Music Recommendation System - Based on Mood and Lyrics

Adithya Suresh babu Sai Sriram Ampapurapu Goutham Reddy Jarugu Sushanth Reddy Marthala

Department of Computer Science

University of Central Florida

ABSTRACT

The popularity of digital music is ever growing and is at an all time high these days. The listeners are craving for good music that suits their tastes and liking. Companies are trying to improve their recommendation systems to give users good recommendations and keep them glued to their services.

In this project, we have designed, implemented and analyzed a recommendation system. We used both Spotify and musixmatch datasets to find correlations between users and songs and to recommend new music to users based on the lyrics and mood or tone of the songs the users are listening to. In this report, we will discuss the problem methods we have implemented, results and their analysis.

I. INTRODUCTION

A. *BACKGROUND*

The rapid growth of mobile devices and the internet has enabled us to freely access many music resources. More music is available to a single person than they can possibly listen to. It can be challenging for people to select music they like from the millions of songs/albums available. Furthermore, music service providers require an efficient approach to organize songs and assist their customers in discovering music by providing quality recommendations. An effective recommendation system is thus highly necessary.

Many internet platforms and services, including music streaming services, now include recommendation systems as a critical component. A recommendation system can help consumers discover music that matches their tastes, making the user experience more engaging and personalized. Furthermore, these systems can help consumers discover new music, broaden their musical horizons, and uncover hidden gems..Many music streaming services, such as Pandora,Gaana and Spotify, are currently focused on developing high-precision commercial music recommendation systems. These

companies generate revenue by assisting their consumers in discovering suitable music and charging them for the quality of their suggestion service. As a result, there is a thriving market for good music recommendation systems.

B. *PROBLEM & IMPORTANCE*

The rapid growth of mobile devices and the internet has enabled us to freely access many music resources. More music is available to a single person than they can possibly listen to. It can be challenging for people to select music they like from the millions of songs/albums available. Furthermore, music service providers require an efficient approach to organize songs and assist their customers in discovering music by providing quality recommendations. An effective recommendation system is thus highly necessary.

Many internet platforms and services, including music streaming services, now include recommendation systems as a critical component. A recommendation system can help consumers discover music that matches their tastes, making the user experience more engaging and personalized. Furthermore, these systems can help consumers discover new music, broaden their musical horizons, and uncover hidden gems..Many music streaming services, such as Pandora,Gaana and Spotify, are currently focused on developing high-precision commercial music recommendation systems. These companies generate revenue by assisting their consumers in discovering suitable music and charging them for the quality of their suggestion service. As a result, there is a thriving market for good music recommendation systems.

C. *EXISTING LITERATURE*

As we know that there is a booming market for music recommendation systems, there is a lot of work done on this topic. The concept of using combined algorithms like Restricted Boltzmann Machines and Collaborative filtering won the Netflix prize for recommender system and spearheaded the development of recommendation systems[1]. There is research that uses Convolutional Neural Networks(CNN's) to recommend music to users based on listening history with some personal information[2]. This approach classifies the audio signal beats present in the music into different genres and also uses collaborative filters in tandem to recommend songs to the users. Much more research has been done on collaborative filtering[3] and Content based filtering[4] in the past.

D. *SYSTEM OVERVIEW*

The ML system has Three major components. Data Preparation,Data Preprocessing, Feature Engineering and Recommendation System.

E. *DATA COLLECTION*

We got the data from two major sources, kaggle and Musixmatchs' million song datasets. The data from Kaggle has around 18000 songs and lyrics and the musixmatch dataset has more than 300k songs data with lyrics in it. Million Song Dataset(MSD) comes in a bag of words format which does affect the confidence of our system but with the amount of data it provides, we would say that it is worth the effort to consider. The Dataset from kaggle has the metadata like energy, dancebility, speech etc.. which helps us find the sentiment and classify the mood of the songs.

F. *COMPONENTS OF ML SYSTEM*

The Machine Learning system we designed has 3 major parts in it. Data collection/preprocessing, Feature engineering, model evaluation and recommendation system. Data collection deals with data preparation, Feature engineering involves removing unwanted features and creating new meaningful features from existing data or combining multiple features together and Model evaluation involves comparing multiple models and picking the best one and recommendation system involves finding similarity between songs based on the features and outputting a list of similar songs to the input song.

II.IMPORTANT DEFINITIONS AND PROBLEM STATEMENTS

A. DATA DEFINITIONS

We are using the Spotify songs dataset and every song's audio features are given in the dataset.

The following audio features are used in this recommendation system:

- Valence: Valence is used to describe the emotions delivered by the track and we can say the track with high valence is positive and the track with low valence is negative.
- Tempo: It refers to the speed or pace of the track. In this dataset, the tempo is measured in terms of beats per minute.
- Speechiness: Speechiness refers to the presence of spoken words in the track recording. In the dataset, it is a metric whose value is close to 1.0 means the more-speech like it is.

