INDUSTRIAL TRAINING REPORT

ON

MUSIC RECCOMENDATION SYSTEM

submitted in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By

NITIKA TYAGI
Roll no: 10815602721
(T2 CSE 3rd year)

Under the guidance of
Ms. Pratibha Dabas, Assistant Professor, CSE Department

Department of Computer Science & Engineering
Dr. Akhilesh Das Gupta Institute of Technology & Management
(Guru Gobind Singh Indraprastha University, Dwarka, Delhi)
New Delhi -110053

CERTIFICATE

I/We hereby certify that the work that is being presented in the industrial training report entitled **Title** to the partial fulfillment of the requirements for the award of the degree of **Bachelor of Computer Science & Engineering** from **Dr. Akhilesh Das Gupta Institute of Technology & Management**, New Delhi. This is an authentic record of our own work carried out during a period from June 2023 to July 2023 under the guidance of **Ms.Pratibha Dabas , Assistant Professor CSE department.**

The matter presented in this project has not been submitted by us for the award of any other degree elsewhere.

Nitika Tyagi
10815602721

Pahul Singh
00815602721

Mayank Singh
05396202721

This is to certify that the above statement made by the candidate is correct to the best of my knowledge. He/She/They are permitted to appear in the External Examination.

Ms. Pratibha Dabas
Assistant Professor
CSE Department

The B. Tech Major Project Viva-Voce Examination of **Name of the Student (Enrollment No: xxx),** has been held on **…….**

Dr. Jyoti Parashar
Training Coordinator, CSE

Prof. (Dr.) Ankit Verma
Head, CSE

ACKNOWLEDGEMENT

I/We would like to acknowledge the contributions of the following persons, without whose help and guidance this report would not have been completed.

I/We acknowledge the counsel and support of our project guide **Ms Pratibha Dabas, Assistant Professor, CSE department,** with respect and gratitude, whose expertise, guidance, support, encouragement, and enthusiasm has made this report possible. Their feedback vastly improved the quality of this report and provided an enthralling experience. I/We are indeed proud and fortunate to be supervised by him.

We are thankful to, **Prof. (Dr.) Ankit Verma, H.O.D. CSE Department, Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi** for his constant encouragement, valuable suggestions and moral support and blessings.

I/We are immensely thankful to our esteemed, **Prof. (Dr.) Sanjay Kumar, Director, Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi** for his never-ending motivation and support.

I/We shall ever remain indebted to**, Dr. Jyoti Parashar, Training Coordinator CSE Department** and faculty and staff members of Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi.

Finally, yet importantly, I/We would like to express our heartfelt thanks to God, our beloved parents for their blessings, our friends/classmates for their help and wishes for the successful completion of this project.

Nitika Tyagi
10815602721

Pahul singh
00815602721

Mayank singh
05396202721

TRAINING CERTIFICATE

Allsoft Solutions
An ISO 9001 : 2008 CERTIFIED COMPANY

PROJECT COMPLETION CERTIFICATE

In recognition of the commitment to achieve professional excellence this is to certify that Ms./Mr.

Nitika Tyagi

has successfully completed an Industry-oriented project.

Project Name	Music recommendation system
Technologies Used	Data Science & Machine Learning - Jupyter notebook,Back end- Machine Learning,Scientific python,GUI-pysimple, Pandas, Numpy, Seaborn, Matplotlib, Scikit learn
Reference No.	CE/2023/B/AUG/0010
Training Date	31st July, 2023 - 31st August, 2023
Training Duration	90 hours
Training Location	Dr. Akhilesh Das Gupta Institute of Technology & Management

Program Co-ordinator
Industry/Academic Alliance

Director
Training and Development
Allsoft Solutions and Services

Business Partner IBM

BIG DATA - ANALYTICS | IoT | ORACLE | J2EE | PHP | CLOUD COMPUTING

ABSTRACT

Music recommendation systems have become increasingly popular in recent years due to the vast amount of music available online. These systems use machine learning algorithms to recommend music to users based on their listening history and preferences this project aims todevelop a machine-learning model that recommends music to users based on their preferences. The model will be trained on a dataset of music metadata and user preferences. The model will use collaborative filtering and content-based techniques to recommend musicto users based on their preferences of other users with similar tastes The model will be developed using Python and the scikit-learn library. The scikit-learn library provides a wide range of machine-learning algorithms that can be used for collaborative filtering. The model will be evaluated using cosine similarity and min-max scaling on a test dataset of music metadata. The results of the evaluation will be used to fine-tune the model and improve its accuracy The dataset used for training the model will consist of music metadata such as artist name, song name, genre and the properties of the song like acousticness, danceability, etc .The dataset willbe used to train the collaborative filtering and content filtering algorithms.

TABLE OF CONTENTS

Certificate i
Acknowledgement ii
Training Certificate iii
Abstract iv
Table of Contents v
List of Figure vi
List of Tables vii

CHAPTER 1: INTRODUCTION AND LITERATURE REVIEW 9

1.1. Introduction
1.1.1 Python
1.1.2 Machine Learning
1.2. Motivation
1.3. Problem Statement
1.4. Objective
1.5. Feasibility Study
1.6. Related Works
1.7. Beneficiary of the system

CHAPTER 2: METHODOLOGY ADOPTED 16

2.1 Importing libraries
2.2 Reading Data
2.3 Data Exploration
2.4 EDA
2.5 Data Transformation
2.6 Model 1(k-clustering)
2.7 Model 2(cosine similarity)

CHAPTER 3: DESIGNING AND RESULT ANALYSIS 20

3.1 Reading and Normalization of data
3.2 EDA
3.3 Model 1(k-Clustering)
3.4 Model 2(cosine similarity)

CHAPTER 4: CONCLUSIONS AND FUTURE SCOPE 28

4.1 CONCLUSION
4.2 FUTURE SCOPE

REFERENCES 29

List of Figures

Figure No.	Title of Figure	Page No.
1.1	Types of machine learning algorithms	11
1.2	working of Machine Learning algorithm	12
3.1	Importing files	20
3.2	Reading Data	20
3.3	step1: removing redundant columns	21
3.4	step2: removing redundant columns	21
3.5	step3: removing redundant columns	21
3.6	EDA: Step 1	22
3.7	EDA: Step 2	22
3.8	EDA: Step 3	22
3.9	Correlation plot	23
3.10	Count Plot	24
3.11	Bar Plot	24
3.12	Line Plot	25
3.13	Model 1: step1	25
3.14	Model 1: main function	26
3.15	Model 1:Output	26
3.16	Model 2:step1	26
3.17	Model 2:Dataframe extraction	27
3.18	Model 2:Main Function	27
3.19	Model 2:Output	27

