MUSIC RECOMMENDATION SYSTEM

MINI PROJECT REPORT

18CSC305J – ARTIFICIAL INTELLIGENCE

Submitted by
ADITYA SUYASH (RA2011003011261)
RITIK RAJ (RA2011003011264)
SANYA RASTOGI (RA2011003011288)

Under the guidance of

Dr. C. Muralidharan

(Assistant Professor, Department of Computing Technologies)

In partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

COLLEGE OF ENGINEERING & TECHNOLOGY

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

S.R.M. NAGAR, KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this project report **“ Music recommendation system ”** is the bonafide work of **“ ADITYA SUYASH (RA2011003011261), RITIK RAJ (RA2011003011264) , and SANYA RASTOGI (RA2011003011288)”** of III Year/VI Sem B.tech(CSE) who carried out the mini project work under my supervision for the course **18CSC305J – ARTIFICIAL INTELLIGENCE** in SRM Institute of Science and Technology during the academic year 2022-2023 (Even sem).

SIGNATURE	**SIGNATURE**
Dr. C. Muralidharan	Dr. M. Pushpalatha
Assistant Professor	Head of the department
Department of Computing Technologies	Department of Computing Technologies

ABSTRACT

The aim of this project is to develop a recommendation system for music that suggests songs to users based on their listening history and various factors such as energy, danceability, key, tempo, etc. The system uses machine learning techniques to analyze the user's listening behavior and preferences to generate personalized recommendations.

The dataset used for training the model consists of music metadata such as genre, duration, time signature which includes various featuressuch as tempo, key, energy, genre, duration, time signature etc, which are used to represent the characteristics of each song. The listening history is used to capture the user's musical preferences and behavior.

The proposed system is a useful tool for music lovers who are looking for personalized song suggestions based on their preferences and listening history. With the increasing popularity of music streaming services, such recommendation systems can enhance the user experience and help users discover new music they may enjoy.

Additionally, Genre prediction is the task of automatically identifying the genre of a given piece of text or media. It plays an important role in various fields, such as music and movie recommendation systems, content tagging, and digital marketing. This task can be approached through different methods, such as machine learning algorithms and natural language processing techniques.

CONTENTS

Figures and Graphs

INTRODUCTION

Music recommendation systems are becoming increasingly popular as a way to help users discover new music that they might enjoy. These systems use machine learning algorithms to analyze user listening habits and recommend songs that are similar to their preferences. The project at hand is a machine learning-based music recommendation system that uses song metadata and user listening history to make personalized song recommendations to users.

The system uses collaborative filtering, a popular machine learning technique, to identify similarities between users and make recommendations based on listening history. Additionally, the system also takes into account song metadata such as key, genre, danceability, mode, and tempo to further improve the accuracy of recommendations. By combining user listening history with song metadata, the system can provide more personalized and accurate recommendations.

The primary goal of the project is to enhance the user experience by providing accurate and engaging recommendations that cater to individual user preferences. The system aims to increase user engagement and satisfaction with the music platform by offering recommendations that are tailored to their unique taste in music. The use of machine learning and song metadata is expected to improve the accuracy of recommendations and provide a more engaging listening experience for users.

PROBLEM STATEMENT

The music industry has seen significant growth in recent years, thanks to the availability of digital platforms and streaming services that offer millions of songs at the touch of a button. However, with so many songs to choose from, users often struggle to find new music that they will enjoy listening to, while also exploring different genres. Additionally, music listeners often have different preferences in terms of the genres they like, and finding new music that aligns with their tastes can be a challenge.

To address this problem, there is a need for an AIML-based music suggestion and genre prediction system that can accurately recommend songs to users based on their listening history and predict the genre of a given song based on its audio features. The system should leverage machine learning algorithms and natural language processing techniques to improve the accuracy of its predictions and recommendations. Additionally, the system should be scalable and able to handle a large number of users and songs.

The proposed music suggestion and genre prediction system should be able to capture a user's listening habits and preferences, and make personalized recommendations based on their listening history. It should also take into account the user's location, time of day, and other relevant factors to provide recommendations that are contextually relevant.

The system should also be able to predict the genre of a given song based on its audio features. This can be achieved through the use of machine learning algorithms that analyze the audio features of a song, such as its tempo, melody, and rhythm, and use this information to predict its genre.

OBJECTIVE

The objective of this project is to develop an AIML-based music suggestion and genre prediction system that can accurately recommend songs to users based on their listening history and predict the genre of a given song based on its audio features. The proposed system should leverage machine learning algorithms and natural language processing techniques to improve the accuracy of its predictions and recommendations.

The system should be able to capture a user's listening habits and preferences, and make personalized recommendations based on their listening history. It should also take into account the user's location, time of day, and other relevant factors to provide recommendations that are contextually relevant. This can help users discover new music that they will enjoy listening to, while also expanding their musical tastes.

To achieve accurate music recommendation and genre prediction, the proposed system should be trained on a large dataset of songs and user listening histories. The system should also incorporate feedback from users to continuously improve its recommendations and predictions. The ultimate goal is to provide a personalized and enjoyable music listening experience for users while helping them discover new music in their preferred genres.

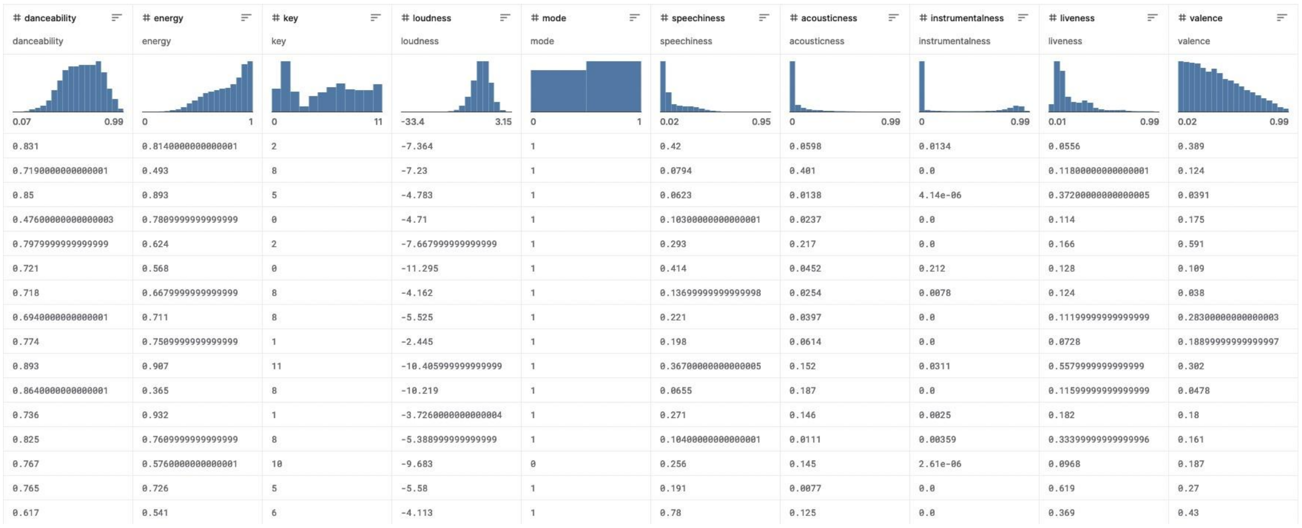
Additionally, the proposed system should be scalable and able to handle a large number of users and songs. This can help music streaming platforms improve user engagement and retention, while also enhancing the music listening experience for users. By achieving this objective, the proposed system can help music streaming platforms increase user satisfaction and improve their bottom line.

To achieve the above objective, the project will involve several tasks, including data collection and cleaning, feature engineering, machine learning model development, and system integration. The project will require expertise in machine learning, natural language processing, data science, and software engineering.

DATASET

Dataset Link: https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify

The project uses the Dataset of Songs on Spotify from Kaggle. It houses a full list of genres that contain Trap, Techno, Techhouse, Trance, Psytrance, Dark Trap, DnB (drums and bass), Hardstyle, Underground Rap, Trap Metal, Emo, Rap, RnB, Pop and Hiphop.

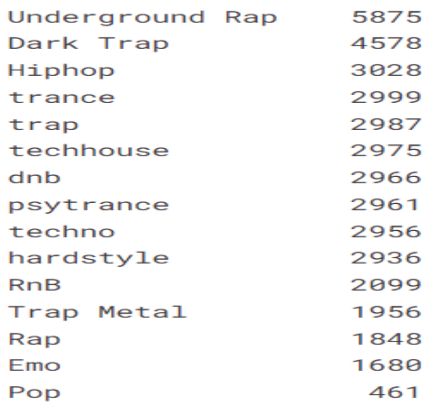


# danceability	# energy	# key	# loudness	# mode	# speechiness	# acousticness	# instrumentalness	# liveness	# valence
danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence
0.07 0.99	0 1	0 11	-33.4 3.15	0 1	0.02 0.95	0 0.99	0 0.99	0.01 0.99	0.02 0.99
0.831	0.8140000000000001	2	-7.364	1	0.42	0.0598	0.0134	0.0556	0.389
0.7190000000000001	0.493	8	-7.23	1	0.0794	0.401	0.0	0.11800000000000001	0.124
0.85	0.893	5	-4.783	1	0.0623	0.0138	4.14e-06	0.37200000000000005	0.0391
0.47600000000000003	0.7809999999999999	0	-4.71	1	0.10300000000000001	0.0237	0.0	0.114	0.175
0.7979999999999999	0.624	2	-7.667999999999999	1	0.293	0.217	0.0	0.166	0.591
0.721	0.568	0	-11.295	1	0.414	0.0452	0.212	0.128	0.109
0.718	0.6679999999999999	8	-4.162	1	0.13699999999999998	0.0254	0.0078	0.124	0.038
0.6940000000000001	0.711	8	-5.525	1	0.221	0.0397	0.0	0.11199999999999999	0.28300000000000003
0.774	0.7509999999999999	1	-2.445	1	0.198	0.0614	0.0	0.0728	0.18899999999999997
0.893	0.907	11	-10.405999999999999	1	0.36700000000000005	0.152	0.0311	0.5579999999999999	0.302
0.8640000000000001	0.365	8	-10.219	1	0.0655	0.187	0.0	0.11599999999999999	0.0478
0.736	0.932	1	-3.7260000000000004	1	0.271	0.146	0.0025	0.182	0.18
0.825	0.7609999999999999	8	-5.388999999999999	1	0.10400000000000001	0.0111	0.00359	0.33399999999999996	0.161
0.767	0.5760000000000001	10	-9.683	0	0.256	0.145	2.61e-06	0.0968	0.187
0.765	0.726	5	-5.58	1	0.191	0.0077	0.0	0.619	0.27
0.617	0.541	6	-4.113	1	0.78	0.125	0.0	0.369	0.43

Fig 1 - Dataset used in the model to train the model

The various factors in consideration include danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence.
By analysing these factors and comparing with other data values in the dataset, the model attempts to predict realistic song suggestions.

