Music Recommendation System

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

By

Aniket Katoch, 181271.

Under the supervision of

Deepak Gupta

to

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Music Recommendation System"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2015 to December 2015 under the supervision of **(Supervisor name)** (Designation and Department name).
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Submitted by:
Aniket Katoch, 181271.

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Supervised by:
Deepak Gupta
Assistant Professor (SG)
Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology
Dated:04-12-2021

ACKNOWLEDGEMENT

Firstly, I express my heartiest thanks and gratefulness to almighty God for His divine blessing makes it possible to complete the project work successfully.

I am really grateful and wish my profound indebtedness to Supervisor Deepak Gupta, Assistant Professor (SG), Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of my supervisor in the field of Machine Learning to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to Deepak Gupta, Department of CSE, for his kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Aniket Katoch, 181271.

TABLE OF CONTENT

ABSTRACT

Recommendation systems have emerged as a result of the large amount of data available on the Internet. Many firms, such as Amazon and Flipkart for e-commerce, wynk music and ganna.com for music streaming, are now employing recommender systems to their advantage. We provide a framework in this particular situation that can then recommend new melodies to clients based on their preferences. This initiative primarily focuses on providing music recommendations to music fans in order to assist them in listening to tracks that they may enjoy. Clients can use this framework to identify new collections of tunes, making the melodic list available for tuning in.

Music is life for music fans, and it has become a larger part of everyone's lives. Music helps us tune in to the cosmos, and the best part about music is that nothing can soothe you like a soothing melody. We chose to do this project because of all the positive aspects of music and the increasing demand for recommender systems on the market. The report comprises a topic description, and a full summary of the work completed thus far. The paper includes thorough explanations of the work completed, including snapshots of implementations, various techniques, and tools used thus far. The project schedule and deliverables are also included in the report. The major goal of music recommendation in this study is to provide strong human-computer interaction and deliver good recommendations to users. It is fluid and can be changed by variables other than the listening history of users or songs

Chapter 01: INTRODUCTION

1.1 Introduction

a) Context

The first suggestion system was created in 1979. Elaine Rich defined her Grundy library system [1] as follows: it is used to offer books to users after a brief interview in which the user is requested to fill in his first and last name, and then Grundy asks them to define themselves in a few key terms in order to discover their preferences and classify them as a "stereotype." Grundy provides an initial suggestion by providing a summary of the book after the data has been recorded. If the user is unhappy with the option, Grundy asks questions to figure out which part of the book it made a mistake on and then proposes a fresh one.

Recommendation systems, which first appeared in the 1990s, have advanced significantly in recent years, particularly with the introduction of Machine Learning and networks. On the one hand, the expanding use of today's digital world, which is characterized by a wealth of data, has enabled us to collect massive user databases. On the other hand, when computing power increased, it became possible to handle these data, particularly using Machine Learning, when human skills were no longer capable of conducting a thorough examination of such a large amount of data.

Unlike search engines, which get queries with specific information about what the user wants, a recommendation system does not receive a direct request from the user, but instead must provide them fresh options based on their past behaviors. E-commerce sites that want to sell as many commodities or services as possible to customers (travel, books, etc.) must swiftly recommend appropriate commodities. The purpose of services that provide streaming music and movies is to keep people on their platform for as long as possible. The recurring theme is that appropriate recommendations are required. Recent advancements in this industry have been significant, and these tips are advantageous to both businesses looking to maximize earnings and customers who are no longer overwhelmed by the quantity of options available. Making decisions is therefore made simple, and a good tip saves a lot of time.

The Recommender System is a software application and algorithm that provides suggestions for items that a user is most interested in. Recommendations are used in a variety of real-world situations, such as deciding what products to buy, listening to music, or reading the latest news. On the other side, there has been a shift in recorded commodity music, particularly after Apple acquired Beats Music in 2014 8. The music industry's economic model has recently shifted from commodity sales to subscriptions and streaming. In comparison to prior eras, the availability of digital music is now abundant due to the new business model in the music industry. As a result, the importance of a music recommender system for music suppliers cannot be overstated. It is foreseeing.Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

It is assumed that if people rate music things similarly or behave similarly, they would rate other music items similarly as well. The sparse evaluation matrix is the major issue in collaborative filtering methods since most users only see a tiny portion of all music libraries, hence most assessments are not decided. Content-based filtering, on the other hand, makes suggestions based on the characteristics of the music pieces.

We will see if we can get better recommendations by using real-time data, such as a user's heart rate and the time of day, when making recommendations in this project. The recommendations will be made by a system that employs several machine learning techniques and is accessible via a mobile application.

The system uses a smart watch to recognise the user’s heart rate in order to give recommendations of songs according to what kind of music is usually associated with that heart rate and time of day for that specific user.

For instance, if a user is out running, the user’s heart rate is probably higher than normal.

Many music firms, such as Amazon Music, Wynk Music, and Gaana.com, now use recommender algorithms, and the old technique of selling music has shifted to a cloud-based one. All of their music resources are now available in the cloud, and customers may listen to tracks directly from there. However, the problem is that the cloud system has a large amount of music. As a result, we must categorize all of the songs based on various genres, artists' regions, age groups, and languages, with the primary purpose of categorizing these songs according to the user's preferences. Because users demand a good return on their time and money, we can attract a large number of clients by offering a variety of valuable services that they are interested in. We're using a variety of machine learning methods as well as data mining techniques for this project. We tested a number of algorithms and compared the results to determine the most effective algorithm for our model.

b) Types of recommendation systems

Collaborative filtering, content-based information retrieval techniques, and context-based recommendation are the three basic recommendation systems that allow users to construct personalized music playlists. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

The purpose of services that provide streaming music and movies is to keep people on their platform for as long as possible. The recurring theme is that appropriate recommendations are required. Recent advancements in this industry have been significant, and these tips are advantageous to both businesses looking to maximize earnings and customers who are no longer overwhelmed by the quantity of options available. Making decisions is therefore made simple, and a good tip saves a lot of time.

It is assumed that if people rate music things similarly or behave similarly, they would rate other music items similarly as well. The sparse evaluation matrix is the major issue in collaborative filtering methods since most users only see a tiny portion of all music libraries, hence most assessments are not decided. Content-based filtering, on the other hand, makes suggestions based on the characteristics of the music pieces. It is possible to combine the preceding strategies, which is referred to as hybrid.

1) Collaborative approach

This kind of recommendation is based on an examination of both the behavior of the listeners and the behavior of all other platform users. The basic premise is that other users' opinions may be utilized to make a credible prediction about another user's preferences for an item that they have not yet rated: a user is given recommendations based on other users who share their tastes. Indeed, for years, we've asked our friends, family, and coworkers for recommendations when it came to music, restaurants, movies, and other entertainment. This mechanism is the one that is being attempted to be replicated here. This strategy (based on stars offered by other users) was pioneered by Netflix, but it is now widely used, notably for Spotify's Discover Weekly.

Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

It is assumed that if various users rate music things similarly or have similar behaviour, they would rate other music items similarly. The sparse evaluation matrix is the major issue in collaborative filtering methods since most users only see a tiny portion of all music libraries and hence most assessments are not decided.

When a new user is added to the system it will not initially have enough ratings for the system to find sufficiently similar users and thus the accuracy of the predictions will be limited. Another example of cold start is when a new item is added to the system. The item will not have any ratings so it won't be recommended to any users. There are many ways so solve this problem, asking new user for initial ratings or recommending the most popular items while gathering more information are two approaches to overcome the new user cold start problem. An approach to solve the new item problem is to implement content-based filtering in a hybrid approach which will be discussed in the coming section.

For example, if user X and user Y have rated a large number of products similarly, their relationship will be strong, and when user X offers a fresh high rating on an item that user Y has not yet reviewed, the system will suggest that item to user Y. A neighbourhood is a collection of people who share similar tastes, and predictions and suggestions may be generated by looking at the neighbor's rankings and user history. Pearson's correlation coefficient might be used to determine how similar two users are.

The fundamental assumption here is that the opinions of other users can be used to provide a reasonable prediction of another user's preferences for an item that they have not yet rated: a user is given recommendations based on users with whom they share the same tastes with. Indeed, during years, in order to choose music, restaurants, movies, etc.. We have been asking our friends, family, and colleagues to recommend something they liked. And it is this mechanism that is attempted to be reproduced here. Netflix was a pioneer of this method (based on stars given by other users) but it is now widely used, including for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

Implementing content-based filtering in a hybrid method to handle the new item problem is one solution, which will be described in the next section. The cold start problem for new users also applies to content-based filtering, because a new user will have no songs to filter and recommend.

As stated in the scope, our system does not rely on collaborative filtering because we do not have the necessary user base to support it. Deep neural networks are the only implementation that uses collaborative filtering, and even then, its use is limited. We had planned to use collaborative filtering in conjunction with content-based filtering and real-time data, but this was not feasible.

It is assumed that if people rate music things similarly or act in similar ways, they would rate other music items similarly. Because most users only encounter a tiny portion of all music libraries, the sparse evaluation matrix is the fundamental issue in collaborative filtering approaches. As a result, most assessments are not decided.

2) Content Based

The investigation of the content of the items candidates for suggestion is what content-based recommendation is all about. This method attempts to deduce the user's tastes in order to suggest goods that are similar in content to those they have previously enjoyed. This method does not require listener feedback; it is only based on sound similarity, which is calculated using information taken from previously heard songs.

The characteristics of each item are used in content-based filtering to locate items that are comparable. We may propose an item based on how similar it is to all other things in the dataset by assigning a score to how similar each item is.

We utilise the properties (loudness, pace, etc.) of each song in a Spotify playlist to calculate the average score of the entire playlist. Then we suggest a song that has a comparable score to the playlist but isn't on it.

Collaboration filtering has concerns with new items, while content-based filtering does not, as will be addressed in the following section. However, one disadvantage of content-based filtering is that it will only propose songs that are similar to what you currently listen to and will not suggest music that are different. For a new user who has only listened to a few songs, the suggestions will be based exclusively on these few listenings, making them incredibly predictable and disappointing.
However, if the extracted information is diverse enough and based on a variety of characteristics, the system may be able to make connections between songs that do not sound similar to the human ear.

A two-stage technique is used in content-based approaches to extract traditional audio content elements and forecast user preferences. In order to identify audio perceptual similarity, several studies have focused on extracting and comparing acoustic variables like as timbre and rhythm. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as content-based music recommendation. We provide music suggestions based on how similar the user's opinions of their favourite music were when they first heard it.

To locate related music, the project uses content-based filtering in addition to user id and real-time parameters.

The difficulty of collecting data does not apply to our project because Spotify offers a wide range of information associated with a track, which will be utilised as the material for comparing songs in our system.
However, due to a lack of user data, we will only be able to use three of the features: volume, mode, and tempo. It's impossible to say whether these characteristics are the best for describing a recording, while pace and loudness have been linked to heart rate.

The theory behind content-based filtering is that if a person is interested in one item, they are also likely to be interested in similar products. As a result, content-based filtering employs labels and characteristics to organise and filter objects. Different qualities can be retrieved depending on the object category, and items with comparable attributes are grouped together. Only the target user's history is required this method. The content of the items in the user's history is compared to the content of other domain items, and the domain items with the highest resemblance to the ones in the user's past are recommended. When proposing news articles, content-based filtering is a popular use.

The similarity between the elements is the basis for this strategy. It's a matter of extracting features to best describe the music in order to evaluate similarities. The Machine Learning algorithms then suggest the item that is the most similar to the ones that the user already likes.

A two-stage technique is used in content-based approaches to extract traditional audio content elements and forecast user preferences. In order to identify audio perceptual similarity, several studies have focused on extracting and comparing acoustic variables like as timbre and rhythm. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as content-based music recommendation. We provide music suggestions based on how similar the user's opinions of their favourite music were when they first heard it.

The main advantage of this approach is that an unknown music is just as
likely to be recommended as a currently popular one, or even a timeless one.
This allows new artists with a few" views" to be brought up as well. Moreover,
the problem of the cold start and in particular of the new items is thus avoided:

when new items are introduced into the system, they can be recommended
directly, without requiring integration time as is the case for recommendation
systems based on a collaborative filtering approach.

3) Context-based approach

We listen to music at a specific time, in a specific emotional state, and under specific conditions (party, job, etc.). And these predispositions will have a significant impact on how we feel about music. Although there are several uses for this sort of recommendation, such as tourist guide apps with adaptive ambient music, there are few actual examples.

The basic premise is that other users' opinions may be utilized to make a credible prediction about another user's preferences for an item that they have not yet rated: a user is given recommendations based on other users who share their tastes. Indeed, for years, we've asked our friends, family, and coworkers for recommendations when it came to music, restaurants, movies, and other entertainment. Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as content-based music recommendation. We provide music suggestions based on how similar the user's opinions of their favourite music were when they first heard it.