List of Graphs

Table No.	Title of Graphs	Page No.
3.1	Feature Correlation plot	23
3.2	Count Plot	24
3.3	Bar Plot	24
3.4	Line plot	25

CHAPTER 1 INTRODUCTION AND LITERATURE REVIEW

1.1 INTRODUCTION

Music recommendation systems have become increasingly popular in recent years due to the vast amount of music available online. These systems use machine learning algorithms to recommend music to users based on their listening history and preferences. Collaborative filtering and content based recommendation are popular techniques used in music recommendation systems that recommends music to users based on the listening histories of other users with similar tastes.. This project aims to develop a machine learning model that recommends music to users based on their preferences. The model will be trained on a dataset of music metadata and user preferences. The model will use collaborative filtering and content based filtering techniques.

1.1.1 PYTHON

Python is an interpreted **high-level general-purpose programming language**. Its design philosophyemphasizes code readability with its use of significant indentation.
Its language constructs as well as **its object-oriented** approach aim to help programmers write clear,logical code for small and large-scale projects. Python is **dynamically-typed** and garbage-collected.

It supports multiple programming paradigms, including structured (particularly, procedural), object- oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000and introduced new features, such as list comprehensions and a cycle-detecting garbage collectionsystem (in addition to reference counting).

Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020. Python consistentlyranks as one of the most popular programming languages.

Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it's relatively easy to learn, Python has been adopted by many non- programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

Python has become a staple in data science, allowing data analysts and other professionals to use thelanguage to conduct complex statistical calculations, create data visualizations, build machine learning algorithms, manipulate and analyze data, and complete other data-related tasks.

Python can build a wide range of different data visualizations, like line and bar graphs, pie charts, histograms, an3D plots. Python also has a number of libraries that enable coders to write

programs for data analysis and machine learning more quickly and efficiently, like TensorFlow andKeras.

Python is popular for a number of reasons. Here's a deeper look at what makes it so versatile andeasy to use for coders.

- It has a **simple syntax** that mimics natural language, so it's easier to read and understand. This makes it quicker to build projects, and faster to improve on them.
- It's **versatile**. Python can be used for many different tasks, from web development tomachine learning.
- It's **beginner friendly**, making it popular for entry-level coders.
- It's **open source**, which means it's free to use and distribute, even for commercialpurposes.
- Python's archive of **modules and libraries**—bundles of code that third-party usershave created to expand Python's capabilities—is vast and growing.
- Python has a **large and active community** that contributes to Python's pool of modules and libraries, and acts as a helpful resource for other programmers. The vastsupport community means that if coders run into a stumbling block, finding a solutionis relatively easy; somebody is bound to have encountered the same problem before.

HISTORY OF PYTHON

Python was conceived in the late 1980s by **Guido van Rossum** at **Centrum Wiskunde & Informatica (CWI)** in the **Netherlands** as a successor to the ABC programming language, whichwas inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system.

Its implementation began in December 1989. Van Rossum shouldered sole responsibility for theproject, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life", a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member "Steering Council" to lead the project.

1.1.2 MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses onthe use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning is important because it gives enterprises a view of trends in customerbehavior and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google, and Uber, make machine learning acentral part of their operations. Machine learning has become a significant competitive
differentiator for many companies.

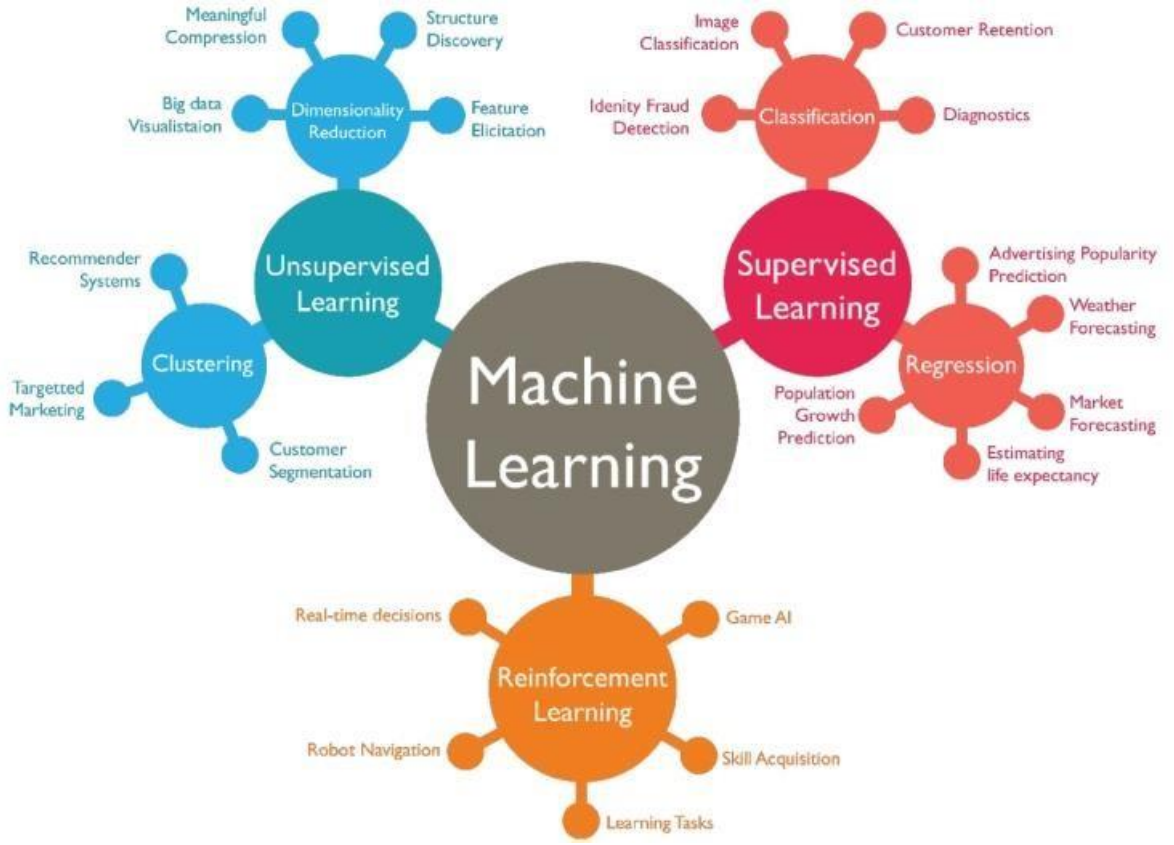


Fig:1.1: Types of machine learning algorithms

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences ontheir own.