Meanwhile for Genre Prediction, the same factors are used to predict genre which plays a crucial role in music recommendation systems.The dataset contains a variety of song genres to train onto as mentioned below:



```
Underground Rap     5875
Dark Trap           4578
Hiphop              3028
trance              2999
trap                2987
techhouse           2975
dnb                 2966
psytrance           2961
techno              2956
hardstyle           2936
RnB                 2099
Trap Metal          1956
Rap                 1848
Emo                 1680
Pop                  461
```

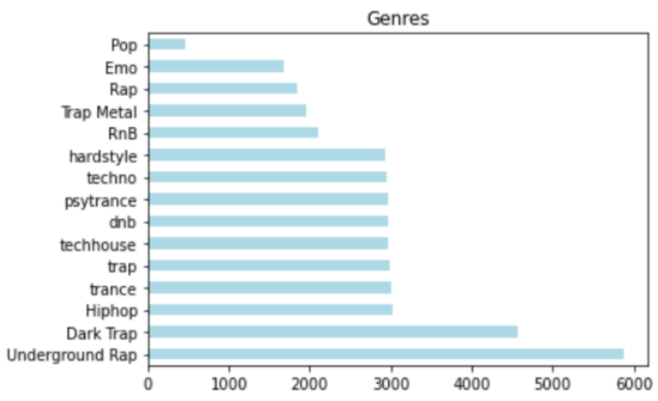


Fig 2,3 - Genre variety present in the dataset and its plot representation

METHODS

MUSIC RECOMMENDATION SYSTEM

DATA PREPROCESSING

Data pre-processing involves cleaning, transforming, and organising raw data in order to prepare it for analysis.
Basic Data Preprocessing and Preparation is done to remove unwanted columns from data set that may cause

EDA (Exploratory Data Analysis)

Exploratory data analysis (EDA) is an approach to analyzing data that involves investigating and summarizing the main characteristics of the data to gain insights and generate hypotheses. EDA is typically performed at the beginning of a data analysis process and involves techniques such as visualization, summary statistics, and data cleaning. The main goals of EDA are to understand the distribution and relationships between variables, detect outliers and anomalies, and identify patterns and trends in the data.

Box Plots

Box plots are a powerful tool in exploratory data analysis (EDA) that provide a visual representation of the distribution and spread of a dataset. A box plot displays the median, quartiles, and outliers of a dataset, making it easy to identify skewness, variability, and potential outliers. By comparing multiple box plots, analysts can quickly identify differences and similarities between groups or datasets. Box plots can also be used to identify patterns and trends in time-series data, as well as to compare distributions across different levels of a categorical variable.

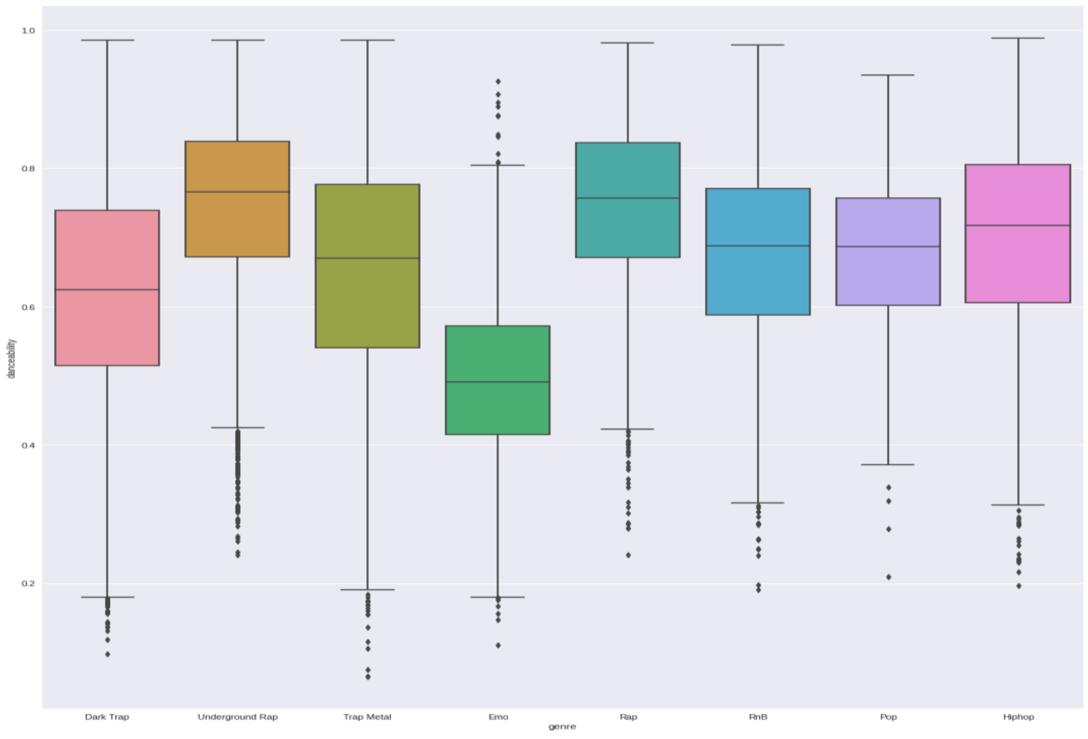


Fig 4 - Box plot of the genre vs danceability of the songs in dataset

Skewness Tests

Skewness tests are necessary because they help us to understand the shape of a distribution and identify departures from normality. Skewness is a measure of the asymmetry of a distribution, with positive skewness indicating a longer tail to the right and negative skewness indicating a longer tail to the left. Skewed distributions can have a significant impact on statistical analyses and can lead to biased results if not taken into account.

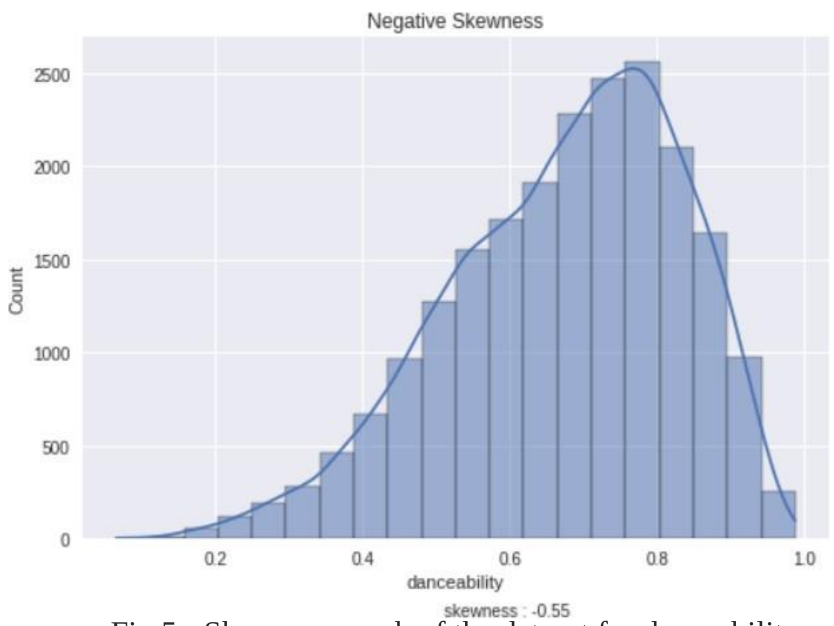
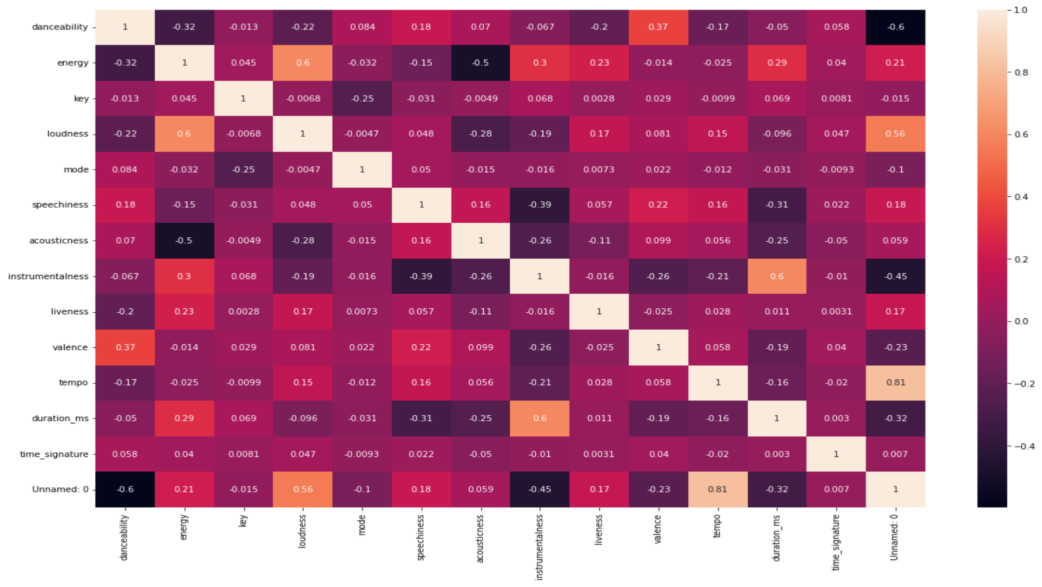


Fig 5 - Skewness graph of the dataset for danceability

Correlation matrix

Correlation matrix is necessary while building machine learning models because it helps us to identify the relationships between the features (variables) in the dataset. In machine learning, we typically have a large number of features, and it can be difficult to identify which features are most relevant to the target variable. Correlation matrix provides a quantitative measure of the strength and direction of the linear relationship between pairs of features, which can help us to identify which features are highly correlated with each other and which features are most strongly correlated with the target variable.



	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	Unnamed: 0
danceability	1	-0.32	-0.013	-0.22	0.084	0.18	0.07	-0.067	-0.2	0.37	-0.17	-0.05	0.058	-0.6
energy	-0.32	1	0.045	0.6	-0.032	-0.15	-0.5	0.3	0.23	-0.014	-0.025	0.29	0.04	0.21
key	-0.013	0.045	1	-0.0068	-0.25	-0.031	-0.0049	0.068	0.0028	0.029	-0.0099	0.069	0.0081	-0.015
loudness	-0.22	0.6	-0.0068	1	-0.0047	0.048	-0.28	-0.19	0.17	0.081	0.15	-0.096	0.047	0.56
mode	0.084	-0.032	-0.25	-0.0047	1	0.05	-0.015	-0.016	0.0073	0.022	-0.012	-0.031	-0.0093	-0.1
speechiness	0.18	-0.15	-0.031	0.048	0.05	1	0.16	-0.39	0.057	0.22	0.16	-0.31	0.022	0.18
acousticness	0.07	-0.5	-0.0049	-0.28	-0.015	0.16	1	-0.26	-0.11	0.099	0.056	-0.25	-0.05	0.059
instrumentalness	-0.067	0.3	0.068	-0.19	-0.016	-0.39	-0.26	1	-0.016	-0.26	-0.21	0.6	-0.01	-0.45
liveness	-0.2	0.23	0.0028	0.17	0.0073	0.057	-0.11	-0.016	1	-0.025	0.028	0.011	0.0031	0.17
valence	0.37	-0.014	0.029	0.081	0.022	0.22	0.099	-0.26	-0.025	1	0.058	-0.19	0.04	-0.23
tempo	-0.17	-0.025	-0.0099	0.15	-0.012	0.16	0.056	-0.21	0.028	0.058	1	-0.16	-0.02	0.81
duration_ms	-0.05	0.29	0.069	-0.096	-0.031	-0.31	-0.25	0.6	0.011	-0.19	-0.16	1	0.003	-0.32
time_signature	0.058	0.04	0.0081	0.047	-0.0093	0.022	-0.05	-0.01	0.0031	0.04	-0.02	0.003	1	0.007
Unnamed: 0	-0.6	0.21	-0.015	0.56	-0.1	0.18	0.059	-0.45	0.17	-0.23	0.81	-0.32	0.007	1

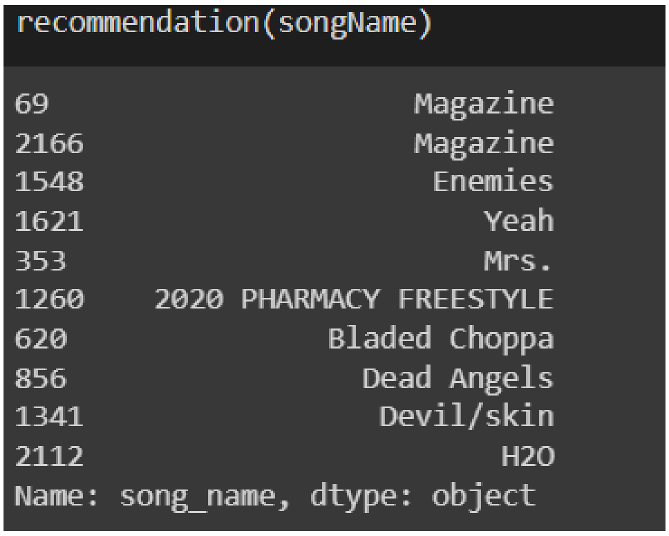
Fig 6 - Confusion matrix for various features of the songs

MACHINE LEARNING MODEL BUILDING

Song Recommendation System

In a music recommendation system, one of the key tasks is to find songs or artists that are similar to the user's preferred music. To accomplish this task, various machine learning models can be used, such as the linear kernel model, Euclidean distance model, and cosine similarity model. These models are often used in conjunction with a normalized matrix that represents the musical attributes of each song or artist.

