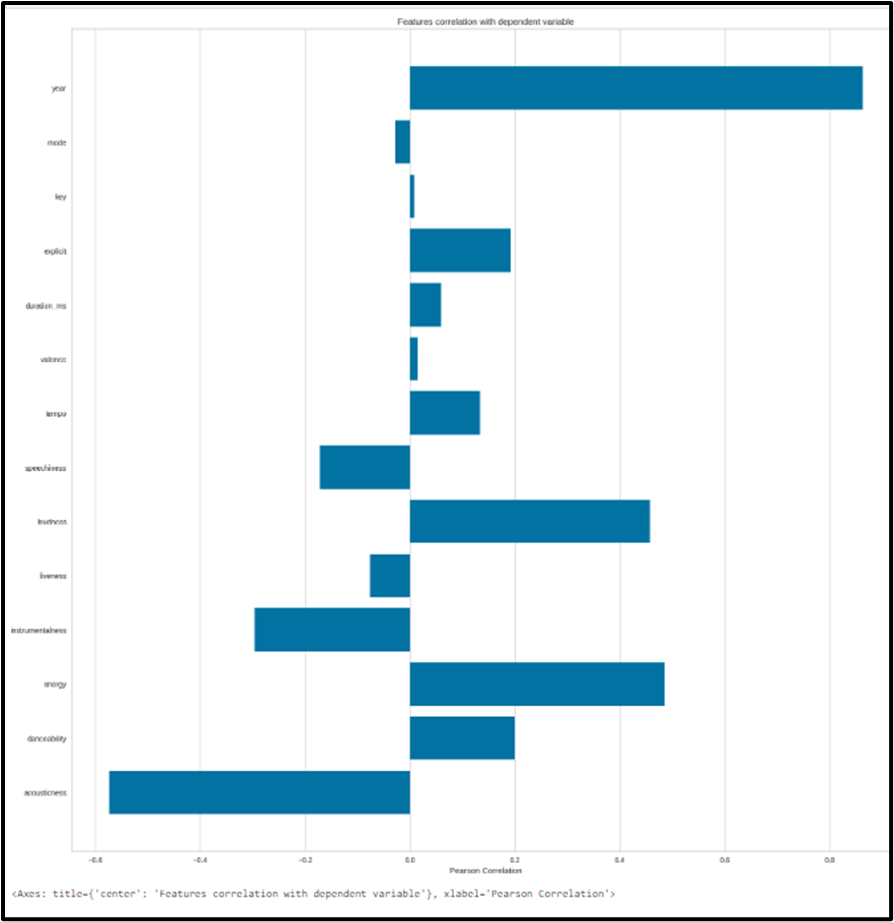
The first set of features similar to those that have been used in the analysis of music for instance; acoustic, danceability, energy, and also popularity among others were established.



**Figure 4.2: Feature correlation analysis**

(Source: Acquired from Jupyter Notebook)

Using Yellowbrick’s FeatureCorrelation tool it has considered how strong the features were and how here correlated to the dependent variable, popularity.



**Figure 4.3: Feature Correlation visualization**

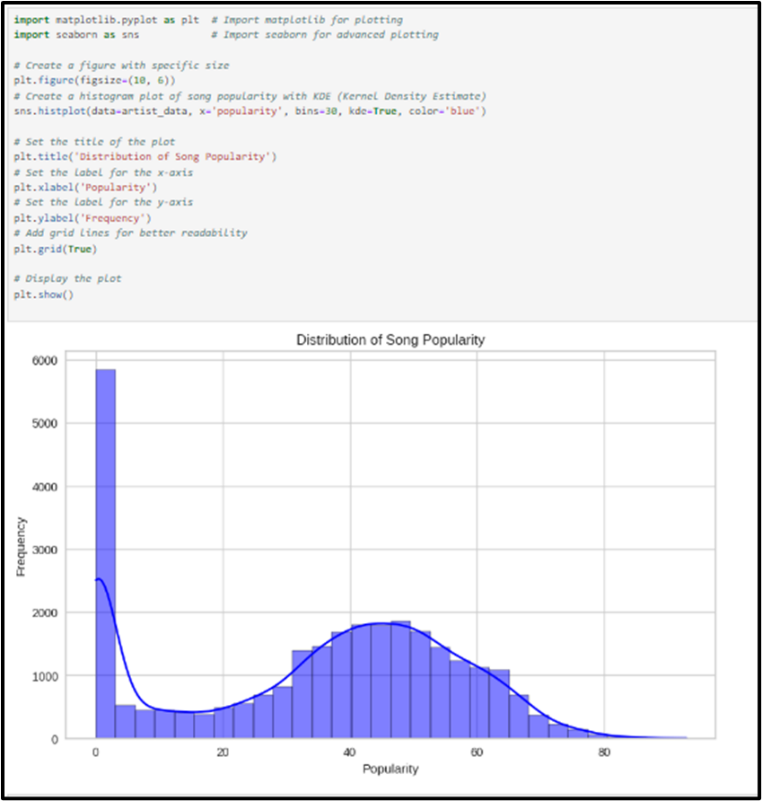
(Source: Acquired from Jupyter Notebook)

The visualizations highlighted great agreement between energy and Danceability with popularity

as it is helpful to determine the features of the recommendation system.

### Exploratory Data Analysis (EDA)

Exploratory Data Analysis commonly known as Data profiling has also been carried out to get inside of the trend of the data. If given, a line plot illustrating how the sound features of the songs changed from year to year in relativity to acoustics, danceability, energy, and so on. Apart from that, a bar plot compared the details of the top 10 genres concerning valence, energy, danceability, and acoustics.

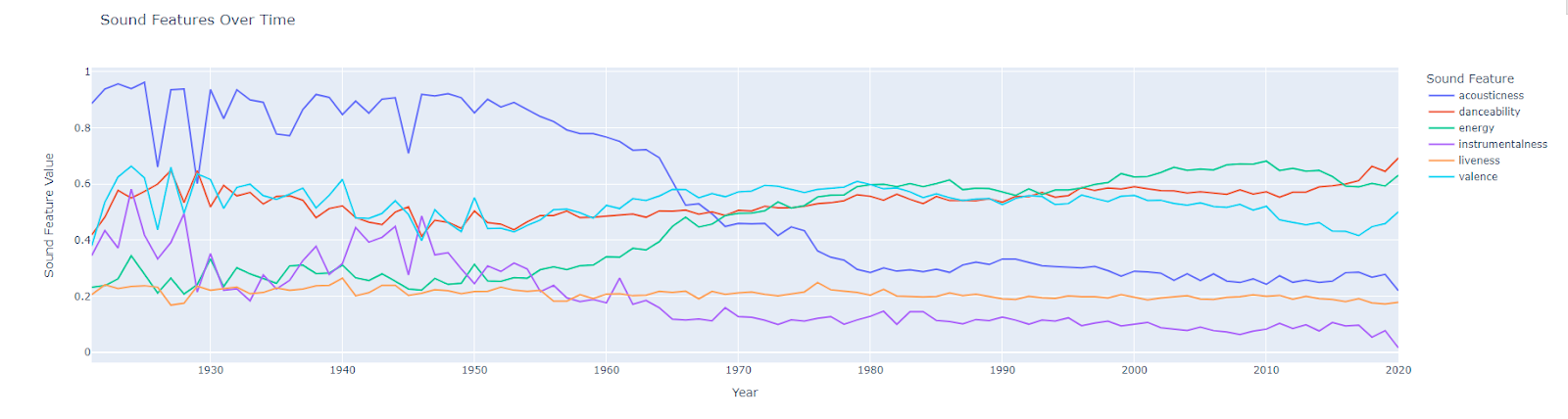


**Figure 4.4: Distribution of Song Popularity**

(Source: Acquired from Jupyter Notebook)

Another kind of the abundance of songs’ popularity has been presented in the form of a histogram with the Kernel Density Estimate (KDE) line lying under it. As has been observed in the given analysis, songs from the middle chart position are dominant in the samples, which is crucial for effective recommendations.