- Loudness: The overall volume of a track is expressed in decibels. The normal level of a track is between -60 and 0 db.
- Liveness: It represents whether the track is recorded in a recording studio or performed live which means audience. If the track's liveness is greater then the track is performed in front of the audience.
- Instrumentalness: The amount of instrumental space on a track indicates how much vocal content there is. The less likely it is that there are vocals in the tape, the closer the value is to 1.0.
- Energy: A scale-based indicator of intensity and activity, the energy level spans from 0.0 to 1.0. The vast majority of upbeat songs are loud and have rapid tempos.
- Acousticness: It is a measure used to say whether the track is acoustic or not and it ranges from 0 to 1.
- Danceability: It is a feature that is used to say whether the track is danceable or not based on the pace, rhythm, and regularity of the track. If it is 0.0 means not danceable or danceable if it is 1.0.
- Track_artist: It indicates the artist who sang the track.

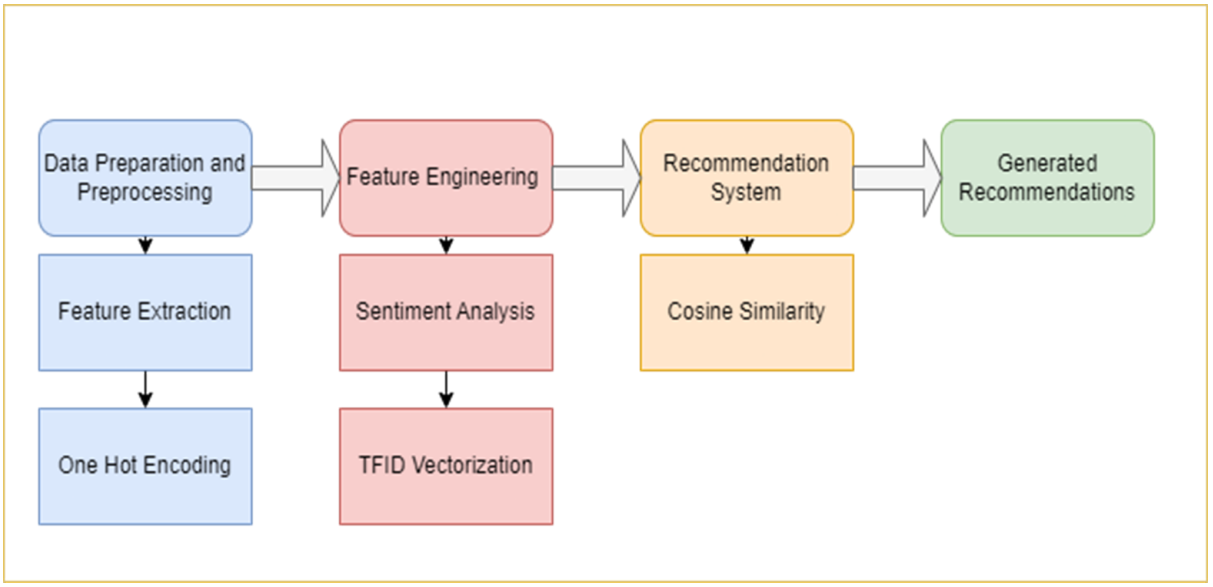
B. *PREDICTION TARGET*

For all models, with the exception of Navies Bayes and Random Forest, prediction targets would be classes like Positive, Negative, and Neutral, implying the mood Happy, Sad, and Neutral. The only possible outcomes for Navies Bayes and Random Forest are Positive and Negative.

C. PROBLEM STATEMENT

The existing music systems use only one aspect of the song to recommend other music. For example they may only listen to the audio and give a recommendation. Or they might just consider the lyrics to give suggestions. Here, we try to combine both these factors and try to come up with a solution to a good recommendation system.

III.OVERVIEW OF THE PROPOSED SYSTEM



Our proposed system comprises four stages:

- Data Preparation and Preprocessing
- Feature Engineering
- Recommendation System
- Generated Recommendations and Evaluation

A.DATA PREPARATION AND PREPROCESSING

In this stage, we collected data from Kaggle and data contains the following features track_id, track_name,track_artist, lyrics, track_popularity,track_album_id,track_album_name, track_album_release_date,playlist_name,playlist_id, playlist_genre, playlist_subgenre, danceability, energy, key, loudness, mode,speechiness,acousticness,instrumentalness, liveness, valence, tempo,duration_ms, language. For Data Preprocessing, we have removed punctuation, tokenization, and word lemmatizer. Finally created two new features called “final” and “final_genre_list” which is track_atrist and preprocessed lyrics combined(feature extraction) and genre and sub_genre combined feature.

B.FEATURE ENGINEERING

In this stage, We have created a new feature from scratch using the feature “lyrics” which is called “sentiment”. We have created this new feature using an existing library called “Sentiment IntensityAnalyzer”. We have done one-hot encoding to the new feature “sentiment” which is a data preprocessing technique and also done one-hot

encoding for subjectivity, key, and mode features using a user-defined function called "ohe_prep" and multiple each one-hot encoding like subjectivity with 0.3, key, mode and sentiment with 0.5. We have applied TF IDF vectorization to the "final_genre_list" feature to quantify its word. Finally created a new data frame called "complete_feature_list we have included all the data preprocessed data and one hot encoded data into it.

C. RECOMMENDATION SYSTEM

The following step is to carry out content-based filtering using the available song features. We combine all of the songs on a playlist into a single summary vector to achieve this. Then, we compare the summarized playlist vector to every song (excluding the songs in the playlist) in the database to see how similar they are. The most pertinent song that isn't in the playlist is then recommended using the similarity metric.