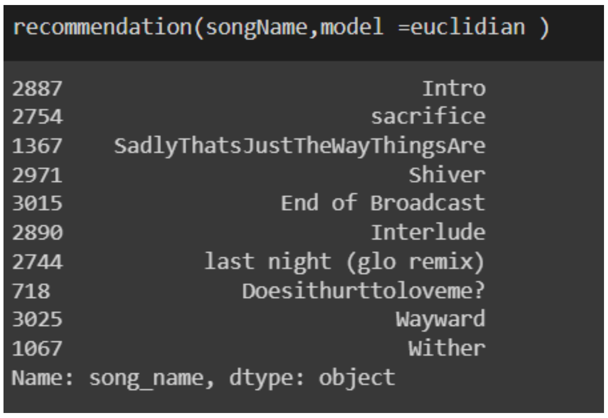
Linear kernel model:
The linear kernel model is a type of Support Vector Machine (SVM) that calculates similarity between songs based on the relationship between their attributes. In this model, the similarity between songs is calculated using a dot product, which measures the similarity between the attributes of two songs. The normalized matrix is used to calculate the dot product and to ensure that the data is scaled consistently.



```
recommendation(songName)

69                        Magazine
2166                      Magazine
1548                       Enemies
1621                          Yeah
353                           Mrs.
1260    2020 PHARMACY FREESTYLE
620                   Bladed Choppa
856                    Dead Angels
1341                    Devil/skin
2112                           H2O
Name: song_name, dtype: object
```

Fig 7 - Song recommendation according to Linear Kernel model

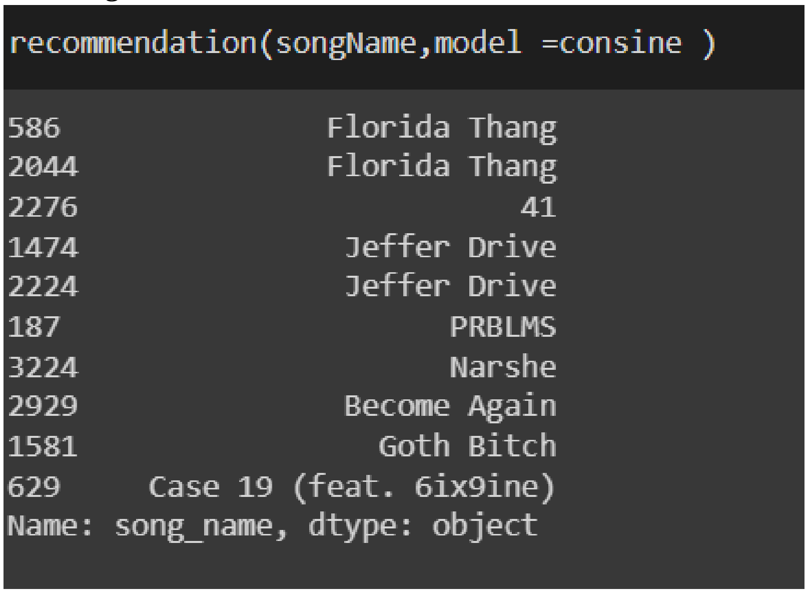
Euclidean distance model:
The Euclidean distance model calculates the distance between songs by measuring the distance between their attributes. The normalized matrix is used to calculate the Euclidean distance, which is the straight-line distance between two points in a multi-dimensional space. This model can be used to find songs that are similar to each other based on their musical attributes.



```
recommendation(songName,model =euclidian )

2887                            Intro
2754                        sacrifice
1367    SadlyThatsJustTheWayThingsAre
2971                           Shiver
3015                 End of Broadcast
2890                       Interlude
2744            last night (glo remix)
718               Doesithurttoloveme?
3025                          Wayward
1067                           Wither
Name: song_name, dtype: object
```

Fig 8 - Song recommendation according to Euclidean distance

Cosine similarity model:
The cosine similarity model is a measure of similarity between two non-zero vectors, which measures the cosine of the angle between them. In the music recommendation system, the cosine similarity model can be used to measure the similarity between songs based on their musical attributes. The normalized matrix is used to calculate the cosine similarity, which can be used to find songs that are similar to each other based on their musical

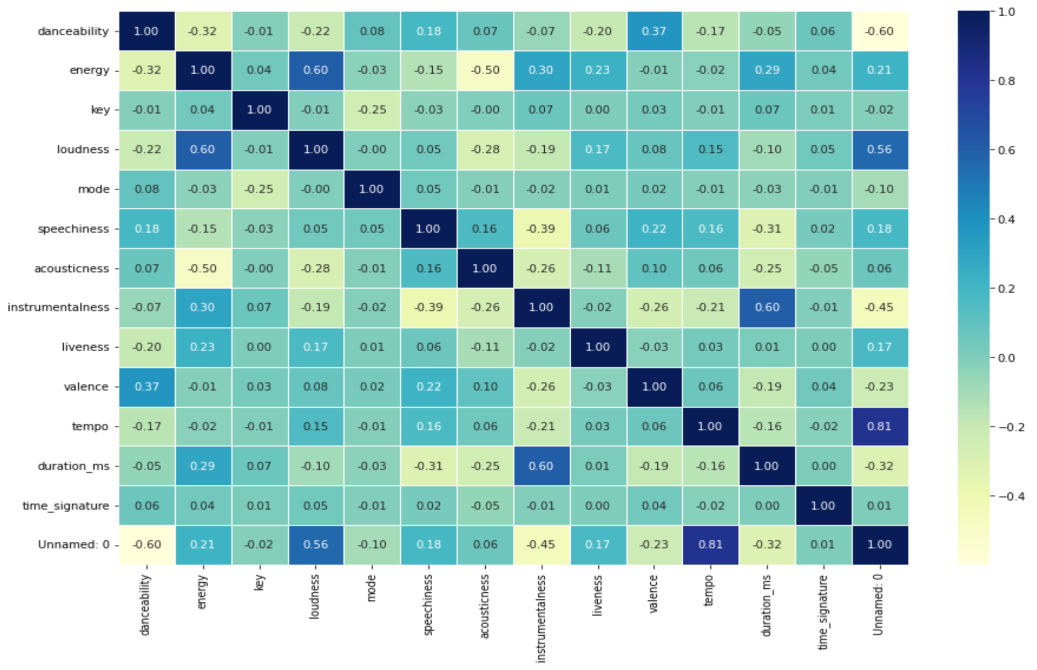


```
recommendation(songName,model =consine )

586                  Florida Thang
2044                 Florida Thang
2276                            41
1474                  Jeffer Drive
2224                  Jeffer Drive
187                         PRBLMS
3224                        Narshe
2929                  Become Again
1581                    Goth Bitch
629        Case 19 (feat. 6ix9ine)
Name: song_name, dtype: object
```

Fig 9 - Song recommendation according to Cosine similarity model

Genre Prediction

Genre prediction is a common task in machine learning that involves predicting the genre of a given piece of media, such as a song or a movie. There are several machine learning algorithms that can be used for genre prediction, including logistic regression, K-nearest neighbors (KNN), and random forest.



	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	Unnamed: 0
danceability	1.00	-0.32	-0.01	-0.22	0.08	0.18	0.07	-0.07	-0.20	0.37	-0.17	-0.05	0.06	-0.60
energy	-0.32	1.00	0.04	0.60	-0.03	-0.15	-0.50	0.30	0.23	-0.01	-0.02	0.29	0.04	0.21
key	-0.01	0.04	1.00	-0.01	-0.25	-0.03	-0.00	0.07	0.00	0.03	-0.01	0.07	0.01	-0.02
loudness	-0.22	0.60	-0.01	1.00	-0.00	0.05	-0.28	-0.19	0.17	0.08	0.15	-0.10	0.05	0.56
mode	0.08	-0.03	-0.25	-0.00	1.00	0.05	-0.01	-0.02	0.01	0.02	-0.01	-0.03	-0.01	-0.10
speechiness	0.18	-0.15	-0.03	0.05	0.05	1.00	0.16	-0.39	0.06	0.22	0.16	-0.31	0.02	0.18
acousticness	0.07	-0.50	-0.00	-0.28	-0.01	0.16	1.00	-0.26	-0.11	0.10	0.06	-0.25	-0.05	0.06
instrumentalness	-0.07	0.30	0.07	-0.19	-0.02	-0.39	-0.26	1.00	-0.02	-0.26	-0.21	0.60	-0.01	-0.45
liveness	-0.20	0.23	0.00	0.17	0.01	0.06	-0.11	-0.02	1.00	-0.03	0.03	0.01	0.00	0.17
valence	0.37	-0.01	0.03	0.08	0.02	0.22	0.10	-0.26	-0.03	1.00	0.06	-0.19	0.04	-0.23
tempo	-0.17	-0.02	-0.01	0.15	-0.01	0.16	0.06	-0.21	0.03	0.06	1.00	-0.16	-0.02	0.81
duration_ms	-0.05	0.29	0.07	-0.10	-0.03	-0.31	-0.25	0.60	0.01	-0.19	-0.16	1.00	0.00	-0.32
time_signature	0.06	0.04	0.01	0.05	-0.01	0.02	-0.05	-0.01	0.00	0.04	-0.02	0.00	1.00	0.01
Unnamed: 0	-0.60	0.21	-0.02	0.56	-0.10	0.18	0.06	-0.45	0.17	-0.23	0.81	-0.32	0.01	1.00

Fig 10 - Genre prediction matrix used to predict the genre of the song

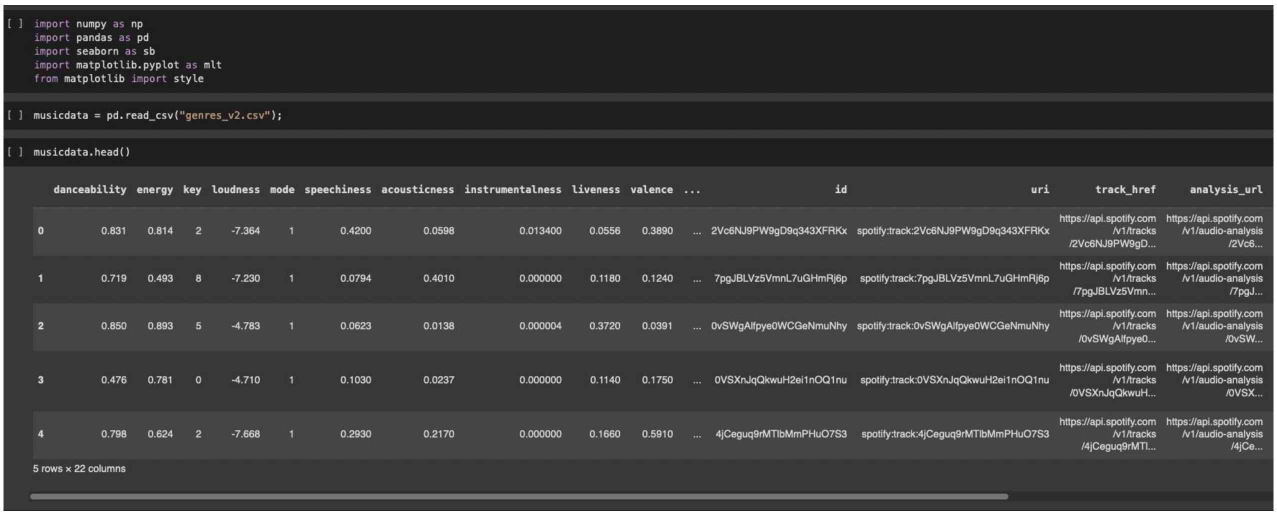
Logistic regression is a supervised learning algorithm that is used for binary classification tasks, where the target variable has two possible outcomes. In genre prediction, logistic regression can be used to predict whether a given piece of media belongs to a particular genre or not. Logistic regression works by estimating the probability of the target variable based on the input features, and then making a binary prediction based on a threshold value.

