**PROJECT REPORT**

On

**Music Recommendation System**

Submitted in partial fulfillment of the requirements

for the award of degree

**MASTER OF COMPUTER APPLICATIONS**

Of

**KLE TECHNOLOGICAL UNIVERSITY**

By

**Ms. Prarthana Charankar** SRN: 01FM19MCA024

**Ms. Kavya Goudar** SRN: 01FM19MCA025

**Under the guidance of**

|  |
| --- |
| Prof. Shivanand Seeri,  Prof. Ashok Chikaraddi. |



**DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS**

**KLE TECHNOLOGICAL UNIVERSITY**

**Vidyanagar, Hubballi-580031 Karnataka.**

**2021- 2022**

**DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS**

**KLE TECHNOLOGICAL UNIVERSITY**



**CERTIFICATE**

This is to certify that the project work entitled

**“Music Recommendation System”**

Submitted in partial fulfillment of the requirements

for the award of degree of

**Master of Computer Applications**

**Of**

**KLE Technological University, Hubballi, Karnataka**

is a result of the bonafide work carried out by

**Ms. Prarthana Charankar** SRN: 01FM19MCA024

**Ms. Kavya Goudar** SRN: 01FM19MCA025

**Academic year 2021-2022**

|  |  |  |  |
| --- | --- | --- | --- |
| **Prof. Shivanand Seeri,**  **Prof. Ashok Chikaraddi.** | **Dr. P. R. Patil M.E, PhD** | | **Prof. N. H. Ayachit** |
| Project Guides,  MCA Dept | Head of Department,  Professor,  MCA Dept. | | Registrar,  KLE Technological University |
|  | ***Viva-Voce Examination*** | |  |
| Name of the Examiners |  | | Signature with Date |
| 1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | :\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| 1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |  | :\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**ACKNOWLEDGEMENT**

Every successful completion of any undertaking would be complete only after we remember and thank the almighty, the parents, the teachers, and the personalities, who directly or indirectly helped and guided during the execution of that work. The success of this work is equally attributed to all well-wishers who have encouraged and guided throughout the execution.

We express our deepest gratitude to **Prof. Shivanand Seeri and Prof. Ashok Chikaraddi** for their guidance and assistance throughout the project with great interest.

We avail this opportunity to express deepest gratitude to **Dr. P. R. Patil, HOD** for encouraging us with his valuable guidelines.

We are grateful to **Prof. N. H. Ayachit,** Registrar for his blessings and **KLE Technological University, Hubballi** that has given us a bright future. We would like to thank all the people for their guidance and valuable suggestions throughout the project.

We also thank all teaching and non-teaching staff members of the MCA department for their invaluable cooperation.

Finally, we would like to express our sincere thanks to our **Parents and Friends** for their enormous encouragement and all other who have extended their helping hands towards completion of our project.

**Ms. Prarthana Charankar**

**Ms. Kavya Goudar**

**ABSTRACT**

*Rapid development of mobile devices and internet has made possible for us to access different music resources freely. While the Music industry may favor certain types of music more than others, it is important to understand that there isn’t a single human culture on earth that has existed without music. In this project, we have designed, implemented and analyzed a song recommendation system. We have used Song Dataset provided to find correlations between users and songs and to learn from the previous listening history of users to provide recommendations for songs which users would prefer to listen most. The dataset contains over thousands of songs and listeners are recommended the best available songs based on the genre, artist and popular songs. With an interactive UI we show the listener the top songs that were played the most and top charts of the year. Listener also have the option to select his/her favorite songs and released year on which songs are recommended to them using the dataset.*

**MUSIC RECOMMENDATION SYSTEM**

**INDEX**

**TABLE OF CONTENTS**

1. **INTRODUCTION**……………………………………………………………………1
   1. Project Overview
   2. Motivation

1.3 Problem Definition

1.4 Objectives of the project

1. **PROPOSED SYSTEM** ………………………………………………………………2

2.1 Proposed System.

2.2 Target Users

2.3 Advantages of Proposed System

2.4 Scope

1. **SOFTWARE REQUIREMENT SPECIFICATION**………………………………3-7

3.1 Overview of SRS

3.2 Requirement Specifications

3.2.1 Functional Requirements

3.2.2 Use case diagrams

3.2.3 Use Case descriptions using scenarios

3.3 Software and Hardware requirement specifications

3.4 GUI of proposed system

1. **SYSTEM DESIGN** …………………………………………………………………..8

4.1 Block Diagram

1. **IMPLEMENTATION** ………………………………………………………………9-13
   1. Proposed Methodology
   2. Modules
      1. Recommendation Module
2. **TESTING**……………………………………………………………………………14
3. **RESULTS & DISCUSSIONS** ……………………………………………………...15-17
4. **CONCLUSION AND FUTURE SCOPE**…………………………………………..18
5. **REFERENCES / BIBILIOGRAPHY**……………………………………………...18

**Chapter 1 - INTRODUCTION**

* 1. **Project Overview**

The “Music Recommendation System” has been developed to recommend the songs based on data that is taken from the user. By the data a recommendation system is created, which is then used to suggest music to the user. Recommendation is an important field strongly related to web business which has been intensely researched in the past years, since electronic commerce web sites started their activity.

* 1. **Motivation**

The continue increasing in connection speed and trends in web development technologies, given rise to large web systems nowadays visited daily. Among these advanced systems, there are systems that allow users to listen music online without the need of downloading it to their personal computer. This issue solves a big problem originated by peer-to-peer software. There is a wide variety of these systems and new alternatives are constantly emerging increasingly improved. Some are simple players providing playlist functionality, others accompany the player with a recommendation system of similar artists (www.spotify.com), also there are complex collaborative systems in which hundreds of people leave comments on songs.

* 1. **Problem Definition**

To design and develop a machine learning project for music recommendation system where the users can get suggestions from the recommendation system to listen to the recommended songs.

**1.4 Objectives of the Project**

1. The goal of a recommendation system would be to provide personalized content by rightly identifying what the user wants.

2.This means that computers will have to think the way humans do; analyzing each of the user’s last choices to predict what they would like in the future.

3.As we have already seen, whenever there is a need for a machine to mimic human behavior, we have to utilize Machine Learning techniques.

4.Therefore, in this project, we will be relying on Machine Learning and Neural Networking techniques to build the ultimate music recommendation system.

**Chapter 2 - PROPOSED SYSTEM**

**2.1 Proposed System**

Recommendation System has been a current trend in today’s stream. The purpose of this project is to give Music Recommendation System. This machine allows the users to get recommendation of songs from the system. The songs are recommended on Spotify by classifying the songs by genre and by year. This is a content-based recommendation system where the songs are recommended based on the data taken from user.