This mechanism is the one that is being attempted to be replicated here. In terms of results, the major goal was to establish a framework for consumers to use in order to assist them find the ideal tunes for them. This project seeks to discover the correlation and similarity between different songs, and then construct a recommendation system framework that suggests new music for your Spotify playlist based on that information.

Many obstacles continue to obstruct study in this subject. Indeed, the type of data to be considered is quite diverse and is dependent on both the environment (time, place, weather, culture, etc.) and the user (motion speed, emotions, heart rate, device luminosity, etc.). An even more serious problem is the scarcity of data for research purposes. In the actual world, retrieving them is difficult as well, because users may not always wish to communicate as much data from their cell phone sensors.

4) Hybrid approach

It is also feasible to develop a hybrid recommendation system by combining the preceding complementing strategies. It can also be based on less well-known techniques like location-based recommendations. This strategy can help with cold start and sparsity issues. Several implementations may be put up, the first of which combines the recommendation systems into one.

A hybrid method also allows for the evaluation of more criteria for each proposal. Demographic filtering, a sort of collaborative filtering that puts people in the same demographic together, can also be applied into such a system. Our recommender system uses both content-based and real-time data to filter results, making it a hybrid method. As previously said, we would have preferred to take a collaborative approach to address some of the difficulties with content-based filtering.

Personalized suggestions in the form of item rankings will arise from both collaborative and content-based screening. Because it compares the similarity of characteristics on audio signals, our method may be classified as content-based music recommendation. Using real-time data does not solve the difficulties with content-based filtering; rather, it is meant to improve the suggestions by customising them depending on the user's present condition.

It's also feasible to maintain many systems distinct and give weights to them, as well as the option to switch between them at anytime. Finally, outputs from one system may be extracted and utilised as input for a subsequent system.

1.2 Problem Statement

With commercial music streaming service which can be accessed from mobile devices, the availability of digital music currently is abundant compared to previous era. Sorting out all this digital music is a very time-consuming and causes information fatigue. Therefore, it is very useful to develop a music recommender system that can search in the music libraries automatically and suggest suitable songs to users.

In the long-term, the goal is not only to recommend existing songs but also to generate songs adapted to the musical taste of the user. During this master thesis I focused on the recommendation part while exchanging with a colleague in charge of the generation part. The music which selected by the user is used as the basis music for recommendations. The features of basis music are extraction vector is obtained from the best genre prediction model in previous step. Next, the values of cosine similarity are sorted from the largest to the smallest value. The future of the project will consisting gathering these two parts in order to have a fully functional recommendation system.

Because real-time data changes rapidly, an algorithm based on it must be efficient.
We want advice that are relevant to the current situation rather than prior situations. Many researchers are presently focusing on machine learning approaches such as neural networks, and they are also becoming more prominent in the field of recommender systems. In terms of results, the major goal was to establish a framework for consumers to use in order to assist them find the ideal tunes for them. This project seeks to discover the correlation and similarity between different songs, and then construct a recommendation system framework that suggests new music for your Spotify playlist based on that information.

They can not only manage the ever-increasing amount of data, but they also increase in quality in proportion to the amount of data evaluated, thanks to the learning algorithms. Indeed, during years, in order to choose music, restaurants, movies, etc.. We have been asking our friends, family, and colleagues to recommend something they liked. And it is this mechanism that is attempted to be reproduced here. Netflix was a pioneer of this method (based on stars given by other users) but it is now widely used, including for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

Machine learning has become significantly more viable than it has been traditionally as the amount of data has expanded and the processing capacity of computers has improved. The algorithms are made to look for patterns.

Because our music choices and our present emotional state are so closely linked, real-time data sources are extremely important for music suggestions. Certain songs or styles of music can affect our mood in various ways, and our musical choices are frequently linked to our mood.

Music selections are also linked to the listener's current activities. Even if we have a certain musical taste, our tastes will shift depending on what we are doing. When a person is working out at the gym, for example, they will likely listen to different tunes than when they are attempting to go asleep at night.

The aim of this thesis is to explore the different recommendation approaches, the available datasets, the ways to take into account the user's preferences and the machine learning methods in order to build a suitable recommendation system. The music which selected by the user is used as the basis music for recommendations. The features of basis music are extraction vector is obtained from the best genre prediction model in previous step. Next, the values of cosine similarity are sorted from the largest to the smallest value. One important part was only dedicated to determine how to evaluate this recommendation system. This project will be introduced to the members of the company and will take the form of an application. The user will be asked to upload a music and the application will recommend some music to be listened to afterwards.

1.3 Objective of the Major Project

The goal of this project was to learn about machine learning and its fundamental concepts, as well as numerous data mining approaches and algorithms. Another goal was to become familiar with a variety of machine learning algorithms and how to use them. Learning algorithms alone does not make you an engineer; the true challenge is determining which method is best for a certain project.

In terms of results, the major goal was to establish a framework for consumers to use in order to assist them find the ideal tunes for them. This project seeks to discover the correlation and similarity between different songs, and then construct a recommendation system framework that suggests new music for your Spotify playlist based on that information.

Long-term, the objective is to not just propose current songs, but also to create songs according to the user's musical preferences. Throughout my master's thesis, I concentrated on the recommendation section while corresponding with a colleague who was in charge of the generating section. The project's future will consist of bringing these two components together to create a fully working recommendation system.

Indeed, during years, in order to choose music, restaurants, movies, etc.. We have been asking our friends, family, and colleagues to recommend something they liked. And it is this mechanism that is attempted to be reproduced here. Netflix was a pioneer of this method (based on stars given by other users) but it is now widely used, including for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

The objective is to recommend the user the types of songs that he would like by comparing his taste i.e., his playlist of music with the songs available in the dataset. By using music recommender system, the music provider can predict and then offer the appropriate songs to their users based on the characteristics of the music that has been heard previously.

Major section was devoted to determining how to assess this recommendation system. This project will be presented to the company's employees in the form of an application. The user will be requested to submit a song, and the program will then offer several songs to listen to.

The idea is not just to recommend current songs but also to produce songs based on the user's musical preferences. During this master's thesis, I concentrated on the suggestion portion while corresponding with a colleague in charge of the generating portion. The project's future will consist of combining these two components to create a fully working recommendation system.

Only one significant section was devoted to determining how to assess this suggestion system. This project will be presented to the company's employees in the form of an application. The user will be invited to upload music, following which the program will suggest some songs to listen to.

The recommendation systems that really emerged in the 1990s have developed strongly in recent years, especially with the introduction of Machine Learning and networks. Indeed, on the one hand, the growing use of the current digital environment, characterized by an overabundance of information has allowed us to obtain large user databases. On the other hand, the increase in computing power made it possible to process these data especially thanks to Machine Learning when human capacities were no longer able to carry out an exhaustive analysis of so much information.

1.4 Methodology

a) Implementation of Music Recommender System

The recommender system is done by calculating cosine similarity of extraction features (equation 1) from one music to another music. The extraction features are in vector form; thus, it is possible to calculate their distance. First, we chose one music for each genre as the basis for the recommender system. Next the prediction of the basis music genre is calculated based on neural networks. The feature vectors that produce before the classification layer are used as a basis for recommendations. After the basis music features are obtained, cosine similarity calculations are performed on other music features.

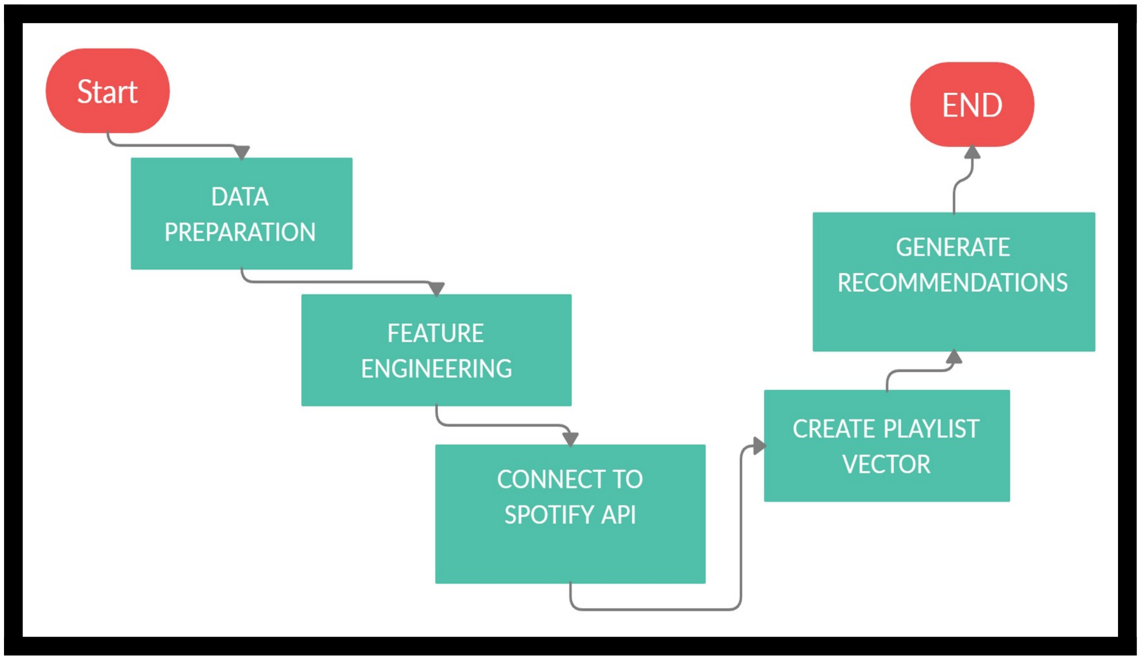
The study of the content of the items considered for suggestion is content-based recommendation. This method attempts to deduce the user's tastes in order to suggest goods that are similar in content to those they have previously enjoyed. This approach does not require listener input; it is only based on sound similarity, which is calculated using information taken from previously heard songs.

The commonalities between the components are used in this strategy. It is a question of extracting characteristics that best characterize the music in order to assess similarities. The Machine Learning algorithms then suggest the item that is most similar to what the consumer already loves.

As a result, item profiles based on characteristics derived from things are required. Furthermore, this strategy necessitates the creation of user profiles based on their preferences as well as their platform history.

In the numerator, the calculation is done by calculating dot product of both vectors and in the denominator, the calculation is done by calculating the vector lengths. The obtained value of cosine similarity is between -1 to 1. By sorting the values from the largest to the smallest, the recommendations can be made by choosing several music with the largest cosine similarity. In this research, the number of recommendations is set to be five music. In our experiments, the recommender system uses two methods. The first method only uses the value of cosine similarity, while the second method uses both the value of cosine similarity and information of music genre.

The music which selected by the user is used as the basis music for recommendations. The features of basis music are extraction vector is obtained from the best genre prediction model in previous step. Next, the values of cosine similarity are sorted from the largest to the smallest value. Finally, the first five music with the greatest value is used as recommendations.



1.5 Organization

We were awestruck with Spotify's recommendation engine. We always wondered how Spotify manages to recommend that perfect song, playlist or even that 'daily mix' . t has been found that CF generally gives better recommendations than CB. However, this is only true if there is usage data available, such as the ratings given to previous tracks. If this is not the case, then it will not prove accurate results and, consequently, suffer from the Cold-Start problem, which includes two categories of problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them,.

We now have more technology than ever before to ensure that if you're the smallest, strangest musician in the world, doing something that only 20 people in the world will dig, we can now find those 20 people and connect the dots between the artist and listeners. This has been the motivation for this project to use various machine learning techniques and to develop a music recommendation engine similar to that of Spotify, which takes music listening experience to another level.

The MRS methods that are used most frequently are Collaborative Filtering (CF), which is the process where the system analyses a user's preferences from his or her historical usage data and Content-Based (CB) techniques which analyze the item descriptions to identify songs that are of particular interest to the user CF recommends songs to people that are based on other users who share the same musical preferences as them. In contrast, CB uses the metadata and content of a song, such as the album, the artist and the genre in order to recommend songs.

It has been found that CF generally gives better recommendations than CB. However, this is only true if there is usage data available, such as the ratings given to previous tracks. If this is not the case, then it will not prove accurate results and, consequently, suffer from the Cold-Start problem, which includes two categories of problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them, e.g., the ratings, is insufficient. The second occurs when a new user joins a system and not much is known about him or her (Crane, 2011). Consequently, the recommendation system is unable to give recommendations that have been personalized for users until they begin to rate different items.