K-nearest neighbors (KNN) is a non-parametric algorithm that is often used for classification and regression tasks. KNN works by finding the k nearest neighbors to a given data point based on a distance metric, and then making a prediction based on the class of the majority of the nearest neighbors. In genre prediction, KNN can be used to identify similar pieces of media based on their genre, and then make a prediction based on the most common genre among the k-nearest neighbors.

Random forest is an ensemble learning algorithm that is often used for classification and regression tasks. Random forest works by creating multiple decision trees on randomly sampled subsets of the data, and then aggregating the predictions of the individual trees to make a final prediction. In genre prediction, random forest can be used to identify the most important features for predicting genre, and then make a prediction based on the collective predictions of the decision trees.

When using these algorithms for genre prediction, it is important to pre-process the data appropriately and choose an appropriate evaluation metric to assess their performance. Additionally, hyperparameter tuning can be used to optimize the performance of these algorithms on a given dataset.

Experiments and results



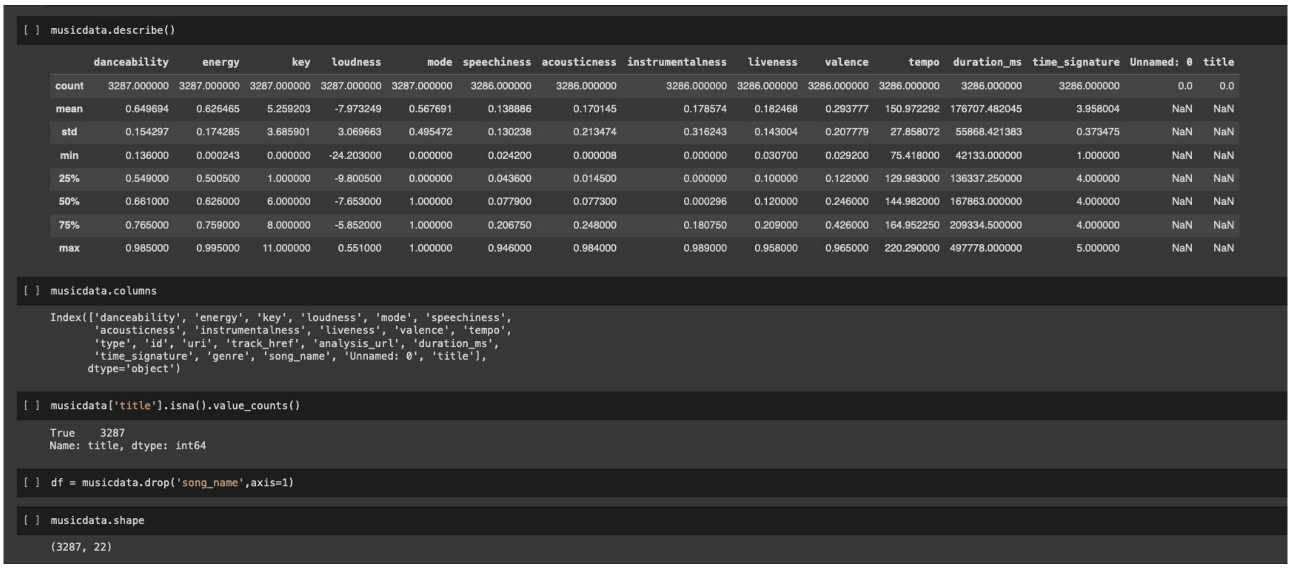
```
[ ] import numpy as np
    import pandas as pd
    import seaborn as sb
    import matplotlib.pyplot as mlt
    from matplotlib import style
```

```
[ ] musicdata = pd.read_csv("genres_v2.csv");
```

```
[ ] musicdata.head()
```

	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	...	id	uri	track_href	analysis_url
0	0.831	0.814	2	-7.364	1	0.4200	0.0598	0.013400	0.0556	0.3890	...	2Vc6NJ9PW9gD9q343XFRKx	spotify:track:2Vc6NJ9PW9gD9q343XFRKx	https://api.spotify.com /v1/tracks /2Vc6NJ9PW9gD...	https://api.spotify.com /v1/audio-analysis /2Vc6...
1	0.719	0.493	8	-7.230	1	0.0794	0.4010	0.000000	0.1180	0.1240	...	7pgJBLVz5VmnL7uGHmRj6p	spotify:track:7pgJBLVz5VmnL7uGHmRj6p	https://api.spotify.com /v1/tracks /7pgJBLVz5Vmn...	https://api.spotify.com /v1/audio-analysis /7pgJ...
2	0.850	0.893	5	-4.783	1	0.0623	0.0138	0.000004	0.3720	0.0391	...	0vSWgAlfpye0WCGeNmuNhy	spotify:track:0vSWgAlfpye0WCGeNmuNhy	https://api.spotify.com /v1/tracks /0vSWgAlfpye0...	https://api.spotify.com /v1/audio-analysis /0vSW...
3	0.476	0.781	0	-4.710	1	0.1030	0.0237	0.000000	0.1140	0.1750	...	0VSXnJqQkwuH2ei1nOQ1nu	spotify:track:0VSXnJqQkwuH2ei1nOQ1nu	https://api.spotify.com /v1/tracks /0VSXnJqQkwuH...	https://api.spotify.com /v1/audio-analysis /0VSX...
4	0.798	0.624	2	-7.668	1	0.2930	0.2170	0.000000	0.1660	0.5910	...	4jCeguq9rMTlbMmPHuO7S3	spotify:track:4jCeguq9rMTlbMmPHuO7S3	https://api.spotify.com /v1/tracks /4jCeguq9rMTl...	https://api.spotify.com /v1/audio-analysis /4jCe...

5 rows × 22 columns



```
[ ] musicdata.describe()
```

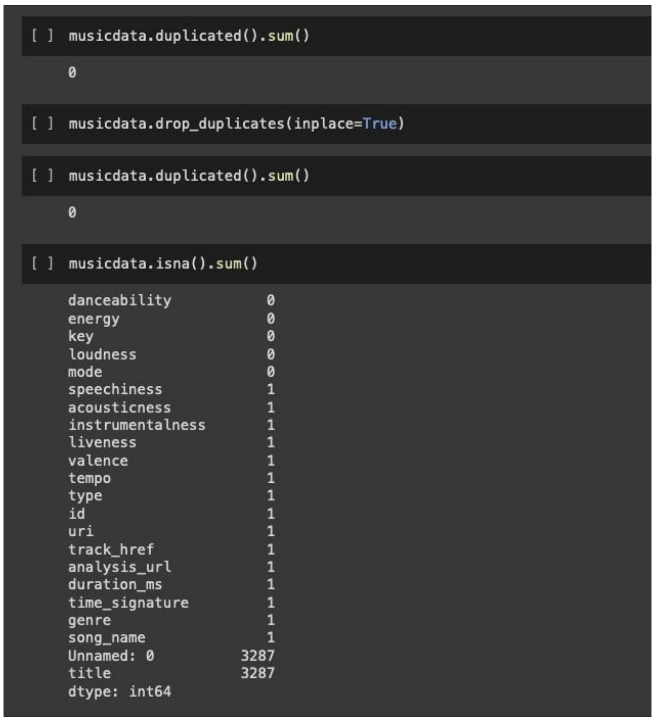
	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	Unnamed: 0	title
count	3287.000000	3287.000000	3287.000000	3287.000000	3287.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	0.0	0.0
mean	0.649694	0.626465	5.259203	-7.973249	0.567691	0.138886	0.170145	0.178574	0.182468	0.293777	150.972292	176707.482045	3.958004	NaN	NaN
std	0.154297	0.174285	3.685901	3.069663	0.495472	0.130238	0.213474	0.316243	0.143004	0.207779	27.858072	55868.421383	0.373475	NaN	NaN
min	0.136000	0.000243	0.000000	-24.203000	0.000000	0.024200	0.000008	0.000000	0.030700	0.029200	75.418000	42133.000000	1.000000	NaN	NaN
25%	0.549000	0.500500	1.000000	-9.800500	0.000000	0.043600	0.014500	0.000000	0.100000	0.122000	129.983000	136337.250000	4.000000	NaN	NaN
50%	0.661000	0.626000	6.000000	-7.653000	1.000000	0.077900	0.077300	0.000296	0.120000	0.246000	144.982000	167863.000000	4.000000	NaN	NaN
75%	0.765000	0.759000	8.000000	-5.852000	1.000000	0.206750	0.248000	0.180750	0.209000	0.426000	164.952250	209334.500000	4.000000	NaN	NaN
max	0.985000	0.995000	11.000000	0.551000	1.000000	0.946000	0.984000	0.989000	0.958000	0.965000	220.290000	497778.000000	5.000000	NaN	NaN

```
[ ] musicdata.columns

Index(['danceability', 'energy', 'key', 'loudness', 'mode', 'speechiness',
       'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo',
       'type', 'id', 'uri', 'track_href', 'analysis_url', 'duration_ms',
       'time_signature', 'genre', 'song_name', 'Unnamed: 0', 'title'],
      dtype='object')
```

```
[ ] musicdata['title'].isna().value_counts()

True    3287
Name: title, dtype: int64
```

```
[ ] df = musicdata.drop('song_name',axis=1)
```

```
[ ] musicdata.shape

(3287, 22)
```