**2.2 Target Users**

The users get the recommended songs on spotify where the music is classified by genre and by year. This is a content-based recommendation system where the songs are recommended based on the data taken from user. The recommendation system takes the music dataset from the Spotify Developers App.

**2.3 Advantages of the proposed system**

1. Provide a way to listen to recommended songs.
2. It is user friendly.
3. The user can easily get recommendations based on their most listened music genres.

**2.4 Scope**

“Music Recommendation System”, a content-based recommendation system is a customized system where we can easily get recommended songs. This system also classifies the music based on genres for easy recommendation. Recommendation is about extending listeners music universe beyond what they know and like. It empowers listeners once they have exhausted all their songs/artists search capabilities with further navigation celerity.

**Chapter 3 - SOFTWARE REQUIREMENT SPECIFICATON**

**3.1 Overview of SRS**

The goal is to design a recommender system on music domain such as Spotify. The Spotify dataset is used to manage the algorithms in order to produce recommendation results. Here the clustering is done using K-means algorithms and is visualized graphically. Generally, a music recommender system consists of three key components: users, items and user-item matching algorithms. So, the background of our product is becoming more important than foreground component. At the background, the system will gather data from both users and items. Using the data grouped by year, we can understand how the overall sound of music has changed over decades.

**3.2 Requirement Specification**

**3.2.1 Functional Requirement**

This application gathers the information from users, investigates some actions of the users, and provides the connection with the dataset. This application is the content-based Music Recommender, so it does not include the functionalities of the host music environment such as playing music etc.

* Requesting recommendations: Content based Recommender must allow user to request recommendations manually, and interact with the dataset to receive recommendations.

Input: User’s songs.

Process: Finding similar songs in dataset.

Result: Request recommendations accordingly.

* Evaluating songs: It must be able to evaluate songs and send appropriate information to the requested user.

Input: User’s songs.

Process: Classifies the songs based on genres.

Result: Evaluates songs accordingly.

* Display recommendations: The application must display the recommendations that are obtained from the data points of the songs a user has listened to and recommending songs corresponding to nearby data points.

Input: User’s songs.

Process: Classifies the songs based on genres.

Result: Display recommendations accordingly.

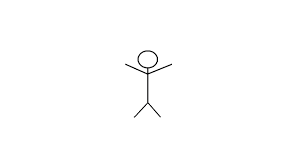
* Recommend using content-based filtering: The recommender system shall be capable of producing recommendations by interpreting the content and evaluations by actual user.

Input: Spotify Dataset.

Process: Content based filtering on dataset.

Result: Produces Recommendations accordingly.

**3.2.2 Use case**



User

**3.2.3 Use case descriptions using scenarios**

**User**

* User initially generates data by listening to the songs.
* User will classify the songs based on his/her interested genres.
* Later the songs are classified by year.
* Based on the types of songs he/she has listened to, the user will get the recommendations.
* Recommended results are of similar genres to the users have most listened to.

**3.3 Software and Hardware Requirement Specifications**

**Hardware Interface:**

The minimum hardware requirements for this application are as follows:

* Processor Intel-Core I3 or above
* 8GB RAM
* 30 GB free space

**Software Interface:**

The development of the application requires the following software’s:

* Windows XP or above
* Google Colab or Jupyter Notebook.

