Music Recommender System

Mario Jacobo & Alexander Baez

1 Introduction

Music has played a central role in human culture for centuries, and continues to do so today. There are many reasons why music is so important, and it can have a profound impact on people's lives. One reason music is important is that it has the ability to evoke strong emotions. Whether it's the joy of listening to a beloved song or the sadness of a mournful ballad, music has the power to move us in a way that few other things can. It can bring people together, helping them to feel a sense of connection and community.

Music also has the ability to help people express themselves and their emotions. Whether it's through lyrics, melodies, or instrumental compositions, music allows people to communicate in a way that words alone cannot. It can be a way to express feelings that are difficult to put into words, and can be a powerful outlet for people who are struggling with difficult emotions.
Music is also important because it has the ability to enhance other activities. Many people listen to music while working out, driving, or even studying, as it can help to improve focus and motivation. In addition, music is often used in a variety of settings, including movies, television shows, and video games, to enhance the overall experience and create a desired mood or atmosphere..

Music recommendation systems are important because they help people discover new music and artists that they may not have otherwise been exposed to. In a world where there is an overwhelming amount of music available, it can be difficult for people to find new music that they enjoy. A music recommendation system can help to narrow down the options and recommend songs and artists that are tailored to an individual's personal tastes.

One way that music recommendation systems work is by using algorithms to analyze an individual's listening habits and preferences. The system can then use this information to recommend songs and artists that are similar to what the person has listened to in the past. This can help people to find new music that they are more likely to enjoy, rather than spending hours sifting through hundreds of songs and artists that may not be a good fit.

Music recommendation systems are also important because they can help people to save time. Rather than spending hours searching for new music, a recommendation system can provide a curated list of songs and artists that are tailored to an individual's tastes. This can help people to find new music more efficiently, allowing them to spend more time listening to and enjoying the music they love.

Overall, music recommendation systems are important because they help people to discover new music and artists and save time by providing a curated list of recommendations. Whether you're a music lover looking for new tunes or an artist trying to reach a larger audience, a music recommendation system can be a valuable tool. The question now is, how can we apply machine learning to music recommendation systems?

2 Problem Definition

Recommendations are constantly being thrown at us in this day and age. Recommender systems are assisting in the process and they are constantly running behind the scenes with the majority of the world not even realizing it. The search for the best recommendation system is never ending with our technology getting better and data scientists working around the clock. Content based filtering systems have been crucial in developing high quality recommender systems. After performing some research on content based filtering we learned the content based model has various advantages. This model does not need any data on other users since the recommendations are solely to that specific user. This means that the model can be scaled to a large number of users. Since the model uses specific user action or direct feedback the model can recommend very specific items to that user; something other models may not be able to do. Unfortunately due to the dependency on the feature representations, the model has some downsides. The feature representations are engineered manually which means that this model requires a deep understanding of the domain being used. This results in a model that can only be as good as the features that were chosen. Collaborative filtering also has its pros and cons and it really comes down to what the problem calls for. Companies like Netflix may go for collaborative filtering, when it comes to recommending shows on a Sunday night to its 200 million subscribers. On the other hand some companies like Nike may keep track of what you search for the most on their website and develop a content based model this way to recommend products like the ones you have been searching for.