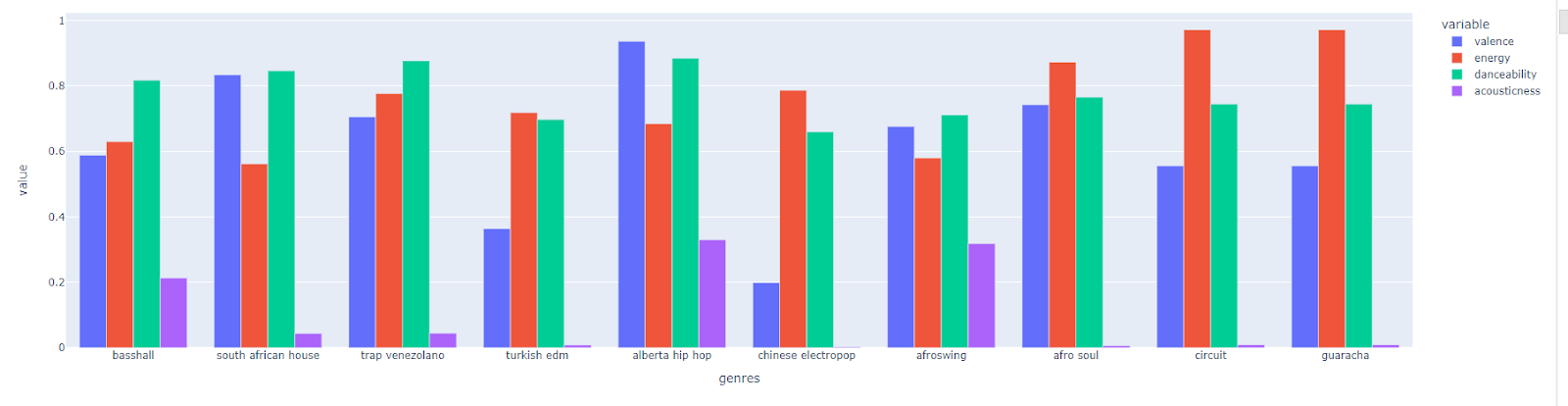
### Sound features over time



**Figure 4.5: Sound features over time**

(Source: Acquired from Jupyter Notebook)

Both the chart and the annotations identify trends in terms of various sound features of music in the given period of 1920, 1950, 1980, and 2020. Explaining the degree to which the music is acoustic, that is, how acoustic the music is, it decreases significantly more in year 1960 and below. Danceability, that quantifies how well a music piece can be danced, shows less fluctuations and a rise after the year 2000. More than complexity and consonance, energy, which reflects the intensity or level of activity of a song, rises but with some variations from the 1950s. Instrumentalness which measures the amount of instrumentation in the track slowly declined after the 60s implying a smaller number of songs with just instrumental components. Liveness, which measures the actual chance for a live performance, remains comparatively constant across the decades.



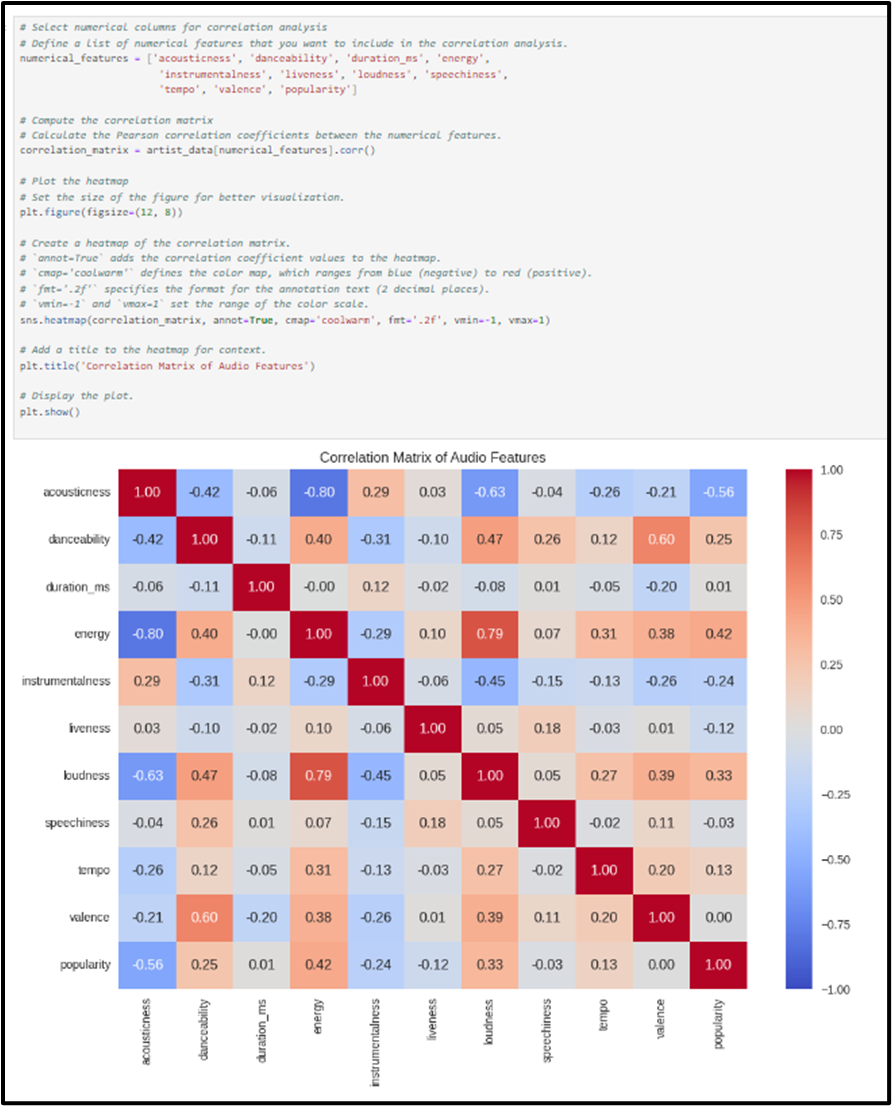
**Figure 4.6: Top 10 generes**

(Source: Acquired from Jupyter Notebook)

In the above bar chart, four defined abstractions of sound are described: valence, energy, danceability, and acoustics concerning different genres of music. The happiness level within each genre, represented by valence, still shows moderate to high figures, with circuit and guaracha enjoying the highest scores. Energy as the intensity of the music differs much; it is higher in Alberta hip hop, Afroswing, and Bass hall, while trap Venezolano and Turkish EDM are less energetic. Danceability, which measures how suitable a song is for dancing, follows a relatively high rating in all forms of music; more so, in Basshall, South African house, and circuit genres.

### Correlation Analysis

For the numerical features, the correlation matrix was calculated, and the results were presented in the form of a heatmap.



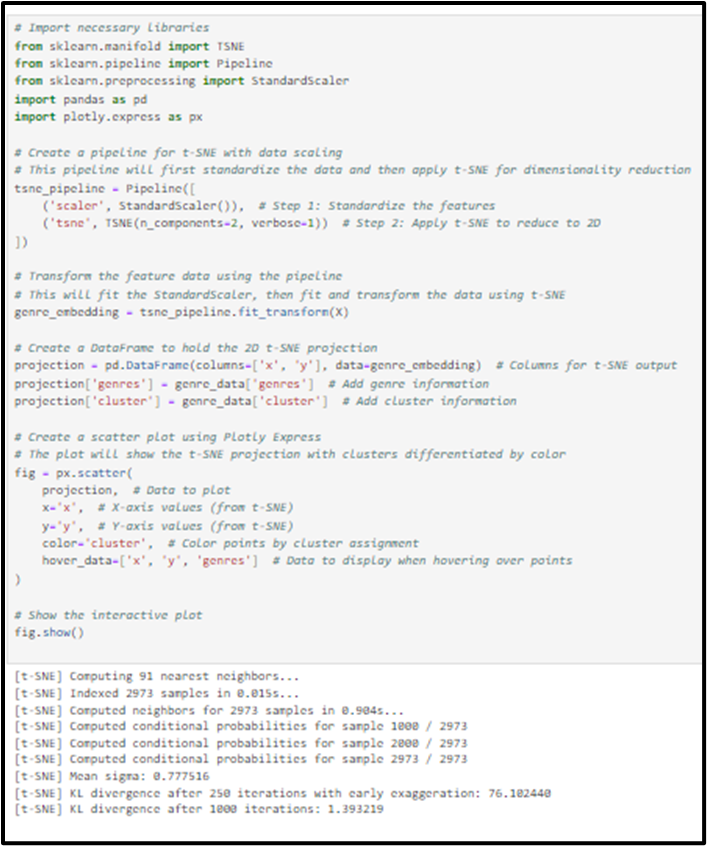
**Figure 4.7: Correlation Analysis**

(Source: Acquired from Jupyter Notebook)

This is possible as one has indicated the relation between features such as there is a positive relation between energy and loudness and a negative relation between the acousticness and energy features. Such correlations are very much a part of the process of feature selection along with the tweaking of the existing models to get the optimal predictions.

### Clustering Analysis

For partitioning of the data, there is one of the most important techniques known as the K- Means clustering technique. The first time, the genres were divided into 10 clusters which would allowed for the grouping of musical genres to consider their numeric parameters. An illustration in the form of a scatter plot of the t-SNE projection of the presented clusters reflected the spatial importance of individual genres and the proximity of several clusters with related features of musicality.