With the help of sample historical data, which is known as training data, machine learning algorithms build a mathematical model that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.

A machine has the ability to learn if it can improve its performance by gaining more data.

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:

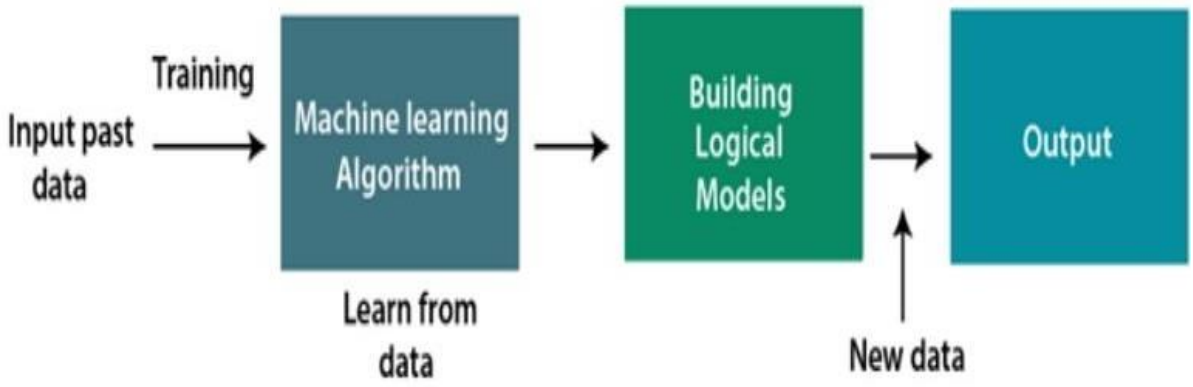


Fig:1.2 working of Machine Learning algorithm

NEED FOR MACHINE LEARNING

The need for machine learning is increasing day by day. The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly. As a human, we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.

We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically. The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function. With the help of machine learning, we can save both time and money.

The importance of machine learning can be easily understood by its use's cases, Currently, machine learning is used in self-driving cars, cyber fraud detection, face recognition, and friend suggestion by Facebook, etc. Various top companies such as Netflix and Amazon have built machine learning models that are using a vast amount of data to analyze the user interest and recommend product accordingly.

FEATURES OF MACHINE LEARNING

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology

1.2 MOTIVATION

"Music is the soundtrack of your life." - Dick Clark

Music recommendation systems have become increasingly popular in recent years due to the vast amount of music available online. However, these systems often fail to take into account the style of music that the user has selected as their favourites. This can lead to recommendations that are not relevant to the user's interests and can result in a poor user experience. By developing a machine learning model that recommends music to users based on their preferences, we can provide more accurate and relevant recommendations to users. This will improve the user experience and increase user engagement with the music recommendation system. Furthermore, this project will contribute to the field of machine learning by exploring the use of collaborative filtering and content filtering techniques for recommendations. The results of this project will be useful for music streaming services and other companies that provide music recommendation systems

1.3 PROBLEM STATEMENT

The problem with existing music recommendation systems is that they often fail to take into account the style of music that the user has selected as their favorites. This can lead to recommendations that are not relevant to the user's interests. By developing a machine-learning model that recommends music to users based on their preferences, we can provide more accurate and relevant recommendations to users. This will improve the user experience and increase user engagement with the music recommendation system.

1.4 OBJECTIVE

The objective of this project is to develop a machine-learning model that recommends music to users based on their preferences. The model will be trained on a dataset of music metadata and user preferences. The model will use collaborative filtering and content based filtering to recommend music to users. The model will also take into account the style of music that the user has selected as their favorite. The model will be evaluated using cosine similarity and min-max scaling on a test dataset of user preferences. The results of the evaluation will be used to fine-tune the model and improve its accuracy

1.5 FEASIBILITY STUDY

THE scope of this project is huge as the number of music listening users are increasing day by day. Many music platforms such as Youtube, Spotify , Apple Music are using new music recommendation techniques every day. They are constantly trying to take into account the style of music that user wants to read to improve user experience and in turn, increase their markets
and profits. The target audience for this industry is the young generation that continuously relies
on music as the best spare time activity and their escape from the real world hence this project
will be very useful in the future.

Hardware Requirements:

A System having Windows 11.

Software Requirements:

Any python running environment such as python IDLE, VS code, JUpyter notebook, PYcham

1.6 RELATED WORKS

Collaborative Filtering for Music Recommendation: A Survey" by Paolo Cremonesi, et al. (2010):
This survey paper provides an in-depth overview of collaborative filtering techniques applied to music recommendation. It discusses user-item interaction data, user-based and item-based approaches, matrix factorization, and challenges like the cold start problem.

"Content-Based Music Recommendation: A Comprehensive Survey" by Li Chen and Shiguang Shan (2012):
Focusing on content-based methods, this survey covers various features such as audio content, lyrics, and metadata for music recommendation. It discusses feature extraction, similarity metrics, and integration of multiple modalities for more accurate recommendations.

"Hybrid Recommender Systems: Survey and Experiments" by Xavier Amatriain and Nuria Oliver (2013):
This paper explores hybrid recommendation systems that combine collaborative filtering and content-based approaches. It discusses different hybrid strategies, such as weighted combination and feature augmentation, along with their advantages.

"A Context-Aware Music Recommendation System for Mobile Devices" by Markus Schedl, et al. (2013):
Focusing on context-aware systems, this study presents a recommendation approach that incorporates contextual factors like time, location, and user activity to provide personalized music recommendations, particularly for mobile users.

"Neural Collaborative Filtering" by Xiangnan He, et al. (2017):
This paper introduces neural network-based approaches for collaborative filtering. It demonstrates how neural networks can capture complex user-item interactions and improve recommendation accuracy.