**3.4 GUI of proposed System**

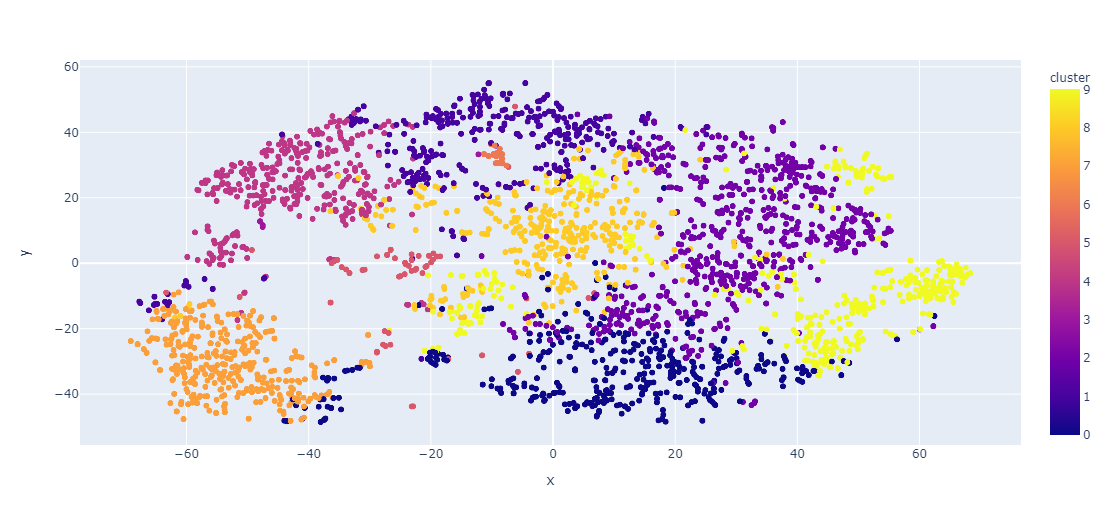
a) Clustering Genres

Fig 3.4.1

b) Recommending music

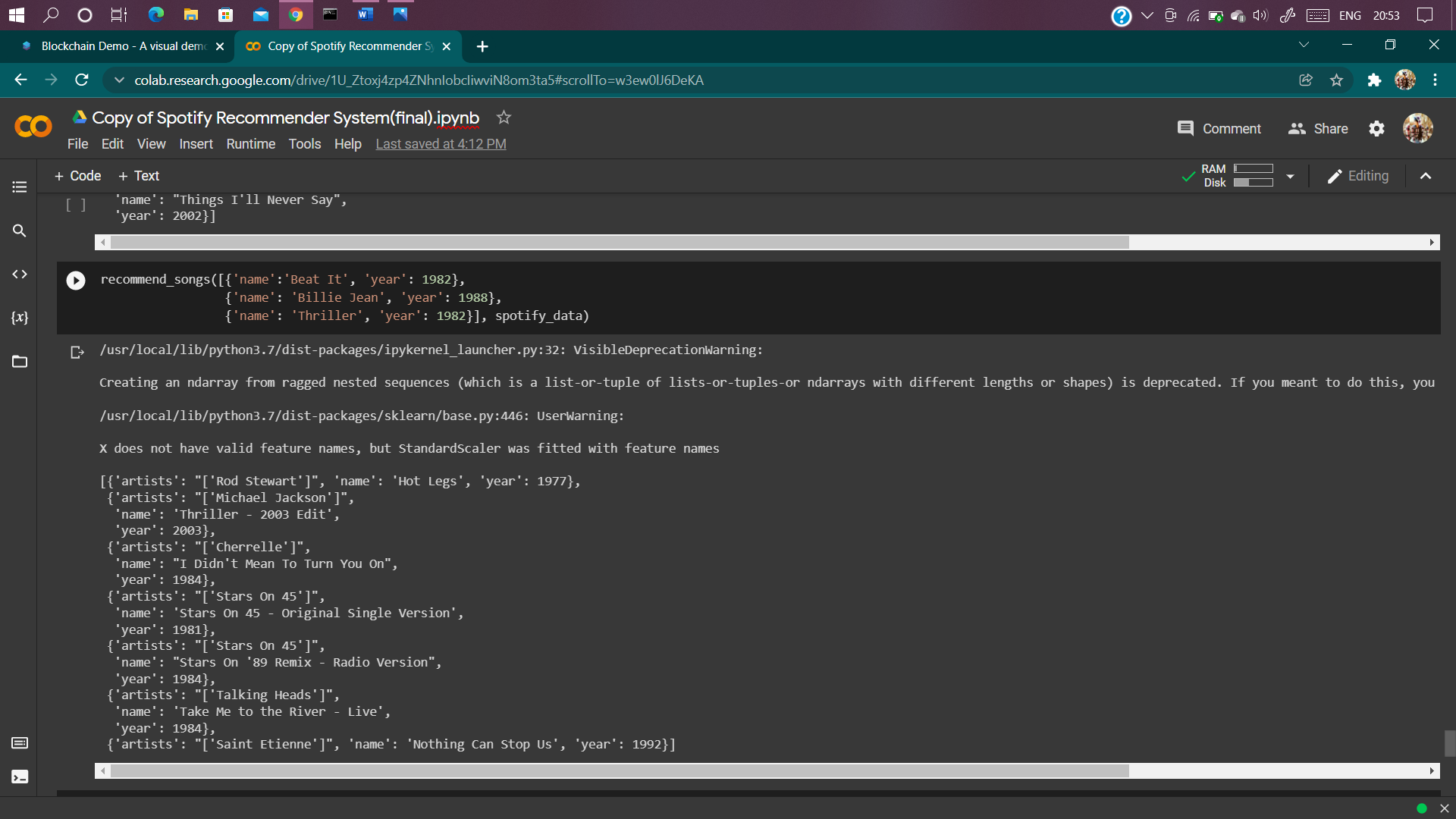


Fig 3.4.2

c) Year And Tempo



Fig 3.4.3

d) Exploratory Data Analysis

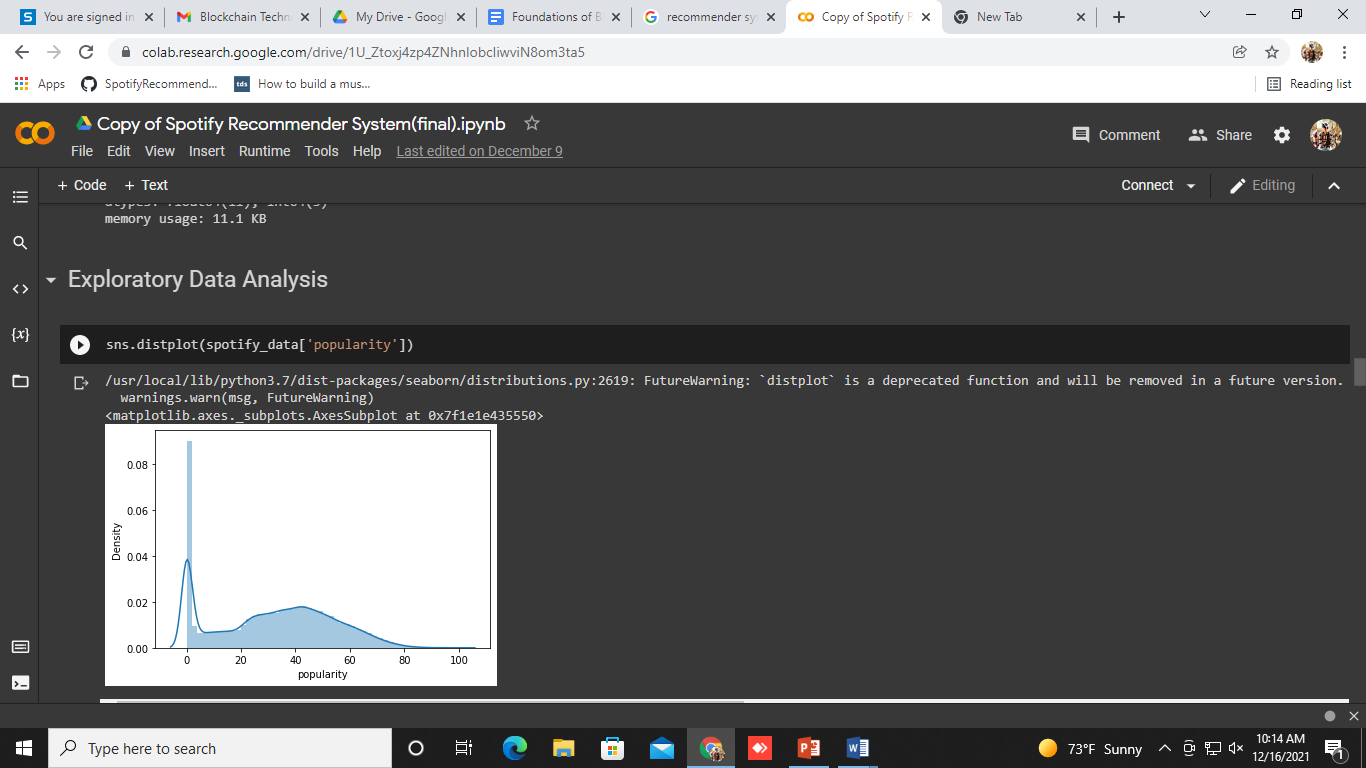


Fig 3.4.4

**Chapter 4 - SYSTEM DESIGN**

**4.1 Block Diagram**

Fig 4.1.1 describes the block diagram of Music Recommendation System. Initially the Music dataset is taken into consideration for the songs clustering and music classification. Here the music clustering includes the music classification system and the songs which are commonly listened by the users. The similar music listeners along with content based filtering helps to build a Music Recommendation System providing the recommended results.