3 Data

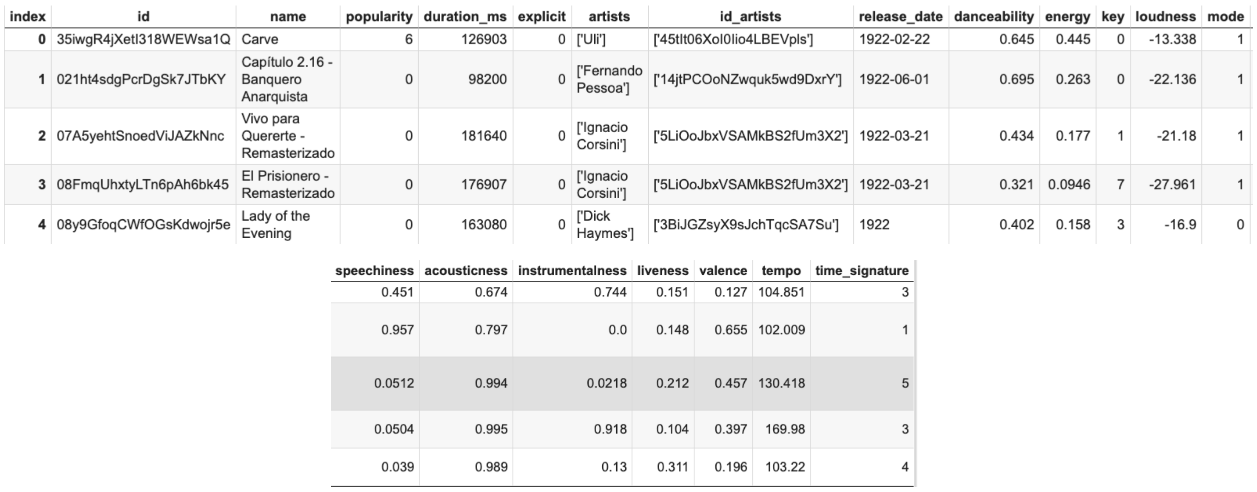
Dataset - The source of the dataset used in our project is https://www.kaggle.com/datasets/yamaerenay/spotify-dataset-19212020-600k-tracks. The dataset consists of over 600 thousand tracks dating all the way back to the year 1921. Each track has 20 features, which are described in more detail below.

Table:

Feature	**Description**
id	The Spotify ID for the track
name	Name of the track
popularity	The popularity of a track is a value between 0 and 100, with 100 being the most popular
duration_ms	The track length in milliseconds
explicit	Whether or not the track has explicit lyrics
artists	Artists in the track
id_artists	The Spotify ID for the track
release_date	When the track was released
danceability	A value of 0.0 is least danceable and 1.0 is most danceable
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity
key	The key the track is in
loudness	The overall loudness of a track in decibels (dB)

mode	Major is represented by 1 and minor is 0
speechiness	Speechiness detects the presence of spoken words in a track
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic
instumentalness	Predicts whether a track contains no vocals
liveness	Detects the presence of an audience in the recording
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track
tempo	The overall estimated tempo of a track in beats per minute (BPM)
time_signature	The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure)