Deep Content-Based Music Recommendation" by Jeffrey A. Nelder and Norman W. Paton (2019):
This work explores the application of deep learning in content-based music recommendation. It investigates the effectiveness of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) in capturing sequential dependencies in music data.

1.7BENEFICIARY OF THE SYSTEM

The beneficiaries of a music recommendation model based on genre can include:

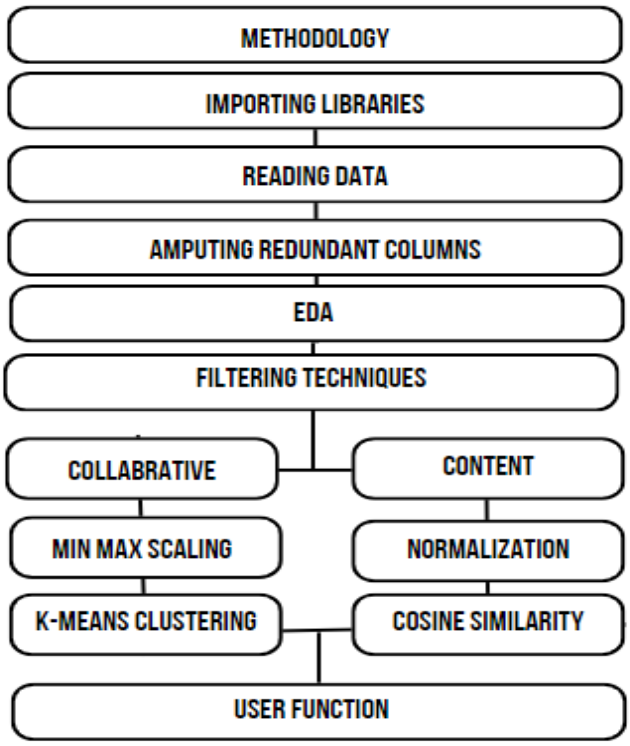
1. **Music Streaming Services:** By providing personalized recommendations and improvingthe user experience, a music recommendation model based on preferences can increase user retention and revenue for music streaming services.

2. **Music Listeners:** A music recommendation model based on preferences can help users discover new music that they may not have found otherwise. This can lead tobetter music discovery and a more diverse listening experience.

3. **Music Artists:** By recommending their music to users who are more likely to enjoy it, a music recommendation model based on preferences can help music artists reach new audiences and increase their fan base.

4. **Music Industry:** A music recommendation model based on preferences can help the music industry by promoting new artists and genres that music

CHAPTER 2 METHEDOLOGY ADAPTED



2.1 Importing libraries:

This step is about importing the necessary libraries that will be used in the project. Libraries imported: **numpy, panda, seaborn, matplotlib, warnings, tqdm, sklearn, scipy**

2.2 Reading data:

In this step we read the necessary csv files required by using pandas read_csv function. This function is responsible for reading csv files.
Dataset : SPOTIFY DATASET/ Kaggle
Link: https://www.kaggle.com/datasets/vatsalmavani/spotify-dataset
Data exploration : in which we used isnull, info and describe to have a better look at data and then we dropped the not useful columns

2.3Dropping not useful column and columns:

In this step we removed the unnecessary columns that were not necessary for the analysis or had a lot of missing values. We used drop method of pandas for the same and found the correlation between numerical colunms

2.4 EDA:

This step is about exploring the data to understand its characteristics. We used descriptive statistics and data visualization techniques to explore the data.
We used the following tecniques:

1. **Feature correlation plot.** – We used Pearson correlation method in this to find correlation of the various parameters of the dataset with popularity.

2. **Count plot**-We used it to show the number of songs produced in each decade from 1920s to presnt.

3.Line plot: We used line plot to show the relation of the various sound features with the year the music was made in .

4. **Bar Plot**: We used bar plot to show the average sound features of the songs belonging to top 10 genres of our dataset.

2.5 Data Transformation :

In this step we transformed the data using normalization to make it more suitable for our model .

Normalization is a technique used to scale numeric data in a way that it falls within a specific
range. The goal of normalization is to change the values of numeric columns in the dataset to use
a common scale, without distorting differences in the ranges of values or losing information.
There are different types of normalization techniques such as:

- Min-Max normalization
- Z-score normalization
- Decimal scaling normalization

MinMaxScaler : is a method used for scaling features in the range [0, 1]. It is a type of normalization technique that scales the data to a fixed range - usually [0, 1]. This scaler works by subtracting the minimum value in the feature and then dividing by the range. The range is the difference between the original maximum and original minimum value. This method is implemented in scikit-learn library which is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, kmeans and DBSCAN.

2.6 MODEL 1 :

K means clustering algorithm: In this step we udes k-clustering algorithm to cluster similar items togeather.

K-means clustering is an unsupervised learning algorithm that is used to solve clustering problems in machine learning or data science. The algorithm takes the unlabeled dataset as

input, divides the dataset into k-number of clusters, and repeats the process until it does not find

the best clusters.

The working of the K-Means algorithm is explained in the following steps[1]:

1. Select the number K to decide the number of clusters.
2. Select random K points or centroids.
3. Assign each data point to their closest centroid, which will form the predefined K clusters.
4. Calculate the variance and place a new centroid of each cluster.
5. Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
6. If any reassignment occurs, then go to step-4 else go to FINISH.
7. The model is ready.

Function for recommend songs to the users: we did this using distance between two colunms using min max scaler and sorted it and recommended top 10 songs

2.7 MODEL 2 :

Content-based Recommendation System Selection the features with song id as index: In this step we select features from the table through the song id. Then we used normalize function from sklearn library.

A content-based recommendation system is a type of recommendation system that recommends items to users based on their past behavior on the website and the similarities between items The goal of a content-based recommendation system is to recommend items to a user that are similar to items that they have previously interacted with. The content-based recommendation system works on two methods, both of them using different models and algorithms. One uses the vector spacing method and is called method 1, while the other uses a classification model and is called method . Recommending items to users based on content can be done using the cosine distance between the vectors of the item and the user to determine its preference.

Cosine similarity: Cosine similarity is a measure of similarity between two vectors, often used to measure document similarity in text analysis. It is calculated as the normalized dot product of the vectors1 . There are several ways to calculate cosine similarity in Python, but one of the fastest and most common ways is to use the **sklearn.metrics.pairwise.cosine_similarity** function from the **scikit-learn** library

Creating new dataframe with id and name: In this step we created a new dataframe with song id and song name so that we can fetch song id of the song by inputing sing name from the user.