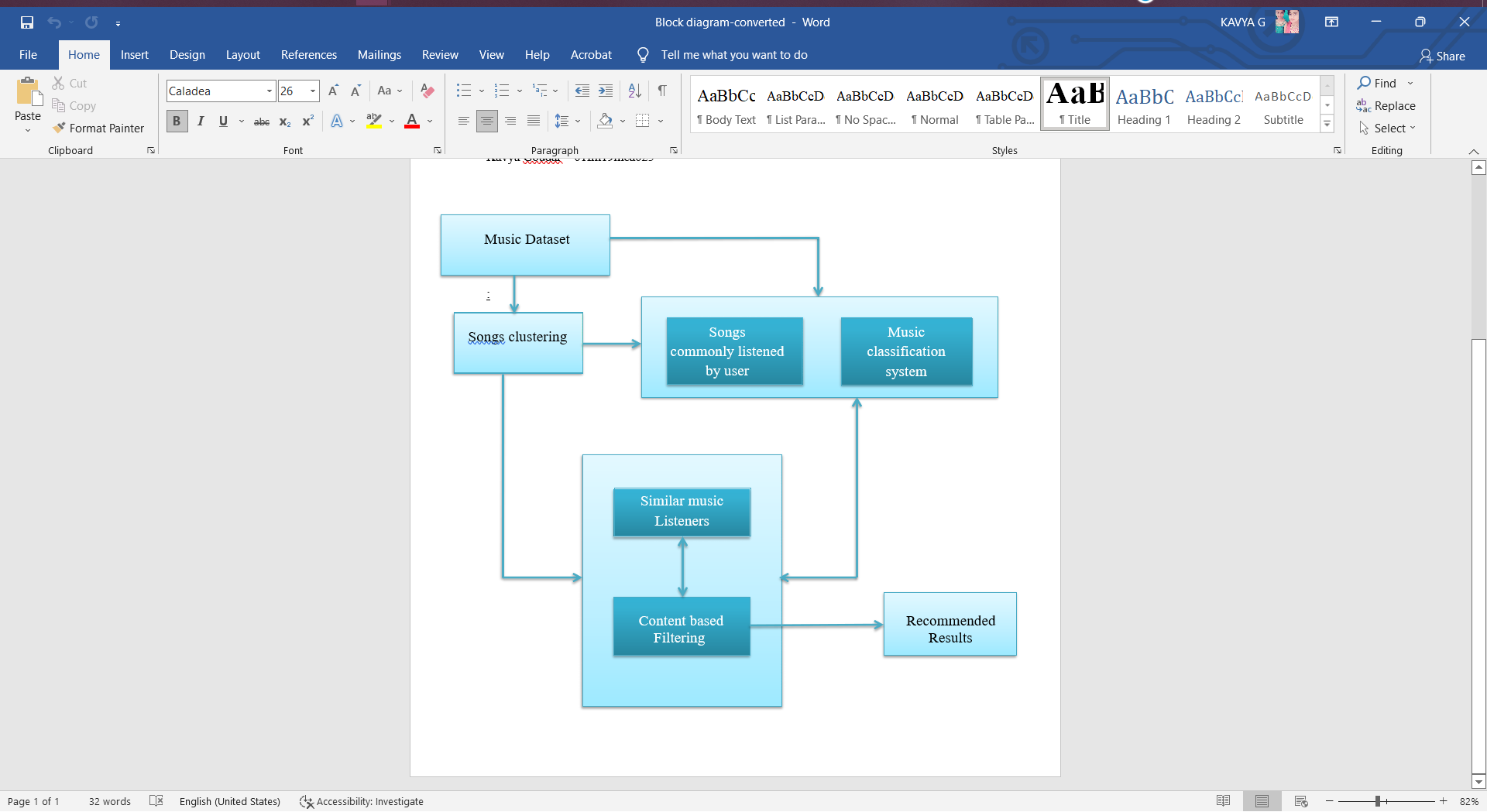


Fig 4.1.1

**Chapter 5 - IMPLEMENTATION**

**5.1 Proposed Methodology**

Recommendation system depends on data come from websites users and companies items. This program has different type of users, so there is functionality differences between users will occur with respect to item data. The recommendation systems used by companies have differences for efficiency. Our recommendation system should work efficiently according to music data. Once the recommendation system finds an accurate result, it will be shown on the interface. In terms of hardware, recommendation system will be embedded in a website. To use or benefit from recommendation system, user should enter from a personal computer, mobile device with internet connection, tablet etc. In terms of software, our recommendation system will run on personal computers, smart phones etc. That is, it will run on any device with internet connection. The system will work both Windows and Unix operating systems.

**Clustering Genres with K-Means**

K-Means Clustering Algorithm:

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

The k-means clustering algorithm mainly performs two tasks:

* Determines the best value for K center points or centroids by an iterative process.
* Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

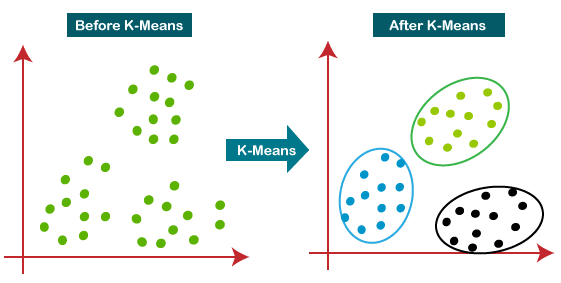


Fig 5.1.1

**Generating Song Recommendations**

The recommendation algorithm used is pretty simple and follows three steps:

1. **Compute the average vector of the audio and metadata features for each song the user has listened to.**
2. **Find the n-closest data points in the dataset (excluding the points from the songs in the user’s listening history) to this average vector.**
3. **Take these n points and recommend the songs corresponding to them**

This algorithm follows a common approach that is used in content-based recommender systems and is generalizable because we can mathematically define the term closest with a wide range of distance metrics ranging from the classic Euclidean distance to the cosine distance. For the purpose of this project, I used the cosine distance, which is defined below for two vectors u and v.

**Cosine distance formula.**

In other words, the cosine distance is one minus the cosine similarity — the cosine of the angle between the two vectors. The cosine distance is commonly used in recommender systems and can work well even when the vectors being used have different magnitudes. If the vectors for two songs are parallel, the angle between them will be zero, meaning the cosine distance between them will also be zero because the cosine of zero is one.

How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

**Step-1:** Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.

**Step-4:** Calculate the variance and place a new centroid of each cluster.

**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

**Step-7**: The model is ready.

* 1. **Modules Description:**

Recommendation Module:

Recommendation module generates recommendation based on the user profile. It analyses the previous listening history and preferences of a user and provides a list of songs that user might prefer to listen. We have used a global popularity model, Content based model and collaborative filtering model.