Figure 1: Dataset Sample

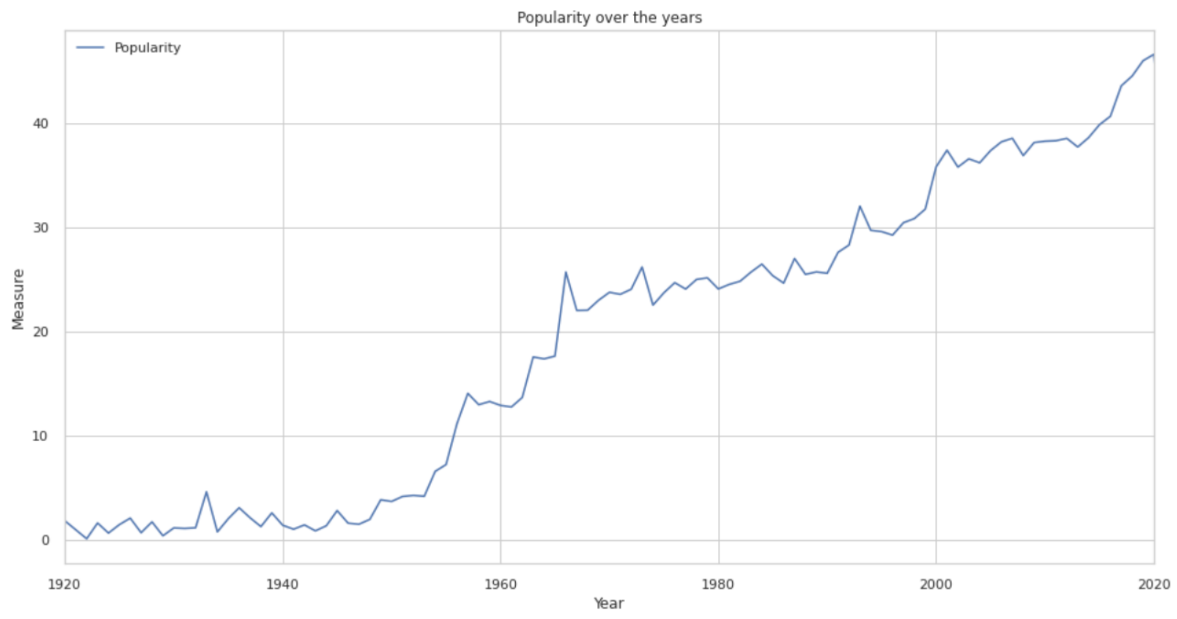


index	id	name	popularity	duration_ms	explicit	artists	id_artists	release_date	danceability	energy	key	loudness	mode
0	35iwgR4jXetl318WEWsa1Q	Carve	6	126903	0	['Uli']	['45tlt06XoI0lio4LBEVpls']	1922-02-22	0.645	0.445	0	-13.338	1
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['14jtPCOoNZwquk5wd9DxrY']	1922-06-01	0.695	0.263	0	-22.136	1
2	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	1922-03-21	0.434	0.177	1	-21.18	1
3	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	1922-03-21	0.321	0.0946	7	-27.961	1
4	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	['Dick Haymes']	['3BiJGZsyX9sJchTqcSA7Su']	1922	0.402	0.158	3	-16.9	0

speechiness	acousticness	instrumentalness	liveness	valence	tempo	time_signature
0.451	0.674	0.744	0.151	0.127	104.851	3
0.957	0.797	0.0	0.148	0.655	102.009	1
0.0512	0.994	0.0218	0.212	0.457	130.418	5
0.0504	0.995	0.918	0.104	0.397	169.98	3
0.039	0.989	0.13	0.311	0.196	103.22	4

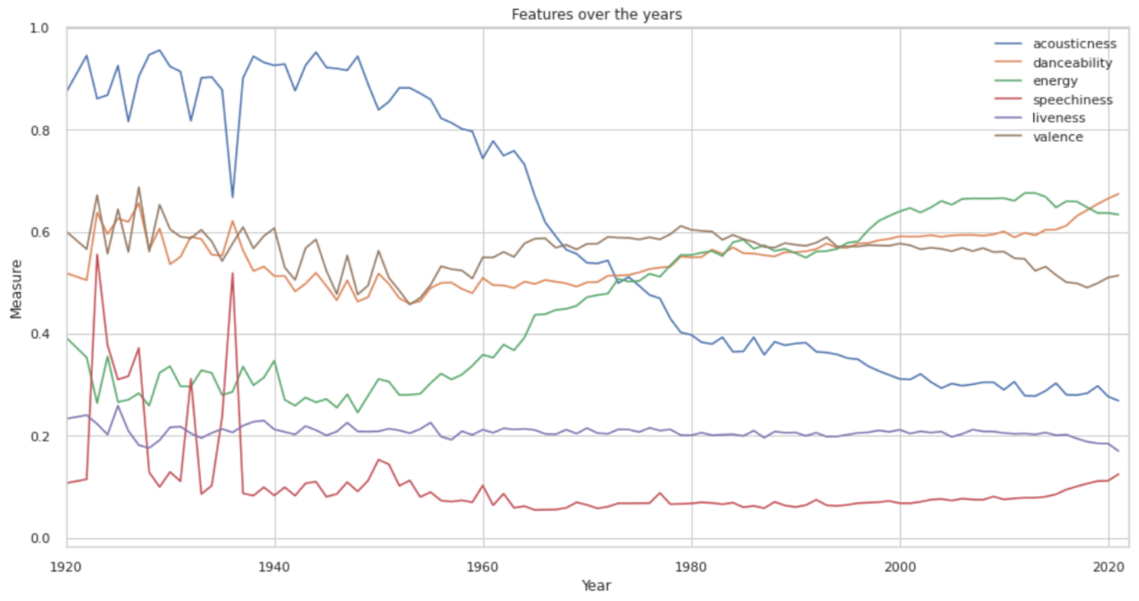
This dataset is not easy to read, to help visualize the data we have provided some graphs in order to assist in exploratory data analysis. We will look at some of the most important metrics of this dataset. Starting off with 'popularity'. The popularity is calculated by an algorithm and is based on the total number of plays the track has had and how recent those plays are. Generally speaking, songs that are being played a lot now will have a higher popularity than songs that were played a lot in the past. Duplicate tracks (e.g. the same track from a single and an album) are rated independently. Accordingly, newer music is much more popular as shown in the graph below.