D. GENERATED RECOMMENDATIONS AND EVALUATION

In this stage, we will generate a recommendation for the user using the above-generated system. For evaluation.

IV.TECHNICAL DETAILS OF PROPOSED SYSTEM

Our proposed system comprises four stages:

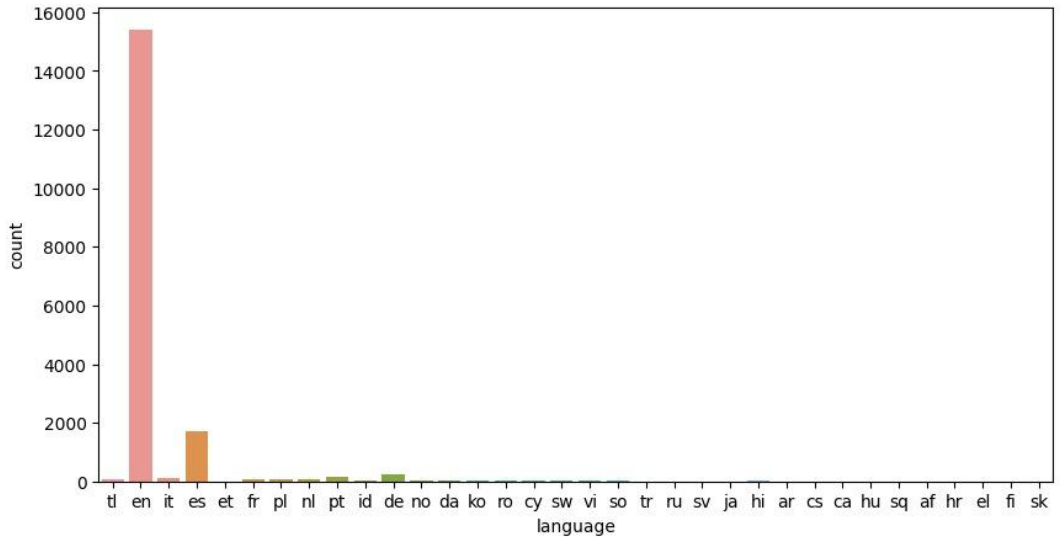
- Data Preparation and Preprocessing
- Feature Engineering
- Recommendation System
- Generated Recommendations and Evaluation

A.DATA PREPARATION AND PREPROCESSING

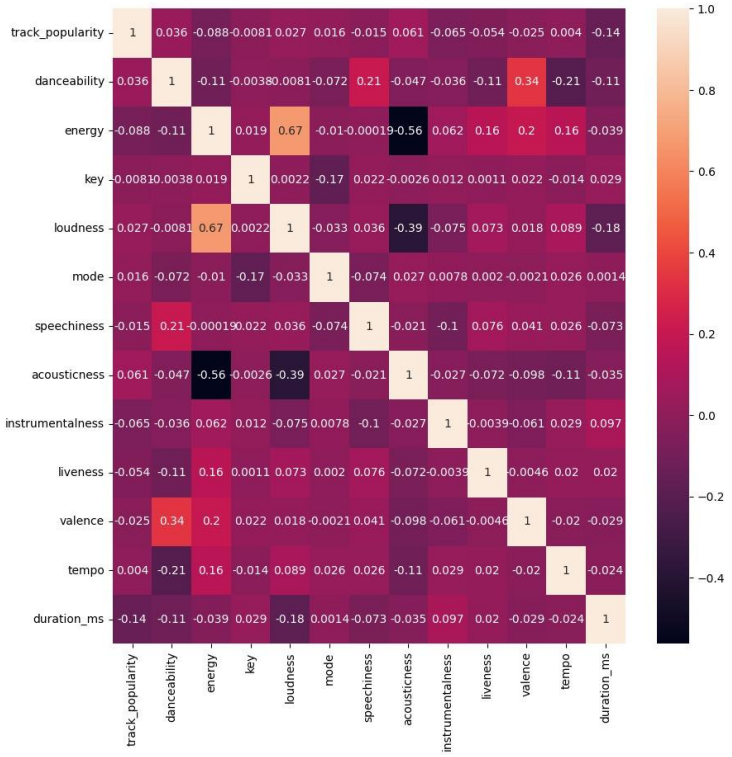
In this stage, we collected data from Kaggle and data contains the following features track_id, track_name,track_artist, lyrics, track_popularity,track_album_id,track_album_name, track_album_release_date,playlist_name,playlist_id, playlist_genre, playlist_subgenre, danceability, energy, key, loudness, mode,speechiness,acousticness,instrumentalness, liveness, valence, tempo,duration_ms, language.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18454 entries, 0 to 18453
Data columns (total 25 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_id                  18454 non-null  object
 1   track_name                18454 non-null  object
 2   track_artist              18454 non-null  object
 3   lyrics                    18194 non-null  object
 4   track_popularity          18454 non-null  int64
 5   track_album_id            18454 non-null  object
 6   track_album_name          18454 non-null  object
 7   track_album_release_date  18454 non-null  object
 8   playlist_name             18454 non-null  object
 9   playlist_id               18454 non-null  object
 10  playlist_genre            18454 non-null  object
 11  playlist_subgenre         18454 non-null  object
 12  danceability              18454 non-null  float64
 13  energy                    18454 non-null  float64
 14  key                       18454 non-null  int64
 15  loudness                  18454 non-null  float64
 16  mode                      18454 non-null  int64
 17  speechiness               18454 non-null  float64
 18  acousticness              18454 non-null  float64
 19  instrumentalness          18454 non-null  float64
 20  liveness                  18454 non-null  float64
 21  valence                   18454 non-null  float64
 22  tempo                     18454 non-null  float64
 23  duration_ms               18454 non-null  int64
 24  language                  18194 non-null  object
dtypes: float64(9), int64(4), object(12)
memory usage: 3.5+ MB
```

We have removed some songs based on the language of the songs. We plotted a bar graph using the languages of the track and we got to know that the majority of the tracks are in English so we have removed all other languages tracks except English tracks.



We wanted to know which features present in the dataset can be useful to the recommendation system so we plotted a heatmap for all the features of the dataset and got to know which features are important for the recommendation system and used those features for the system.



For Data Preprocessing, we have done

Removal of Punctuation:

Since the dataset contains lyrics in which there can be punctuation marks like “, . ;” we have to remove punctuation because the machine learning model does not need the punctuation to understand the lyrics and it is also unnecessary for the model.

Tokenization:

Since the dataset contains lyrics that contain long sentences which cannot be understood by the recommendation, we are going to break the lyrics into recommendation system understandable data. For that purpose, we are going to use Tokenization. Tokenization is used to break down the raw data with long sentences into smaller data or smaller chunks which are called tokens. These tokens help in understanding the context or developing the recommendation system. What we are doing is word tokenization. For example, the text “It was nice ” can be tokenized into ‘It’, ‘was’, ‘nice’.

Before:

```
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
i really cant stay baby cold outside ive got go away baby cold this evening been hoping youd drop so nice i
```

After:

```
['i', 'really', 'cant', 'stay', 'baby', 'cold', 'outside', 'ive', 'got', 'go', 'away', 'baby', 'cold', 'this', 'evenin
```

Word Lemmatization:

Since the dataset contains lyrics that changed into tokens by using tokenization we still improve the lyrics data into better data by using word lemmatization. Word lemmatization is a text pre-processing technique used to reduce a given word in the lyrics to a root word. Basically, we can say lemmatization and stemming are the same but they are different in many ways. Stemming also does the same thing as stemming but it is different because stemming does not understand the meaning of the word before chopping the word but in lemmatization, every word before it understands the meaning of it and chops it if needed.

Finally created two new features called “final” and “final_genre_list” which is track_atrist and preprocessed lyrics combined(feature extraction) and genre and sub_genre combined feature.

```
1    Steady Rollin the tree singing wind the sky bl...
2    Bell Biv DeVoe na yeah spyderman freeze full e...
3    CeeLo Green i really cant stay baby cold outsi...
4    KARD get business you dont keep turning 모두 다 여...
5    James TW hold breath dont look keep trying dar...
6    Diddy all i want somebody who gonna love someb...
7    Babyface feel good everybody tender lover tend...
8    Blasterjaxx dont run away getting colder our h...
9    Rocco Hunt ho una cosa da dirti da tempo ma no...
Name: final, dtype: object
1            [rock, hard, rock]
2       [r&b, new, jack, swing]
3              [r&b, neo, soul]
4             [pop, dance, pop]
5    [r&b, urban, contemporary]
6       [r&b, new, jack, swing]
7       [r&b, new, jack, swing]
8              [edm, big, room]
9               [r&b, hip, pop]
Name: final_genre_list, dtype: object
```

The second dataset we are using is the million songs dataset. The lyrics in this data is in the form of a bag of words where each track is described as the word-counts for a dictionary of the top 5,000 words across the set.

B.FEATURE ENGINEERING

In this stage, We have created a new feature from scratch using the feature “lyrics” which is called “sentiment”. We have created this new feature using an existing library called “SentimentIntensityAnalyzer”.

SentimentIntensityAnalyzer:

It is NLTK library from VADER SentimentIntensityAnalyzer which relies on a dictionary that maps lexical features to emotion intensity known as sentiment scores. The sentiment score of a text can be obtained by summing up the intensity of each word in the text. For example, words like “good”, and “better” are considered positive but VADER is intelligent enough to know that “it was not good” is a negative statement. It takes the preprocessed lyrics as input and converts them into a dictionary of scores like “positive”, “negative”, “neutral” and “compound”. VADER is also sensitive to both polarity and strong emotional intensity.

track_album_release_date	playlist_name	playlist_id	...	liveness	valence	tempo	duration_ms	language	final	negative	neutral	positive	compound
2001-01-01	Pinoy Classic Rock	37i9dQZF1DWYDQ8wBxd7xt	...	0.0887	0.566	97.091	235440	tl	Barbie's Cradle minsan pa nang akoy napalingon...	0.000	1.000	0.000	0.0000
2017-11-21	Hard Rock Workout	3YouF0u7waJnolytf9JCXf	...	0.3470	0.404	135.225	373512	en	Steady Rollin the tree singing wind the sky bl...	0.059	0.505	0.436	0.9850
2005-01-01	Back in the day - R&B, New Jack Swing, Swingbe...	3a9y4eeCJRmG9p4YKfqYIx	...	0.4890	0.650	111.904	262467	en	Bell Biv DeVoe na yeah spyderman freeze full e...	0.306	0.527	0.167	-0.9969
2012-10-29	Christmas Soul	6FZYc2BvF7tColxO8PBShV	...	0.0664	0.405	118.593	243067	en	CeeLo Green i really cant stay baby cold outsi...	0.113	0.713	0.173	0.8234
2019-09-22	K-Party Dance Mix	37i9dQZF1DX4RDXswvP6Mj	...	0.1380	0.240	130.018	193160	en	KARD get business you dont keep turning 모두 다 여...	0.400	0.544	0.057	-0.9992

We have done one-hot encoding to the new feature “sentiment” which is a data preprocessing technique and also done one-hot encoding for subjectivity, key, and mode features using a user-defined function called “ohe_prep” and multiple each one-hot encoding like subjectivity with 0.3, key, mode and sentiment with 0.5.