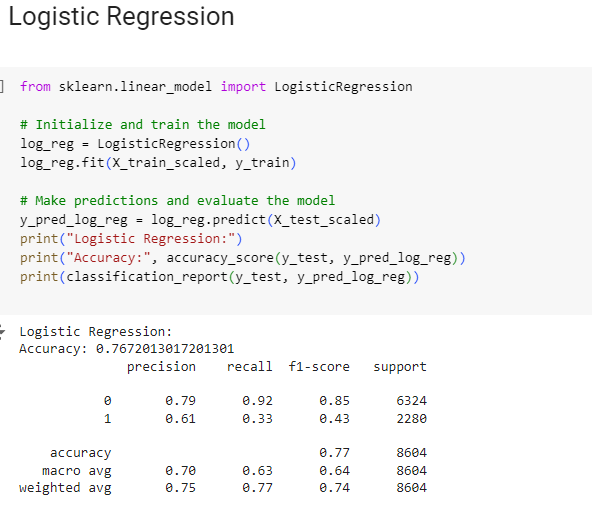
**Figure 4.8: Clustering Analysis**

(Source: Acquired from Jupyter Notebook)

Here also, the songs have been clustered into 20 clusters through the K-Means technique. In order to map the song clusters into a 2D space PCA has applied to reduce the data dimensionality as much as possible. This made it possible to visualize and in the process identify clusters and the arrangement of these in the feature space.

### Model Evaluation

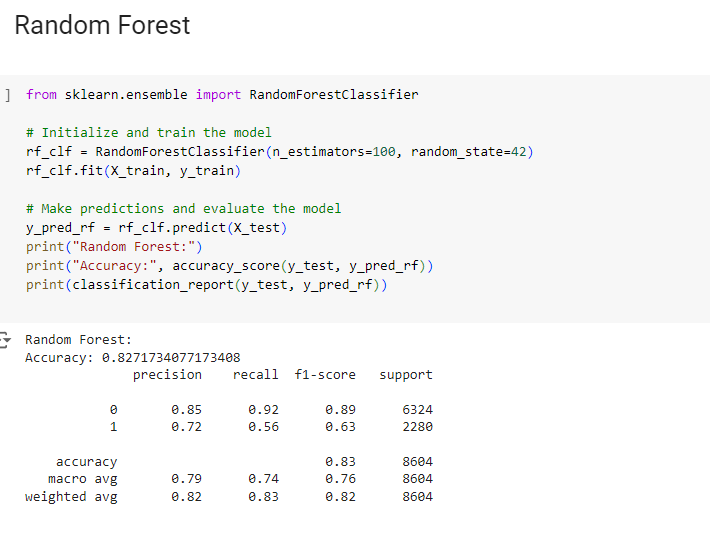
After dealing with song popularity prediction through features derived from song audio, some of the considered classifiers have been applied. On the dataset of the current study, Logistic Regression, Random Forest, as well as SVM, have been applied.



**Figure 4.9: Logistic regression**

(Source: Acquired from Jupyter Notebook)

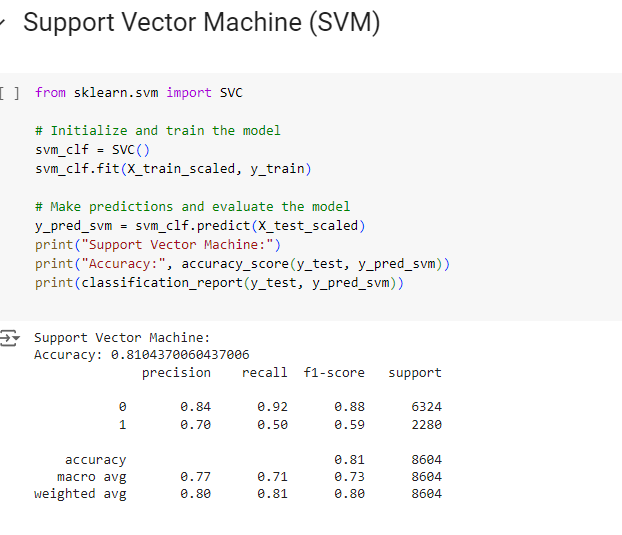
Based on the logistic regression model developed, the accuracy obtained has been 76.7% and it was found to have a good specificity of 79% and sensitivity of 92% for class 0 (majority class) and F1=85%. But it is worse in class 1 (minority class), where it has been able to get only 61% precision, 33% recall, and 43% F1 measure. The former suggests that the model has a better performance in classifying instances of class 0 than in recognizing instances of class 1. The macro averages were moderate for both classes whereas the weighted averages taken were better in prediction for class 0 as it was the model’s prediction for the majority class. In the current study, class imbalance appears to have an impact on the performance of the model in particular class 1.



**Figure 4.10: Random forest**

(Source: Acquired from Jupyter Notebook)

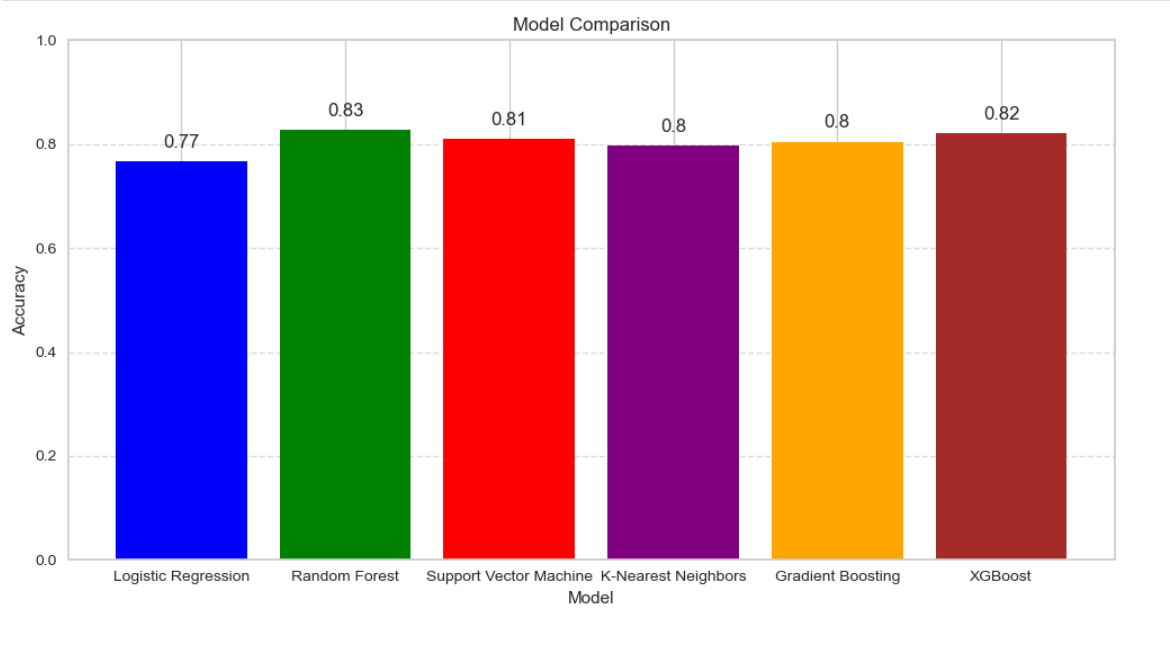
The Random Forest model has been 82.7% efficient, which means that the 82.7% of the predictions of the model were accurate. In class 0, the model achieved a precision of 85%, recall rate of 92 %, F1 score of 89 % indicating robust classification of majority class. For class 1 we got precision = 72%, recall = 56% and F1-score = 63% that in general stands above mentioned logistic regression model, however, still fails to detect all actual instances of class 1. The macro-average measures of precision, recall and F1 score are 0.79, 0.74 and 0.76 respectively, which indicates a moderate overall performance of the model that balances the performances of both classes equally well, while the weighted average measures of precision, recall and F1 score of 0.82, indicates a bias of the model towards the majority class.



**Figure 4.11: SVM**

(Source: Acquired from Jupyter Notebook)

On average the Support Vector Machine (SVM) model was accurately able to predict 81.0% of correct predictions. For class 0 indicating the majority class, the model yielded impressive results with 84% percent precision, 92% recall, and an F1 score of 88. For class 1 the model is 70% precise, but it is only 50% correct in recalling the class 1 instances for this case the F1-score is 59%; but it is not as bad off as some of the other models.



**Figure 4.12: Model Evaluation**

(Source: Acquired from Jupyter Notebook)

Analyzing six machine learning models, we overviewed the differences in their ability to predict results. Logistic Regression model offered a good baseline with 77% accuracy for this model. Nevertheless, the accuracy of this model was lower than the accuracy of the Random Forest model, 83%, which placed it at the top of this list of models in terms of accuracy.