Figure 2: Popularity vs. Years



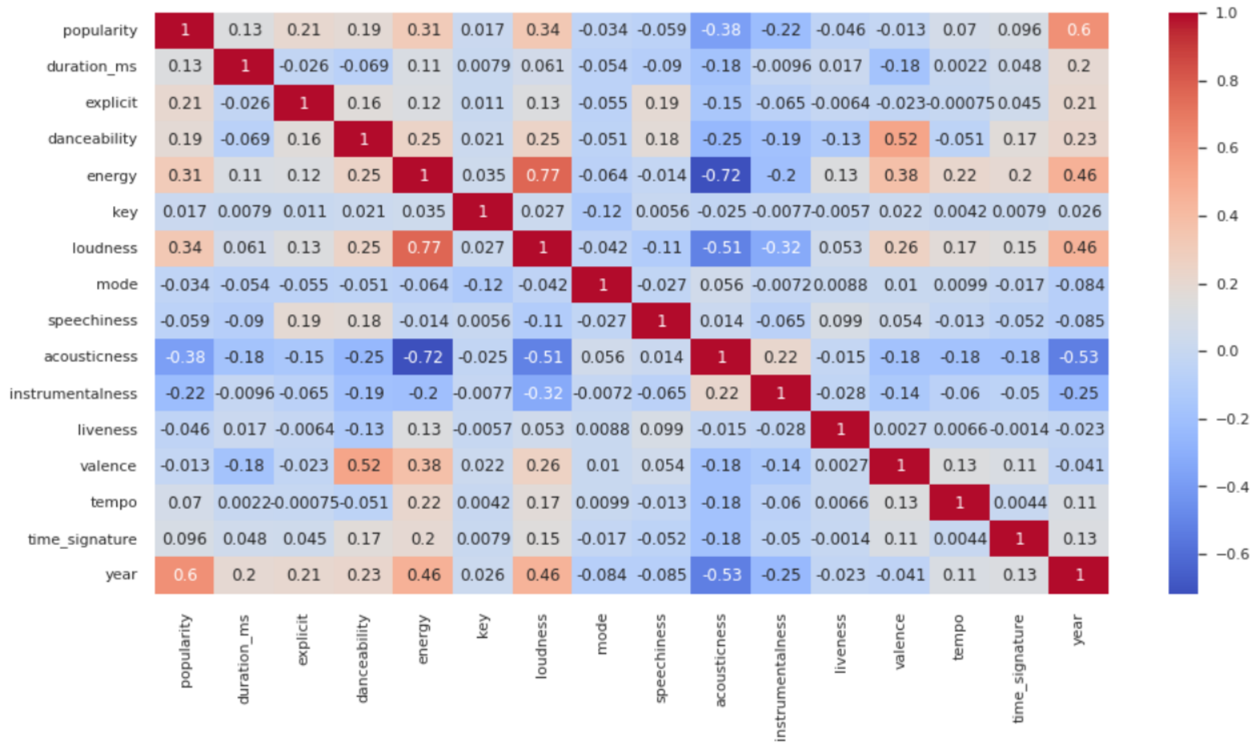
The following graph shows just how much certain features have changed over the last hundred years.

Figure 3: Different Features vs. Years



Now that we took a closer look at some of the individual features of the data, let's observe the correlation between the features.