```
0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
        ...
18449    0.5
18450    0.0
18451    0.0
18452    0.0
18453    0.0
Name: key|10, Length: 18454, dtype: float64
```

```
0        0.0
1        0.0
2        0.5
3        0.5
4        0.0
        ...
18449    0.5
18450    0.0
18451    0.5
18452    0.5
18453    0.0
Name: mode|0, Length: 18454, dtype: float64
```

```
0        0.5
1        0.5
2        0.5
3        0.5
4        0.5
        ...
18449    0.5
18450    0.5
18451    0.5
18452    0.5
18453    0.5
Name: polar|1, Length: 18454, dtype: float64
```

```
0        0.0
1        0.3
2        0.3
3        0.3
4        0.3
        ...
18449    0.3
18450    0.3
18451    0.3
18452    0.3
18453    0.0
Name: subject|high, Length: 18454, dtype: float64
```

TF IDF Vectorization:

Term Frequency-Inverse Document Frequency, often known as TF-IDF, is a technique for counting the number of words in a group of texts. The objective of TF-IDF is to highlight a word's significance within the corpus and documents. The following is the general formula to calculate TF-IDF:

Term Frequency×Inverse Document Frequency

- **Term Frequency (TF)**: The number of times a term appears in each document divided by the total word count in the document.
- **Inverse Document Frequency (IDF)**: The log value of the document frequency. Document frequency is the total number of documents where one term is present.

TFIDF

For a term i in document j:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Finding words that are significant in each document and encompass the entire corpus is the goal. A big N would result in an extremely large IDF compared to TF, hence the log number was chosen to lessen its impact. IDF focuses on the significance of a word across papers, whereas TF concentrates on the significance of a word within a document.

The documents in this project are comparable to tunes. Therefore, to establish the weight of the genre, we are analyzing the most prevalent genre in each song as well as their prevalence across songs. Since there are no

weights to account for the importance and popularity of each genre, which could result in overweighting on uncommon genres, this is far better than one-hot encoding.

We have applied TF IDF vectorization to the “final_genre_list” feature to quantify its word.

	genre|album	genre|big	genre|classic	genre|contemporary	genre|dance	genre|edm	genre|electro	genre|electropop	genre|gangster	genre|hard	...	genre|rock	ge
0	0.0	0.0	0.600851	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.799361	
1	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.600643	...	0.799517	
2	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.000000	
3	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.000000	
4	0.0	0.0	0.000000	0.0	0.682981	0.000000	0.000000	0.000000	0.0	0.000000	...	0.000000	
...	...	...	...	...	...	...	...	...	...	...	...	...	
18449	0.0	0.0	0.000000	0.0	0.000000	0.420954	0.498251	0.000000	0.0	0.000000	...	0.000000	
18450	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.875566	0.0	0.000000	...	0.000000	
18451	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.000000	