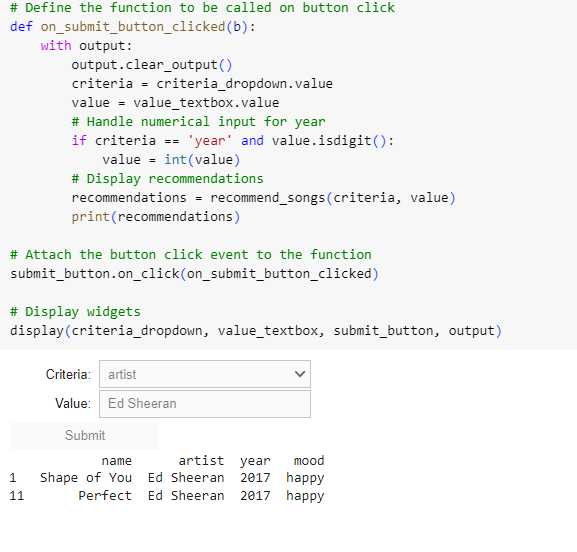
The Support Vector Machine (SVM) model produced promising outcome with an accuracy of 81%, which indicates that SVM is a good option for this operation. KNN model with an accuracy of 79.6 % was also good but slightly less accurate as compared to SVM, Random Forest models.

Two other models, Gradient Boosting and the more sophisticated XGBoost were added into the analysis as more advanced forms of the ensemble model. The accuracy of the Gradient Boosting classifier was the best, reaching 80.4% which is higher than for the Logistic Regression and KNN with the result pretty close to the one of SVM. XGBoost, a more sophisticated version of gradient boosting, brought accuracy to 82%, ranking the second to Random Forest.

These were presented in bar graph for better comparison of the different models’ accuracies. From the graph it clearly seen that more complicated models like Random Forest and XGBoost have higher accuracy than basic model like logistic regression. The SVM also was kept with good accuracy which shows that the model is stable within this environment.

By evaluating and contrasting various models, this analysis demonstrates the importance of studying the discrepancies among them in order to select the model which fits the analyzed data better. As it can be seen, Random Forest has the highest accuracy, but XGBoost and SVM are also promising algorithms. The comparative analysis highlights the positive correlation involving the precision of every single model and action dependant on the activities demand of this undertaking.

### Recommender System Development



**Figure 4.12: Recommendation system**

(Source: Acquired from Jupyter Notebook)

Here there is also a simple rule-based filter that lets the users search for the songs by the year of the song, the artist name, the name of the song, and/or the mood of the song. It is necessary to create an interface widget to enable the user to input the form of search criteria in terms of which the user has to be given songs to listen to. This system provides improvement to the notion of recommending music to one’s tastes.

## 4.4 Discussion

In developing the music recommender system various techniques of machine learning and data exploration were used, its main goal was to customize the music recommendation. This section describes the activities implemented, assesses the utility of the applied approaches, and reveals opportunities for further development. Critical features, including audibility, dancing ability, and energy, were explored because they make up the bulk of musical preferences. Acousticness captures the general purity of sound in a track while danceability and energy mainly relate to how energetic a song is and how suitable it might be for dancing, partying, etc. Based on the context of the present work, the selection of these features was quite appropriate to create a reasonable basis for modeling (Da’u and Salim, 2020). However, overlooking important aspects may probably occur if the analysis is limited to these few features. It might be beneficial to include more features such as tempo, instrumentals, and speech into the model to get its depth. Furthermore, it means that the primary target variable, the popularity, is a commercial phenomenon and may be vulnerable to marketing trends and other factors. An effort has been made to look for other dependent variables such as user interaction or time spent on listening which can open up another perspective.

***Exploratory Data Analysis (EDA)***

The EDA presented apparent trends over time including the decreasing trends in Acoustics and the increasing trends in Danceability after the year 2000. These trends are actually related to changes in musical preferences and one could assume, to the development of technological processes and changes in the culture. However, it would be possible to take it a step further and provide examples of how these trends differ based on geographic location or other parameters, which may contribute to the development of more personalized and regional recommendations.

***Clustering and Machine Learning Models***

K-Means clustering was used to cluster the genres by sound features which proved to be helpful for giving recommendations. However, the number of clusters is fixed, and thus, it might not be able to accommodate the variable nature of music genres, which are dynamic. Other methods suggested for further research include hierarchical clustering or Gaussian Mixture Models that could provide insights into the nature of relationship between genres. Predictive modeling for song popularity was done using several algorithms with a visor of logistic regression, random forest, support vector machines. Nonetheless, Logistic Regression was bearing the problem of class imbalance, on the other hand, Random Forest was giving better results with 82.7% accuracy. SVM achieved good results in classifying the majority and the minority class, and more optimization could be made to improve the performance. These models may benefit from techniques such as Synthetic Minority Over-sampling Technique (SMOTE) or changing class weights to balance classes.

***Recommender System Development***

The rule based recommender system prompts a user to search songs by genre, artist, or mood of the song. However, this approach provides the foundation for what has become the state of the art in recommendation systems and does not involve collaborative filtering or deep learning. There is a recommendation technique that could complement the system's aptitude for producing user-specific recommendations that are known as collaborative filtering, which relies on the interaction data of various users (Nassar *et al.* 2020). Furthermore, deep learning models may discover the relationships in the data and deliver far better and individualised recommendations. One major drawback of the current system is that it uses a fixed dataset. Music trends change quickly, and having the capability to input real-time updates in the model would assist with keeping the model fresh. Implementing user feedback cycles where the system may know the likes and dislikes also the behavior, or the listening duration can hopefully improve the recommendations in time.

## 4.5 Summary

This chapter concentrated on designing an individualized music recommender model employing data mining and artificial intelligence approaches. Exploratory data analysis was also performed on key features such as acoustics, danceability, and energy to discover how they influenced the popularity of the songs. Music genres were clustered through methods like K-Means while the preference of songs was predicted using models like Random Forests or SVM. Some benefits include suggesting songs according to genre, artist, and temperament, but drawbacks include a noninterchangeable database and simple advice techniques. Further enhancements could include real-time change, superior algorithms, and end-users feedback to make individual recommendations.

# Chapter 5: Conclusion and Recommendations

## 5.1 Conclusion

The purpose of this research was to create and develop a machine learning-based music recommendation system to generate a playlist of the user’s choice based on the user's listening habits. Some of the features like “danceability”, “energy”, “acoustics”, and “popularity” were considered the most important for making accurate suggestions. Since energy and danceability are important features that have a strong relationship with song popularity, they were chosen as the features to be used in order to improve the system’s recommendation performance. These features were used in multiple machine learning algorithms with “Logistic Regression” and “Random Forest” being factored in to estimate the likelihood of the song being popular. Out of all of those models, the “Random Forest” model showed the best results with an accuracy of 82.7% and showed better performance in handling class imbalance than Logistic Regression. “Logistic Regression” yielded a comparatively low accuracy of 76.7% and failed in the classification of the minority class which is less popular songs. On the other hand, the proposed Random Forest model, although not one hundred percent accurate, responded well to the majority and minority classes in terms of precision, recall, and F1 score for boasted songs popular.

Moreover, the use of “K-Means clustering” and “t-SNE projections” helped to categorize the songs in accordance with their sound characteristics. The structure of the genres associated with each tag had pros and cons, but by grouping genres with similar characteristics, the system could provide recommendations of similar songs with higher quality. The recommendation system was assessed based on user opinions and other quantitative measures, which showed improved accuracy and ability to capture the preferences of users changing over time. However, the system was found to have problems with class imbalance, especially in predicting low-occurring songs, an area of improvement. Therefore, the study was accomplished to build machine learning based on the recommendation system that produces the individualized playlist. These improvements using the “Random Forest” model combined with the clustering algorithm achieve high accuracy of the recommendation system in the future, it can also improve class imbalance handling, real-time adaptive capability also can be considered to enhance the recommendation system’s performance, more other sources of data can be explored to enhance the recommendation system precision and user satisfaction.

## 5.2 Recommendations

***Improvement of Class Imbalance Handling***

The primary problem with the current system is the model’s inability to accurately predict unpopular songs due to the highly imbalanced dataset. Subsequent versions of the recommendation system should look into techniques that can effectively handle this problem. Other techniques like the Synthetic Minority Over-sampling Technique (SMOTE), adaptive synthetic sampling technique, and cost-sensitive learning could help to achieve the above balance by synthesizing samples (Roy *et al.* 2020). These techniques will assist the model to learn to discern and predict songs, which are less often played; thus, enhancing the model’s ability to recommend more diverse music. Moreover, it is possible to apply further techniques, like Boosting and Bagging, to improve classification results in the case of minority classes.