Figure 4: Correlation Matrix



	popularity	duration_ms	explicit	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	time_signature	year
popularity	1	0.13	0.21	0.19	0.31	0.017	0.34	-0.034	-0.059	-0.38	-0.22	-0.046	-0.013	0.07	0.096	0.6
duration_ms	0.13	1	-0.026	-0.069	0.11	0.0079	0.061	-0.054	-0.09	-0.18	-0.0096	0.017	-0.18	0.0022	0.048	0.2
explicit	0.21	-0.026	1	0.16	0.12	0.011	0.13	-0.055	0.19	-0.15	-0.065	-0.0064	-0.023	-0.00075	0.045	0.21
danceability	0.19	-0.069	0.16	1	0.25	0.021	0.25	-0.051	0.18	-0.25	-0.19	-0.13	0.52	-0.051	0.17	0.23
energy	0.31	0.11	0.12	0.25	1	0.035	0.77	-0.064	-0.014	-0.72	-0.2	0.13	0.38	0.22	0.2	0.46
key	0.017	0.0079	0.011	0.021	0.035	1	0.027	-0.12	0.0056	-0.025	-0.0077	-0.0057	0.022	0.0042	0.0079	0.026
loudness	0.34	0.061	0.13	0.25	0.77	0.027	1	-0.042	-0.11	-0.51	-0.32	0.053	0.26	0.17	0.15	0.46
mode	-0.034	-0.054	-0.055	-0.051	-0.064	-0.12	-0.042	1	-0.027	0.056	-0.0072	0.0088	0.01	0.0099	-0.017	-0.084
speechiness	-0.059	-0.09	0.19	0.18	-0.014	0.0056	-0.11	-0.027	1	0.014	-0.065	0.099	0.054	-0.013	-0.052	-0.085
acousticness	-0.38	-0.18	-0.15	-0.25	-0.72	-0.025	-0.51	0.056	0.014	1	0.22	-0.015	-0.18	-0.18	-0.18	-0.53
instrumentalness	-0.22	-0.0096	-0.065	-0.19	-0.2	-0.0077	-0.32	-0.0072	-0.065	0.22	1	-0.028	-0.14	-0.06	-0.05	-0.25
liveness	-0.046	0.017	-0.0064	-0.13	0.13	-0.0057	0.053	0.0088	0.099	-0.015	-0.028	1	0.0027	0.0066	-0.0014	-0.023
valence	-0.013	-0.18	-0.023	0.52	0.38	0.022	0.26	0.01	0.054	-0.18	-0.14	0.0027	1	0.13	0.11	-0.041
tempo	0.07	0.0022	-0.00075	-0.051	0.22	0.0042	0.17	0.0099	-0.013	-0.18	-0.06	0.0066	0.13	1	0.0044	0.11
time_signature	0.096	0.048	0.045	0.17	0.2	0.0079	0.15	-0.017	-0.052	-0.18	-0.05	-0.0014	0.11	0.0044	1	0.13
year	0.6	0.2	0.21	0.23	0.46	0.026	0.46	-0.084	-0.085	-0.53	-0.25	-0.023	-0.041	0.11	0.13	1

Some important observations to point out are:

- Acousticness and Energy have a strong negative correlation
- Loudness and Energy have a strong positive correlation
- Acousticness and Year have a negative correlation
- Popularity and Year have a positive correlation (as observed in Figure 2)

4 Methods

We decided to try K-Means clustering and KNN for our recommendation models. K-Means is an unsupervised learning algorithm, used for when you have unlabeled data like in this case. The goal of this algorithm is to find clusters in the data. It will assign each point to a certain cluster based on the euclidean distance metric. For the K-Means clustering we decided to choose the MinMaxScaler provided by scikit-learn, this scaler resulted in the lowest distortion score which we will dive into deeper in the next section. We used scikit-learn for several methods during our project such as K-Means, scalers, PCA, and pipelines. We also used a submodule from scikit-learn, metrics pairwise to calculate euclidean distances in our experiments. We also experimented with PCA and created some graphs to help visualize the clusters. Lastly we used scikit-learn's NearestNeighbors, an unsupervised learner for neighbor searches, and we used it with a kd-tree neighbors search algorithm. Sklearn's neighbor module provides functionality for unsupervised and supervised neighbors-based learning methods.

One advantage of k-means clustering is that it is relatively simple to implement and can be used with a wide range of data types. It is also relatively fast, making it well-suited for use with large datasets. Overall, k-means clustering is a useful tool for predicting data by identifying patterns and grouping similar data points together. Because of these benefits, we found it suitable to use for programs to recommend songs.