**Source code:**

from spotipy.oauth2 import SpotifyClientCredentials

from collections import defaultdict

from spotipy.oauth2 import SpotifyOAuth

sp = spotipy.Spotify(auth\_manager=SpotifyClientCredentials(client\_id="cf0ed68e536848da870c979cdcf39ac7",

                                                        client\_secret="9e244b94ec3e4551a03be2a55278a4f4"))

def find\_song(name, year):

    song\_data = defaultdict()

    results = sp.search(q= 'track: {} year: {}'.format(name,

                                                       year), limit=1)

    if results['tracks']['items'] == []:

        return None

    results = results['tracks']['items'][0]

    track\_id = results['id']

    audio\_features = sp.audio\_features(track\_id)[0]

    song\_data['name'] = [name]

    song\_data['year'] = [year]

    song\_data['explicit'] = [int(results['explicit'])]

    song\_data['duration\_ms'] = [results['duration\_ms']]

    song\_data['popularity'] = [results['popularity']]

    for key, value in audio\_features.items():

        song\_data[key] = value

    return pd.DataFrame(song\_data)

from collections import defaultdict

from sklearn.metrics import euclidean\_distances

from scipy.spatial.distance import cdist

import difflib

number\_cols = ['valence', 'year', 'acousticness', 'danceability', 'duration\_ms', 'energy', 'explicit',

 'instrumentalness', 'key', 'liveness', 'loudness', 'mode', 'popularity', 'speechiness', 'tempo']

def get\_song\_data(song, spotify\_data):

    try:

        song\_data = spotify\_data[(spotify\_data['name'] == song['name'])

                                & (spotify\_data['year'] == song['year'])].iloc[0]

        return song\_data

    except IndexError:

 return find\_song(song['name'], song['year'])

def get\_mean\_vector(song\_list, spotify\_data):

    song\_vectors = []

    for song in song\_list:

        song\_data = get\_song\_data(song, spotify\_data)

        if song\_data is None:

            print('Warning: {} does not exist in Spotify or in database'.format(song['name']))

            continue

        song\_vector = song\_data[number\_cols].values

        song\_vectors.append(song\_vector)

    song\_matrix = np.array(list(song\_vectors))

    return np.mean(song\_matrix, axis=0)

def flatten\_dict\_list(dict\_list):

    flattened\_dict = defaultdict()

    for key in dict\_list[0].keys():

        flattened\_dict[key] = []

    for dictionary in dict\_list:

        for key, value in dictionary.items():

            flattened\_dict[key].append(value)

    return flattened\_dict

def recommend\_songs( song\_list, spotify\_data, n\_songs=10):

    metadata\_cols = ['name', 'year', 'artists']

    song\_dict = flatten\_dict\_list(song\_list)

    song\_center = get\_mean\_vector(song\_list, spotify\_data)

    scaler = song\_cluster\_pipeline.steps[0][1]

    scaled\_data = scaler.transform(spotify\_data[number\_cols])

    scaled\_song\_center = scaler.transform(song\_center.reshape(1, -1))

    distances = cdist(scaled\_song\_center, scaled\_data, 'cosine')

    index = list(np.argsort(distances)[:, :n\_songs][0])

    rec\_songs = spotify\_data.iloc[index]

    rec\_songs = rec\_songs[~rec\_songs['name'].isin(song\_dict['name'])]

    return rec\_songs[metadata\_cols].to\_dict(orient='records')

recommend\_songs([{'name': 'Come As You Are', 'year':1991},

                {'name': 'Smells Like Teen Spirit', 'year': 1991},

                {'name': 'Lithium', 'year': 1992},

                {'name': 'All Apologies', 'year': 1993},

                {'name': 'Stay Away', 'year': 1993}],  spotify\_data)

**Chapter 6 – TESTING**

Testing is process of executing a program with the intent of finding an error. A good test case is one that high probability of finding an as yet undiscovered error. A successful test is one that uncovers an as yet undiscovered error. Testing cannot show the absence of defect, it can only show that software errors are present.

**Test case:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl. No | Input | Expected Output | Actual Output | Status | Reference |
| 1 | Clustering Genres | It should display the clustered graph by genre. | https://lh4.googleusercontent.com/37dp57w7mtT4u1D_qA8bGy3kp9SGGoAw5uz8IsbTD6M25X05fxx8cEOJucsYavHx3ZJPk8F1pe-7ZWTYgTyiPsuL4Ju908_NZnIaLzRRHBN1sayVGX0YnjzEqGInJI89u0lycYCU | Passed | Fig 7.3 |
| 2 | Clustering Songs | It should display the clustered graph by songs. | https://lh6.googleusercontent.com/FqFIX89eBv7ZUSAzFocx_7-1GL66NN7ObvYykOl1j2UWSAIZ8OIgN7j8Viuhd6K5t0AlJfgF4z-h9hDED8_8k9wTXcumsgmzW2c0fd2fHtrynVK8137Q8Cq-Yt1ZZSsKr0mHeVNQ | Passed | Fig 7.4 |
| 3 | Recommend songs of similar genres. | It should display the list of recommended songs of similar genres. | It will display the list of recommended songs of similar genres. | Passed | Fig 7.5 |
| 4 | Recommend songs. | It should display the list of recommended songs. | It will display the list of recommended songs. | Passed | Fig 7.5 |

**Chapter 7 – RESULTS AND DISCUSSIONS**

1. In order to build a music recommendation system, we used the [Spotify Dataset](https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks), which is publicly available on [Kaggle](https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks) and contains metadata and audio features for over 170,000 different songs.