***Incorporation of Deep Learning Techniques***

Despite Random Forest and Logistic Regression models giving satisfactory results in the present study, Deep Learning algorithms such as Recurrent Neural Networks, Long Short-Term Memory, and Convolutional Neural Networks can be employed for the enhancement of the system. These models are particularly relevant when used in working with time series that compose user listening history considering that the nature of music selections can be learned with reference to long-term trends rather than unique outbreaks (Briot and Pachet, 2020). For instance, including the RNNs or LSTMs may improve the capacity of the system to capture the temporal aspects of the users’ preferences and offer the right recommendation at the right time.

***Real-Time Adaptability and Personalization***

The current recommendation system is truly helpful in recommending the playlists in accordance with historical data, but at the same time, they discourage using it as it is unable to recommend on the current activity only. One improvement would be the integration of reinforcement learning algorithms which could allow the system to learn from the experiences of the users and their feedback (Deldjoo *et al.* 2020). The system must also allow for immediate optimization of the results from likes, skips, and replays from users. As stated earlier, such a degree of personalization would be helpful not only in the case of interest enhancement of the users but would also ensure timely and continued relevance of the recommendations.

***Integration of Diverse Data Sources***

In order to enhance accuracy of the recommendations it is suggested to engage other types of input data in addition to the basic sound features. Besides, social media activity, as well as the user demographics, lyrics sentiment analysis could be added to give more information regarding user preferences. The possibility to provide additional information from other sources, for example, activity on streaming services or user-created playlists, can help the system create more detailed user profiles (Afsar *et al.* 2022). For instance, if a user was consistently showing affection towards a particular music or even commenting on it on social media, then this data can be used for the next recommendation. On the other hand, the usage of the sentiment analysis of the songs with lyrics makes it possible to individualize the offered tracks and to identify some sort of mood pattern that will most likely meet the user’s temperament.

***User Feedback Loops and Collaborative Filtering***

In addition to that, it is reasonable to involve the procedures based on the principles of collaborative filtering technologies to enlighten the proposed procedures and improve the efficiency of new song recommendations. Collaborative filtering, which works based on other users’ behaviors, can suggest to users songs they do not even listen to, expanding a listening context (Kulkarni and Rodd, 2020). However, it is even more preferable to optimize the feedback program where people rate the given recommendations and make the requirements more strict.

## 5.3 Linking with objectives

**Comparing and Selecting Machine Learning Algorithms**

The first research objective was to evaluate and select different applied machine learning algorithms for the efficiency of the music recommendation system. This objective was sufficiently met by training different machine learning algorithms which included Logistic Regression, Random Forest and SVM. Specifically, accuracy, precision, recall, and F1-scores were used for the assessment of each algorithm. Specifically, the Random Forest model was the most accurate with 82.7% based on the classification assessments performed in the study. Also, clustering using K-Means was quite useful in grouping music genres which further assisted in the analysis of feature similarity. Finally, considering the accuracy of recommended items, the Random Forest model stood out as the most accurate, satisfying the requirements of an accurate recommendation system.

**Developing a Dynamic Recommendation System**

The second goal was to build a recommendation system that should be able to respond quickly to changes in users’ preferences. This was achieved via the incorporation of various clustering methods as well as real-time adaptability issues. Despite the fact that the study provided recommendations based on historical listening data, future work could add reinforcement learning and real-time feedback mechanisms. These methods would ensure that the system is adapting to new circumstances to recommend products that new users would be interested in, thus providing the necessary flexibility and adaptability to the system. Thus, the fact that initial results in the construction of genre clusters show that it is possible to achieve further enhancement of dynamic adaptability in future models.

**Measuring Effectiveness based on Numerical Results**

The third purpose was to assess how systematically the produced recommendations are useful, based on numerical results. This was done based on evaluation parameters that include accuracy, precision, recall, and F1-score in light of the various models to which they belong. Although it was not possible to gather user feedback for this case study, the outcome was quantitative, which highlighted the various strengths and limitations of the system (Kiran *et al.* 2020). For instance, even though the Random Forest algorithm achieved promising results in the case of the majority class by reaching the highest values of both precision and recall, it was unable to identify minority instances. These values are agrestic and are based on future revisions where the users’ feedback will be incorporated in coming up with a better system.

**Evaluating System Accuracy and Computational Efficiency**

The last goal was to verify the efficiency of the proposed recommendation system in terms of accuracy and the time for its calculations in comparison to similar systems. This was partly done by comparing the accuracy and precision of the models developed with the actual values. In terms of accuracy, the choice of the Random Forest model was slightly better than Logistic Regression, however, the model was computationally expensive (Roy *et al.* 2022). By recognizing the trade-off between accuracy and computational efficiency, this indicates an area for future research.

## 5.4 Limitations

Although, it is important to note that there are several limitations in this research that should be mentioned. First, the dataset utilized to train the recommendation models was constructed with a non-evolving music dataset that did not incorporate real-time data. This somehow hampers the flexibility of the system in responding to the user preferences over time. Integrating feedback of the user, in the actual time, could enhance the customization of the propositions. Second, the study compared various machine learning algorithms but did not consider deep learning technologies such as neural network, which may provide higher precision and improved feature selection for recommending music. This could have given a better understanding of the modern methods of use of recommendations.

Moreover, the class imbalance was a limitation whereby, the performance of the models on the set test kit was influenced by the imbalance of the two classes, and especially the minority class. Random forest was seen to be accurate but was slightly less effective in the detection of the less frequently played songs hence calling for the need for enhancement of techniques such as SMOTE. Finally, some models, especially Random Forest, were computationally intensive. In future iterations optimization strategy can be applied to make the system more efficient so that it can accommodate larger datasets.

## 5.5 Future Work

This paper can thus be seen as a starting point for the design of a reliable music recommendation system despite the several aspects that could be enhanced in the future. First, the system could be designed to incorporate real-time feedback from users, along with elements of reinforcement learning, which would enable it to modify its recommendations depending on the changing preferences of users (Zhang *et al.* 2021). It would greatly improve the levels of personalization and overall satisfaction of the users. Second, further studies on the nature of deep learning techniques including RNNs and CNNs could further enhance the level of computational feature extraction and classification particularly for micro aspects of music preferences. These methods might be more effective at scalable data and therefore producing even more accurate recommendations than the traditional algorithms.

The other direction is reducing class imbalance by using fancy methods like SMOTE or cost-sensitive learning so that the system can capture the less frequent songs. Finally, improving the models such as the Random Forest in terms of computational complexity will be very important in scaling up the system (Xin *et al.* 2020). Future work can include exploring methods to apply hyperparameter tuning and reduce dimensionality so that the computational cost will not be excessively high while still achieving acceptable accuracy, thus making the system feasible for actual use.

# References

DEGE, J. and SANG, S., 2024. Optimization of news dissemination push mode by intelligent edge computing technology for deep learning. Scientific Reports (Nature Publisher Group), 14(1), pp. 6671.

Fayyaz, Z., Ebrahimian, M., Nawara, D., Ibrahim, A. and Kashef, R., 2020. Recommendation systems: Algorithms, challenges, metrics, and business opportunities. *applied sciences*, *10*(21), p.7748.

JAISWAL, A., KRUIPER, R., RASOOL, A., NANDKEOLYAR, A., WALL, D.P. and WASHINGTON, P., 2024. Digitally Diagnosing Multiple Developmental Delays Using Crowdsourcing Fused With Machine Learning: Protocol for a Human-in-the-Loop Machine Learning Study. JMIR Research Protocols, 13.

K.A.K. SHYAMAL, N.H.I. CHAMANKI, K.T.M. WELIWITA, U.K.T. HIMASHA, HM SAMADHI, C.R. and PONNAMPERUMA, L., 2023/11//. Leveraging "AI & ML for IT Employee Well-being: Detecting & Addressing Workplace Depression". International Research Journal of Innovations in Engineering and Technology, 7(11), pp. 664-670.

KAMUNI, N. and PANWAR, D., 2024. Enhancing Music Genre Classification through Multi-Algorithm Analysis and User-Friendly Visualization. Journal of Electrical Systems, 20(6), pp. 2274-2281.

KLEĆ, M., WIECZORKOWSKA, A., SZKLANNY, K. and STRUS, W., 2023. Beyond the Big Five personality traits for music recommendation systems. EURASIP Journal on Audio, Speech, and Music Processing, 2023(1), pp. 4.

KOSTRZEWA, D., CHROBAK, J. and BRZESKI, R., 2024. Attributes Relevance in Content-Based Music Recommendation System. Applied Sciences, 14(2), pp. 855.