18452	0.0	0.0	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.000000	
18453	0.0	0.0	0.600851	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	...	0.799361	

Finally created a new data frame called “complete_feature_list we have included all the data preprocessed data and one hot encoded data into it.

	genre|album	genre|big	genre|classic	genre|contemporary	genre|dance	genre|edm	genre|electro	genre|electropop	genre|gangster	genre|hard	...	key|5	key|6	key|7
0	0.0	0.0	0.600851	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0
1	0.0	0.0	0.000000	0.0	0.000000	0.0	0.0	0.0	0.0	0.600643	...	0.0	0.0	0.0
2	0.0	0.0	0.000000	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.5	0.0
3	0.0	0.0	0.000000	0.0	0.000000	0.0	0.0	0.0	0.0	0.000000	...	0.5	0.0	0.0
4	0.0	0.0	0.000000	0.0	0.682981	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0

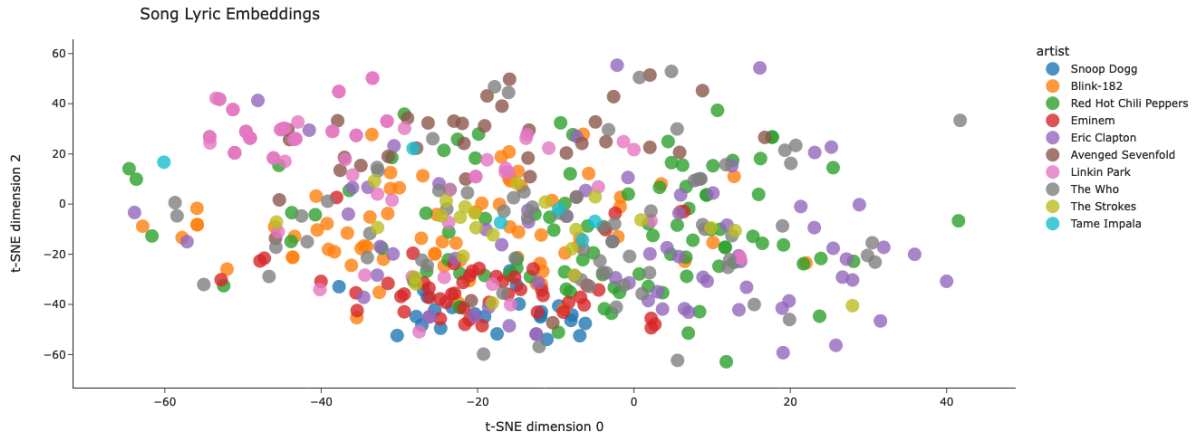
5 rows × 65 columns

Song Lyrics Embedding

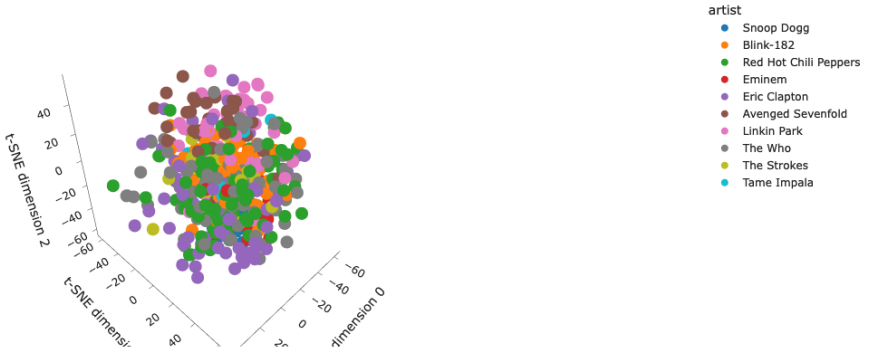
Word embedding is a powerful natural language processing tool that helps us make words into numbers. When we are dealing with numbers, we can convert them to vectors. When we can turn sentences into vectors, then we can also turn songs into vectors and try to visualize them in a different way. Using this visualization, we can get some insights on how different songs are correlated or how different artists may compose similar types of songs. A modeling of our song dataset can be done using paragraph vector-Distributed bag of words model to get the vectors of lyrics data of the song.

Visualization of song Lyrics

To visualize the songs, we use a technique called t-Distributed Stochastic Neighbor Embedding (t-SNE). It is a dimensionality reduction technique which is very helpful in visualizing high dimensional data. One of the advantages of t-SNE is it plots local relationships which we can use to plot data in low dimensions to get a view of it in high dimensions. One of the low dimension examples is shown below



As we can see, the different colors indicate different artists and the dots on the graph show the songs they make. To the human eye, we can see that they don't make much sense. We can combine multiple of these plots into 3d plot and try to observe some patterns



Here we can see that some of the artists produce similar songs to other artists and we can use this information to recommend not only similar songs but also similar artists.

C. RECOMMENDATION SYSTEM

The following step is to carry out content-based filtering using the available song features. We combine all of the songs on a playlist into a single summary vector to achieve this. Then, we compare the summarized playlist vector to every song (excluding the songs in the playlist) in the database to see how similar they are. The most pertinent song that isn't in the playlist is then recommended using the similarity metric.

We have used cosine similarity in our recommendation system. Cosine similarity is a mathematical value that measures the similarity between two vectors. For example, if we are considering our songs to be two-dimensional then the degree of similarity will be for the two songs is 0 for similar, 90 for nearly similar, and 180 for opposite songs.