CB is not usually as sensitive to the Cold-Start problem, as it has the capacity to still recommend items, even if it does not have enough ratings. CB has its own issues, however, and so is not without problems The biggest of these is that it will recommend songs the user is familiar with already. The objective of this thesis is to put forward a 3 solution to solve the Cold-Start problem in MRS and also offer users new music recommendations they have never heard before. Therefore, the motivation for this thesis is to increase the performance of a music recommender system which includes the diversity and the novelty by reducing the effects of the Cold-Start problem through combining three recommendation techniques into hybrid approaches. Thus, people will be provided with a solution that will enable them to receive more accurate and better recommendations that are based on their own music preferences.

Chapter 02: LITERATURE SURVEY

We were awestruck with Spotify's recommendation engine. We always wondered how Spotify manages to recommend that perfect song, playlist or even that 'daily mix' . We now have more technology than ever before to ensure that if you're the smallest, strangest musician in the world, doing something that only 20 people in the world will dig, we can now find those 20 people and connect the dots between the artist and listeners. This has been the motivation for this project to use various machine learning techniques and to develop a music recommendation engine similar to that of Spotify, which takes music listening experience to another level. Music Recommendation Systems.

Recommender systems help consumers deal with the problem of information overload by providing them with individualised, unique content and service suggestions. Various methods for developing recommendation systems have recently been created, including collaborative filtering, content-based filtering, and hybrid filtering. The collaborative filtering approach is the most developed and widely used. Collaborative filtering suggests things by locating other users who have similar tastes to the current user and using their recommendations. Collaborative recommender systems have been used in a variety of settings. problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them,.

The common characteristics in these systems are constant when using users' preferences compared with users' context (location, mood, weather, etc.). For instance, in the library when people are sitting there maybe they need quiet and melodious music to listen according to the environment where they are in. Last.fm, All music, Spotify, Pandora and Shazam are commercial music recommendation systems which are considered to be excellent systems by focusing on the music already played in order to help the users to find more music. Users are able to connect to a web-based music streaming service to access the recommendations. All the tracks that are played on this stream are recommended.

It is Based on songs or artists which users either upload from your iTunes playlists or add as favourites on the site where users start managing their library of music with tags and keep tracking of the music the friends who listening to and getting multiple recommendations per song played. Additionally, this app filters recommendations by decade, genre, and popularity, as well as builds fabulous playlists (Song et al., 2012).t has been found that CF generally gives better recommendations than CB. However, this is only true if there is usage data available, such as the ratings given to previous tracks. If this is not the case, then it will not prove accurate results and, consequently, suffer from the Cold-Start problem, which includes two categories of problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them,.

Meanwhile, many researchers have used social media (Twitter & Facebook) to identify user's mood (tension, depression, anger, vigor, fatigue, confusion) and also identify user's personality (openness, conscientiousness, extraversion, agreeableness, neuroticism) where these are very important factors which influence on user's music taste (Wang et al., 2014; Roberts et al., 2012; Pandarachalil et al., 2015; Ross et al., 2009; Bachrach et al., 2012; Back et al., 2010) and also contextual features (location & event) can lead to different emotional effects due to objective features of the situation or subjective perceptions of the listeners (Scherer et al., 2001).

Music lyrics are also considered to be one of emotional presentation because they include some kinds of implicit thinking, thus we can fully understand emotions and their associated thinking in each song (Nunes and Jannach, 2017; Tintarev and Masthoff, 2008). Cano et al. (2017) mentioned that there is a strong relation between the user mood and listening to the music. The people may want to listen to music which has the same mood of them when they are in specific mood and in contrast the people want to listen to different kind of music which encourage them to enhance their mood and this thing depend on the psychological studies and therefore, the author produced a contextual mood-based music recommender system which is able to regulate the driver's mood and also try to put the driver in a positive mood when driving because listening to the music while driving has always been one of the most favourite activities carried out by people. Finally, similarly, active learning approaches suffer from various limitations.

Chapter 03: MAJOR PROJECT SDLC

3.1 Requirements on Major Project

3.1.1 Functional Requirements

The project's functional requirement definition is divided into three categories: user needs, security requirements, and device requirements, each of which is discussed in depth below:

Requirement of the user: To explore the identification for the music suggestion, the user must have an account on the framework and have listened to at least one song.

The project's functional requirement definition is divided into three categories: user needs, security requirements, and device requirements, each of which is discussed in depth below:

Requirement of the user: To explore the identification for the music suggestion, the user must have an account on the framework and have listened to at least one song.

The project's functional requirement definition is divided into three categories: user needs, security requirements, and device requirements, each of which is discussed in depth below:

Requirement of the user: To explore the identification for the music suggestion, the user must have an account on the framework and have listened to at least one song.

The project's functional requirement definition is divided into three categories: user needs, security requirements, and device requirements, each of which is discussed in depth below:

Requirement of the user: To explore the identification for the music suggestion, the user must have an account on the framework and have listened to at least one song.

The project's functional requirement definition is divided into three categories: user needs, security requirements, and device requirements, each of which is discussed in depth below:

Requirement of the user: To explore the identification for the music suggestion, the user must have an account on the framework and have listened to at least one song.

3.1.2 Non-Functional Requirements

i. Performance: The framework will produce results that are quick, precise, and trustworthy.

ii. Capacity and Scalability: The framework will be able to store identification that has been registered in the database.

iii. Availability: The framework will be available to clients anytime an Internet connection is available.

iv. Recovery: In the event of a server failure or inaccessibility, the framework should be able to recover and store any data loss or excess.

v. Flexibility and Portability: The system will be accessible at any time and from any location

i. Performance: The framework will produce results that are quick, precise, and trustworthy.

ii. Capacity and Scalability: The framework will be able to store identification that has been registered in the database.

iii. Availability: The framework will be available to clients anytime an Internet connection is available.

iv. Recovery: In the event of a server failure or inaccessibility, the framework should be able to recover and store any data loss or excess.

v. Flexibility and Portability: The system will be accessible at any time and from any location

i. Performance: The framework will produce results that are quick, precise, and trustworthy.

ii. Capacity and Scalability: The framework will be able to store identification that has been registered in the database.

iii. Availability: The framework will be available to clients anytime an Internet connection is available.

iv. Recovery: In the event of a server failure or inaccessibility, the framework should be able to recover and store any data loss or excess.

v. Flexibility and Portability: The system will be accessible at any time and from any location

i. Performance: The framework will produce results that are quick, precise, and trustworthy.

ii. Capacity and Scalability: The framework will be able to store identification that has been registered in the database.

iii. Availability: The framework will be available to clients anytime an Internet connection is available.

iv. Recovery: In the event of a server failure or inaccessibility, the framework should be able to recover and store any data loss or excess.

v. Flexibility and Portability: The system will be accessible at any time and from any locationi. Performance: The framework will produce results that are quick, precise, and trustworthy.

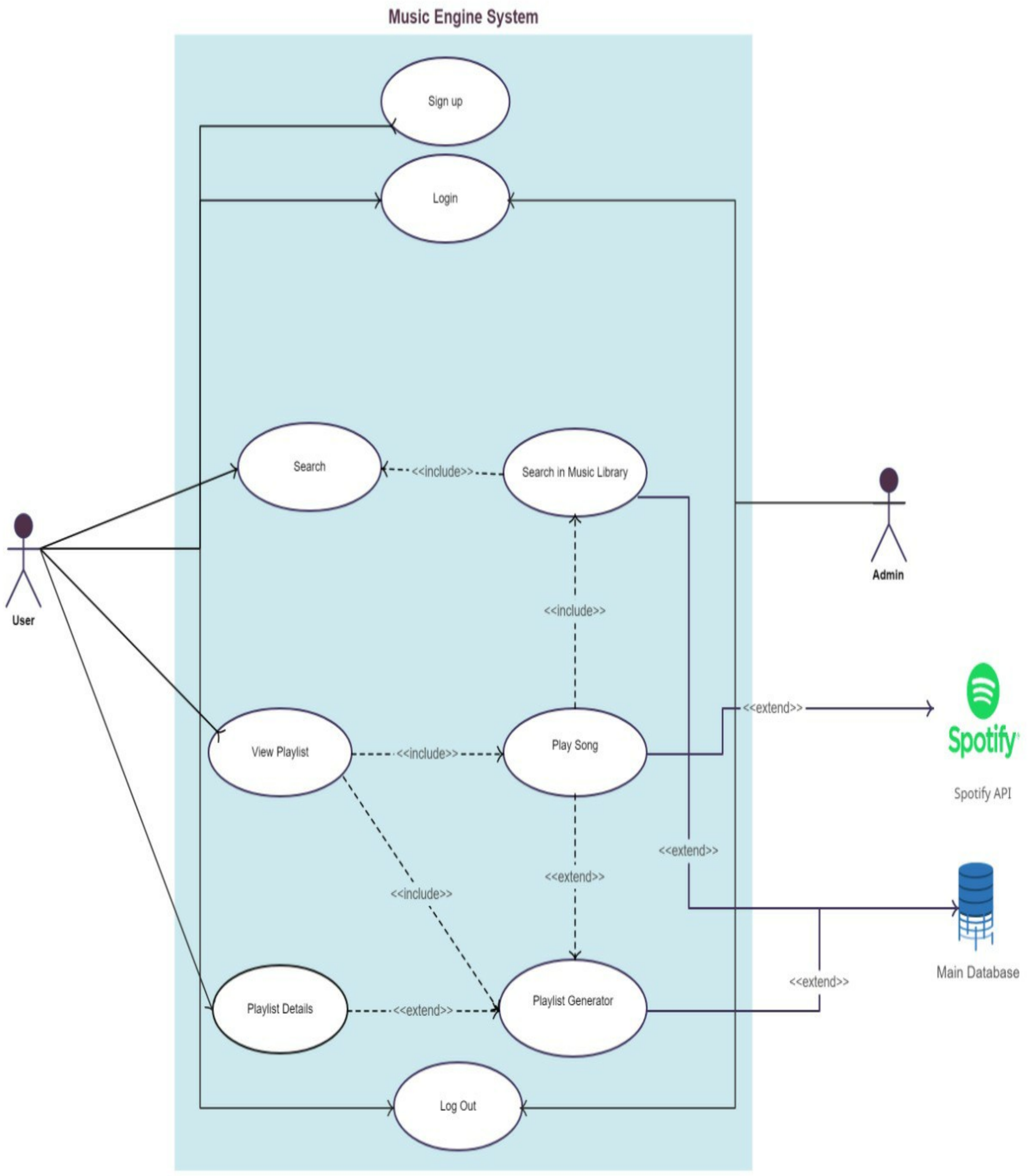
ii. Capacity and Scalability: The framework will be able to store identification that has been registered in the database.

iii. Availability: The framework will be available to clients anytime an Internet connection is available.

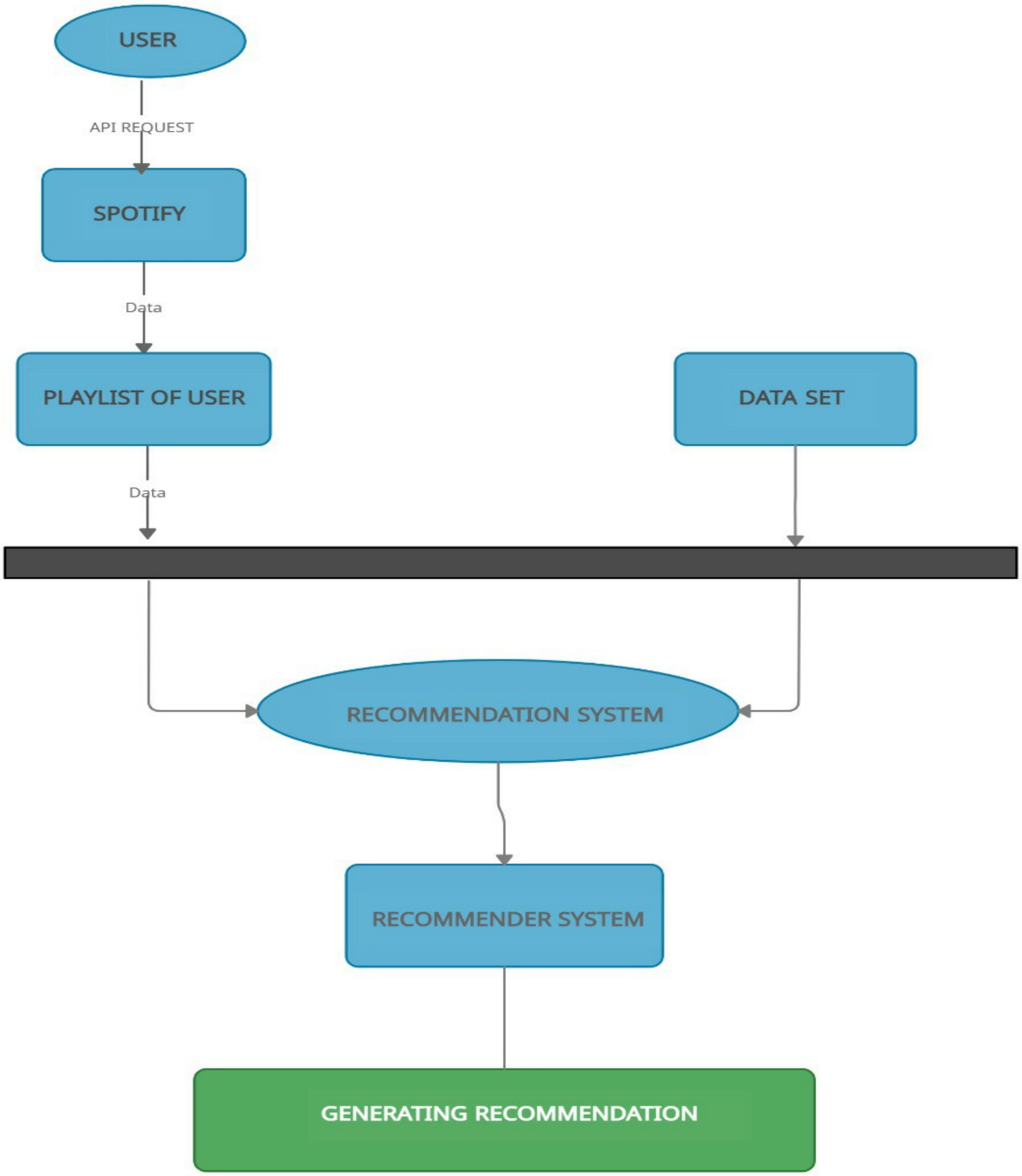
iv. Recovery: In the event of a server failure or inaccessibility, the framework should be able to recover and store any data loss or excess.