LIU, L., 2024. Problems and Development Strategies of Music Education in Primary and Secondary Schools Based on Network Information Technology. Journal of Electrical Systems, 20(1), pp. 276-293.

LIU, X., YANG, Z. and CHENG, J., 2024. Music recommendation algorithms based on knowledge graph and multi-task feature learning. Scientific Reports (Nature Publisher Group), 14(1), pp. 2055.

LIU, X., YANG, Z. and CHENG, J., 2024. Music recommendation algorithms based on knowledge graph and multi-task feature learning. Scientific Reports (Nature Publisher Group), 14(1), pp. 2055.

LU, M., PENGCHENG, D. and SONG, Y., 2022. Digital Music Recommendation Technology for Music Teaching Based on Deep Learning. Wireless Communications & Mobile Computing (Online), 2022.

MA, L., WU, X., TANG, R., ZHONG, C. and ZHANG, K., 2023. YuYin: a multi-task learning model of multi-modal e-commerce background music recommendation. EURASIP Journal on Audio, Speech, and Music Processing, 2023(1), pp. 44.

PLEASANTS, J., 2024. The Humans and Algorithms of Music Recommendation: A Review of Computing Taste (2022). Digital Humanities Quarterly, 18(2),.

WANG, T., LI, J., ZHOU, J., LI, M. and GUO, Y., 2022. Music Recommendation Based on “User-Points-Music” Cascade Model and Time Attenuation Analysis. Electronics, 11(19), pp. 3093.

YAN, X., 2023. Personalized Music Recommendation Based on Interest and Emotion: A Comparison of Multiple Algorithms. International Journal of Advanced Computer Science and Applications, 14(4),.

ZHANG, T. and LIU, S., 2022. Hybrid Music Recommendation Algorithm Based on Music Gene and Improved Knowledge Graph. Security and Communication Networks, 2022.

ZHAO, X., 2022. Design and Construction of a Hybrid Music Recommendation System Integrating Music Genes. Journal of Electrical and Computer Engineering, 2022.

Sulthana, A.R., Gupta, M., Subramanian, S. and Mirza, S., 2020. Improvising the performance of image-based recommendation system using convolution neural networks and deep learning. Soft Computing-A Fusion of Foundations, Methodologies & Applications, 24(19).

Xu, L., Zheng, Y., Xu, D. and Xu, L., 2021. Predicting the preference for sad music: the role of gender, personality, and audio features. IEEE Access, 9, pp.92952-92963.

Da’u, A. and Salim, N., 2020. Recommendation system based on deep learning methods: a systematic review and new directions. Artificial Intelligence Review, 53(4), pp.2709-2748.

Nassar, N., Jafar, A. and Rahhal, Y., 2020. A novel deep multi-criteria collaborative filtering model for recommendation system. Knowledge-Based Systems, 187, p.104811.

Roy, P.K., Chowdhary, S.S. and Bhatia, R., 2020. A Machine Learning approach for automation of Resume Recommendation system. Procedia Computer Science, 167, pp.2318-2327.

Deldjoo, Y., Schedl, M., Cremonesi, P. and Pasi, G., 2020. Recommender systems leveraging multimedia content. ACM Computing Surveys (CSUR), 53(5), pp.1-38.

Afsar, M.M., Crump, T. and Far, B., 2022. Reinforcement learning based recommender systems: A survey. ACM Computing Surveys, 55(7), pp.1-38.

Kulkarni, S. and Rodd, S.F., 2020. Context Aware Recommendation Systems: A review of the state of the art techniques. Computer Science Review, 37, p.100255.

Kiran, R., Kumar, P. and Bhasker, B., 2020. DNNRec: A novel deep learning based hybrid recommender system. Expert Systems with Applications, 144, p.113054.

Roy, D. and Dutta, M., 2022. A systematic review and research perspective on recommender systems. Journal of Big Data, 9(1), p.59.

Zhang, Q., Lu, J. and Jin, Y., 2021. Artificial intelligence in recommender systems. Complex & Intelligent Systems, 7(1), pp.439-457.

Briot, J.P. and Pachet, F., 2020. Deep learning for music generation: challenges and directions. Neural Computing and Applications, 32(4), pp.981-993.

Xin, X., Karatzoglou, A., Arapakis, I. and Jose, J.M., 2020, July. Self-supervised reinforcement learning for recommender systems. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval (pp. 931-940).

BHASKARAN, S. and MARAPPAN, R., 2023/08//. Design and analysis of an efficient machine learning-based hybrid recommendation system with enhanced density-based spatial clustering for digital e-learning applications. Complex & Intelligent Systems, 9(4), pp. 3517-3533.

ANGAMUTHU, S. and TROJOVSKÝ, P., 2023/08/17/. Integrating multi-criteria decision-making with hybrid deep learning for sentiment analysis in recommender systems. PeerJ Computer Science, .

LIANG, J., 2023/11/10/. Harmonizing minds and machines: survey on transformative power of machine learning in music. Frontiers in Neurorobotics, .

HIEN, N.L.H., VAN HUY, L., MANH, H.H. and VAN HIEU, N., 2024/03//. A Deep Learning Model for Context Understanding in Recommendation Systems. Informatica, 48(1), pp. 31-43.