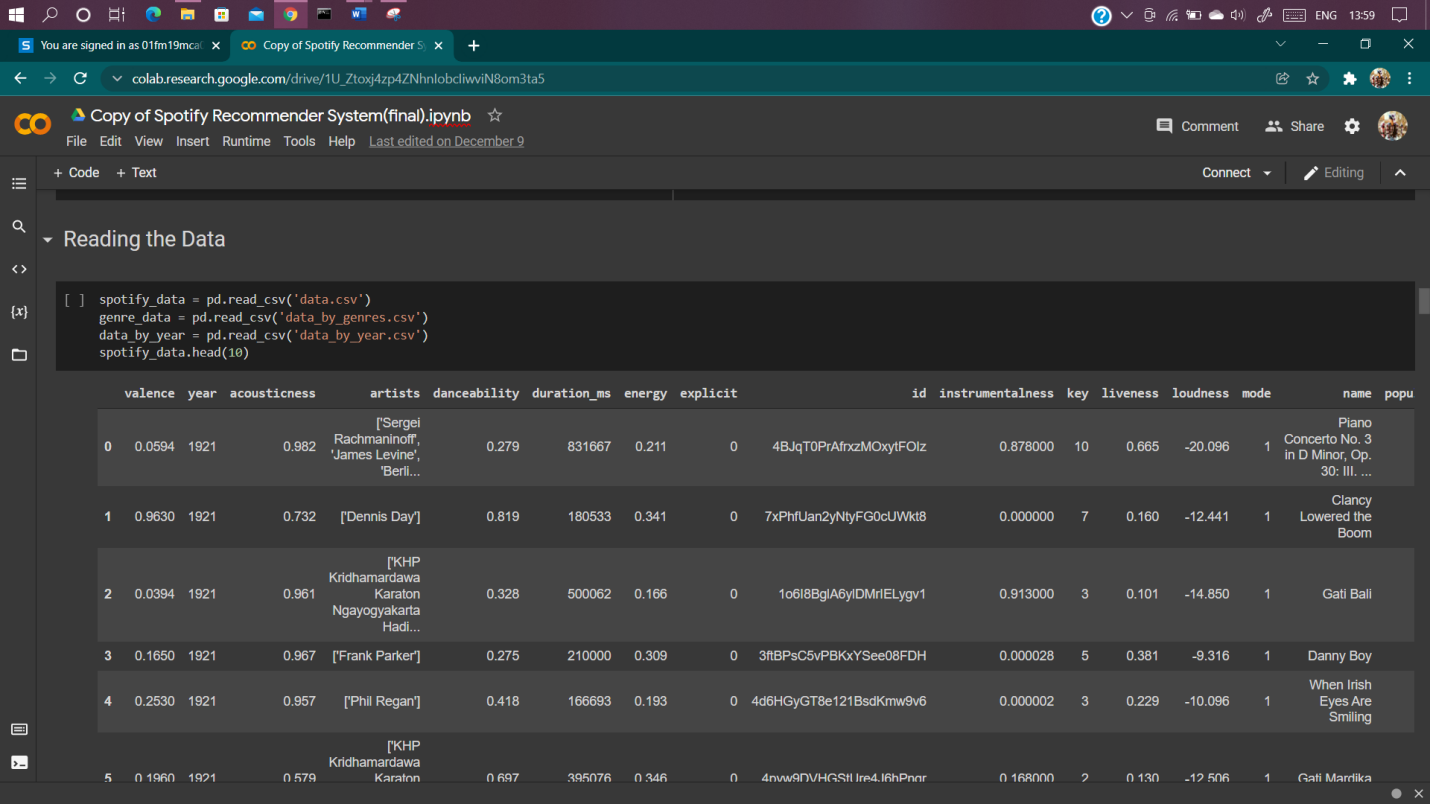


Fig 7.1

2. Using the data grouped by year, we can understand how the overall sound of music has changed from 1921 to 2020.

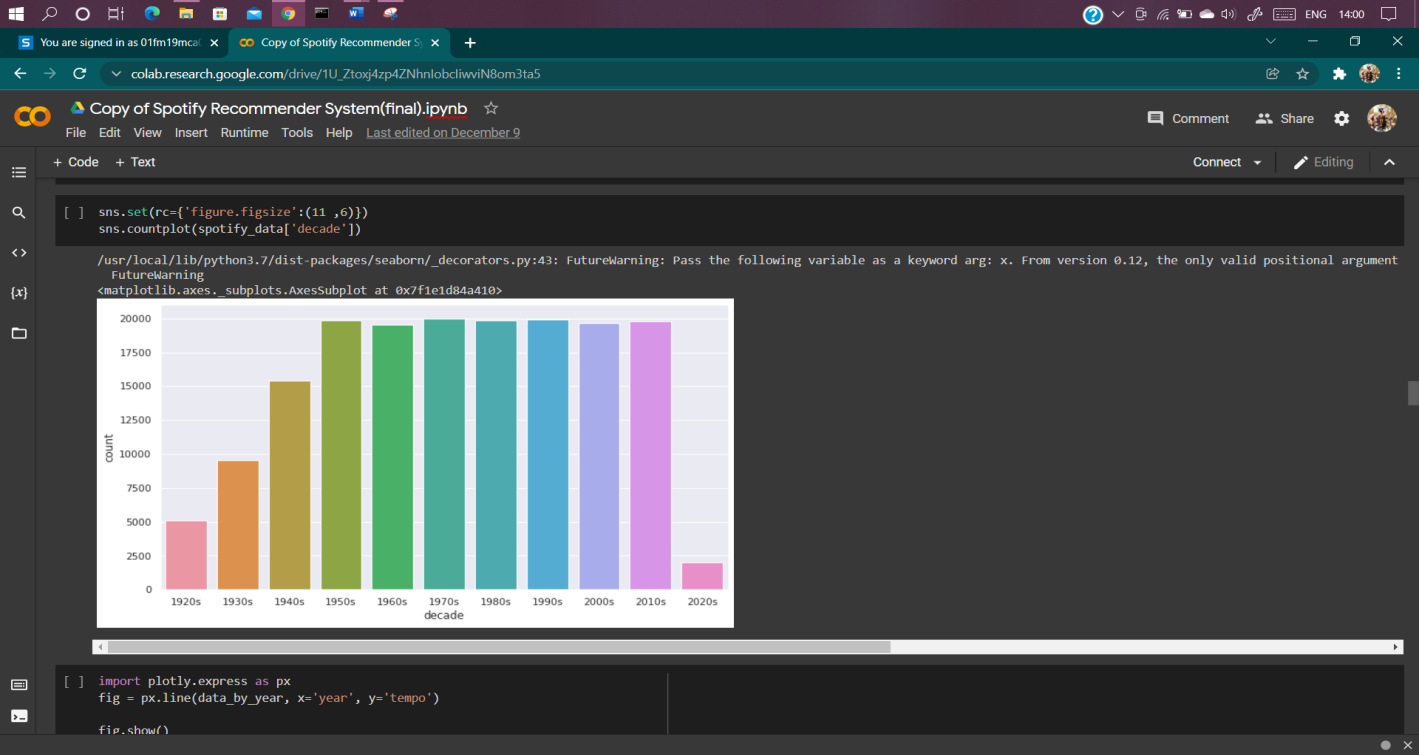


Fig 7.2

3. There are many audio features for each genre and it is difficult to visualize clusters in a high-dimensional space. However, we can use a dimensionality reduction technique known as [t-Distributed Stochastic Neighbor Embedding](https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf) to compress the data into a two-dimensional space.