Function for recommend songs to the users: This is the function that takes the input of song name from the user and gives similar songs as output. We used distance method using cosin and made a different data frame using all songs with normalized value

CHAPTER 3 DESIGNING AND RESULT ANALYSIS

3.1 Reading and normalization of data.

Spotify Recommendation System using Python

```
###Importing useful libraries

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

from scipy import sparse
import random

from tqdm import tqdm
sns.set()
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans

import warnings
warnings.filterwarnings("ignore")

import ast
from scipy.spatial.distance import cosine, euclidean, hamming
from sklearn.preprocessing import normalize
```

Fig 3.1

READING DATASET

```
### reading the dataset
data = pd.read_csv("D:/PROGRAMMING/JAVA/Notes/data.csv")
```

```
data.head()
```

	valence	year	acousticness	artists	danceability	duration_ms	energy	explicit	id	instrumentalness	key	liveness	loudness
0	0.0594	1921	0.982	['Sergei Rachmaninoff', 'James Levine', 'Berli...	0.279	831667	0.211	0	4BJqT0PrAfrxzMOxytFOIz	0.878000	10	0.665	-20.096
1	0.9630	1921	0.732	['Dennis Day']	0.819	180533	0.341	0	7xPhfUan2yNtyFG0cUWkt8	0.000000	7	0.160	-12.441
2	0.0394	1921	0.961	['KHP Kridhamardawa Karaton Ngayogyakarta Hadi...	0.328	500062	0.166	0	1o6I8BglA6ylDMrIELygv1	0.913000	3	0.101	-14.850
3	0.1650	1921	0.967	['Frank Parker']	0.275	210000	0.309	0	3ftBPsC5vPBKxYSee08FDH	0.000028	5	0.381	-9.316
4	0.2530	1921	0.957	['Phil Regan']	0.418	166693	0.193	0	4d6HGyGT8e121BsdKmw9v6	0.000002	3	0.229	-10.096

Fig 3.2

```
In [122]: ### Data Exploration
          data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 170653 entries, 0 to 170652
Data columns (total 19 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   valence           170653 non-null  float64
 1   year              170653 non-null  int64
 2   acousticness      170653 non-null  float64
 3   artists           170653 non-null  object
 4   danceability      170653 non-null  float64
 5   duration_ms       170653 non-null  int64
 6   energy            170653 non-null  float64
 7   explicit          170653 non-null  int64
 8   id                170653 non-null  object
 9   instrumentalness  170653 non-null  float64
 10  key               170653 non-null  int64
 11  liveness          170653 non-null  float64
 12  loudness          170653 non-null  float64
 13  mode              170653 non-null  int64
 14  name              170653 non-null  object
 15  popularity        170653 non-null  int64
 16  release_date      170653 non-null  object
 17  speechiness       170653 non-null  float64
 18  tempo             170653 non-null  float64
dtypes: float64(9), int64(6), object(4)
memory usage: 24.7+ MB
```

Fig 3.3

```
In [123]: data.describe()
```

Out[123]:

	valence	year	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	key	liveness	
count	170653.000000	170653.000000	170653.000000	170653.000000	1.706530e+05	170653.000000	170653.000000	170653.000000	170653.000000	170653.000000	1
mean	0.528587	1976.787241	0.502115	0.537396	2.309483e+05	0.482389	0.084575	0.167010	5.199844	0.205839	
std	0.263171	25.917853	0.376032	0.176138	1.261184e+05	0.267646	0.278249	0.313475	3.515094	0.174805	
min	0.000000	1921.000000	0.000000	0.000000	5.108000e+03	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.317000	1956.000000	0.102000	0.415000	1.698270e+05	0.255000	0.000000	0.000000	2.000000	0.098800	
50%	0.540000	1977.000000	0.516000	0.548000	2.074670e+05	0.471000	0.000000	0.000216	5.000000	0.136000	
75%	0.747000	1999.000000	0.893000	0.668000	2.624000e+05	0.703000	0.000000	0.102000	8.000000	0.261000	
max	1.000000	2020.000000	0.996000	0.988000	5.403500e+06	1.000000	1.000000	1.000000	11.000000	1.000000	

Fig 3.4

```
In [125]: ### correlation between the feature
          df = data.drop(columns=['id', 'name', 'artists', 'release_date','year'])
          df.corr()
```

Out[125]:

	valence	acousticness	danceability	duration_ms	energy	explicit	instrumentalness	key	liveness	loudness	mode	popularity
valence	1.000000	-0.184101	0.558946	-0.191813	0.353876	-0.018613	-0.198501	0.028473	0.003832	0.313512	0.015641	0.014200
acousticness	-0.184101	1.000000	-0.266852	-0.076373	-0.749393	-0.246007	0.329819	-0.020550	-0.024482	-0.561696	0.047168	-0.573162
danceability	0.558946	-0.266852	1.000000	-0.139937	0.221967	0.241757	-0.278063	0.024439	-0.100193	0.285057	-0.045956	0.199606
duration_ms	-0.191813	-0.076373	-0.139937	1.000000	0.042119	-0.048880	0.084770	-0.004266	0.047168	-0.003037	-0.046085	0.059597
energy	0.353876	-0.749393	0.221967	0.042119	1.000000	0.132723	-0.281101	0.027705	0.126192	0.782362	-0.039260	0.485005
explicit	-0.018613	-0.246007	0.241757	-0.048880	0.132723	1.000000	-0.140987	0.005432	0.039640	0.140300	-0.078872	0.191543
instrumentalness	-0.198501	0.329819	-0.278063	0.084770	-0.281101	-0.140987	1.000000	-0.014591	-0.047193	-0.408611	-0.036543	-0.296750
key	0.028473	-0.020550	0.024439	-0.004266	0.027705	0.005432	-0.014591	1.000000	0.000205	0.017385	-0.116260	0.007826
liveness	0.003832	-0.024482	-0.100193	0.047168	0.126192	0.039640	-0.047193	0.000205	1.000000	0.056422	0.002641	-0.076464
loudness	0.313512	-0.561696	0.285057	-0.003037	0.782362	0.140300	-0.408611	0.017385	0.056422	1.000000	-0.010727	0.457051
mode	0.015641	0.047168	-0.045956	-0.046085	-0.039260	-0.078872	-0.036543	-0.116260	0.002641	-0.010727	1.000000	-0.028897
popularity	0.014200	-0.573162	0.199606	0.059597	0.485005	0.191543	-0.296750	0.007826	-0.076464	0.457051	-0.028897	1.000000
speechiness	0.046381	-0.043980	0.235491	-0.084604	-0.070555	0.414070	-0.121700	0.023784	0.134667	-0.139296	-0.057796	-0.171979
tempo	0.171689	-0.207120	0.001801	-0.025472	0.250865	0.011969	-0.105361	0.002629	0.007714	0.209774	0.011637	0.133310