v. Flexibility and Portability: The system will be accessible at any time and from any location

3.2 Use Case Diagram of the Major Project



3.3 DFD Diagram of the Major Project



Chapter 04: IMPLEMENTATION OF THE MAJOR PROJECT

4.1 Date Set Used in the Major Project

Spotify Dataset 1922-2021, ~600k Tracks Audio features of ~600k songs released in between 1922 and 2021. this dataset contains audio features and

metadata of each song.

This dataset then went through different data mining techniques to make it suitable for further analysis .In simple terms data mining means firstly extracting the useful data and then making that data suitable to be used in the algorithms by cleaning the data and transforming it. This dataset then went through different data mining techniques to make it suitable for further analysis This dataset then went through different data mining techniques to make it suitable for further analysis .

source: https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

limitations :

1. only genre metadata was provided
2. Since, the dataset consists of only audio features and no data related to the user's listening history , therefore we cannot perform collaborative filtering techniques .
3. kaggle dataset doesn’t consist of all of the songs in my playlist .

This dataset then went through different data mining techniques to make it suitable for further analysis .

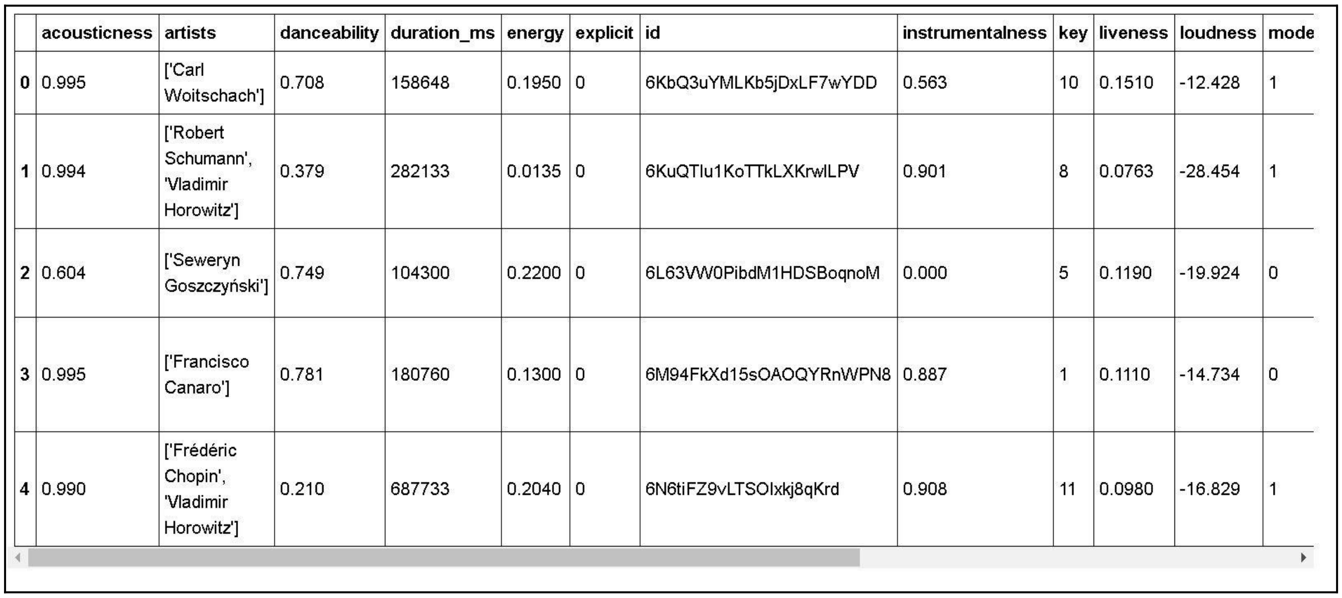
4.2 Date Set Features

4.2.1 Types of Data Set

The dataset is in the form of csv files which have been taken from kaggle and then various data mining techniques are applied to it to extract the information. Dataset had various attributes like user id, song counts, language, artist, genre, and year. The dataset was further divided in four parts namely data by genre, data by artist, data by year and data and were stored in individual csv files to analyze and to train the model.

4.2.2 Number of Attributes, fields, description of the data set

- audio features of tracks 170k rows
- audio features of artists, 30k rows
- audio features of years,100 rows
- audio features of genres



	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode
0	0.995	['Carl Woitschach']	0.708	158648	0.1950	0	6KbQ3uYMLKb5jDxLF7wYDD	0.563	10	0.1510	-12.428	1
1	0.994	['Robert Schumann', 'Vladimir Horowitz']	0.379	282133	0.0135	0	6KuQTIu1KoTTkLXKrwlLPV	0.901	8	0.0763	-28.454	1
2	0.604	['Seweryn Goszczyński']	0.749	104300	0.2200	0	6L63VW0PibdM1HDSBoqnoM	0.000	5	0.1190	-19.924	0
3	0.995	['Francisco Canaro']	0.781	180760	0.1300	0	6M94FkXd15sOAOQYRnWPN8	0.887	1	0.1110	-14.734	0
4	0.990	['Frédéric Chopin', 'Vladimir Horowitz']	0.210	687733	0.2040	0	6N6tiFZ9vLTSOIxkj8qKrd	0.908	11	0.0980	-16.829	1

4.3 Design of Problem Statement

We used numerous methods to create, build, and assess music recommendation systems in this study. Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs.

Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs. Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs. Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs.

We'll take the following method to solving the problem:

i. Collecting information (choosing dataset)

ii. Using data mining techniques to analyse a dataset (data pre-processing and data cleaning)

iii. Preparing the dataset for usage in various methods

iv. Choosing the best algorithms for the job v. Using different machine learning algorithms

vi. Analyzing the data and selecting the best model

4.4 Algorithm / Pseudo code of the Project Problem

The algorithm used for the Project Problems is:

Cosine Similarity

The metric cosine similarity assesses how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

Cosine Similarity is a statistic that determines how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

The metric cosine similarity assesses how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

The metric cosine similarity assesses how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

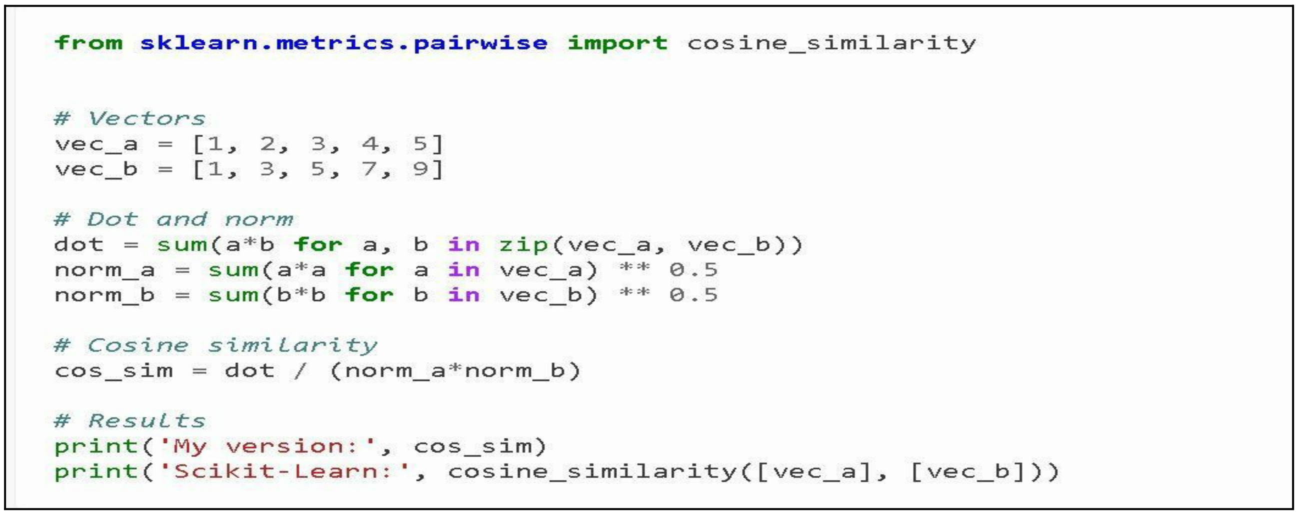
Cosine Similarity is a statistic that determines how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the

vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

Cosine similarity is a commonly used technique for determining similarity that can be found in major libraries and tools such as Matlab, SciKit-Learn, and TensorFlow.

In Swift, here's a fast implementation of the cosine similarity logic.



```
from sklearn.metrics.pairwise import cosine_similarity


# Vectors
vec_a = [1, 2, 3, 4, 5]
vec_b = [1, 3, 5, 7, 9]

# Dot and norm
dot = sum(a*b for a, b in zip(vec_a, vec_b))
norm_a = sum(a*a for a in vec_a) ** 0.5
norm_b = sum(b*b for b in vec_b) ** 0.5

# Cosine similarity
cos_sim = dot / (norm_a*norm_b)

# Results
print('My version:', cos_sim)
print('Scikit-Learn:', cosine_similarity([vec_a], [vec_b]))
```

The metric cosine similarity assesses how similar two or more vectors are. The cosine similarity is the cosine of the angle between vectors. In most cases, the vectors are non-zero and belong to the inner product space. The cosine similarity is formally described as the difference between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

The range of cosine similarity values is limited to 0 to 1. Similarity is calculated using the cosine of the angle between the two non-zero vectors A and B.The characteristics of each item are used in content-based filtering to locate items that are comparable. We may propose an item based on how similar it is to all other things in the dataset by assigning a score to how similar each item is.

We utilise the properties (loudness, pace, etc.) of each song in a Spotify playlist to calculate the average score of the entire playlist. Then we suggest a song that has a comparable score to the playlist but isn't on it.

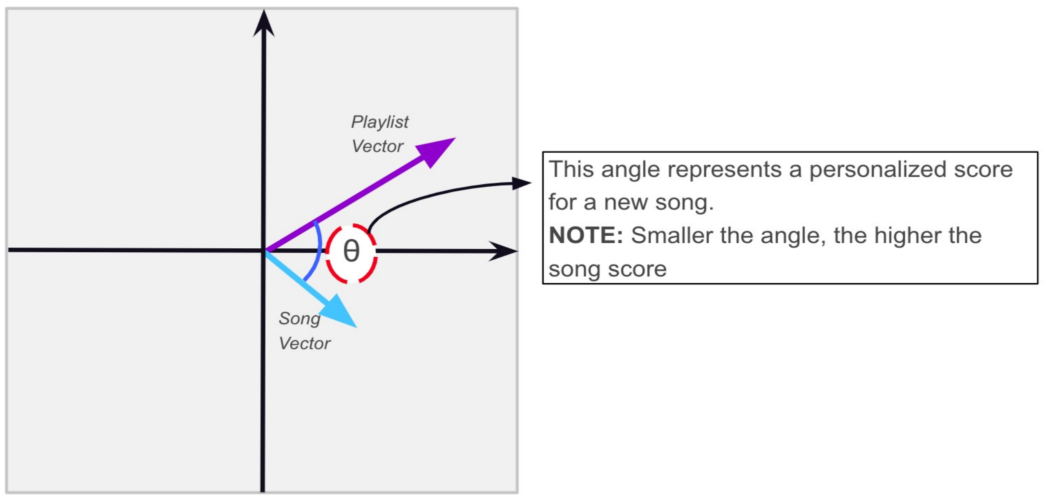
Assume the two vectors intersect at a 90-degree angle. In such case, the cosine similarity will be 0, indicating that the two vectors are orthogonal or perpendicular.

similarity measurement approaches 1. The photographs below show this in further detail.