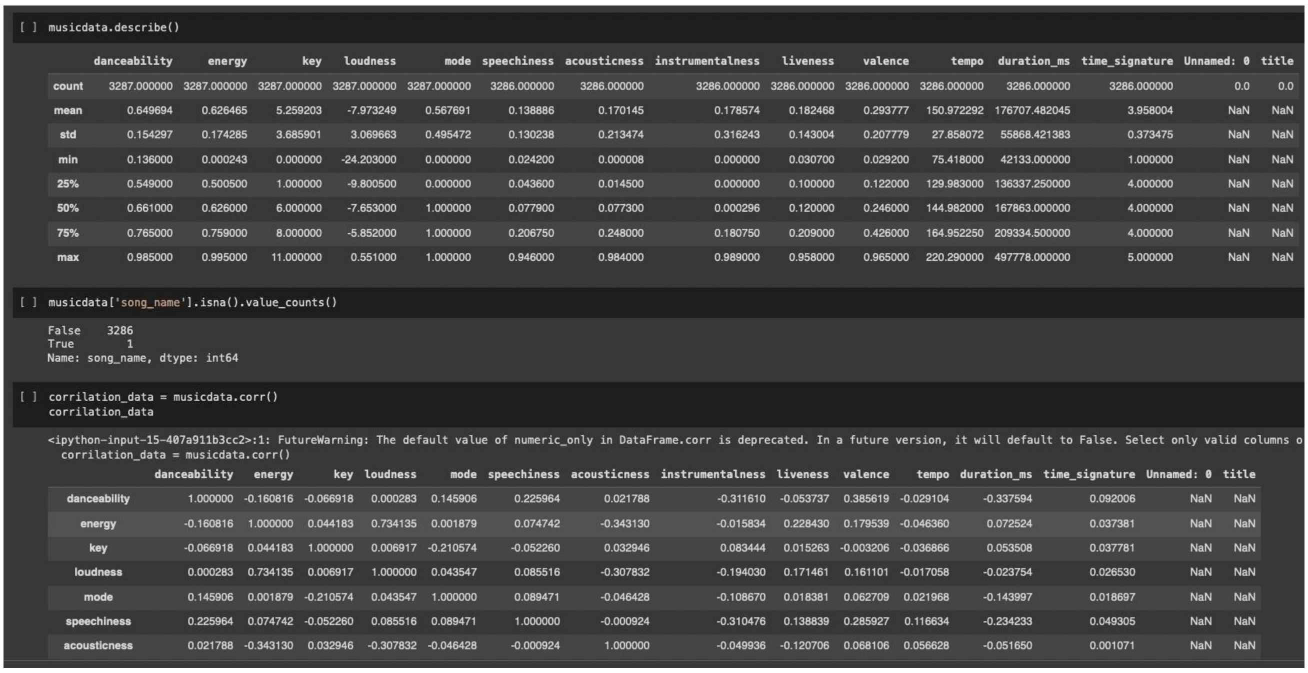


```
[ ] musicdata.duplicated().sum()

0
```

```
[ ] musicdata.drop_duplicates(inplace=True)
```

```
[ ] musicdata.duplicated().sum()

0
```

```
[ ] musicdata.isna().sum()

danceability           0
energy                 0
key                    0
loudness               0
mode                   0
speechiness            1
acousticness           1
instrumentalness       1
liveness               1
valence                1
tempo                  1
type                   1
id                     1
uri                    1
track_href             1
analysis_url           1
duration_ms            1
time_signature         1
genre                  1
song_name              1
Unnamed: 0          3287
title               3287
dtype: int64
```



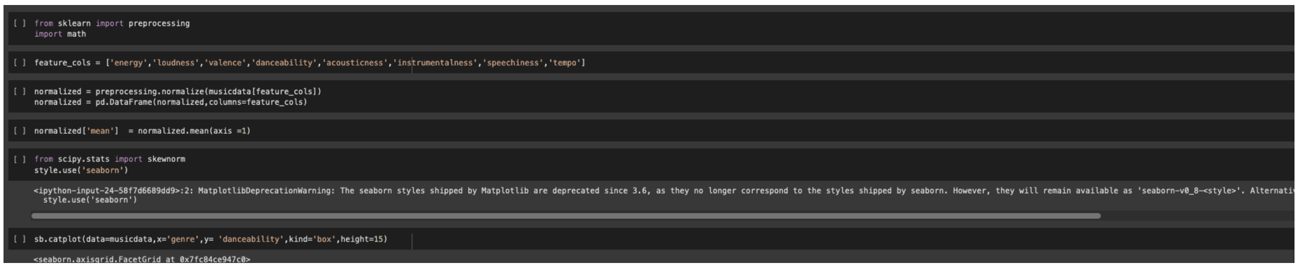
```
[ ] musicdata.describe()
```

	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	Unnamed: 0	title
count	3287.000000	3287.000000	3287.000000	3287.000000	3287.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	3286.000000	0.0	0.0
mean	0.649694	0.626465	5.259203	-7.973249	0.567691	0.138886	0.170145	0.178574	0.182468	0.293777	150.972292	176707.482045	3.958004	NaN	NaN
std	0.154297	0.174285	3.685901	3.069663	0.495472	0.130238	0.213474	0.316243	0.143004	0.207779	27.858072	55868.421383	0.373475	NaN	NaN
min	0.136000	0.000243	0.000000	-24.203000	0.000000	0.024200	0.000008	0.000000	0.030700	0.029200	75.418000	42133.000000	1.000000	NaN	NaN
25%	0.549000	0.500500	1.000000	-9.800500	0.000000	0.043600	0.014500	0.000000	0.100000	0.122000	129.983000	136337.250000	4.000000	NaN	NaN
50%	0.661000	0.626000	6.000000	-7.653000	1.000000	0.077900	0.077300	0.000296	0.120000	0.246000	144.982000	167863.000000	4.000000	NaN	NaN
75%	0.765000	0.759000	8.000000	-5.852000	1.000000	0.206750	0.248000	0.180750	0.209000	0.426000	164.952250	209334.500000	4.000000	NaN	NaN
max	0.985000	0.995000	11.000000	0.551000	1.000000	0.946000	0.984000	0.989000	0.958000	0.965000	220.290000	497778.000000	5.000000	NaN	NaN

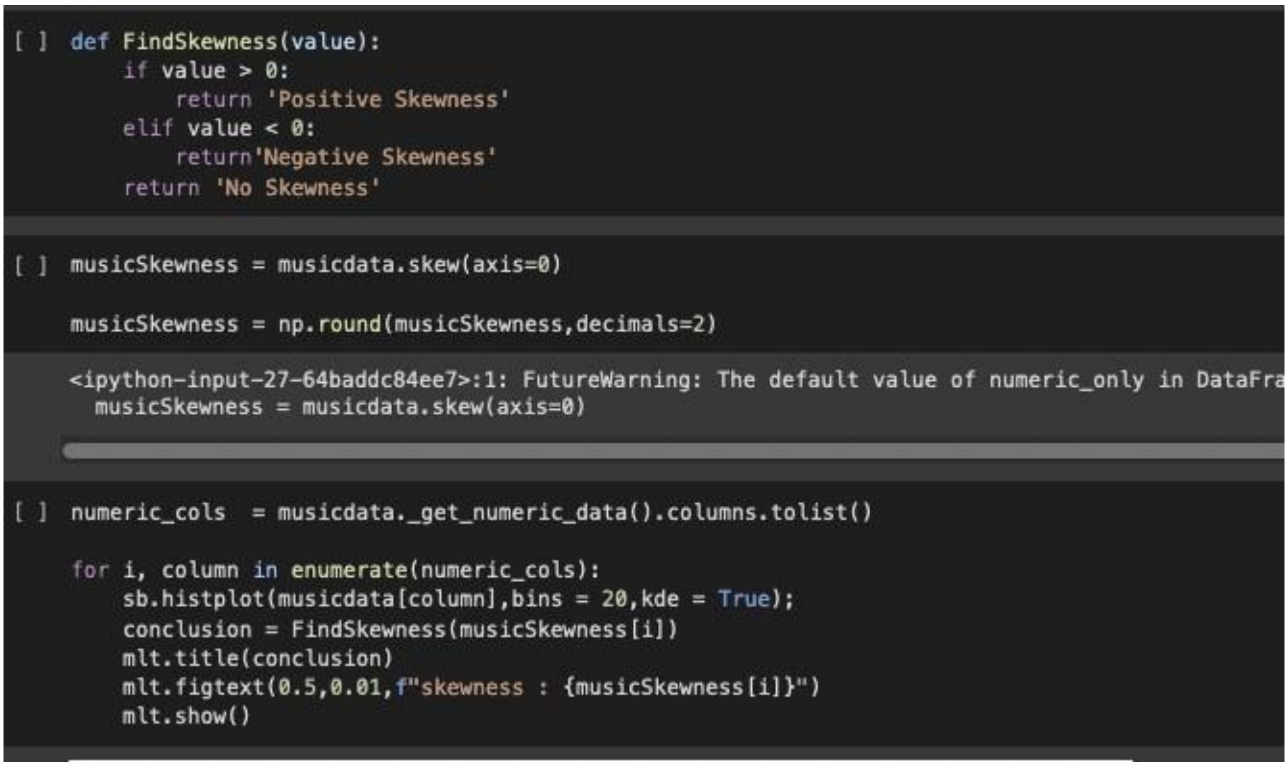
```
[ ] musicdata['song_name'].isna().value_counts()

    False    3286
    True        1
    Name: song_name, dtype: int64
```

```
[ ] corrilation_data = musicdata.corr()
    corrilation_data

    <ipython-input-15-407a911b3cc2>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns o
      corrilation_data = musicdata.corr()
```

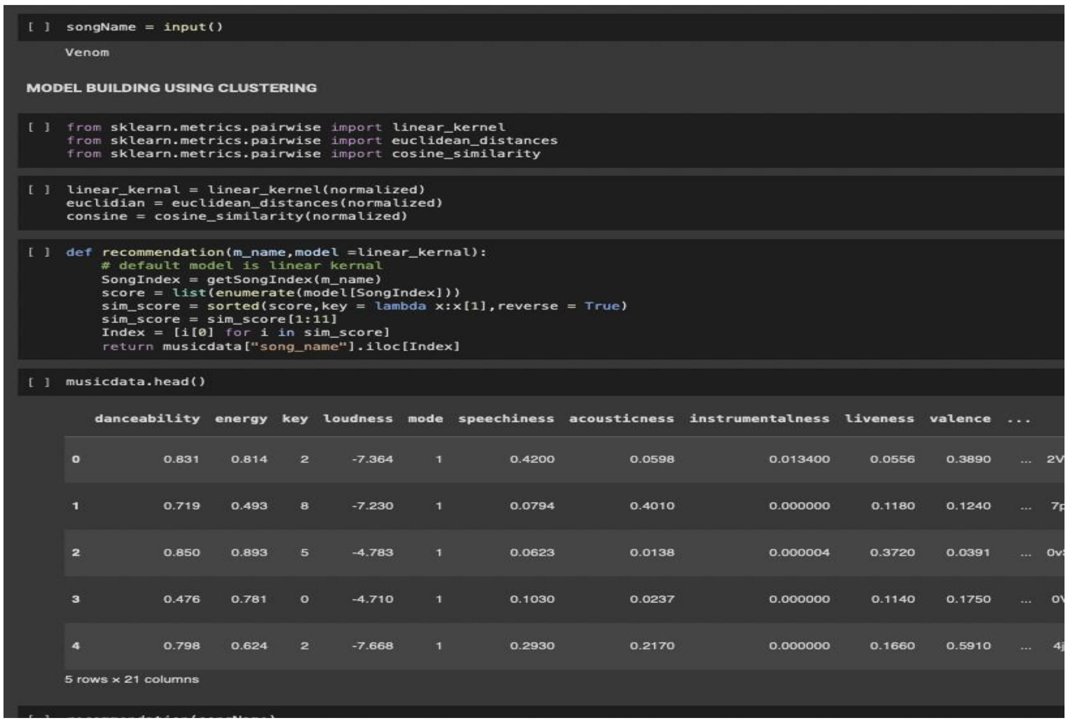
	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	Unnamed: 0	title
danceability	1.000000	-0.160816	-0.066918	0.000283	0.145906	0.225964	0.021788	-0.311610	-0.053737	0.385619	-0.029104	-0.337594	0.092006	NaN	NaN
energy	-0.160816	1.000000	0.044183	0.734135	0.001879	0.074742	-0.343130	-0.015834	0.228430	0.179539	-0.046360	0.072524	0.037381	NaN	NaN
key	-0.066918	0.044183	1.000000	0.006917	-0.210574	-0.052260	0.032946	0.083444	0.015263	-0.003206	-0.036866	0.053508	0.037781	NaN	NaN
loudness	0.000283	0.734135	0.006917	1.000000	0.043547	0.085516	-0.307832	-0.194030	0.171461	0.161101	-0.017058	-0.023754	0.026530	NaN	NaN
mode	0.145906	0.001879	-0.210574	0.043547	1.000000	0.089471	-0.046428	-0.108670	0.018381	0.062709	0.021968	-0.143997	0.018697	NaN	NaN
speechiness	0.225964	0.074742	-0.052260	0.085516	0.089471	1.000000	-0.000924	-0.310476	0.138839	0.285927	0.116634	-0.234233	0.049305	NaN	NaN
acousticness	0.021788	-0.343130	0.032946	-0.307832	-0.046428	-0.000924	1.000000	-0.049936	-0.120706	0.068106	0.056628	-0.051650	0.001071	NaN	NaN



```
[ ] from sklearn import preprocessing
    import math

[ ] feature_cols = ['energy','loudness','valence','danceability','acousticness','instrumentalness','speechiness','tempo']

[ ] normalized = preprocessing.normalize(musicdata[feature_cols])
    normalized = pd.DataFrame(normalized,columns=feature_cols)

[ ] normalized['mean']  = normalized.mean(axis =1)

[ ] from scipy.stats import skewnorm
    style.use('seaborn')

    <ipython-input-24-58f7d6689dd9>:2: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'. Alternati
      style.use('seaborn')

[ ] sb.catplot(data=musicdata,x='genre',y= 'danceability',kind='box',height=15)

    <seaborn.axisgrid.FacetGrid at 0x7fc84ce947c0>
```



```
[ ] def FindSkewness(value):
        if value > 0:
            return 'Positive Skewness'
        elif value < 0:
            return'Negative Skewness'
        return 'No Skewness'

[ ] musicSkewness = musicdata.skew(axis=0)

    musicSkewness = np.round(musicSkewness,decimals=2)

    <ipython-input-27-64baddc84ee7>:1: FutureWarning: The default value of numeric_only in DataFra
      musicSkewness = musicdata.skew(axis=0)

[ ] numeric_cols  = musicdata._get_numeric_data().columns.tolist()

    for i, column in enumerate(numeric_cols):
        sb.histplot(musicdata[column],bins = 20,kde = True);
        conclusion = FindSkewness(musicSkewness[i])
        mlt.title(conclusion)
        mlt.figtext(0.5,0.01,f"skewness : {musicSkewness[i]}")
        mlt.show()
```



```
[ ]  songName = input()

     Venom
```