Fig 3.5

3.2 EXPLORATORY DATA ANALYSIS

Reading the data

```
In [ ]: data=pd.read_csv('C:/Summer Training/Project/spotify-dataset/data/data.csv')
        genre_data=pd.read_csv('C:/Summer Training/Project/spotify-dataset/data/data_by_genres.csv')
        year_data=pd.read_csv('C:/Summer Training/Project/spotify-dataset/data/data_by_year.csv')

In [ ]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 170653 entries, 0 to 170652
Data columns (total 19 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   valence           170653 non-null  float64
 1   year              170653 non-null  int64
 2   acousticness      170653 non-null  float64
 3   artists           170653 non-null  object
 4   danceability      170653 non-null  float64
 5   duration_ms       170653 non-null  int64
 6   energy            170653 non-null  float64
 7   explicit          170653 non-null  int64
 8   id                170653 non-null  object
 9   instrumentalness  170653 non-null  float64
 10  key               170653 non-null  int64
 11  liveness          170653 non-null  float64
 12  loudness          170653 non-null  float64
 13  mode              170653 non-null  int64
 14  name              170653 non-null  object
 15  popularity        170653 non-null  int64
 16  release_date      170653 non-null  object
 17  speechiness       170653 non-null  float64
 18  tempo             170653 non-null  float64
dtypes: float64(9), int64(6), object(4)
memory usage: 24.7+ MB
```

Fig: 3.6

```
genre_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2973 entries, 0 to 2972
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   mode              2973 non-null   int64
 1   genres            2973 non-null   object
 2   acousticness      2973 non-null   float64
 3   danceability      2973 non-null   float64
 4   duration_ms       2973 non-null   float64
 5   energy            2973 non-null   float64
 6   instrumentalness  2973 non-null   float64
 7   liveness          2973 non-null   float64
 8   loudness          2973 non-null   float64
 9   speechiness       2973 non-null   float64
 10  tempo             2973 non-null   float64
 11  valence           2973 non-null   float64
 12  popularity        2973 non-null   float64
 13  key               2973 non-null   int64
dtypes: float64(11), int64(2), object(1)
memory usage: 325.3+ KB
```

Fig: 3.7

```
year_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   mode              100 non-null    int64
 1   year              100 non-null    int64
 2   acousticness      100 non-null    float64
 3   danceability      100 non-null    float64
 4   duration_ms       100 non-null    float64
 5   energy            100 non-null    float64
 6   instrumentalness  100 non-null    float64
 7   liveness          100 non-null    float64
 8   loudness          100 non-null    float64
 9   speechiness       100 non-null    float64
 10  tempo             100 non-null    float64
 11  valence           100 non-null    float64
 12  popularity        100 non-null    float64
 13  key               100 non-null    int64
dtypes: float64(11), int64(3)
```

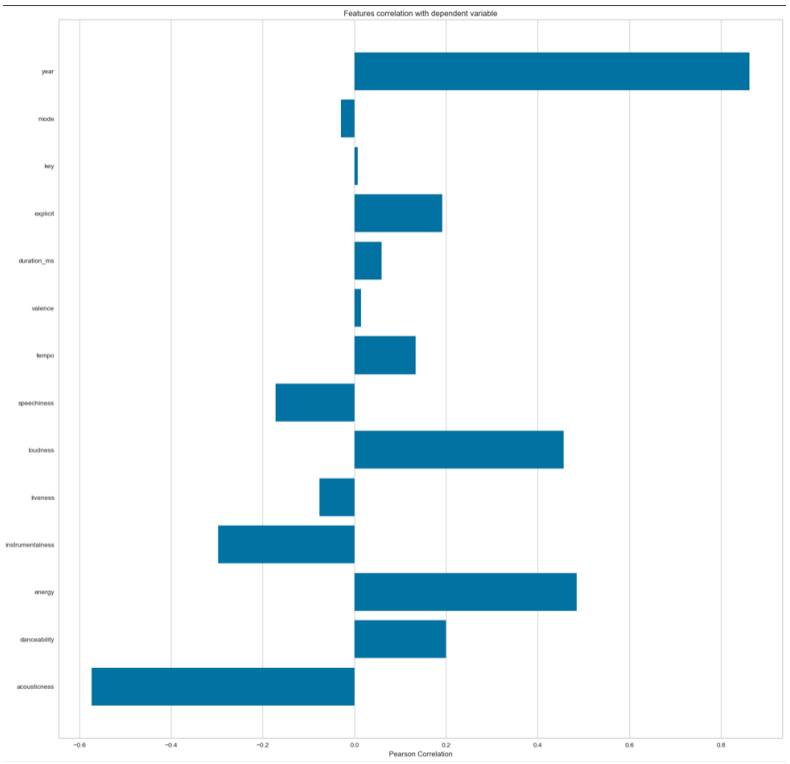
Fig:3.8

I .**Feature correlation plot.** – We used Pearson correlation method in this to find correlation of the various parameters of the dataset with popularity

Data Visualization and EDA

```
feature_names = ['acousticness', 'danceability', 'energy', 'instrumentalness',
       'liveness', 'loudness', 'speechiness', 'tempo', 'valence','duration_ms','explicit','key','mode','year']

X, y = data[feature_names], data['popularity']

# Create a list of the feature names
features = np.array(feature_names)

# Instantiate the visualizer
visualizer = FeatureCorrelation(labels=features)

plt.rcParams['figure.figsize']=(20,20)
visualizer.fit(X, y)     # Fit the data to the visualizer
visualizer.show()
```

Fig : 3.9



Graph:3.1

II. Count plot-We used it to show the number of songs produced in each decade from

1920s to presnt.