Calculating Scores for New Songs

Playlist Vector is compared to individual Song Vectors using *cosine similarity* to generate recommendations:

Cosine Similarity Explained:



This angle represents a personalized score for a new song.
NOTE: Smaller the angle, the higher the song score

TF-IDF

The abbreviation for term frequency is TF-IDF. Records having a Document Frequency that is inverse. It is the process of identifying the importance of a word in a sequence or corpus to a text. The number of times a word appears in the text increases its significance, although this is countered by the corpus's word frequency (data-set).

The TF-IDF determines the relevance of a phrase by considering its value in a single text and scaling it by its importance across all documents.

The most commonly used recommendation algorithm .We call it a “user-user” algorithm because it recommends an item to a user if similar users liked this item before. The similarity between two users is computed from the amount of items they have in common in the dataset.

The term Frequency is shortened as TF-IDF. Records with an inverted Document Frequency. It is the process of identifying the importance of a word in a sequence or corpus to a text. The meaning of a word rises in proportion to how many times it appears in the text, but this is neutralised by the corpus's word frequency.

The TF-IDF determines the relevance of a phrase by considering its value in a single text and scaling it by its importance across all documents.

Term Frequency: The acronym TF-IDF stands for Frequency. Records with a Document Frequency that is reversed. It is the process of determining the significance of a word in a sequence or corpus in relation to a text. The number of times a word appears in the text increases its significance, although this is offset by the corpus's word frequency.

The TF-IDF determines the relevance of a phrase by considering its value in a single text and scaling it by its importance across all documents.

The acronym TF-IDF stands for Frequency. Records with a Document Frequency that is reversed. It is the process of determining the significance of a word in a sequence or corpus in relation to a text. The significance of a word increases in proportion to how many times it appears in the text, but this is offset by the corpus's word frequency.

Document Frequency: This is similar to TF in that it verifies the meaning of the text across the full corpus collection. The only difference is that in document d, TF represents the frequency counter for a term t, but in document set N, df reflects the number of times the phrase t appears. To put it another way, the number of times the phrase appears in publications is DF.

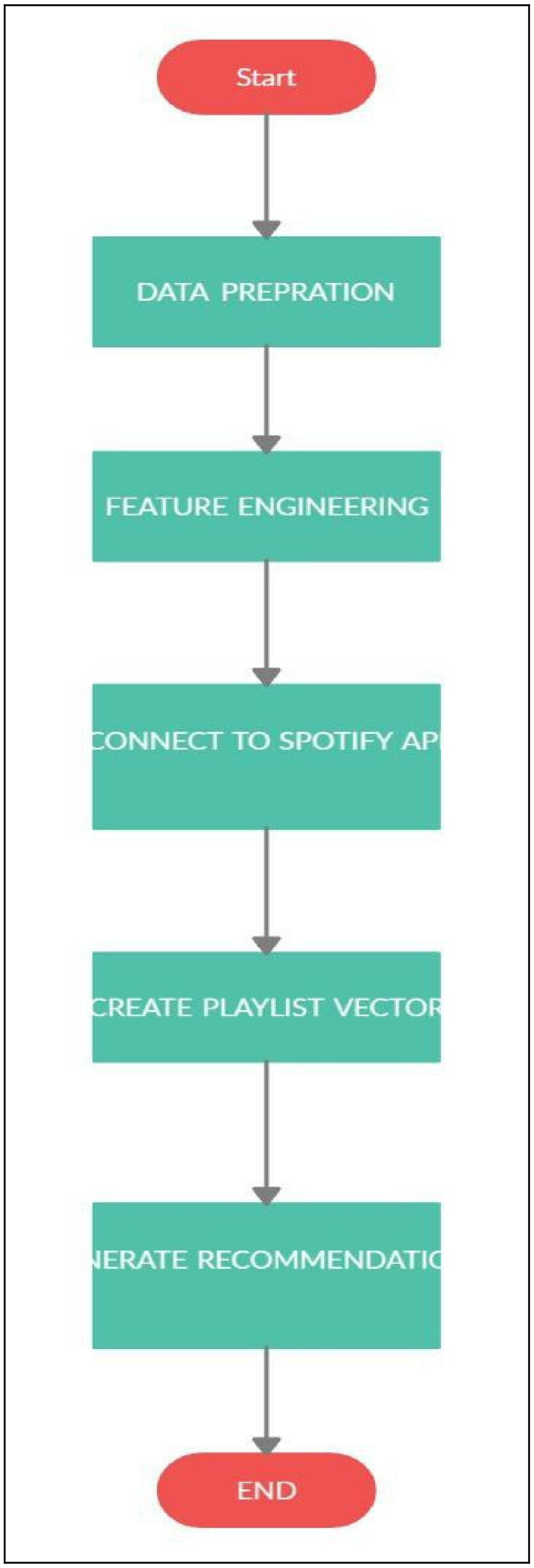
```
df(t) = occurrence of t in documents
```

Inverse Document Frequency (IDF) is a test that assesses a word's relevance. The primary purpose of the search is to locate records that are related to the requirement. The term frequencies cannot be used to assess the weight of a phrase in the document since tf considers all terms to be equally significant.Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs. Thanks to significant improvements in the disciplines of music information retrieval and (audio) signal processing over the last decades, content-based characteristics retrieved from music audio signals have traditionally played a significantly larger role than in other domains. DL approaches can work on a significantly wider and more comprehensive collection of low- and mid-level audio properties, thanks to established tools and expertise from these domains.

In contrast to the movie or product domains, where people frequently dislike repetitive suggestions of the same products, listeners can like repeated recommendations. The network's output is commonly a vector over items (or playlists) that provides the probability of fit, thanks to the probabilistic handling of items in DL architectures.

Music suggestion is a difficult challenge since we must structure music in such a way that we can propose users' favourite songs, which is never a sure thing. It is dynamic, and it is occasionally influenced by variables other than the listening history of users or songs. To find the document frequency of the keyword t, start by counting the number of documents that include it:

4.5 Flow graph of the Major Project Problem

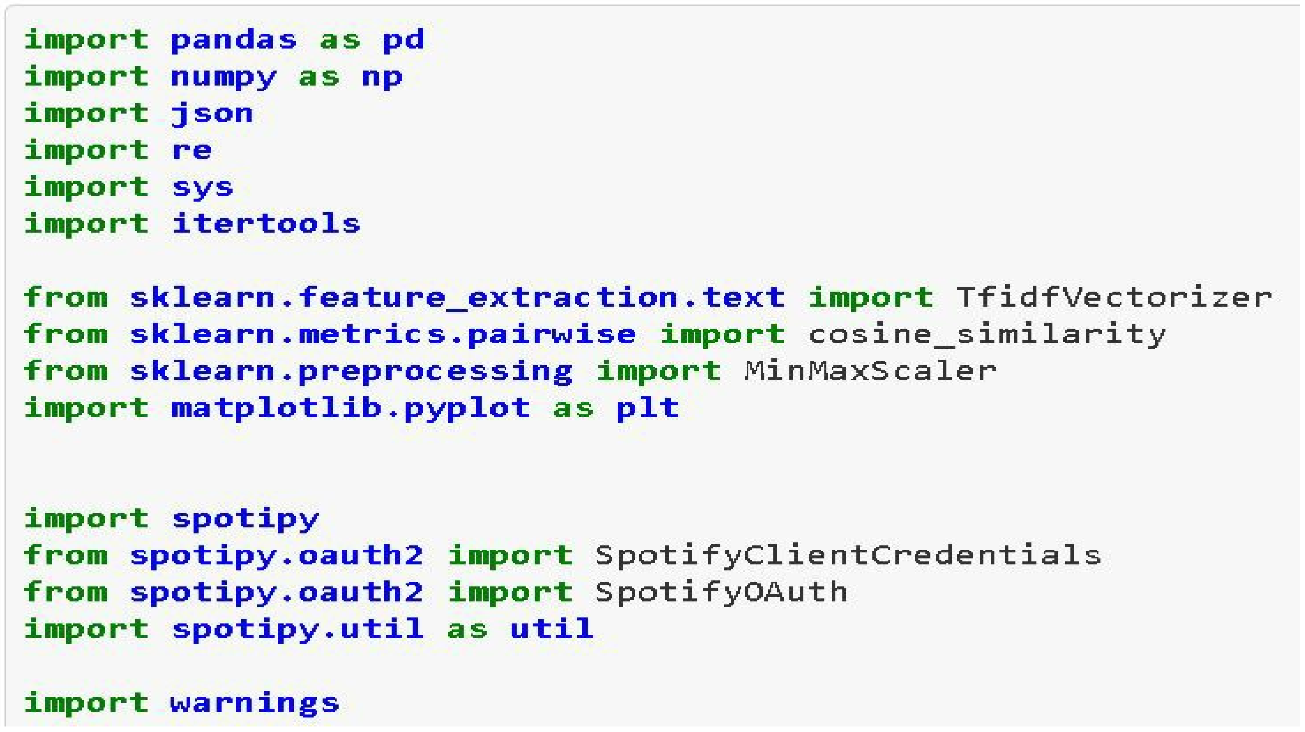

Start
DATA PREPRATION
FEATURE ENGINEERING
CONNECT TO SPOTIFY API
CREATE PLAYLIST VECTOR
NERATE RECOMMENDATIO
END

4.6 Screenshots of the various stages of the Project

Step 1- Importing libraries

TensorFlow

- NumPy
- SciPy
- Pandas
- Matplotlib
- Keras
- SciKit-Learn

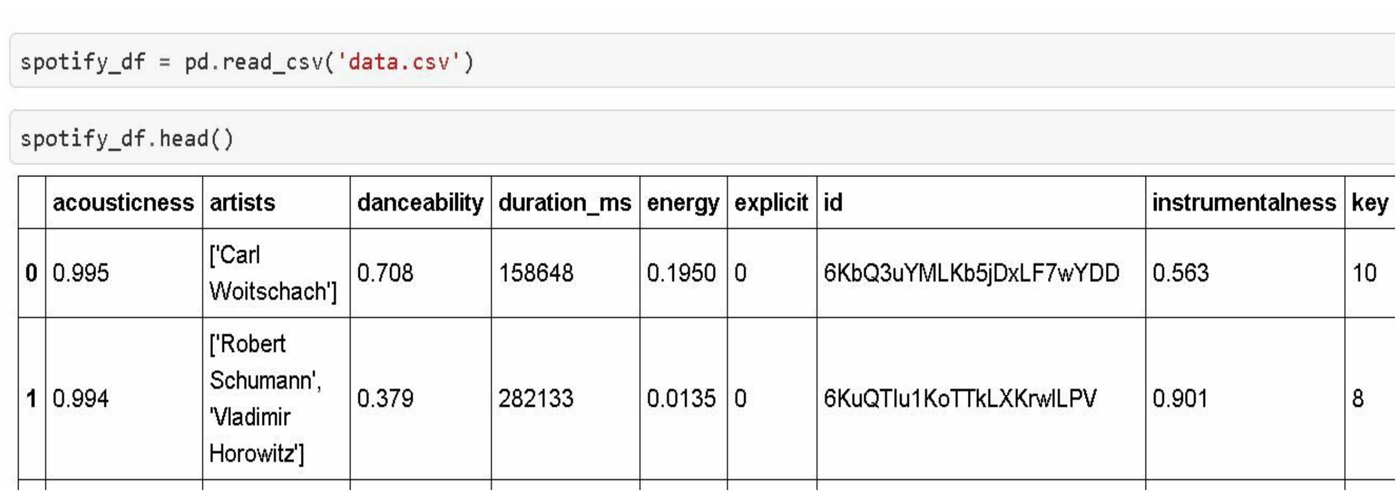


```
import pandas as pd
import numpy as np
import json
import re
import sys
import itertools

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt


import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
from spotipy.oauth2 import SpotifyOAuth
import spotipy.util as util

import warnings
```

Libraries used

1. Numpy
2. Pandas
3. Matplotlib
4. scikit-learn
5. sklearn.metrics
6. sklearn.model_selection
7. sklearn.feature_extraction.text
8. sklearn.ensemble
9. Spotipy is a little Python library for accessing the Spotify Web API. Spotipy gives you complete access to all of the Spotify platform's music data.