5 Experiments

When implementing the K-Means algorithm the most important metric we looked at was the distortion score. The distortion score is the sum of squared distances from each point to its corresponding centroid. We want to choose a model that results in a low distortion score. To do this we test with different numbers of clusters and also different forms of scaling the data. The best way to visualize this test is to use the elbow method and that is exactly what we did. We performed the elbow method on 3 different versions of our data. The first one was implementing K-Means on the raw data(no scaling done to data), the second test was using MinMaxScaler and the last test was using StandardScaler. These were our results:

Figure 5: KMeans on raw data

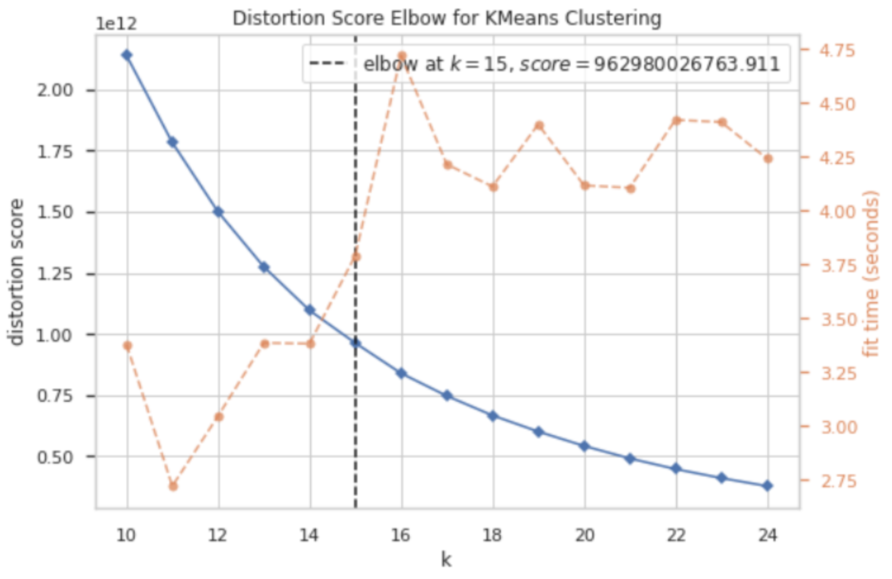


Figure 6: KMeans using MinMaxScaler

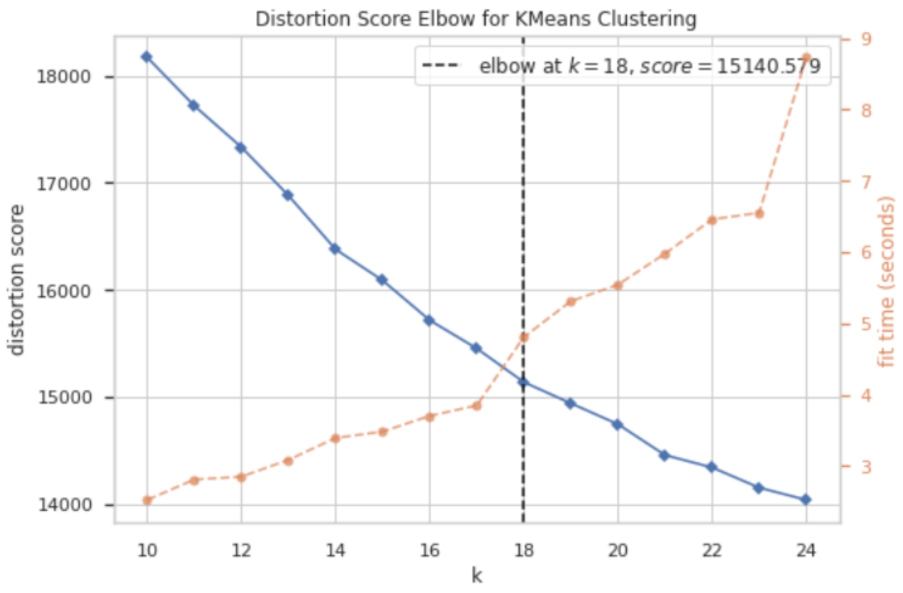
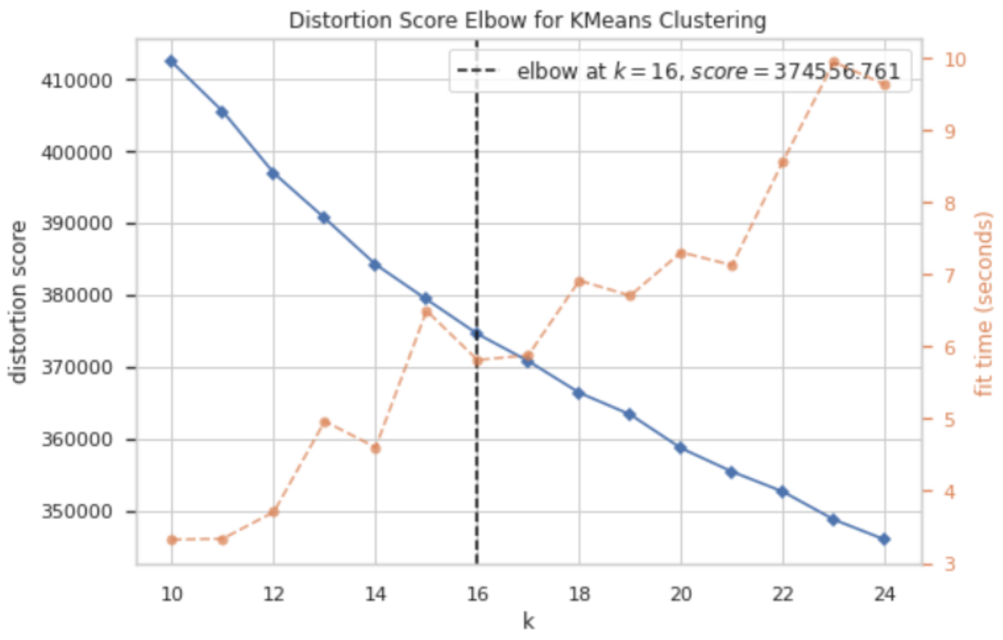