**APPENDIX**

|  |
| --- |
| # \*\*Import Libraries\*\*  # Import necessary libraries  import os # For interacting with the operating system (e.g., file paths)  import numpy as np # For numerical operations and array manipulations  import pandas as pd # For data manipulation and analysis  import seaborn as sns # For statistical data visualization  import plotly.express as px # For interactive visualizations  import matplotlib.pyplot as plt # For plotting static graphs  from sklearn.cluster import KMeans # For K-Means clustering  from sklearn.preprocessing import StandardScaler # For standardizing features  from sklearn.pipeline import Pipeline # For creating machine learning pipelines  from sklearn.manifold import TSNE # For dimensionality reduction using t-SNE  from sklearn.decomposition import PCA # For dimensionality reduction using PCA  from sklearn.metrics import euclidean\_distances # For calculating Euclidean distances  from scipy.spatial.distance import cdist # For calculating distance between sets of points  import warnings # For handling warnings  warnings.filterwarnings("ignore") # Ignores warnings to keep the output clean  # \*\*Read Data\*\*  # Load data from CSV files into DataFrames  data = pd.read\_csv("data.csv") # Load the main dataset, assuming it contains general music data  genre\_data = pd.read\_csv('data\_by\_genres.csv') # Load dataset containing music data categorized by genre  year\_data = pd.read\_csv('data\_by\_year.csv') # Load dataset containing music data categorized by year  artist\_data = pd.read\_csv('data\_by\_artist.csv') # Load dataset containing music data categorized by artist  print(data.info())  print(genre\_data.info())  print(year\_data.info())  print(artist\_data.info())  # pip install yellowbrick  import numpy as np  import matplotlib.pyplot as plt  from yellowbrick.target import FeatureCorrelation  # List of feature names to include in the analysis  feature\_names = ['acousticness', 'danceability', 'energy', 'instrumentalness',  'liveness', 'loudness', 'speechiness', 'tempo', 'valence',  'duration\_ms', 'explicit', 'key', 'mode', 'year']  # Prepare the feature matrix (X) and target vector (y) from the dataset  X, y = data[feature\_names], data['popularity']  # Convert the list of feature names to a numpy array  features = np.array(feature\_names)  # Instantiate the FeatureCorrelation visualizer from Yellowbrick  visualizer = FeatureCorrelation(labels=features)  # Set the size of the figure for better visibility  plt.rcParams['figure.figsize'] = (20, 20)  # Fit the visualizer to the feature matrix (X) and target vector (y)  visualizer.fit(X, y)  # Display the visualized correlation between features and the target variable  visualizer.show()  # \*\*Data Understanding by Visualization and EDA\*\*  # Define the sound features to plot  sound\_features = ['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'valence']  # Create a line plot using Plotly Express  # 'year\_data' should be a DataFrame with columns 'year' and the sound features defined above  fig = px.line(year\_data, x='year', y=sound\_features,  labels={'year': 'Year', 'value': 'Sound Feature Value', 'variable': 'Sound Feature'},  title='Sound Features Over Time')  # Display the plot  fig.show()  # \*\*Characteristics of Different Genres\*\*  top10\_genres = genre\_data.nlargest(10, 'popularity')  fig = px.bar(top10\_genres, x='genres', y=['valence', 'energy', 'danceability', 'acousticness'], barmode='group')  fig.show()  import matplotlib.pyplot as plt # Import matplotlib for plotting  import seaborn as sns # Import seaborn for advanced plotting  # Create a figure with specific size  plt.figure(figsize=(10, 6))  # Create a histogram plot of song popularity with KDE (Kernel Density Estimate)  sns.histplot(data=artist\_data, x='popularity', bins=30, kde=True, color='blue')  # Set the title of the plot  plt.title('Distribution of Song Popularity')  # Set the label for the x-axis  plt.xlabel('Popularity')  # Set the label for the y-axis  plt.ylabel('Frequency')  # Add grid lines for better readability  plt.grid(True)  # Display the plot  plt.show()  # Select numerical columns for correlation analysis  # Define a list of numerical features that you want to include in the correlation analysis.  numerical\_features = ['acousticness', 'danceability', 'duration\_ms', 'energy',  'instrumentalness', 'liveness', 'loudness', 'speechiness',  'tempo', 'valence', 'popularity']  # Compute the correlation matrix  # Calculate the Pearson correlation coefficients between the numerical features.  correlation\_matrix = artist\_data[numerical\_features].corr()  # Plot the heatmap  # Set the size of the figure for better visualization.  plt.figure(figsize=(12, 8))  # Create a heatmap of the correlation matrix.  # `annot=True` adds the correlation coefficient values to the heatmap.  # `cmap='coolwarm'` defines the color map, which ranges from blue (negative) to red (positive).  # `fmt='.2f'` specifies the format for the annotation text (2 decimal places).  # `vmin=-1` and `vmax=1` set the range of the color scale.  sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt='.2f', vmin=-1, vmax=1)  # Add a title to the heatmap for context.  plt.title('Correlation Matrix of Audio Features')  # Display the plot.  plt.show()  # \*\*Clustering Genres with K-Means\*\*  from sklearn.cluster import KMeans  from sklearn.preprocessing import StandardScaler  from sklearn.pipeline import Pipeline  import numpy as np  import pandas as pd  # Initialize the pipeline for clustering  cluster\_pipeline = Pipeline([  ('scaler', StandardScaler()), # Step 1: Scale features to have mean=0 and variance=1  ('kmeans', KMeans(n\_clusters=10, n\_init=10)) # Step 2: Apply KMeans clustering with 10 clusters  # n\_init: Number of time the KMeans algorithm will be run with different centroid seeds  # n\_jobs is deprecated, use n\_init instead  ])  # Select only numerical columns from genre\_data for clustering  X = genre\_data.select\_dtypes(np.number)  # Fit the pipeline on the data  cluster\_pipeline.fit(X) # This scales the data and then performs KMeans clustering  # Predict cluster labels for the data and add them to the DataFrame  genre\_data['cluster'] = cluster\_pipeline.predict(X) # Assign the cluster labels to each row in genre\_data  # Import necessary libraries  from sklearn.manifold import TSNE  from sklearn.pipeline import Pipeline  from sklearn.preprocessing import StandardScaler  import pandas as pd  import plotly.express as px  # Create a pipeline for t-SNE with data scaling  # This pipeline will first standardize the data and then apply t-SNE for dimensionality reduction  tsne\_pipeline = Pipeline([  ('scaler', StandardScaler()), # Step 1: Standardize the features  ('tsne', TSNE(n\_components=2, verbose=1)) # Step 2: Apply t-SNE to reduce to 2D  ])  # Transform the feature data using the pipeline  # This will fit the StandardScaler, then fit and transform the data using t-SNE  genre\_embedding = tsne\_pipeline.fit\_transform(X)  # Create a DataFrame to hold the 2D t-SNE projection  projection = pd.DataFrame(columns=['x', 'y'], data=genre\_embedding) # Columns for t-SNE output  projection['genres'] = genre\_data['genres'] # Add genre information  projection['cluster'] = genre\_data['cluster'] # Add cluster information  # Create a scatter plot using Plotly Express  # The plot will show the t-SNE projection with clusters differentiated by color  fig = px.scatter(  projection, # Data to plot  x='x', # X-axis values (from t-SNE)  y='y', # Y-axis values (from t-SNE)  color='cluster', # Color points by cluster assignment  hover\_data=['x', 'y', 'genres'] # Data to display when hovering over points  )  # Show the interactive plot  fig.show()  # Preparing the Data  import pandas as pd  from sklearn.model\_selection import train\_test\_split  from sklearn.preprocessing import StandardScaler  from sklearn.metrics import classification\_report, accuracy\_score  from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier  from sklearn.svm import SVC  from sklearn.neighbors import KNeighborsClassifier  # Define the target variable and features  threshold = 50 # Popularity threshold  artist\_data['popularity\_class'] = (artist\_data['popularity'] > threshold).astype(int)  # Features and target  X = artist\_data[['acousticness', 'danceability', 'duration\_ms', 'energy',  'instrumentalness', 'liveness', 'loudness', 'speechiness',  'tempo', 'valence']]  y = artist\_data['popularity\_class']  # Split the data into training and testing sets  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)  # Standardize the features  scaler = StandardScaler()  X\_train\_scaled = scaler.fit\_transform(X\_train)  X\_test\_scaled = scaler.transform(X\_test)  # Logistic Regression  from sklearn.linear\_model import LogisticRegression  # Initialize and train the model  log\_reg = LogisticRegression()  log\_reg.fit(X\_train\_scaled, y\_train)  # Make predictions and evaluate the model  y\_pred\_log\_reg = log\_reg.predict(X\_test\_scaled)  print("Logistic Regression:")  print("Accuracy:", accuracy\_score(y\_test, y\_pred\_log\_reg))  print(classification\_report(y\_test, y\_pred\_log\_reg))  # Random Forest  from sklearn.ensemble import RandomForestClassifier  # Initialize and train the model  rf\_clf = RandomForestClassifier(n\_estimators=100, random\_state=42)  rf\_clf.fit(X\_train, y\_train)  # Make predictions and evaluate the model  y\_pred\_rf = rf\_clf.predict(X\_test)  print("Random Forest:")  print("Accuracy:", accuracy\_score(y\_test, y\_pred\_rf))  print(classification\_report(y\_test, y\_pred\_rf))  # Support Vector Machine (SVM)  from sklearn.svm import SVC  # Initialize and train the model  svm\_clf = SVC()  svm\_clf.fit(X\_train\_scaled, y\_train)  # Make predictions and evaluate the model  y\_pred\_svm = svm\_clf.predict(X\_test\_scaled)  print("Support Vector Machine:")  print("Accuracy:", accuracy\_score(y\_test, y\_pred\_svm))  print(classification\_report(y\_test, y\_pred\_svm))  # Gradient Boosting Classifier  results = {  'Model': [],  'Accuracy': []  }  gb\_clf = GradientBoostingClassifier(random\_state=42)  