Step-2 Uploading



```
spotify_df = pd.read_csv('data.csv')
```

```
spotify_df.head()
```

	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key
0	0.995	['Carl Woitschach']	0.708	158648	0.1950	0	6KbQ3uYMLKb5jDxLF7wYDD	0.563	10
1	0.994	['Robert Schumann', 'Vladimir Horowitz']	0.379	282133	0.0135	0	6KuQTIu1KoTTkLXKrwlLPV	0.901	8

Dataset- Step-3 Cleaning the data

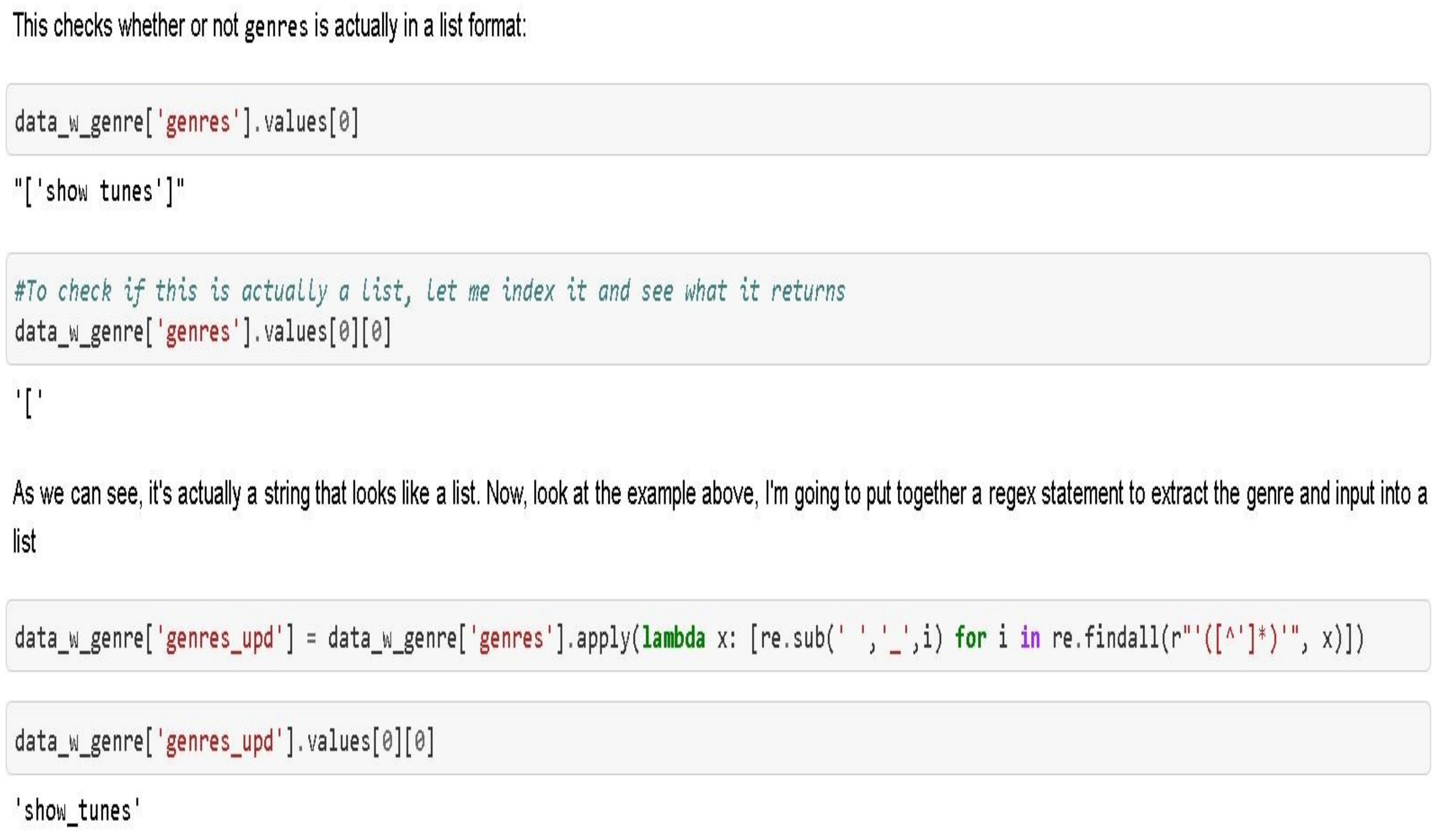
The "Comma-Separated List," or CSV, is the most basic and well-supported file type for tabular data on Kaggle. A header row with human-readable field names should be included in CSVs uploaded to Kaggle. For example, a CSV representation of a shopping list with a header row looks like this:

id,type,quantity

0,bananas,12

1,apples,7

CSVs are the most popular file type on Kaggle and the best option for tabular data.



This checks whether or not `genres` is actually in a list format:

```
data_w_genre['genres'].values[0]
```

```
"['show tunes']"
```

```
#To check if this is actually a list, let me index it and see what it returns
data_w_genre['genres'].values[0][0]
```

```
'['
```

As we can see, it's actually a string that looks like a list. Now, look at the example above, I'm going to put together a regex statement to extract the genre and input into a list

```
data_w_genre['genres_upd'] = data_w_genre['genres'].apply(lambda x: [re.sub(' ','_',i) for i in re.findall(r"'([^']*)'", x)])
```

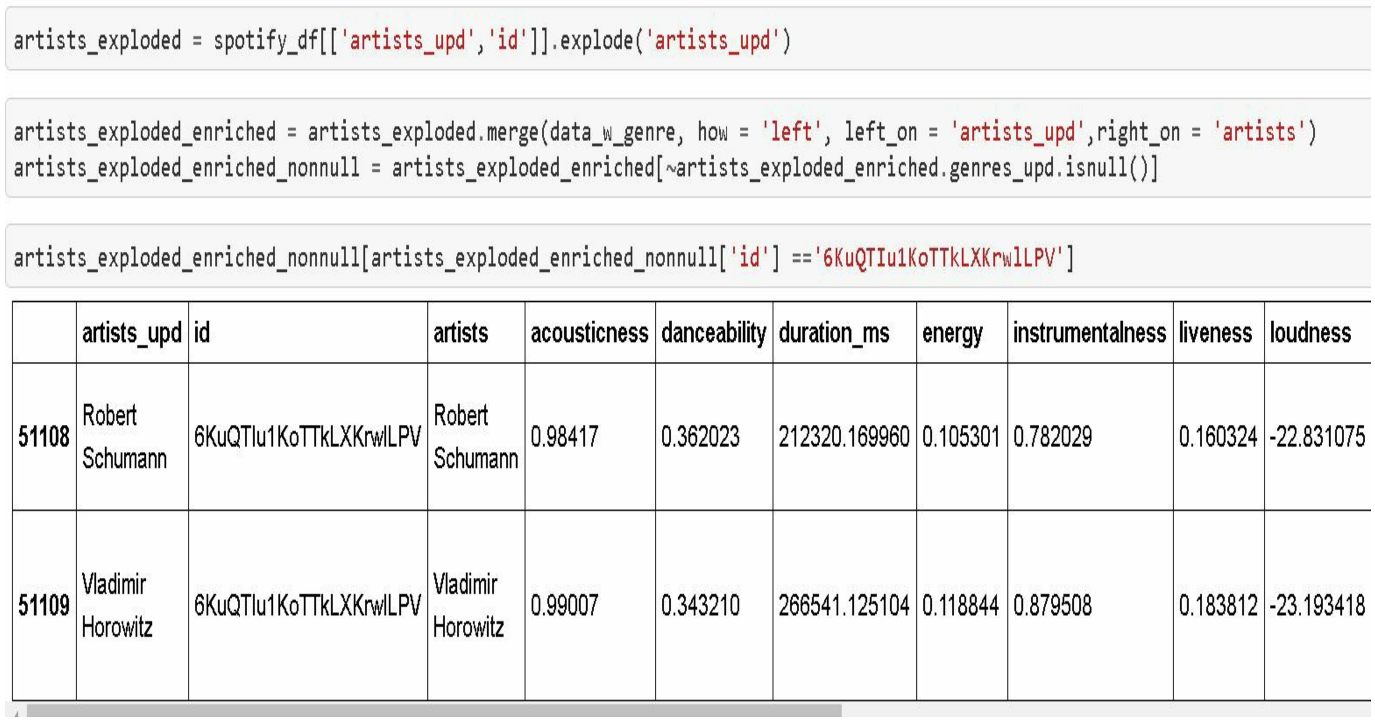
```
data_w_genre['genres_upd'].values[0][0]
```

```
'show_tunes'
```

Step-4 Making dataset ready to be used in various algorithms

Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.

Burst evious dataset's artists column so that each artist inside a song has their own row, then merge data w genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset. Expand the previous dataset's artists column so that each artist inside a song has their own row, then combine data w genre with the exploded dataset in Step 1 so that the previous dataset is enriched with the genre dataset.



```
artists_exploded = spotify_df[['artists_upd','id']].explode('artists_upd')
```

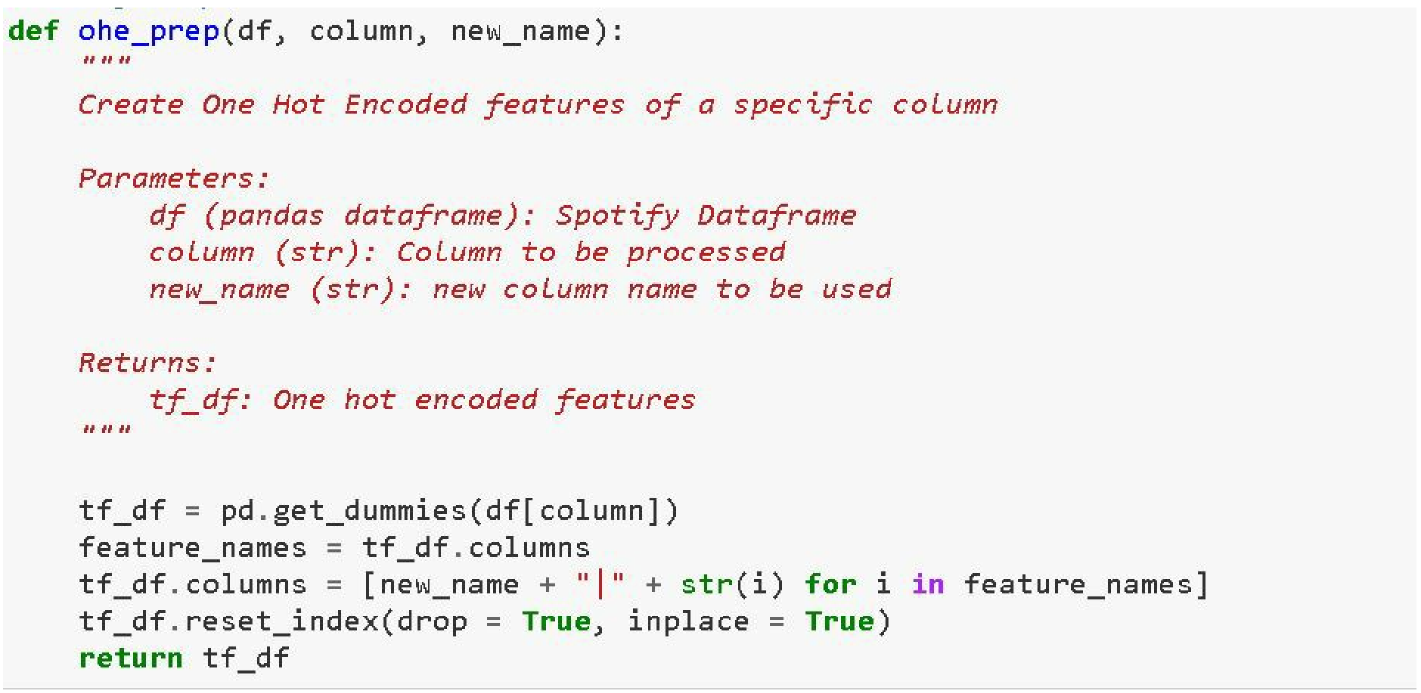
```
artists_exploded_enriched = artists_exploded.merge(data_w_genre, how = 'left', left_on = 'artists_upd',right_on = 'artists')
artists_exploded_enriched_nonnull = artists_exploded_enriched[~artists_exploded_enriched.genres_upd.isnull()]
```

```
artists_exploded_enriched_nonnull[artists_exploded_enriched_nonnull['id'] =='6KuQTIu1KoTTkLXKrwlLPV']
```

	artists_upd	id	artists	acousticness	danceability	duration_ms	energy	instrumentalness	liveness	loudness
51108	Robert Schumann	6KuQTIu1KoTTkLXKrwlLPV	Robert Schumann	0.98417	0.362023	212320.169960	0.105301	0.782029	0.160324	-22.831075
51109	Vladimir Horowitz	6KuQTIu1KoTTkLXKrwlLPV	Vladimir Horowitz	0.99007	0.343210	266541.125104	0.118844	0.879508	0.183812	-23.193418