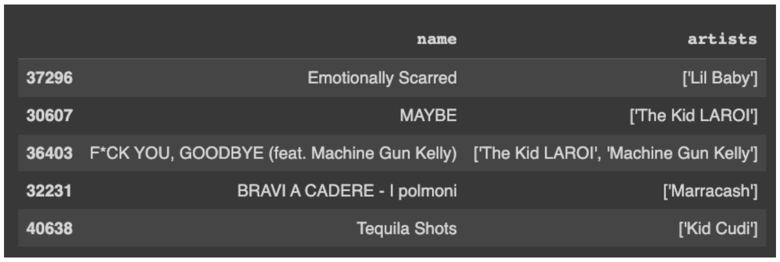
Figure 7: KMeans using StandardScaler



The graphs above show just how much of a difference scaling your data can do. Although the first test(figure 5) had the best performance in time, the distortion score was astonishingly high. The best performing model in this case was the second one, using MinMaxScaler. All three of the tests also iterated through different numbers of clusters, from 10 to 24 clusters to be specific. Our winning model gave us the parameter of using 18 clusters and had the lowest distortion score of 15140.

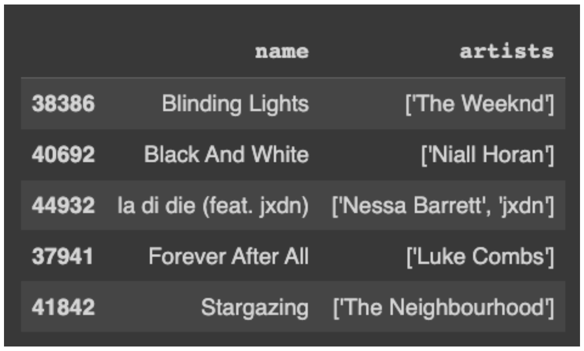
After performing those tests we came to the conclusion that we would be using the second model, so we created a pipeline implementing the necessary parameters. Now that our model was created it was time to test it. The following are examples of two different tests on the model:

Figure 8: KMeans Model Test



	name	artists
37296	Emotionally Scarred	['Lil Baby']
30607	MAYBE	['The Kid LAROI']
36403	F*CK YOU, GOODBYE (feat. Machine Gun Kelly)	['The Kid LAROI', 'Machine Gun Kelly']
32231	BRAVI A CADERE - I polmoni	['Marracash']
40638	Tequila Shots	['Kid Cudi']