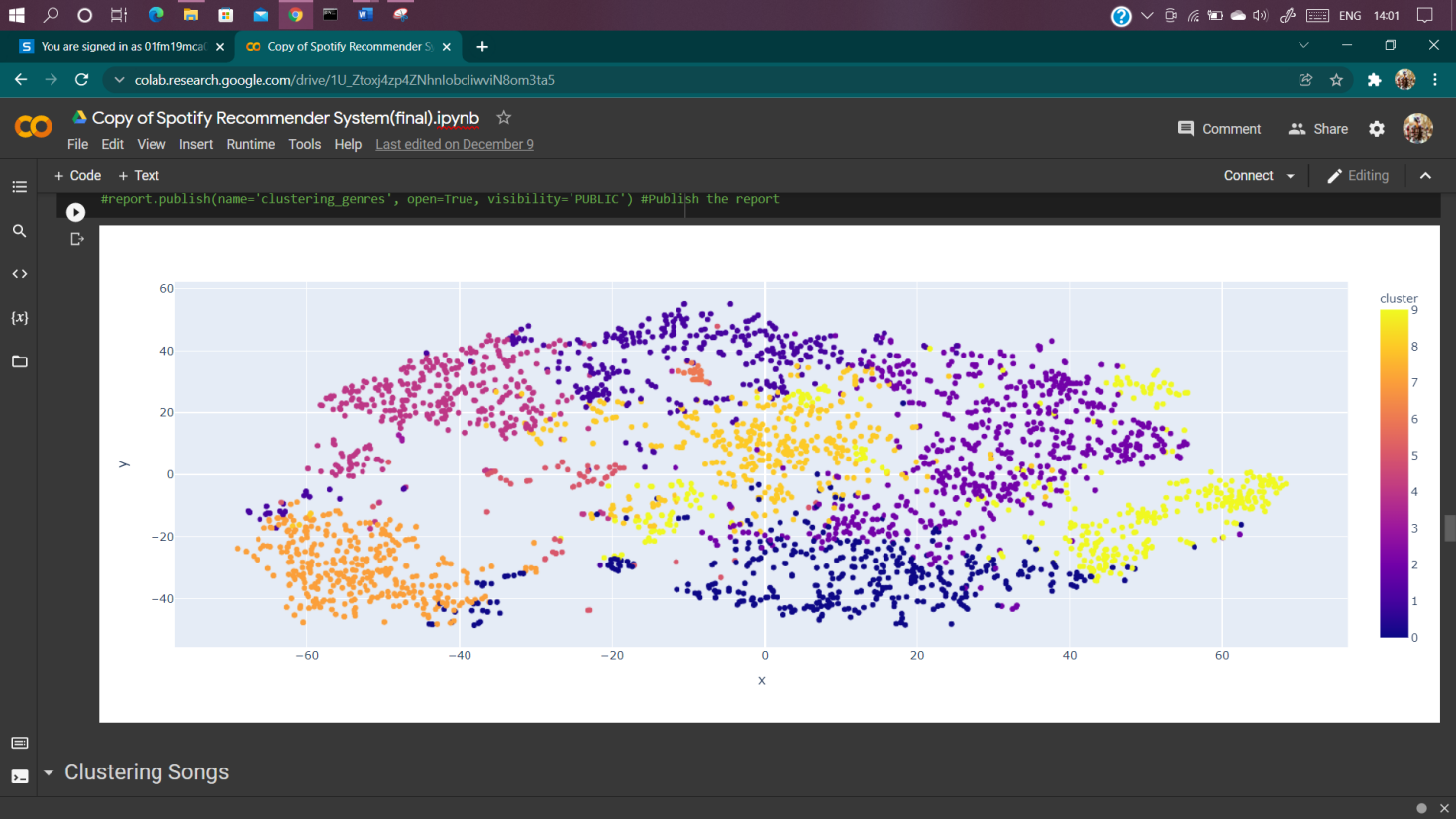


Fig 7.3

4.Visualizes the genre clusters in a two-dimensional coordinate plane by using Plotly’s scatter function.

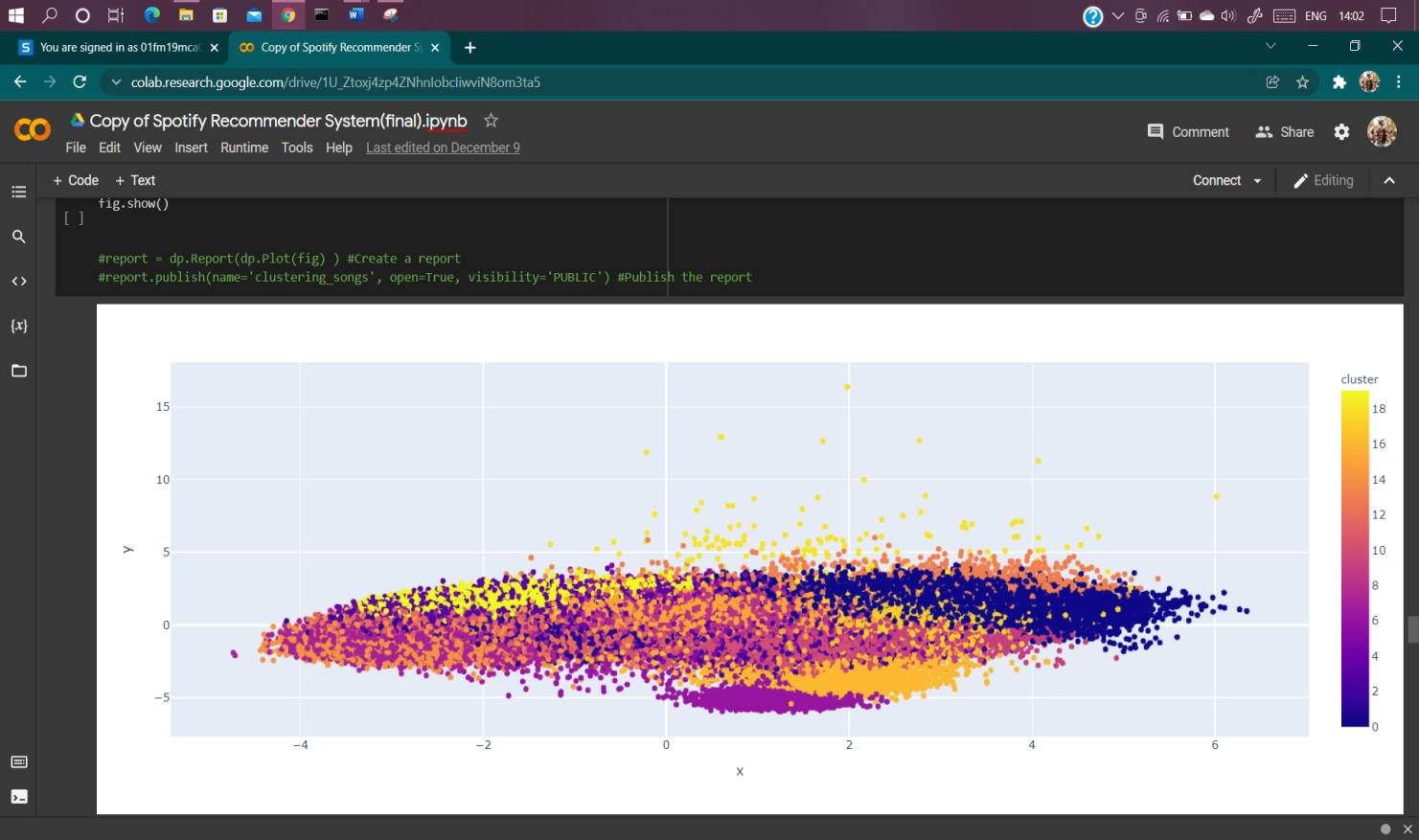


Fig 7.4

5.The **find\_song** function that we defined below fetches the data for any song from Spotify’s catalog given the song’s name and release year. The results are returned as a Pandas Dataframe with the data fields present in the original dataset that we downloaded from [Kaggle](https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks).