Step-5 Creating one hot encoded features

simple function to create OHE features



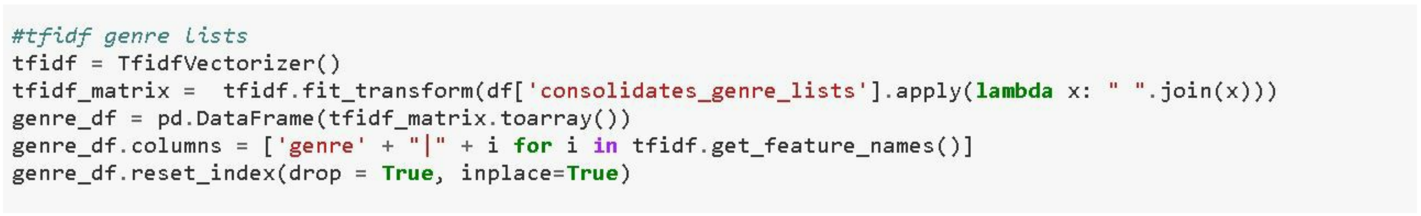
```python
def ohe_prep(df, column, new_name):
    """
    Create One Hot Encoded features of a specific column

    Parameters:
        df (pandas dataframe): Spotify Dataframe
        column (str): Column to be processed
        new_name (str): new column name to be used

    Returns:
        tf_df: One hot encoded features
    """

    tf_df = pd.get_dummies(df[column])
    feature_names = tf_df.columns
    tf_df.columns = [new_name + "|" + str(i) for i in feature_names]
    tf_df.reset_index(drop = True, inplace = True)
    return tf_df
```

Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.

Burst evious dataset's artists column so that each artist inside a song has their own row, then merge data w genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset. Expand the previous dataset's artists column so that each artist inside a song has their own row, then combine data w genre with the exploded dataset in Step 1 so that the previous dataset is enriched with the genre dataset.

Step-6 Use of TF-IDF in genre lists

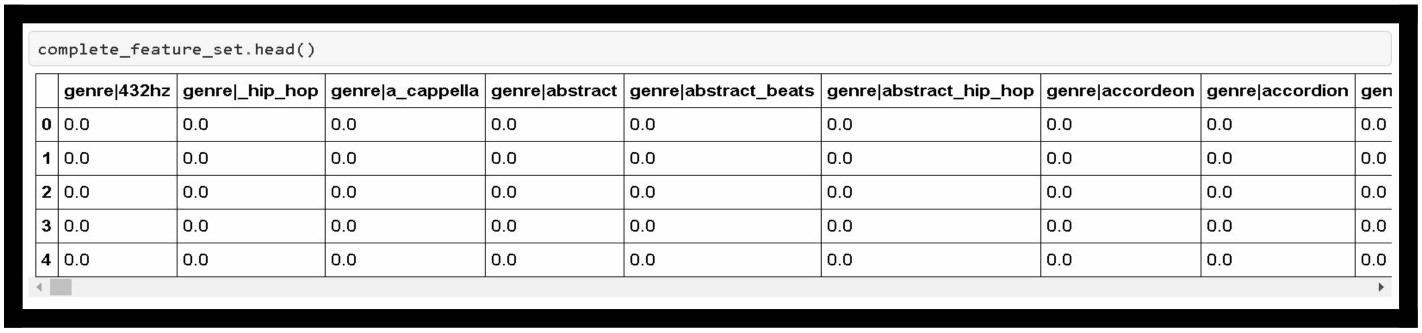
It ssigns a value to a term based on its importance in a document scaled by its importance across all documents in your corpus, TF-IDF is a popular approach for NLP tasks because it mathematically eliminates naturally occurring words in the English language and selects words that are more descriptive of your text. Text summarization, information retrieval, and sentiment classification are just a few of the NLP tasks that make use of TF-strong IDF's weighting operation.



```
#tfidf genre lists
tfidf = TfidfVectorizer()
tfidf_matrix =  tfidf.fit_transform(df['consolidates_genre_lists'].apply(lambda x: " ".join(x)))
genre_df = pd.DataFrame(tfidf_matrix.toarray())
genre_df.columns = ['genre' + "|" + i for i in tfidf.get_feature_names()]
genre_df.reset_index(drop = True, inplace=True)
```

Step-7 Complete Feature set

When developing a predictive model using machine learning or statistical modelling, feature engineering refers to the process of leveraging domain expertise to choose and convert the most important variables from raw data. Feature engineering, also known as feature extraction, is the process of extracting features from raw data using domain expertise. When opposed to sending merely raw data to a machine learning process, the objective is to leverage these extra characteristics to improve the quality of outcomes from a machine learning process.



```
complete_feature_set.head()
```

	genre|432hz	genre|_hip_hop	genre|a_cappella	genre|abstract	genre|abstract_beats	genre|abstract_hip_hop	genre|accordeon	genre|accordion	gen
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Step- 8 Connecting to Spotify api

Here, we have connected to the spotify api with the credentials client id and client secret obtained from spotify for developers .Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.Explode artists column in the previous so each artist within a song will have their own row and merge data_w_genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset.

Burst evious dataset's artists column so that each artist inside a song has their own row, then merge data w genre to the exploded dataset in Step 1 so that the previous dataset no is enriched with genre dataset. Expand the previous dataset's artists column so that each artist inside a song has their own row, then combine data w genre with the exploded dataset in Step 1 so that the previous dataset is enriched with the genre dataset.
Ater connecting to the spotify servers we can obtain live data from the account of the user which then can be used to generate recommendations.When developing a predictive model using machine learning or statistical modelling, feature engineering refers to the process of leveraging domain expertise to choose and convert the most important variables from raw data. Feature engineering, also known as feature extraction, is the process of extracting features from raw data using domain expertise. When opposed to sending merely raw data to a machine learning process, the objective is to leverage these extra characteristics to improve the quality of outcomes from a machine learning process.

```python
#client id and secret for my application
client_id = 'id'
client_secret= 'secret'
```

```python
scope = 'user-library-read'

if len(sys.argv) > 1:
    username = sys.argv[1]
else:
    print("Usage: %s username" % (sys.argv[0],))
    sys.exit()
```

```python
auth_manager = SpotifyClientCredentials(client_id=client_id, client_secret=client_secret)
sp = spotipy.Spotify(auth_manager=auth_manager)
```

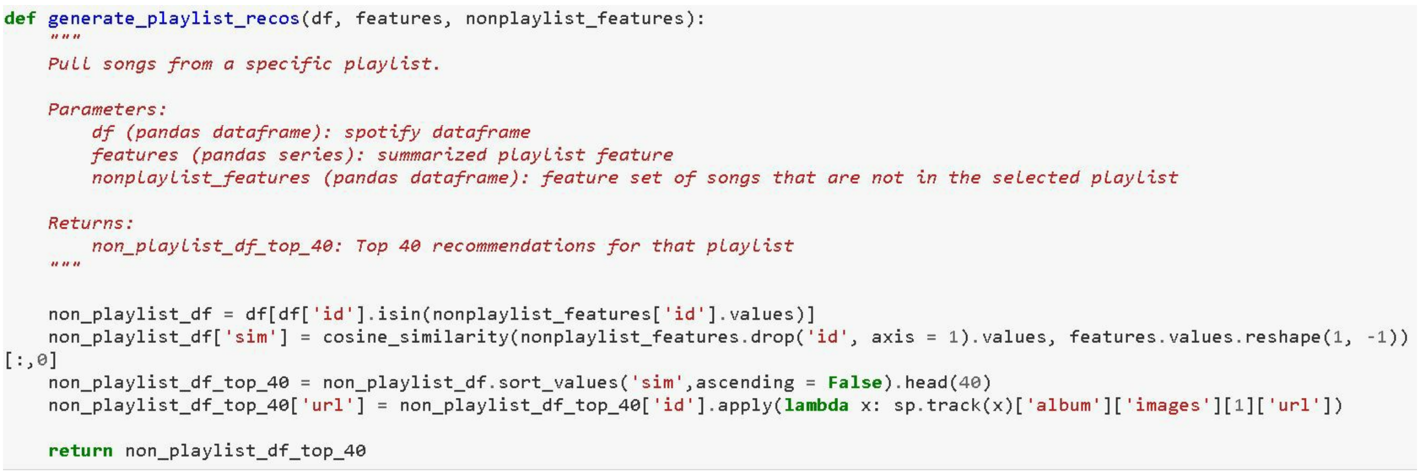
```python
token = util.prompt_for_user_token(scope, client_id= client_id, client_secret=client_secret, redirect_uri='http://localhost:888
1/')
```

```python
sp = spotipy.Spotify(auth=token)
```

Step-9 Generate Recommendation

Calculate the average vector of audio and metadata attributes for each song listened to by the user. Find the n-closest data points in the dataset to this average vector (excluding points from the user's listening history). Take these n points and suggest songs that go along with them.

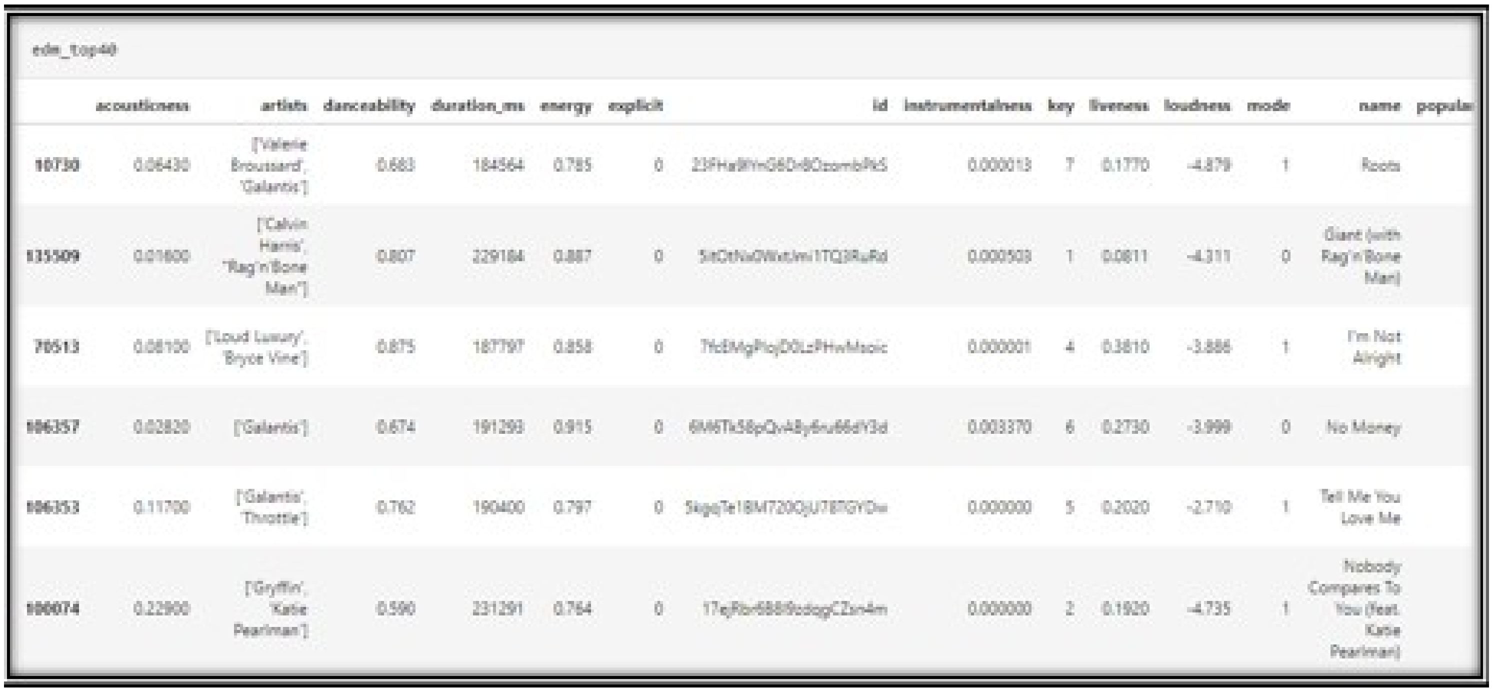
The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them.



```
def generate_playlist_recos(df, features, nonplaylist_features):
    """
    Pull songs from a specific playlist.

    Parameters:
        df (pandas dataframe): spotify dataframe
        features (pandas series): summarized playlist feature
        nonplaylist_features (pandas dataframe): feature set of songs that are not in the selected playlist

    Returns:
        non_playlist_df_top_40: Top 40 recommendations for that playlist
    """

    non_playlist_df = df[df['id'].isin(nonplaylist_features['id'].values)]
    non_playlist_df['sim'] = cosine_similarity(nonplaylist_features.drop('id', axis = 1).values, features.values.reshape(1, -1))
[:,0]
    non_playlist_df_top_40 = non_playlist_df.sort_values('sim',ascending = False).head(40)
    non_playlist_df_top_40['url'] = non_playlist_df_top_40['id'].apply(lambda x: sp.track(x)['album']['images'][1]['url'])

    return non_playlist_df_top_40
```