Figure 9: KMeans Model Test

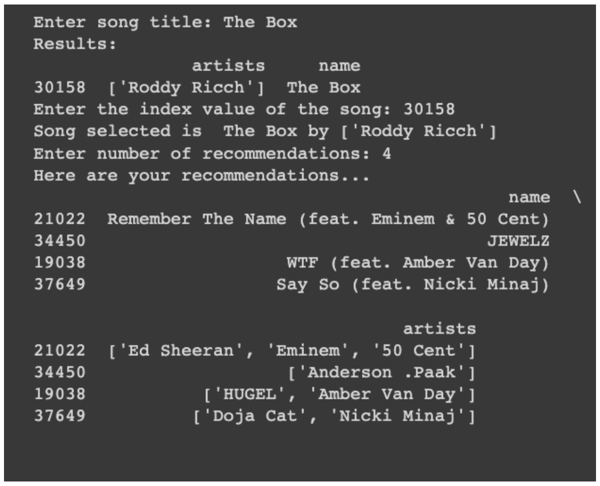


	name	artists
38386	Blinding Lights	['The Weeknd']
40692	Black And White	['Niall Horan']
44932	la di die (feat. jxdn)	['Nessa Barrett', 'jxdn']
37941	Forever After All	['Luke Combs']
41842	Stargazing	['The Neighbourhood']

The top row is the input song and the rest are recommendations based on the input song. The algorithm uses the euclidean distance metric to calculate its nearest point(in this case a song) in the corresponding cluster and return the index of that song, which then would display the name or names of similar songs to the one the user likes.

Our other experiment was regarding nearest neighbors and we used the kd tree algorithm for this implementation. Scikit-learn's NearestNeighbors applies unsupervised nearest neighbors learning, acting as a uniform interface to three different nearest neighbor algorithms - BallTree, KDTree, and brute-force. The algorithm we chose to implement was kd tree, which is used for fast generalized n point problems. Here is an example of a model test we performed by using the song 'The Box' by Roddy Rich as an example:

Figure 10: Nearest Neighbor Model Test



```
Enter song title: The Box
Results:
               artists     name
30158  ['Roddy Ricch']  The Box
Enter the index value of the song: 30158
Song selected is  The Box by ['Roddy Ricch']
Enter number of recommendations: 4
Here are your recommendations...
                                              name  \
21022  Remember The Name (feat. Eminem & 50 Cent)
34450                                       JEWELZ
19038                   WTF (feat. Amber Van Day)
37649                   Say So (feat. Nicki Minaj)

                                   artists
21022  ['Ed Sheeran', 'Eminem', '50 Cent']
34450                   ['Anderson .Paak']
19038           ['HUGEL', 'Amber Van Day']
37649          ['Doja Cat', 'Nicki Minaj']
```

After entering the song as an input, the model returns 4 songs that are similar to the input song.

6 Conclusion

In conclusion, our team has just completed a machine learning project that has been both challenging and rewarding. We began by thoroughly understanding the concept of music recommendation and we were trying to solve and identify the appropriate machine learning techniques to use. We then gathered and preprocessed the necessary data, trained and evaluated our models, and fine-tuned them to achieve the best possible results.

Throughout this project, we encountered various obstacles and had to adapt our approach multiple times. However, we were able to overcome these challenges through hard work, dedication, communication, and problem-solving skills.
We are proud of the results we have achieved and the possible impact this might have on our careers. However, we also recognize that there is always room for improvement and we will continue to learn and evolve as we take on new projects in the future.

Overall, our group is grateful for the opportunity to work on this project and we believe that our experience has not only helped us grow as individuals, but also as a cohesive unit. We are both excited to see what the future holds and curious to see the impact that this machine learning course will have on our work performance.

Link to GitHub Rep - https://github.com/mjacobo6/SongRecommender/tree/main

References

[]Google. (n.d). *Introduction | machine learning | google developers.* Google. Retrieved December 20,2022, from https://developers.google.com/machine-learning/recommendation
[]Muvi. (2021, May 20). *The difference between collaborative and content based recommendation engines*. Medium. Retrieved December 20, 2022, from https://muvi-com.medium.com/the-difference-between-collaborative-and-content-based-recommendation-engines-ade24b5147f2
[]Roy, A. (2020, July 31). *Introduction to Recommender systems- 1: Content-based filtering and collaborative filtering*. Medium. Retrieved December 20, 2022, from https://towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421
[]*1.6. nearest neighbors*. scikit. (n.d.). Retrieved December 20, 2022, from https://scikit-learn.org/stable/modules/neighbors.html