MODEL BUILDING USING CLUSTERING

```
[ ]  from sklearn.metrics.pairwise import linear_kernel
     from sklearn.metrics.pairwise import euclidean_distances
     from sklearn.metrics.pairwise import cosine_similarity
```

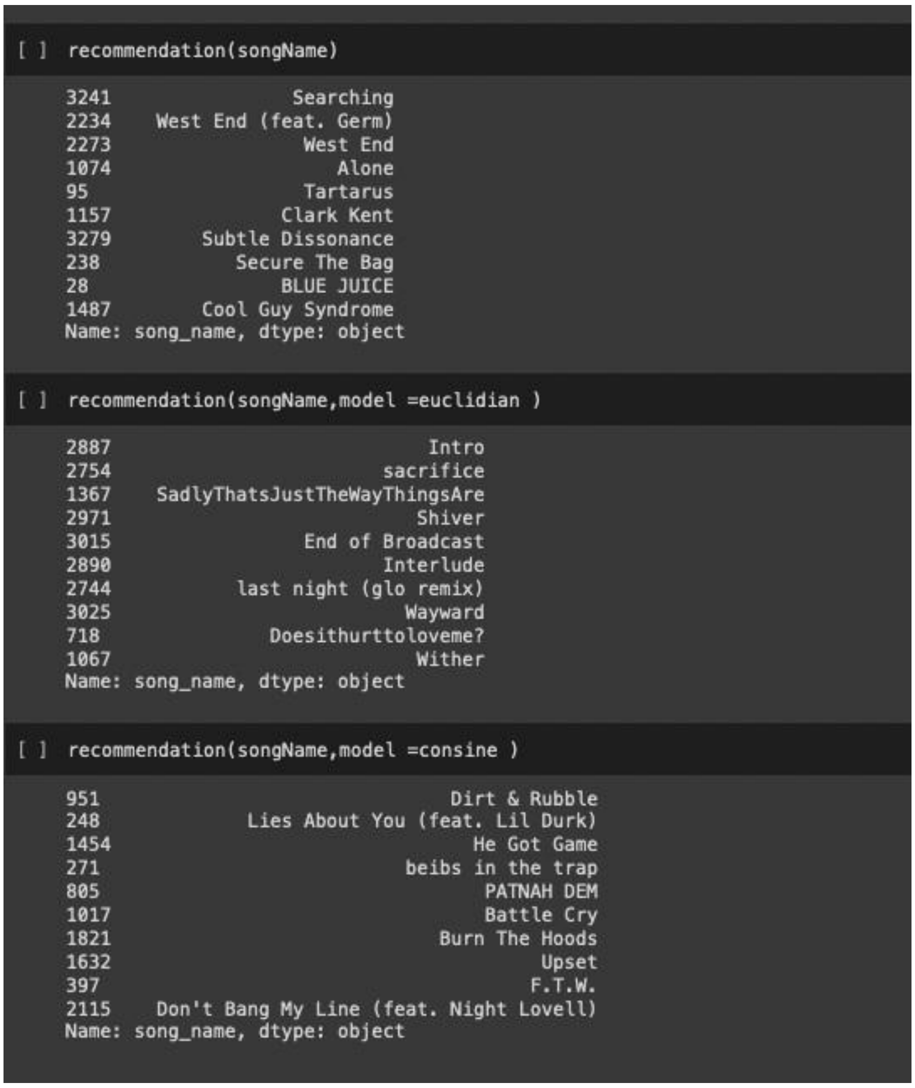
```
[ ]  linear_kernal = linear_kernel(normalized)
     euclidian = euclidean_distances(normalized)
     consine = cosine_similarity(normalized)
```

```
[ ]  def recommendation(m_name,model =linear_kernal):
         # default model is linear kernal
         SongIndex = getSongIndex(m_name)
         score = list(enumerate(model[SongIndex]))
         sim_score = sorted(score,key = lambda x:x[1],reverse = True)
         sim_score = sim_score[1:11]
         Index = [i[0] for i in sim_score]
         return musicdata["song_name"].iloc[Index]
```

```
[ ]  musicdata.head()
```

	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	...	
0	0.831	0.814	2	-7.364	1	0.4200	0.0598	0.013400	0.0556	0.3890	...	2V
1	0.719	0.493	8	-7.230	1	0.0794	0.4010	0.000000	0.1180	0.1240	...	7p
2	0.850	0.893	5	-4.783	1	0.0623	0.0138	0.000004	0.3720	0.0391	...	0v
3	0.476	0.781	0	-4.710	1	0.1030	0.0237	0.000000	0.1140	0.1750	...	0V
4	0.798	0.624	2	-7.668	1	0.2930	0.2170	0.000000	0.1660	0.5910	...	4j

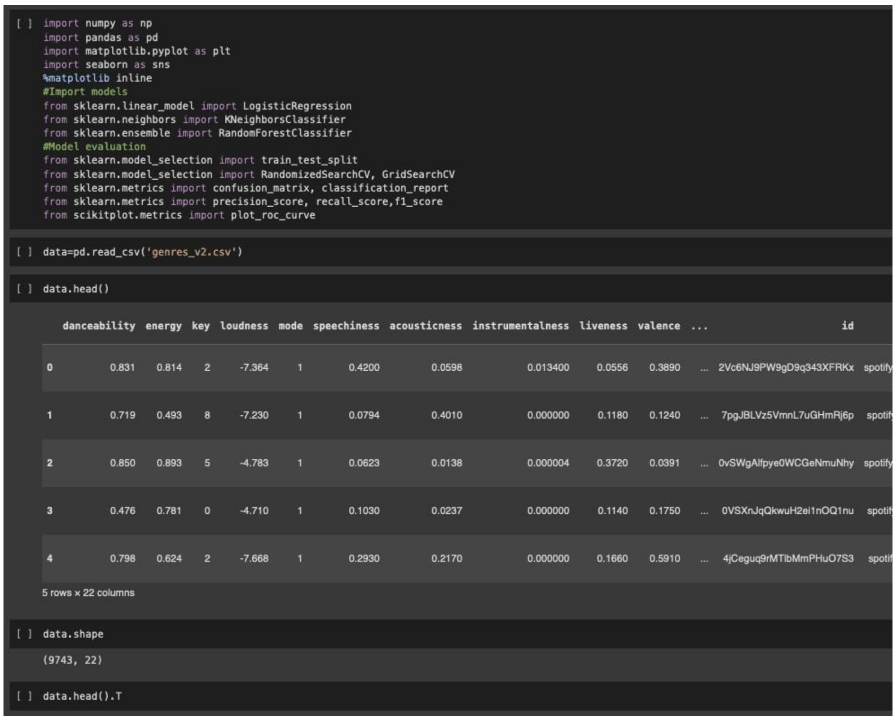
5 rows × 21 columns



```
[ ]  recommendation(songName)

     3241                Searching
     2234     West End (feat. Germ)
     2273                 West End
     1074                    Alone
     95                   Tartarus
     1157               Clark Kent
     3279         Subtle Dissonance
     238             Secure The Bag
     28                 BLUE JUICE
     1487         Cool Guy Syndrome
     Name: song_name, dtype: object
```

```
[ ]  recommendation(songName,model =euclidian )

     2887                             Intro
     2754                         sacrifice
     1367    SadlyThatsJustTheWayThingsAre
     2971                            Shiver
     3015                  End of Broadcast
     2890                         Interlude
     2744             last night (glo remix)
     3025                          Wayward
     718                 Doesithurttoloveme?
     1067                            Wither
     Name: song_name, dtype: object
```

```
[ ]  recommendation(songName,model =consine )

     951                                 Dirt & Rubble
     248               Lies About You (feat. Lil Durk)
     1454                                  He Got Game
     271                              beibs in the trap
     805                                   PATNAH DEM
     1017                                   Battle Cry
     1821                               Burn The Hoods
     1632                                        Upset
     397                                        F.T.W.
     2115     Don't Bang My Line (feat. Night Lovell)
     Name: song_name, dtype: object
```



```
[ ] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
    %matplotlib inline
    #Import models
    from sklearn.linear_model import LogisticRegression
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.ensemble import RandomForestClassifier
    #Model evaluation
    from sklearn.model_selection import train_test_split
    from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
    from sklearn.metrics import confusion_matrix, classification_report
    from sklearn.metrics import precision_score, recall_score,f1_score
    from scikitplot.metrics import plot_roc_curve
```

```
[ ] data=pd.read_csv('genres_v2.csv')
```

