PROJECT DELIVERABLE 2

Problem Statement

This project aims to develop a music recommendation system using a large dataset. The problem entails preprocessing the dataset, creating a machine learning model for content-based music recommendations, and evaluating the system's performance subjectively. The ultimate goal is to integrate this recommendation system into a web application, allowing users to input artists, songs and receive embedded recommendations.

Data Preprocessing

https://www.kaggle.com/datasets/amitanshjoshi/spotify-1million-tracks/

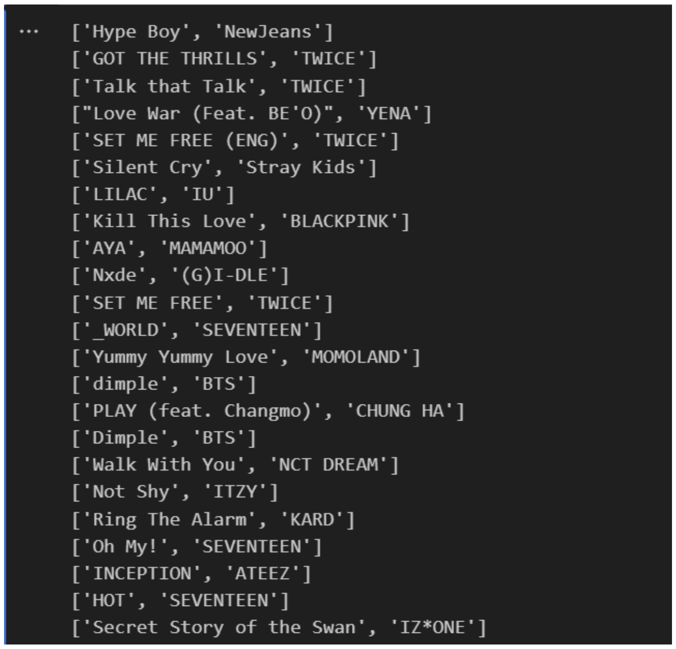
I used the same dataset, linked above, that I planned to use in Deliverable 1. It contains over a million unique tracks released between 2000 and 2023 extracted from Spotify. For each track, the dataset provides the name, artist, year, genre and thirteen other characteristics, such as loudness and valence. To process the data, I started by ensuring that there were no null or duplicate tracks. I then scaled the numerical columns using MinMaxScaler to ensure that each feature was correctly accounted for in the distance calculations. I then encoded the genre data using OneHotEncoder to create a new column for each unique genre, increasing my total number of columns to 100.

Machine learning model

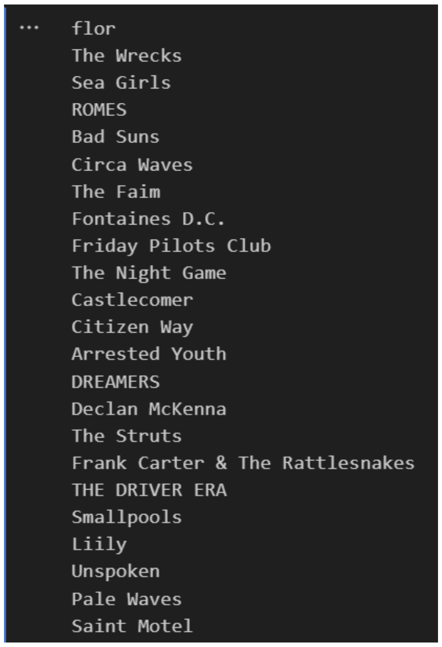
As mentioned in Deliverable 1, I started by implementing cosine similarity using sklearn's cosine_similarity method. However, this was very slow in providing similar songs when given all rows and dimensions of the dataset. I ended up switching to scipy's cdist method, using Euclidean distance as the comparison metric, which was able to produce a list of similar songs in approximately 1 second on my computer. While I am satisfied with the accuracy and temporal performance of the algorithm, would like to reduce the amount of memory it uses when creating the dataframe and computing the Euclidean distance from all songs in the dataset.

Preliminary Results

As stated in my first deliverable, since music taste is inherently subjective, I would be evaluating the model primarily based on my own musical tastes and knowledge along with the opinion of my friends on the model's output. For this deliverable, I did some preliminary tests by inputting a song and judging the results, as seen below. On the left, I gave a k-pop song and received a list of similar k-pop songs, demonstrating that the genre encoding was working as intended. I also implemented a function that finds similar artists, an example of which can be found on the right.



```
...  ['Hype Boy', 'NewJeans']
     ['GOT THE THRILLS', 'TWICE']
     ['Talk that Talk', 'TWICE']
     ["Love War (Feat. BE'O)", 'YENA']
     ['SET ME FREE (ENG)', 'TWICE']
     ['Silent Cry', 'Stray Kids']
     ['LILAC', 'IU']
     ['Kill This Love', 'BLACKPINK']
     ['AYA', 'MAMAMOO']
     ['Nxde', '(G)I-DLE']
     ['SET ME FREE', 'TWICE']
     ['_WORLD', 'SEVENTEEN']
     ['Yummy Yummy Love', 'MOMOLAND']
     ['dimple', 'BTS']
     ['PLAY (feat. Changmo)', 'CHUNG HA']
     ['Dimple', 'BTS']
     ['Walk With You', 'NCT DREAM']
     ['Not Shy', 'ITZY']
     ['Ring The Alarm', 'KARD']
     ['Oh My!', 'SEVENTEEN']
     ['INCEPTION', 'ATEEZ']
     ['HOT', 'SEVENTEEN']
     ['Secret Story of the Swan', 'IZ*ONE']
```



```
...  flor
     The Wrecks
     Sea Girls
     ROMES
     Bad Suns
     Circa Waves
     The Faim
     Fontaines D.C.
     Friday Pilots Club
     The Night Game
     Castlecomer
     Citizen Way
     Arrested Youth
     DREAMERS
     Declan McKenna
     The Struts
     Frank Carter & The Rattlesnakes
     THE DRIVER ERA
     Smallpools
     Lily
     Unspoken
     Pale Waves
     Saint Motel
```

Next Steps

The next step is to work on integrating the model into an application. I plan to use Streamlit to create and host a public web app where people can get their own customized recommendations without having to download anything locally. However, I may have to modify the dataset or parameters that I am currently using when I make the web app, as Streamlit Community Cloud has resource limits that could prevent me from using my entire dataset. I will continue optimizing my algorithm to reduce its resource utilization, and may look into KMeans clustering to create clusters of similar songs if it can help with that.