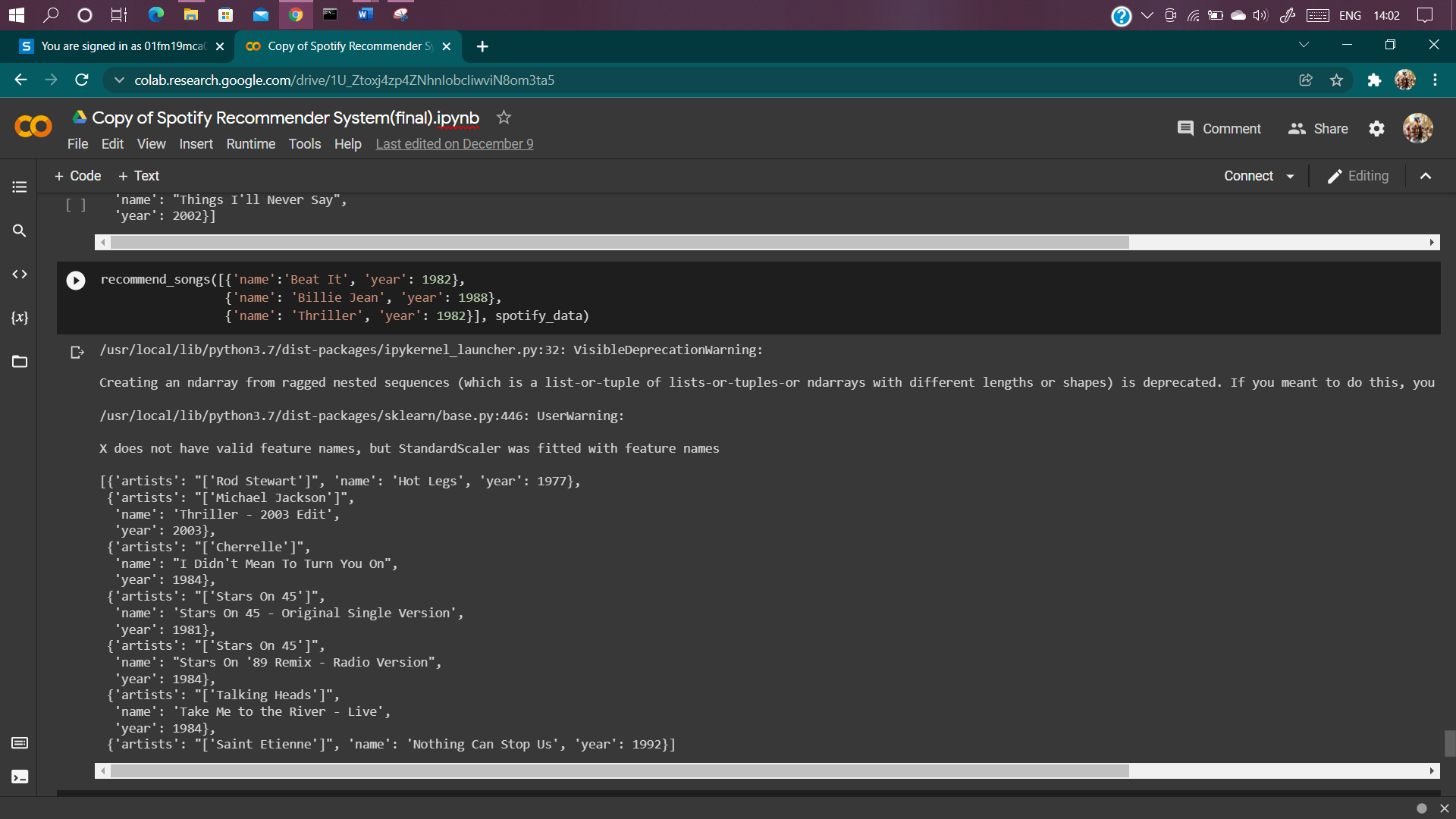


Fig 7.5

**CONCLUSION AND FUTURE SCOPE**

Customer relationship management is focused on the creation and maintenance of

long-term, mutually beneficial relationships with strategically important markets.

Customer relationship management is focused on the creation and maintenance of

long-term, mutually beneficial relationships with strategically important markets.

Customer relationship management is focused on the creation and maintenance of

long-term, mutually beneficial relationships with strategically important markets.

Customer relationship management is focused on the creation and maintenance of

long-term, mutually beneficial relationships with strategically important markets. It is

based on the premise that customers with the highest life-time value potential are

those in whom the company should invest their retention resources. Other customers

might be fired. For others, it may be possible to re-engineer or nurture the

relationship to create new sources of value.

Customer relationship management is focused on the creation and maintenance of

long-term, mutually beneficial relationships with strategically important markets

Software is said to have attained its objective only when it meet all requirements of the user, further the user himself is the person to judge the success of the system. Every attempt has been made to ensure that the system is fully functional & works effectively & efficiently. The system has been tested with simple data to cover all possible options & checked for al outputs. Since the system is flexible & modular, further modifications of this package can be easily incorporated.

* In the future, we will try to add a greater number of artists and languages which will make the recommendation stronger giving even better playlists for the users.
* We can try the system with other machine learning models as well to compare the results and look for better results.
* For future applications, an emotional detector system that will recommend the songs by recognizing our facial emotion can be developed.

**Reference/Bibliography**

1. https://www.ijert.org/music-recommendation-system-using-content-and-collaborative-filtering-methods
2. https://towardsdatascience.com/how-to-build-an-amazing-music-recommendation-system-4cce2719a572
3. https://senior.ceng.metu.edu.tr/2014/cenghistoryx/documents/CengHistoryX\_SRS.pdf
4. Balabanovic, M. and Shoham, Y. (1997). Fab: Content-based Collaborative Filtering Recommendation. *Communications of the ACM*, 40, 66–72.
5. Basu, C., Haym, H., and Cohen, W.W. (1998).” Recommendation as Classification: Using Social and Content-Based Information in Recommendation”. In *Proceedings of National Conference on Artificial Intelligence*, pp. 714–720.