Mathematical Formula for Cosine Similarity:

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

D. GENERATED RECOMMENDATIONS

In this stage, we will generate a recommendation for the user using the above-generated system.

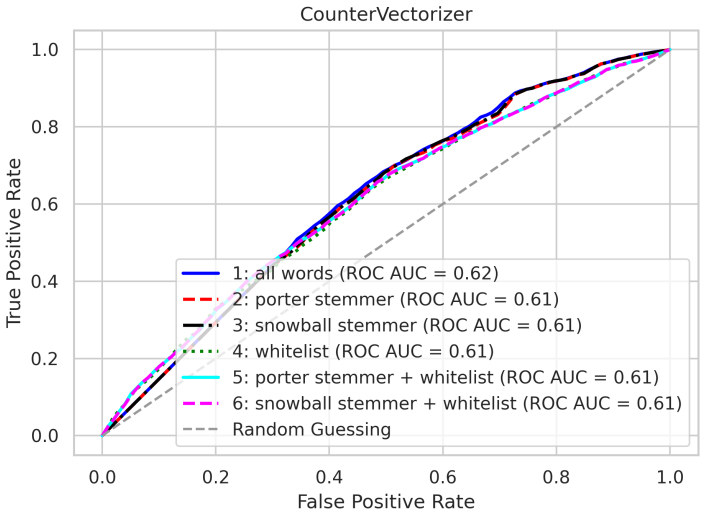
	track_artist	track_name
0	Barbie's Cradle	Pangarap
12111	The Cranberries	All Over Now
12307	Cidtronyck	Esta Noche (feat. Portavoz, Niel & Evelyn Corn...
12306	Bruce Springsteen	Hello Sunshine
12305	Jason Derulo	In My Head
12304	Ari Lennox	Shea Butter Baby (with J. Cole)
12303	SaberZ	Take On The World
12302	Righeira	Vamos a la Playa
12301	BROCKHAMPTON	FACE
12300	Elton John	Don't Go Breaking My Heart

	title	all_artists
0	Adhana	['Vini Vici', 'Astrix']
1	White Dwarf	['Off Limits', 'Wilders']
2	Deep Jungle Walk	['Astrix']
3	He.art	['Astrix']
4	Pranava - Ranji & Mind Spin Remix	['Ace Ventura', 'Astrix', 'Ranji', 'Mind Spin']
5	Creatures	['Alpha Portal']
6	We Are In The Shadows	['Outsiders', 'Imagine Mars']
7	Our Moment Has Arrived - Original Mix	['Outsiders']
8	Science Is God - Outsiders Vs. Xerox Remix	['Xerox', 'Passenger']
9	Space Travel - Original Mix	['Outsiders', 'Killerwatts']
10	Colors - UK Psychedelic Remix	['Avalon', 'Tristan', 'Vini Vici', 'Killerwatts']
11	We Are the Creators	['Vini Vici', 'Bryan Kearney']
12	Psychedelic - Original Mix	['Outsiders']
13	Stimulator - Off Limits Remix	['Ace Ventura', 'Intelabeam', 'Off Limits']
14	Secrets of the Universe	['Ranji', 'Mind Spin']
15	The Escape - Original Mix	['Outside The Universe', 'Tristan']
16	Action Jackson	['Day Din', 'WAIO']
17	The Force	['Sonic Species']

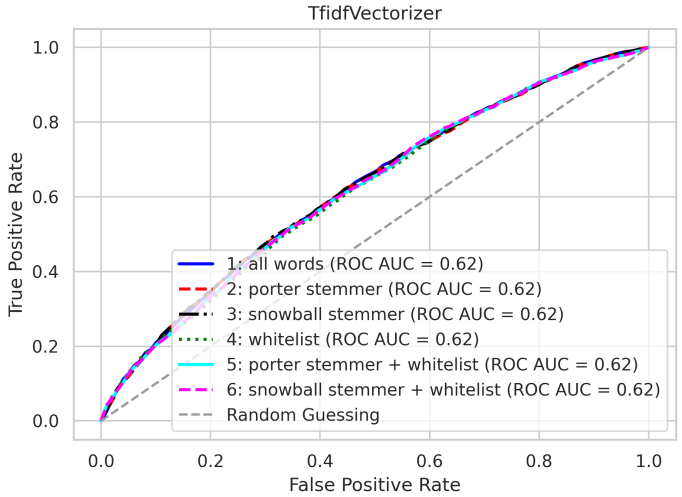
V. EXPERIMENTS

We tried to incorporate a baseline model in Naive bias classifier to classify songs based on the lyrics and got some results. Why Naive Bias? As we were taught in class, sometimes naive bias can simply outperform some advanced models because of the simplicity of the algorithm. It is faster and we can also get different metrics to evaluate the model that we used. We used the same kaggle dataset as mentioned above to generate the results and we are using ROC curves as evaluation metrics.

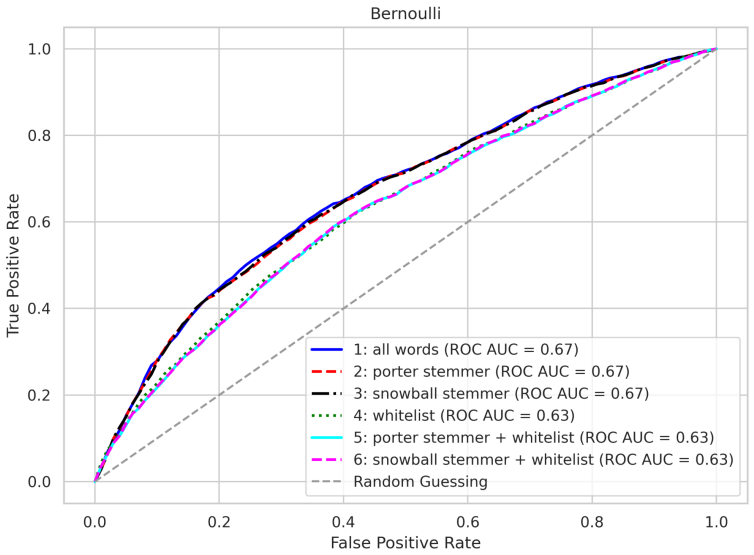
We also used different vectorizers on the same multinomial Naive bias model to evaluate their prediction accuracy.



The above graph shows the ROC curve for counter vectorizer. Each unique word is represented by a column in the matrix that is created by CountVectorizer, and each sample of text from the document is represented by a row in the matrix. Each cell's value is just the number of words in that specific text sample.



The above graph shows the ROC curve for the method we used for cosine similarity, the TfIDF vectorizer. We have already briefed about TFIDF vectorization in the above sections.

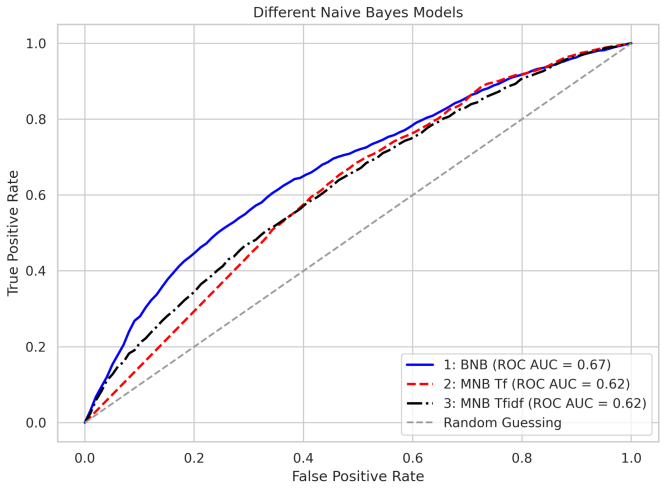


The above graph shows the ROC for Bernoulli Naive Bias. The Naive Bayes family includes Bernoulli Naive Bayes. It uses the Bernoulli Distribution as its foundation and only accepts binary data, i.e., 0 or 1. We may presume that Bernoulli Naive Bayes will be the method to apply if the dataset's characteristics are binary. The equations of BernoulliNB are similar to that of classic NB but with Bernoulli distribution.