Relation of music with time

```
In [ ]: def get_decade(year):
            period_start = int(year/10) * 10
            decade = '{}s'.format(period_start)
            return decade

        data['decade'] = data['year'].apply(get_decade)

        sns.set(rc={'figure.figsize':(11 ,6)})
        sns.countplot(x=data['decade'])
```

```
Out[ ]: <Axes: xlabel='decade', ylabel='count'>
```

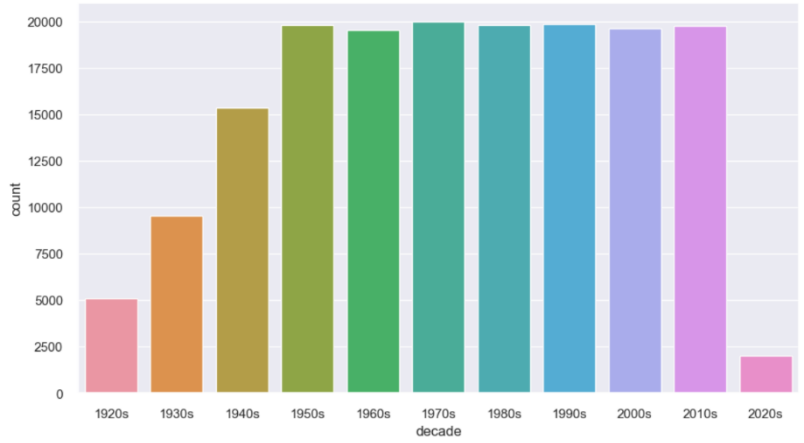


Fig 3.10 Graph:3.2

III. Bar Plot: We used bar plot to show the average sound features of the songs belonging to top 10 genres of our dataset.

Relation of Top 10 popular genres with sound features

```
In [ ]: top10_genres = genre_data.nlargest(10, 'popularity')

        fig = px.bar(top10_genres, x='genres', y=sound_features, barmode='group')
        fig.show()
```

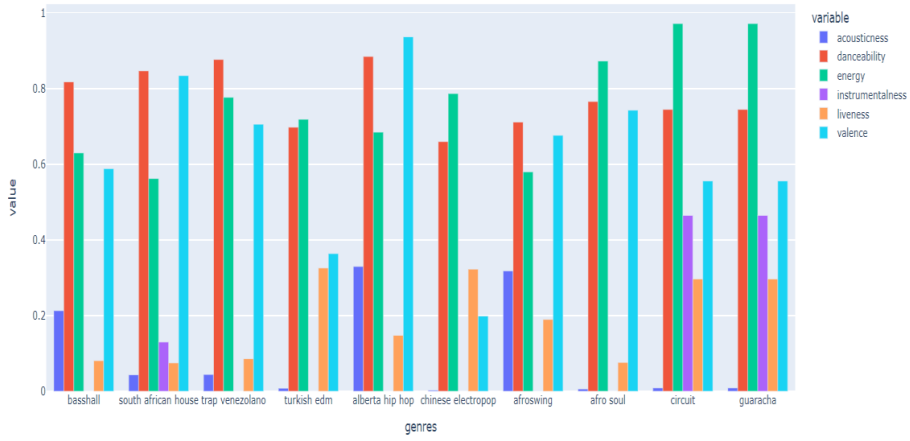


Fig 3.11 Graph 3.3

IV.Line plot: We used line plot to show the relation of the various sound features with the year the music was made in .

Relation of time with sound features

```
In [ ]: sound_features = ['acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'valence']
fig = px.line(year_data, x='year', y=sound_features)
fig.show()
```

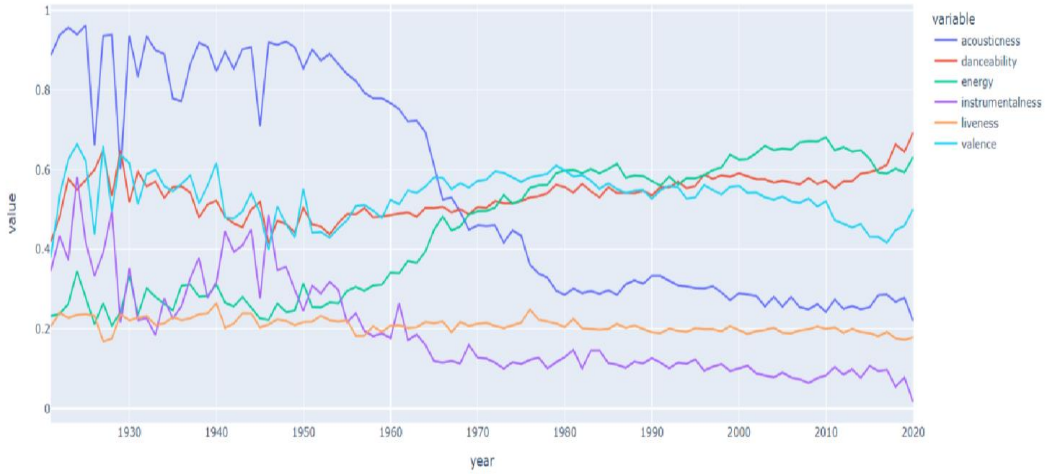


Fig 3.12 Graph:3.4

3.3 MODEL 1:

COLLABRATIVE FILTERING BASED MODEL USING K MEANS CLUSTERING

System Using collaborative filtering

```
In [126]: ###Data Transformation

          ### Normalize the dataset by using the MinMaxScaler method provided by the Scikit-Learn library in Python.
          datatypes = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
          normarization = data.select_dtypes(include=datatypes)
          for col in normarization.columns:
              MinMaxScaler(col)

In [127]: ### K means clustering algorithm
          kmeans = KMeans(n_clusters=10)
          features = kmeans.fit_predict(normarization)
          data['features'] = features
          MinMaxScaler(data['features'])

Out[127]:
```

	MinMaxScaler
2	3
3	7
4	1
..	..
170648	0
170649	1
170650	7
170651	6
170652	1

Name: features, Length: 170653, dtype: int32)

Fig:3.13

FUNCTION TO RECOMMEND SONGS

```
In [131]: ### function for recommend songs to the users
def recommend(songs):
        distance = []
        song = data[(data.name.str.lower() == songs.lower())].head(1).values[0]
        rec = data[data.name.str.lower() != songs.lower()]
        for songs in tqdm(rec.values):
            d = 0
            for col in np.arange(len(rec.columns)):
                if not col in [1, 3, 8, 14,16]:
                    d = d + np.absolute(float(song[col]) - float(songs[col]))
            distance.append(d)
        rec['distance'] = distance
        rec = rec.sort_values('distance')
        columns = ['artists', 'name']
        return rec[columns][:10]
```