Step-10 Recommend songs

The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them.



edm_top40

	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popula
10730	0.06430	['Valerie Broussard', 'Galantis']	0.683	184564	0.785	0	23FHaBYnG6Dr8OzombPkS	0.000013	7	0.1770	-4.879	1	Roots	
135509	0.01600	['Calvin Harris', "Rag'n'Bone Man"]	0.807	229184	0.887	0	5itOtNx0WxtJmi1TQ3RuRd	0.000503	1	0.0811	-4.311	0	Giant (with Rag'n'Bone Man)	
70513	0.08100	['Loud Luxury', 'Bryce Vine']	0.875	187797	0.858	0	7fcEMgPlojD0LzPHwMsoic	0.000001	4	0.3810	-3.886	1	I'm Not Alright	
106357	0.02820	['Galantis']	0.674	191293	0.915	0	6M6Tk58pQvABy6ru66dY3d	0.003370	6	0.2730	-3.999	0	No Money	
106353	0.11700	['Galantis', 'Throttle']	0.762	190400	0.797	0	5kgqTe1BM720OjU7BTGYDw	0.000000	5	0.2020	-2.710	1	Tell Me You Love Me	
100074	0.22900	['Gryffin', 'Katie Pearlman']	0.590	231291	0.764	0	17ejRbr6B8l9zdqgCZsn4m	0.000000	2	0.1920	-4.735	1	Nobody Compares To You (feat. Katie Pearlman)	

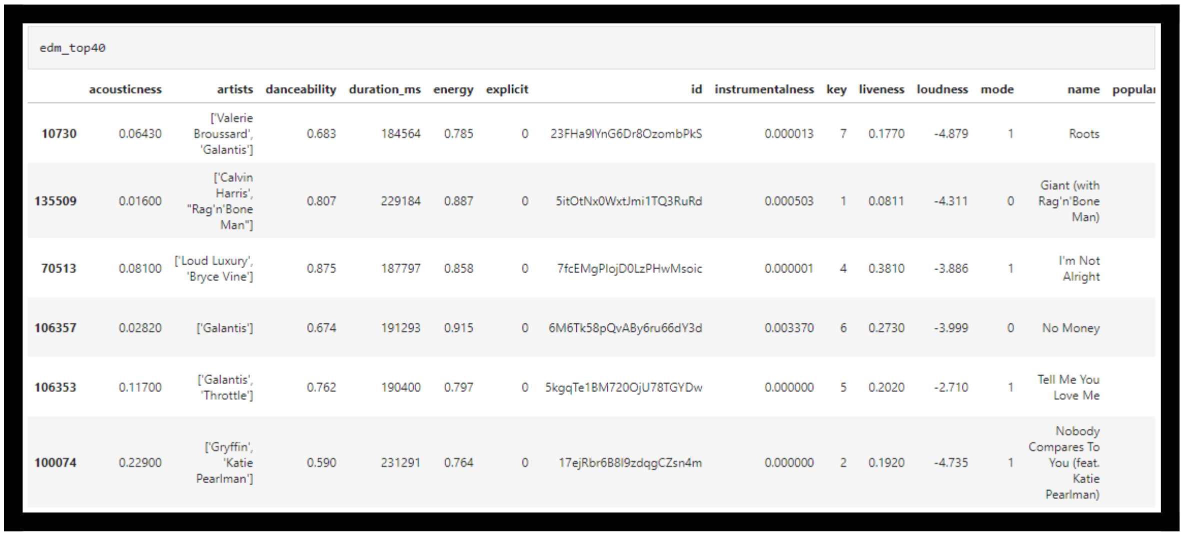
Chapter 05: CONCLUSION

5.1 Discussion on the Results Achieved

This assignment provided us with a fantastic learning opportunity. We've studied data mining and data cleansing.The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them.

A research on the limits of an interactive music recommendation service based on artificial audio similarity calculation was provided. A number of computer experiments, as well as a review of real download data, reveal that a large chunk of the audio collection is only never or never suggested. A number of computer experiments are used to investigate this issue in depth, including the investigation of various audio similarity functions and comparisons with real download data. Our music recommendation service uses Gaussian mixtures as statistical models to determine timbre similarity. This is the de facto standard method for computing audiosimilarity, and it is recognised to produce high-quality results.

A machine learning model's first goal is to eliminate all problem-causing objects from the dataset. Data cleansing and exploration were quite beneficial in getting the dataset algorithm ready. We learned how to design a machine learning model, train it, and then test it.

- The songs which scored highest have been recommended in the result given below.
- Smaller the angle, the higher the song score.
- 

```
edm_top40
```

	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness	mode	name	popula
10730	0.06430	['Valerie Broussard', 'Galantis']	0.683	184564	0.785	0	23FHa9lYnG6Dr8OzombPkS	0.000013	7	0.1770	-4.879	1	Roots	
135509	0.01600	['Calvin Harris', "Rag'n'Bone Man"]	0.807	229184	0.887	0	5itOtNx0WxtJmi1TQ3RuRd	0.000503	1	0.0811	-4.311	0	Giant (with Rag'n'Bone Man)	
70513	0.08100	['Loud Luxury', 'Bryce Vine']	0.875	187797	0.858	0	7fcEMgPlojD0LzPHwMsoic	0.000001	4	0.3810	-3.886	1	I'm Not Alright	
106357	0.02820	['Galantis']	0.674	191293	0.915	0	6M6Tk58pQvABy6ru66dY3d	0.003370	6	0.2730	-3.999	0	No Money	
106353	0.11700	['Galantis', 'Throttle']	0.762	190400	0.797	0	5kgqTe1BM720OjU78TGYDw	0.000000	5	0.2020	-2.710	1	Tell Me You Love Me	
100074	0.22900	['Gryffin', 'Katie Pearlman']	0.590	231291	0.764	0	17ejRbr6B8l9zdqgCZsn4m	0.000000	2	0.1920	-4.735	1	Nobody Compares To You (feat. Katie Pearlman)	

5.2 Application of the Major Project

In a later version, the goal is that the application will also be able to record an extract of a music being played. The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them.

The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music.
From the music or extract, the application will offer the possibility to listen to recommended songs by the algorithm developed in project.

The programme will eventually be able to record an excerpt of music being played. Using a music recommender system based on the attributes of previously heard music, the music provider may foresee and then provide suitable songs to its customers.
The application will allow users to listen to songs selected by the algorithm built in the project based on the music or extract.The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-

closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them.

For user convenience and technological simplicity, a dataset can be generated and versioned fully from a single data source. That is, data sources in a dataset cannot be mixed and matched at this time (for example, a dataset built from a GitHub repository cannot include files uploaded from your local system).
The programme will allow users to listen to recommended tracks based on the music or extract.

5.3 Limitation of the Major Project

This project due to the nature of the dataset fails to provide accurate recommendations as the dataset does not consist of all of the songs in the playlist.This is noteworthy that a dataset can be built and versioned entirely from one data source for user convenience and technical simplicity. That is, data sources in a dataset cannot presently be mixed and matched (for example, a dataset built from a GitHub repository cannot include files uploaded from your local system). You can build numerous datasets and add them both to a Notebook if you want to leverage multiple distinct data sources in it.

Songs that are identical to a large number of other songs and hence appear unnecessarily frequently in recommendation lists prevent a big section of the audio library from being recommended at all. A number of computer experiments are used to investigate this issue in depth, including the investigation of various audio similarity functions and comparisons with real download data.

For user convenience and technological simplicity, a dataset can be generated and versioned fully from a single data source. That is, data sources in a dataset cannot be mixed and matched at this time (for example, a dataset built from a GitHub repository cannot include files uploaded from your local system).

The programme will allow users to listen to recommended tracks based on the music or extract.
A dataset can be built and versioned entirely from one data source for user experience and technical simplicity.The music supplier can forecast and then give acceptable songs to its consumers using a music recommender system based on the qualities of previously heard music. for each song listened to by the user, the average vector of audio and metadata attributes Find the n-closest data points in the dataset (excluding points from the user's listening history) to this average vector. Take these n points and come up with some tunes to go with them. That is, data sources in a dataset cannot presently be mixed and matched (for example, a dataset built from a GitHub repository cannot also include files uploaded from your local workstation). You may build numerous datasets and add them both to a Notebook if you want to leverage different data sources in it.

5.4 Future Work

. The range of characteristics covered by the recommender system is extensive. In today's generation of e-services and commerce, it is growing and evolving. However, there is a requirement to create and optimise the working and output of the recommender system at the same time. For user convenience and technological simplicity, a dataset can be generated and versioned fully from a single data source. That is, data sources in a dataset cannot be mixed and matched at this time (for example, a dataset built from a GitHub repository cannot include files uploaded from your local system).
The programme will allow users to listen to recommended tracks based on the music or extract.

Several service providers provide consumers with a shopping list. However, this is insufficient since consumers have varying preferences and decisions that are influenced by a variety of circumstances and restrictions. It may also be impossible to propose specific things to individual users in many circumstances. As a result, there is potential for combining several dimensions into music recommender systems in particular.

We were unable to create a model utilising singular value decomposition and support vector machines due to a lack of time. Because popularity-based models are adept at making suggestions, we'll aim to utilise it to forecast the top-N songs for the users who are most popular at any given time.

ndeed, during years, in order to choose music, restaurants, movies, etc.. We have been asking our friends, family, and colleagues to recommend something they liked. And it is this mechanism that is attempted to be reproduced here. Netflix was a pioneer of this method (based on stars given by other users) but it is now widely used, including for Spotify's Discover Weekly. Collaborative filtering makes suggestions based on the collaborative power of the available evaluation by users.

Customers are less likely to use the majority of the items and services offered by various e commerce sites since they are pricey. As a result, you won't be able to accurately and properly rank an item or collection of things. As a result, typical recommender system strategies are inadequate. This paves the path for more research and development in the form of an efficient recommender system that also considers constraints.we'll aim to utilise it to forecast the top-N songs for the users who are most popular at any given time.

Discover Weekly is a 30-song playlist that includes music that are similar to what the user is listening to. This, like its daily mixes and tailored playlists, is made possible by AI and big data. The system also considers the user's streaming history and playlists, as well as their current music preferences, to improve this suggestion.

Spotify is planning to launch a live audio streaming function in order to improve its users' 'tailored' experience. Locker Room, a live audio app for creating conversations about music and culture, was recently bought by Spotify.

It will allow users to listen to recommended tracks based on the music or extract.
Several service providers provide consumers with a shopping list. However, this is insufficient since consumers have varying preferences and decisions that are influenced by a variety of circumstances and restrictions. It may also be impossible to propose specific things to individual users in many circumstances. As a result, there is potential for combining several dimensions into music recommender systems in particular.

REFERENCES

1. Aiolli, F. (2012). A preliminary study on a recommender system for the million songs dataset challenge. PREFERENCE LEARNING: PROBLEMS AND APPLICATIONS IN AI

2. Spotify. (2013, March) Spotify Music for everyone. [Online].

3. https://www.spotify.com/us/ [10] (2013, March) Wikipedia. [Online].

4. Peter Knees and Markus Schedl. Music retrieval and recommendation – a tutorial overview. In Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), Santiago, Chile, August 2015.

5. Peter Knees and Markus Schedl. Music Similarity and Retrieval: An Introduction to Audio- and Web-based Strategies. 2016.

6. Chris Johnson. From idea to execution: Spotify's discover weekly. November 2015.

7. Zhiwei Gu Li Guo and Tianchi Liu. Music genre classification via machine learning.

8. [11] Vikas Sindhwani Prem Melville, "Recommender Systems," in Encyclopedia of Machine Learning. Yorktown Heights, USA: IBM T. J. Watson Research Center, 2010, ch. 00338.

9. Beatriz C. F. de Azevedo Glaucia M. Bressan and Elisangela Ap. S. Lizzi.