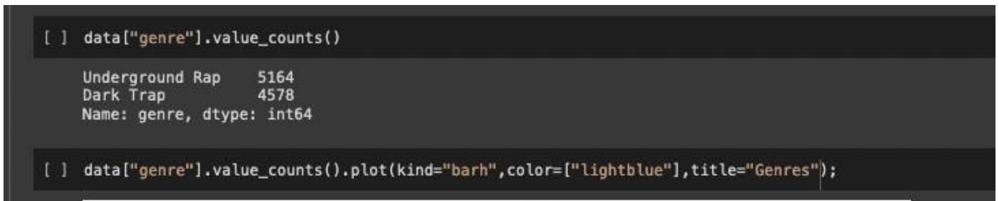
```
[ ] data.head()
```

	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	...	id	
0	0.831	0.814	2	-7.364	1	0.4200	0.0598	0.013400	0.0556	0.3890	...	2Vc6NJ9PW9gD9q343XFRKx	spotify
1	0.719	0.493	8	-7.230	1	0.0794	0.4010	0.000000	0.1180	0.1240	...	7pgJBLVz5VmnL7uGHmRj6p	spotif
2	0.850	0.893	5	-4.783	1	0.0623	0.0138	0.000004	0.3720	0.0391	...	0vSWgAlfpye0WCGeNmuNhy	spotify
3	0.476	0.781	0	-4.710	1	0.1030	0.0237	0.000000	0.1140	0.1750	...	0VSXnJqQkwuH2ei1nOQ1nu	spotif
4	0.798	0.624	2	-7.668	1	0.2930	0.2170	0.000000	0.1660	0.5910	...	4jCeguq9rMTlbMmPHuO7S3	spotif

5 rows × 22 columns

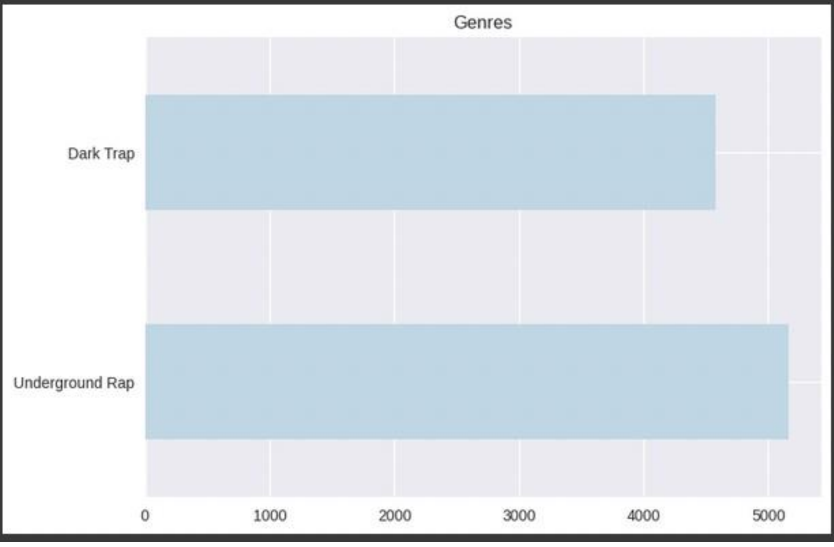
```
[ ] data.shape

    (9743, 22)
```

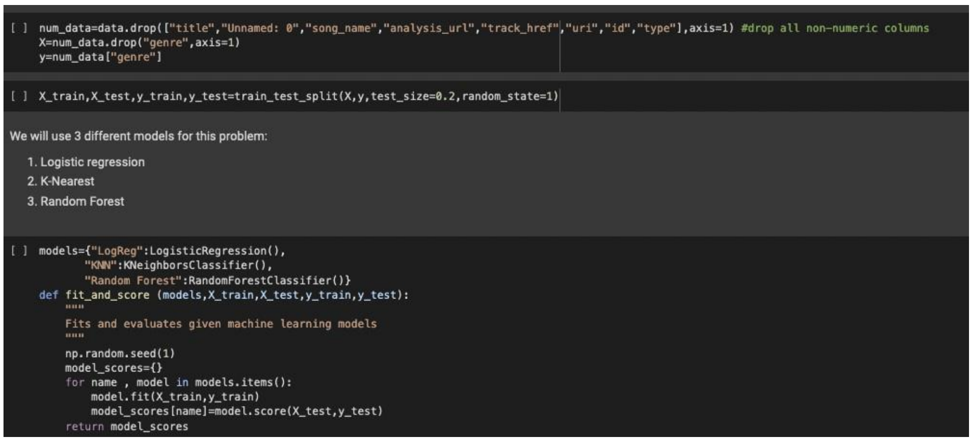
```
[ ] data.head().T
```



```
[ ] data["genre"].value_counts()

    Underground Rap    5164
    Dark Trap          4578
    Name: genre, dtype: int64
```

```
[ ] data["genre"].value_counts().plot(kind="barh",color=["lightblue"],title="Genres");
```





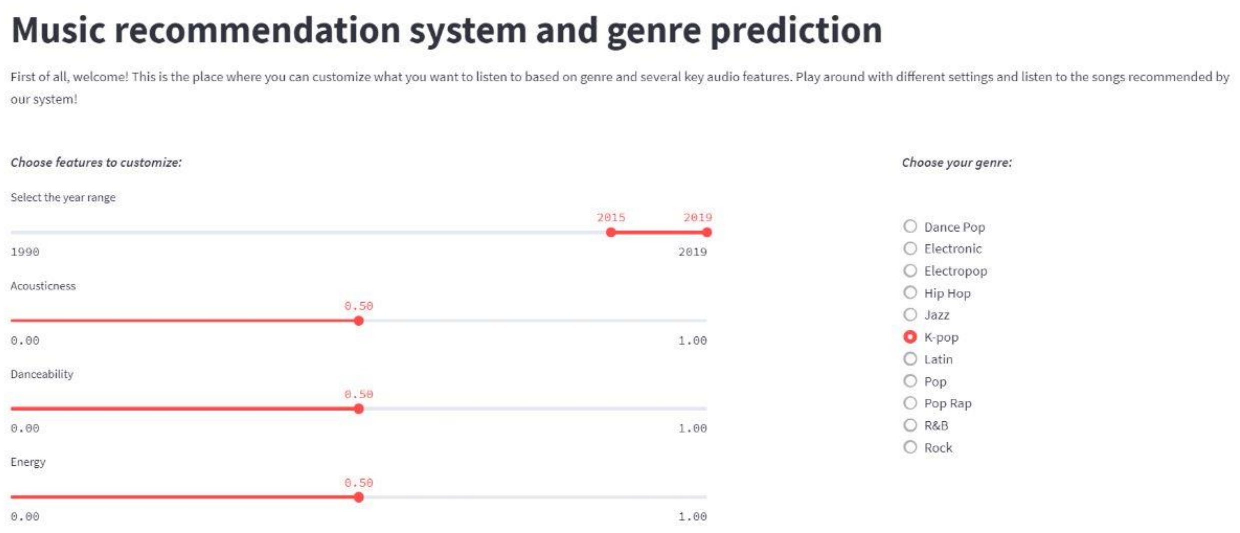
```
[ ] num_data=data.drop(["title","Unnamed: 0","song_name","analysis_url","track_href","uri","id","type"],axis=1) #drop all non-numeric columns
    X=num_data.drop("genre",axis=1)
    y=num_data["genre"]
```

```
[ ] X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=1)
```

We will use 3 different models for this problem:

1. Logistic regression
2. K-Nearest
3. Random Forest

```
[ ] models={"LogReg":LogisticRegression(),
           "KNN":KNeighborsClassifier(),
           "Random Forest":RandomForestClassifier()}
    def fit_and_score (models,X_train,X_test,y_train,y_test):
        """
        Fits and evaluates given machine learning models
        """
        np.random.seed(1)
        model_scores={}
        for name , model in models.items():
            model.fit(X_train,y_train)
            model_scores[name]=model.score(X_test,y_test)
        return model_scores
```



Music recommendation system and genre prediction

First of all, welcome! This is the place where you can customize what you want to listen to based on genre and several key audio features. Play around with different settings and listen to the songs recommended by our system!

Choose features to customize:

Select the year range: 2015 – 2019 (1990 – 2019)

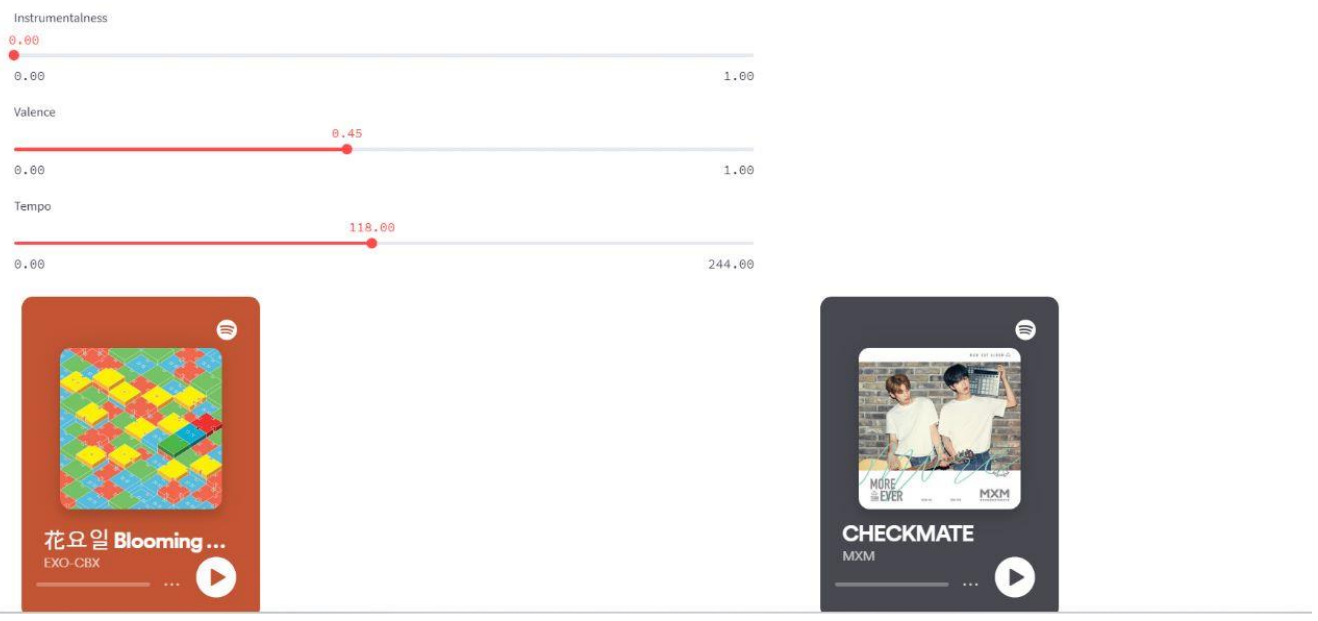
Acousticness: 0.50 (0.00 – 1.00)

Danceability: 0.50 (0.00 – 1.00)

Energy: 0.50 (0.00 – 1.00)

Choose your genre:

- Dance Pop
- Electronic
- Electropop
- Hip Hop
- Jazz
- [✓ K-pop]
- Latin
- Pop
- Pop Rap
- R&B
- Rock