Fig:3.14

OUTPUT MODEL 1:

```
In [132]: recommend("save me")
100%|| 170637/170637 [00:15<00:00, 11321.31it/s]
```

Out[132]:

	artists	name
28564	['Elton John']	Border Song
67104	['Camilo Sesto']	Amor No Me Ignores
90241	['Gorillaz', 'Daley']	Doncamatic (feat. Daley)
47250	['George Harrison']	Beware Of Darkness - 1st Version Recorded At A...
66900	['Dolly Parton']	You're the Only One
140090	['Big Smo', 'Upchurch']	Where You From
13441	['Red Hot Chili Peppers']	Higher Ground - Remastered
51779	['Bad Religion']	Generator
165151	['Rodney O', 'Joe Cooley']	You Don't Hear Me Tho' - Radio Mix
84225	['America']	All Around

Fig:3.15

3.4 MODEL 2:

CONTENT BASED RECOMMENDATION SYSTEM USING COSINE SIMILARITY

Content based Recommadation System

```
In [133]: ### Selection the features with song id as index
data['id']=data.index
df = data[['danceability','energy',"valence","speechiness","instrumentalness","acousticness"]]
df.index = data['id']

# normalized  data by columns
df_normalized = pd.DataFrame(normalize(df, axis=1))
df_normalized.columns = df.columns
df_normalized.index = df.index
df_normalized.head()
```

Out[133]:

id	danceability	energy	valence	speechiness	instrumentalness	acousticness
0	0.204439	0.154611	0.043526	0.026819	0.643359	0.719566
1	0.526206	0.219092	0.618725	0.266637	0.000000	0.470308
2	0.238274	0.120590	0.028622	0.024627	0.663245	0.698114
3	0.258165	0.290084	0.154899	0.033233	0.000026	0.907802
4	0.382654	0.176680	0.231606	0.034787	0.000002	0.876076

Fig: 3.16

```
In [134]: ### creking new dataframe with id and name
          df1 = data[["id","name"]]
```

```
In [135]: df1
```

Out[135]:

	id	name
0	0	Piano Concerto No. 3 in D Minor, Op. 30: III. ...
1	1	Clancy Lowered the Boom
2	2	Gati Bali
3	3	Danny Boy
4	4	When Irish Eyes Are Smiling
...	...	...
170648	170648	China
170649	170649	Halloweenie III: Seven Days
170650	170650	AYA
170651	170651	Darkness
170652	170652	Billetes Azules (with J Balvin)

170653 rows × 2 columns

Fig:3.17

```
In [139]: ### function for recommend songs to the users

def recommend(song_name):
    matching_rows = df1[df1['name'] == song_name.title()]

    if len(matching_rows) > 0:
        id= matching_rows['id'].iloc[0]
        distance_method = cosine
        allSongs = pd.DataFrame(df_normalized.index)
        allSongs = allSongs[allSongs.id != id]
        allSongs["distance"] = allSongs["id"].apply(lambda x: distance_method(df_normalized.loc[id], df_normalized.loc[x]))
        # sort by distance then recipe id, the smaller value of recipe id will be picked
        TopNRecommendation = allSongs.sort_values(["distance"]).head(5).sort_values(by=['distance', 'id'])

        Recommendation = pd.merge(TopNRecommendation , data, how='inner', on='id')
        SongName = Recommendation['name']
        for i in SongName:
            print(SongName)
            break

    else:
        return None
```

Fig:3.18

OUTPUT MODEL 2:

```
In [141]: recommend("Pause Track")
```

```
0    Piano Concerto No. 3 in D Minor, Op. 30: III. ...
1                                You Can't Love 'Em All
2    Dixie Breakdown - Recorded at the Mecca, Los A...
3                           Charu Theme - Instrumental
4                         Mama Don't Whip Little Buford
Name: name, dtype: object
```

Fig:3.19

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 CONCLUSION

This project used a content-based and collabrative filtering based recommendation system to recommend songs to users based on their past behavior and the similarities between items. The steps for creating such a system include importing libraries, reading data, dropping not useful columns, performing EDA, normalizing the data, applying the K means clustering algorithm, selecting features with song id as an index for building a content-based recommendation system, creating a new dataframe with id and name for building a content based recommendation system, and creating a function that recommends songs to users based on their past behavior on the website and the similarities between items. Once the project is complete, it can be used to provide personalized recommendations to users based on their past behavior and prefrences This can help users discover new songs that they might not have found otherwise.

4.2 THE FUTURE SCOPE

This model can improve the accuracy of the recommendations by using a hybrid recommendation system that combines content-based and collaborative filtering techniques. This can help to overcome the limitations of each technique and provide more accurate recommendations to users. Another future scope can be to use deep learning techniques such as neural networks to build a recommendation system that can learn from user behavior and provide even more personalized recommendations. Additionally, the model can be extended to include other features such as genre, artist, and album information to provide more comprehensive recommendation

REFERENCES

Music Recommendation Systems: A Survey by Mariusz Kleć & Alicja Wieczorkowska Part of the Studies in Computational Intelligence book series (SCI,volume 946)
https://link.springer.com/chapter/10.1007/978-3-030-66450-3_7
Music Recommendation Systems: Techniques, Use Cases, and Challenges by Markus Schedl, Peter Knees, Brian McFee & Dmitry Bogdanov
https://link.springer.com/chapter/10.1007/978-1-0716-2197-4_24
Content-Based Recommendation Systems by Michael J. Pazzani & Daniel Billsus
https://link.springer.com/chapter/10.1007/978-3-540-72079-9_10
A collaborative filtering method for music recommendation using playing coefficients for artists and users by Author links open overlay panelDiego Sánchez-Moreno, Ana B. Gil González, M. Dolores Muñoz Vicente, Vivian F. López Batista, María N. Moreno García
https://www.sciencedirect.com/science/article/abs/pii/S0957417416304973
Music Recommendation System Using Machine Learning by Varsha Verma Ninad Marathe Parth Sanghavi Dr. Prashant Nitnaware
https://www.researchgate.net/publication/357600972_Music_Recommendation_System_Using_M

1. achine_Learning