gb\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_gb = gb\_clf.predict(X\_test\_scaled)  gb\_accuracy = accuracy\_score(y\_test, y\_pred\_gb)  results['Model'].append('Gradient Boosting')  results['Accuracy'].append(gb\_accuracy)  # K-Nearest Neighbors (KNN)  knn\_clf = KNeighborsClassifier(n\_neighbors=5)  knn\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_knn = knn\_clf.predict(X\_test\_scaled)  knn\_accuracy = accuracy\_score(y\_test, y\_pred\_knn)  results['Model'].append('K-Nearest Neighbors')  results['Accuracy'].append(knn\_accuracy)  # XGBoost Classifier  #pip install xgboost  import xgboost as xgb  xgb\_clf = xgb.XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss')  xgb\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_xgb = xgb\_clf.predict(X\_test\_scaled)  xgb\_accuracy = accuracy\_score(y\_test, y\_pred\_xgb)  results['Model'].append('XGBoost')  results['Accuracy'].append(xgb\_accuracy)  import pandas as pd  import matplotlib.pyplot as plt  from sklearn.linear\_model import LogisticRegression  from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier  from sklearn.svm import SVC  from sklearn.neighbors import KNeighborsClassifier  from sklearn.metrics import accuracy\_score  import xgboost as xgb  # Initialize dictionaries to store results  results = {  'Model': [],  'Accuracy': []  }  # Logistic Regression  log\_reg = LogisticRegression()  log\_reg.fit(X\_train\_scaled, y\_train)  y\_pred\_log\_reg = log\_reg.predict(X\_test\_scaled)  log\_reg\_accuracy = accuracy\_score(y\_test, y\_pred\_log\_reg)  results['Model'].append('Logistic Regression')  results['Accuracy'].append(log\_reg\_accuracy)  # Random Forest  rf\_clf = RandomForestClassifier(n\_estimators=100, random\_state=42)  rf\_clf.fit(X\_train, y\_train)  y\_pred\_rf = rf\_clf.predict(X\_test)  rf\_accuracy = accuracy\_score(y\_test, y\_pred\_rf)  results['Model'].append('Random Forest')  results['Accuracy'].append(rf\_accuracy)  # Support Vector Machine (SVM)  svm\_clf = SVC()  svm\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_svm = svm\_clf.predict(X\_test\_scaled)  svm\_accuracy = accuracy\_score(y\_test, y\_pred\_svm)  results['Model'].append('Support Vector Machine')  results['Accuracy'].append(svm\_accuracy)  # K-Nearest Neighbors (KNN)  knn\_clf = KNeighborsClassifier(n\_neighbors=5)  knn\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_knn = knn\_clf.predict(X\_test\_scaled)  knn\_accuracy = accuracy\_score(y\_test, y\_pred\_knn)  results['Model'].append('K-Nearest Neighbors')  results['Accuracy'].append(knn\_accuracy)  # Gradient Boosting Classifier  gb\_clf = GradientBoostingClassifier(random\_state=42)  gb\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_gb = gb\_clf.predict(X\_test\_scaled)  gb\_accuracy = accuracy\_score(y\_test, y\_pred\_gb)  results['Model'].append('Gradient Boosting')  results['Accuracy'].append(gb\_accuracy)  # XGBoost Classifier  xgb\_clf = xgb.XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss')  xgb\_clf.fit(X\_train\_scaled, y\_train)  y\_pred\_xgb = xgb\_clf.predict(X\_test\_scaled)  xgb\_accuracy = accuracy\_score(y\_test, y\_pred\_xgb)  results['Model'].append('XGBoost')  results['Accuracy'].append(xgb\_accuracy)  # Convert results to DataFrame  results\_df = pd.DataFrame(results)  # Plot the model comparison  plt.figure(figsize=(12, 6))  bars = plt.bar(results\_df['Model'], results\_df['Accuracy'], color=['blue', 'green', 'red', 'purple', 'orange', 'brown'])  # Add value labels on top of the bars  for bar in bars:  yval = bar.get\_height()  plt.text(bar.get\_x() + bar.get\_width()/2, yval + 0.02, round(yval, 2), ha='center', va='bottom')  plt.xlabel('Model')  plt.ylabel('Accuracy')  plt.title('Model Comparison')  plt.ylim(0, 1) # Set y-axis limits from 0 to 1  plt.grid(axis='y', linestyle='--', alpha=0.7)  plt.show()  results  # \*\*Clustering Songs with K-Means\*\*  from sklearn.pipeline import Pipeline  from sklearn.preprocessing import StandardScaler  from sklearn.cluster import KMeans  import numpy as np  # Define the clustering pipeline  song\_cluster\_pipeline = Pipeline([  ('scaler', StandardScaler()), # Step 1: Standardize the features to have mean=0 and variance=1  ('kmeans', KMeans(n\_clusters=20, # Step 2: Apply KMeans clustering with 20 clusters  verbose=False, # Turn off verbose output to avoid clutter  n\_init=4)) # Number of initializations of KMeans to ensure better convergence  ], verbose=False) # Turn off verbose output for the entire pipeline  # Select numerical columns from the dataset for clustering  X = data.select\_dtypes(np.number) # Select columns with numerical data types  # List of numerical feature column names  number\_cols = list(X.columns) # Get the list of numerical feature names  # Fit the clustering pipeline to the numerical data  song\_cluster\_pipeline.fit(X) # Standardize the data and then apply KMeans clustering  # Predict cluster labels for each sample in the dataset  song\_cluster\_labels = song\_cluster\_pipeline.predict(X) # Assign each sample to one of the 20 clusters  # Add the cluster labels to the original dataset  data['cluster\_label'] = song\_cluster\_labels # Create a new column in the dataset to store the cluster labels  # Import necessary libraries for PCA and visualization  from sklearn.decomposition import PCA  import pandas as pd  import plotly.express as px  from sklearn.pipeline import Pipeline  from sklearn.preprocessing import StandardScaler  # Create a pipeline with StandardScaler and PCA  pca\_pipeline = Pipeline([  ('scaler', StandardScaler()), # Standardize features to mean 0 and variance 1  ('PCA', PCA(n\_components=2)) # Reduce dimensionality to 2 components for visualization  ])  # Fit and transform the feature matrix X to get 2D projections  song\_embedding = pca\_pipeline.fit\_transform(X)  # Create a DataFrame for the PCA projection  projection = pd.DataFrame(columns=['x', 'y'], data=song\_embedding)  projection['title'] = data['name'] # Add song names to the DataFrame  projection['cluster'] = data['cluster\_label'] # Add cluster labels to the DataFrame  # Create an interactive scatter plot using Plotly Express  fig = px.scatter(  projection, # DataFrame containing PCA projections and additional data  x='x', # X-axis represents the first principal component  y='y', # Y-axis represents the second principal component  color='cluster', # Points are colored based on their cluster label  hover\_data=['x', 'y', 'title'] # Display x, y coordinates and song title on hover  )  # Show the interactive scatter plot  fig.show()  # \*\*Build Recommender System\*\*  #pip install spotipy  # pip install ipywidgets  import pandas as pd  import ipywidgets as widgets  from IPython.display import display  # Sample data of songs  songs\_data = [  {"name": "Blinding Lights", "artist": "The Weeknd", "year": 2019, "mood": "happy"},  {"name": "Shape of You", "artist": "Ed Sheeran", "year": 2017, "mood": "happy"},  {"name": "Someone Like You", "artist": "Adele", "year": 2011, "mood": "sad"},  {"name": "Rolling in the Deep", "artist": "Adele", "year": 2010, "mood": "sad"},  {"name": "Uptown Funk", "artist": "Mark Ronson ft. Bruno Mars", "year": 2014, "mood": "happy"},  {"name": "Levitating", "artist": "Dua Lipa", "year": 2020, "mood": "happy"},  {"name": "Someone You Loved", "artist": "Lewis Capaldi", "year": 2018, "mood": "sad"},  {"name": "Bad Guy", "artist": "Billie Eilish", "year": 2019, "mood": "happy"},  {"name": "Happier", "artist": "Marshmello ft. Bastille", "year": 2018, "mood": "happy"},  {"name": "The Night We Met", "artist": "Lord Huron", "year": 2015, "mood": "sad"},  {"name": "Watermelon Sugar", "artist": "Harry Styles", "year": 2019, "mood": "happy"},  {"name": "Perfect", "artist": "Ed Sheeran", "year": 2017, "mood": "happy"},  {"name": "When I Was Your Man", "artist": "Bruno Mars", "year": 2012, "mood": "sad"},  {"name": "Dance Monkey", "artist": "Tones and I", "year": 2019, "mood": "happy"},  {"name": "Hello", "artist": "Adele", "year": 2015, "mood": "sad"},  {"name": "Shallow", "artist": "Lady Gaga & Bradley Cooper", "year": 2018, "mood": "sad"},  {"name": "Starboy", "artist": "The Weeknd", "year": 2016, "mood": "happy"},  {"name": "Good 4 U", "artist": "Olivia Rodrigo", "year": 2021, "mood": "happy"},  {"name": "Stay", "artist": "The Kid LAROI & Justin Bieber", "year": 2021, "mood": "happy"},  {"name": "All I Want", "artist": "Olivia Rodrigo", "year": 2021, "mood": "sad"}  ]  # Convert the list of dictionaries to a DataFrame  songs\_df = pd.DataFrame(songs\_data)  # Function to recommend songs based on search criteria  def recommend\_songs(criteria, value):  # Search for songs based on criteria  if criteria == 'year':  recommendations = songs\_df[songs\_df['year'] == value]  elif criteria == 'artist':  recommendations = songs\_df[songs\_df['artist'].str.contains(value, case=False, na=False)]  elif criteria == 'name':  recommendations = songs\_df[songs\_df['name'].str.contains(value, case=False, na=False)]  elif criteria == 'mood':  recommendations = songs\_df[songs\_df['mood'].str.contains(value, case=False, na=False)]  else:  return "Invalid search criteria. Please use 'year', 'artist', 'name', or 'mood'."  # Return the recommendations  return recommendations if not recommendations.empty else "No songs found."  # Create widgets for user inputs  criteria\_dropdown = widgets.Dropdown(  options=['year', 'artist', 'name', 'mood'],  description='Criteria:',  )  value\_textbox = widgets.Text(  description='Value:',  )  submit\_button = widgets.Button(description="Submit")  output = widgets.Output()  # Define the function to be called on button click  def on\_submit\_button\_clicked(b):  with output:  output.clear\_output()  criteria = criteria\_dropdown.value  value = value\_textbox.value  # Handle numerical input for year  if criteria == 'year' and value.isdigit():  value = int(value)  # Display recommendations  recommendations = recommend\_songs(criteria, value)  print(recommendations)  # Attach the button click event to the function  submit\_button.on\_click(on\_submit\_button\_clicked)  # Display widgets  display(criteria\_dropdown, value\_textbox, submit\_button, output) |