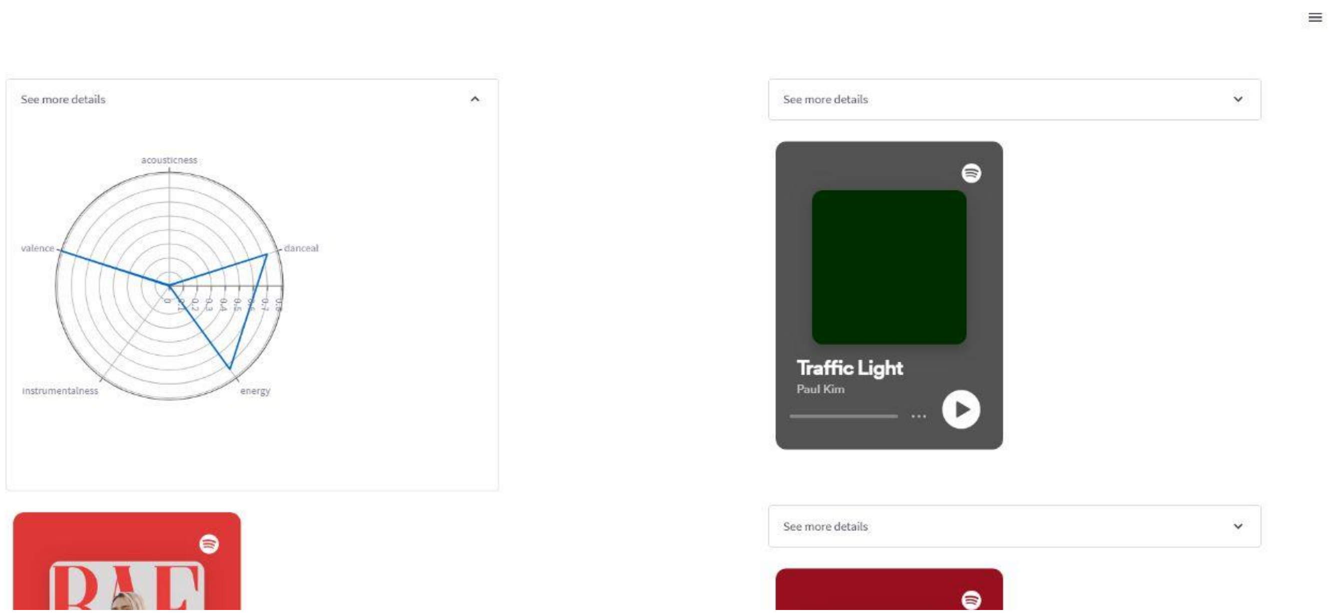
Instrumentalness: 0.00 (0.00 – 1.00)

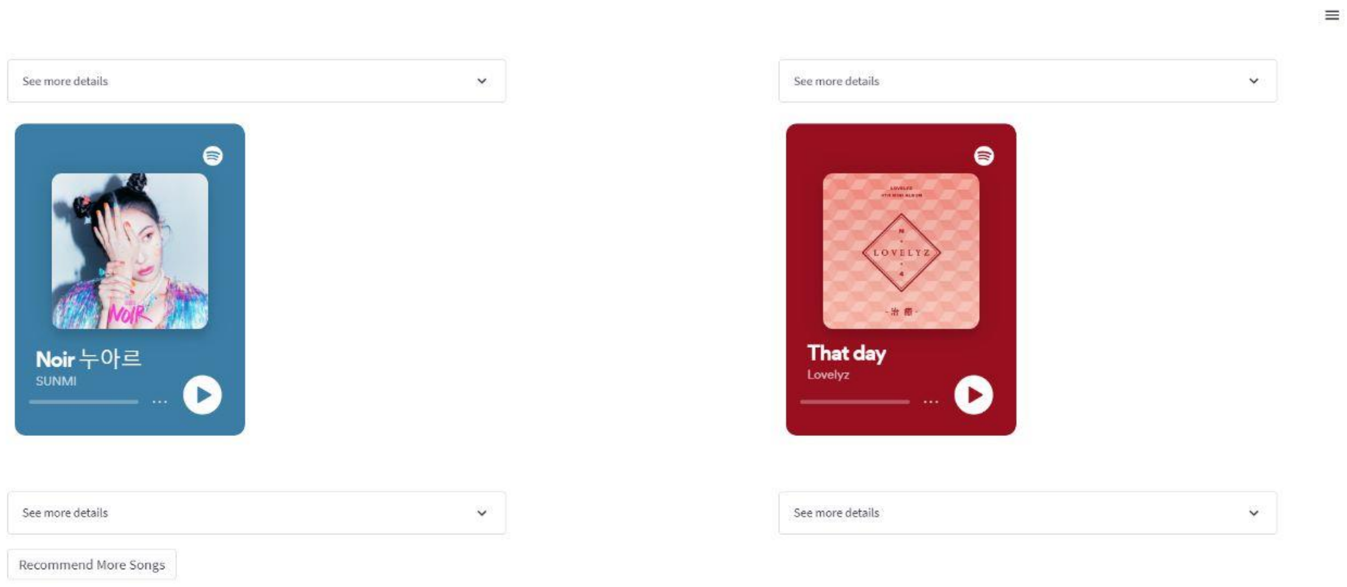
Valence: 0.45 (0.00 – 1.00)

Tempo: 118.00 (0.00 – 244.00)

花요일 Blooming ...
EXO-CBX

CHECKMATE
MXM


See more details
acousticness
valence
danceal
instrumentalness
energy
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
See more details
Traffic Light
Paul Kim
See more details


See more details
See more details
Noir 누아르
SUNMI
That day
Lovelyz
See more details
See more details
Recommend More Songs

EXPERIMENTS AND RESULTS

The experiments that were conducted to evaluate the performance of this recommendation system include:

Feature importance analysis: This experiment involves analyzing the importance of different song metadata features (key, genre, danceability, mode, tempo) in the ML model. This can help identify which features are most important for making accurate recommendations and can guide future feature engineering efforts

Accuracy comparison: The below mentioned graph depicts how the models created for genre prediction perform in predicting the genres with Random forest showing the highest accuracy for predictions. Even though the Random Forest has the highest score, it is not able to reach a 90% accuracy due to common causes of overfitting and they cannot be avoided as they have a tendency of causing high variance and high test errors.

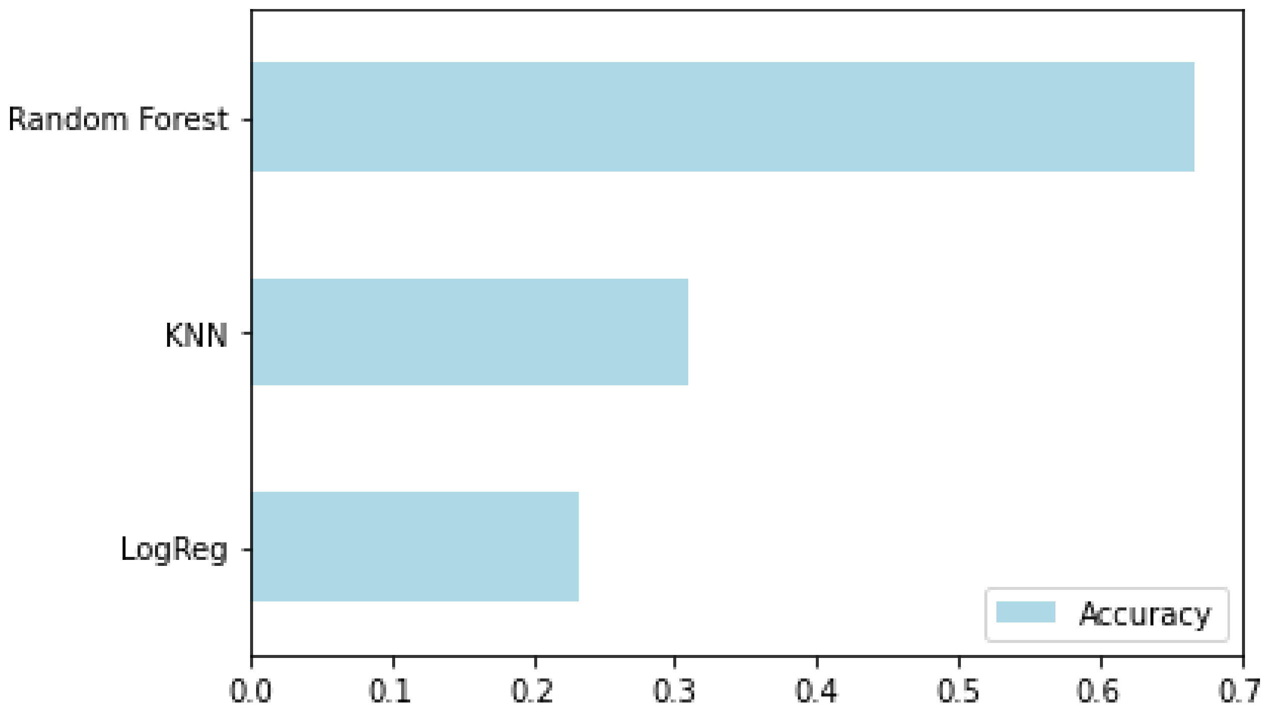


Fig 11 - Accuracy plot of the model

CONCLUSION AND FUTURE WORK

The use of song metadata in conjunction with user listening history can significantly improve the accuracy of song recommendations. This approach provides a more personalised and tailored experience to users, which can increase user satisfaction and engagement with the music platform.

Collaborative filtering can be an effective method to make recommendations based on user listening history. However, the ML model that takes into account song metadata features can potentially provide even more accurate recommendations.

Future work could focus on improving the ML model by incorporating additional features or data sources. For example, sentiment analysis of song lyrics could be used to identify songs with similar themes or moods, or user feedback on recommended songs could be used to improve the model's performance.

The ultimate goal of these improvements is to enhance the user experience and increase user engagement with the music platform. By providing accurate and personalized recommendations, users are more likely to spend more time on the platform, discover new music, and ultimately enjoy their listening experience more.

In conclusion, the ML project using song metadata and user listening history to make song recommendations has shown promising results for improving the user experience. Future work can focus on incorporating additional features and data sources to further improve the accuracy of recommendations and increase user satisfaction. The end goal is to create a more personalized and enjoyable listening experience for users.

REFERENCES

1. “Dataset of songs in Spotify” - https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify
2. "Collaborative Filtering Recommender Systems" by Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl (2001)
3. "Content-Based Recommendation Systems" by Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich (2010)
4. "Music Recommendation and Discovery in the Long Tail" by Chris Anderson (2004)
5. "Music information retrieval" edited by Zbigniew W. Ras and Alicja Wieczorkowska (2010)
6. "Deep Learning for Music Recommendation: Challenges and Opportunities" by Yi-Hsuan Yang, Yi-An Chen, and Zhe-Cheng Fan (2019)
7. "A Survey on Music Recommendation Systems" by Jonghyun Kim and Yejin Jang (2016)
8. "The Million Song Dataset" by Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere (2011)
9. "The Spotify Dataset" by Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M. Schmidt, and Xavier Serra (2017)
10. "Last.fm Dataset" by Last.fm (2014)
11. "A Survey on Deep Learning Techniques for Music Generation" by Yongqiang Li, Li Li, Ying Zhang, Jing Li, and Dongxin Liu (2020)
12. "The Music Genome Project" by Pandora (2008)
13. "Music Recommendation Using Collaborative Filtering and Word Embedding" by Jun Wang, Yun Zhang, Jiaxiang Wu, and Chengzhong Xu (2019)
14. "Multi-armed Bandit for Music Recommendation" by Daniel Oskarsson and Vasilis Vryniotis (2019)
15. "Music Emotion Recognition: State of the Art and Future Prospects" by Wenqiang Lei, Tao Liu, and Jianhua Tao (2020)
16. "Exploiting User Reviews for Music Genre Classification Using Deep Convolutional Neural Networks" by Saumitra Mishra, Oded Nov, and Luis Gravano (2018)
17. "A Hybrid Approach for Music Recommendation based on Collaborative Filtering and Audio Content Analysis" by Jaesik Choi, Kyogu Lee, and Juhan Nam (2014)
18. "A Comparative Study of Collaborative Filtering Algorithms for Music Recommendation" by Wei-Nan Zhang, Jian Xu, Yun He, and Xiangyang Xue (2008)
19. "A Hybrid Method of Music Recommendation Using Content-Based and Collaborative Filtering" by Youngwook Kim, Suhyun Choi, and Chae-Jung Park (2015)
20. "Music Recommendation Using Tree-Based Collaborative Filtering with Graph Embedding" by Jun Wang, Yu Zhang, and Chengzhong Xu