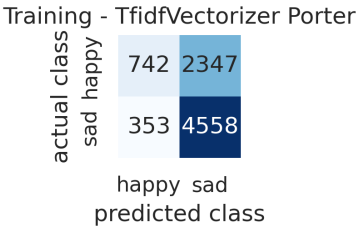
$$p(x) = P[X = x] = \begin{cases} q = 1 - p & x = 0 \\ p & x = 1 \end{cases}$$

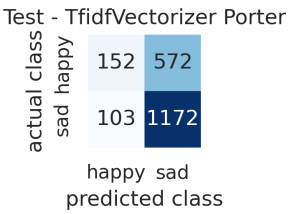
$$X = \begin{cases} 1 & \text{Bernoulli trial} = \mathbf{S} \\ 0 & \text{Bernoulli trial} = \mathbf{F} \end{cases}$$

To compare the three models we used, we can plot a combined ROC with the best words. The graph is shown below



From the above graph, we can observe that BernoulliNB performed better than MNB with an avg roc with 0.67.





The images above show the confusion matrices of train and test data from the TfIDF vectorizer.

VI. CONCLUSION

In this project we worked on getting recommendations of songs based on different features in the songs. We mainly used lyrics and metadata of the song to compute the sentiment and give a proper prediction of what a user will like if they are listening to a particular song. We also observed from different visualizations that songs overlap each other and there is no best or worst recommendation for a given song. Most of the songs overlap with each other and any one of them can make a good recommendation. There is a possibility of fine tuning based on mixing in other techniques like Collaborative filtering and Convolutional Neural Networks. In experiments, we also observed that different variations of Naive Bias can give different results not only based on the algorithm but also with different vectorization techniques we used.

REFERENCES

[1] https://en.wikipedia.org/wiki/Netflix_Prize

[2]S. -H. Chang, A. Abdul, J. Chen and H. -Y. Liao, "A personalized music recommendation system using convolutional neural networks approach," 2018 IEEE International Conference on Applied System Invention (ICASI), 2018, pp. 47-49, doi: 10.1109/ICASI.2018.8394293.

[3].J.B. Schafer, D. Frankowski, J. Herlocker and S. Sen, "Collaborative filtering recommender systems", The Adaptive Web: Methods and Strategies of Web Personalization, pp. 291-324, 2007

[4]M.J. Pazzani and D. Billsus, "Content based recommendation systems", The Adaptive Web: Methods and Strategies of Web Personalization, pp. 325-341, 2007.

[5] P. N, D. Khanwelkar, H. More, N. Soni, J. Rajani and C. Vaswani, "Analysis of Clustering Algorithms for Music Recommendation," 2022 IEEE 7th International conference for Convergence in Technology (I2CT), Mumbai, India, 2022, pp. 1-6, doi: 10.1109/I2CT54291.2022.9824160.

[6] S. -H. Chang, A. Abdul, J. Chen and H. -Y. Liao, "A personalized music recommendation system using convolutional neural networks approach," 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba, Japan, 2018, pp. 47-49, doi: 10.1109/ICASI.2018.8394293.

[7] Visualizing Data using t-SNE Laurens van der Maaten, Geoffrey Hinton; 9(86):2579−2605, 2008.

[8]D. Rani, R. Kumar and N. Chauhan, "Study and Comparision of Vectorization Techniques Used in Text Classification," 2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2022, pp. 1-6, doi: 10.1109/ICCCNT54827.2022.9984608.

Links to code

https://colab.research.google.com/drive/1RiAGbbkv_d8sLe4vkuU_uSdxpxLGXjgJ?usp=sharing