Members: Shirley Mach, Amy Meiyi Kuang

Problem Statement

The purpose of this project is to develop a music recommendation system using existing Spotify data to gain insights into the workings of various machine learning algorithms. We will explore and implement machine learning algorithms on a comprehensive Spotify dataset to evaluate their effectiveness in generating music recommendations.

Data Information

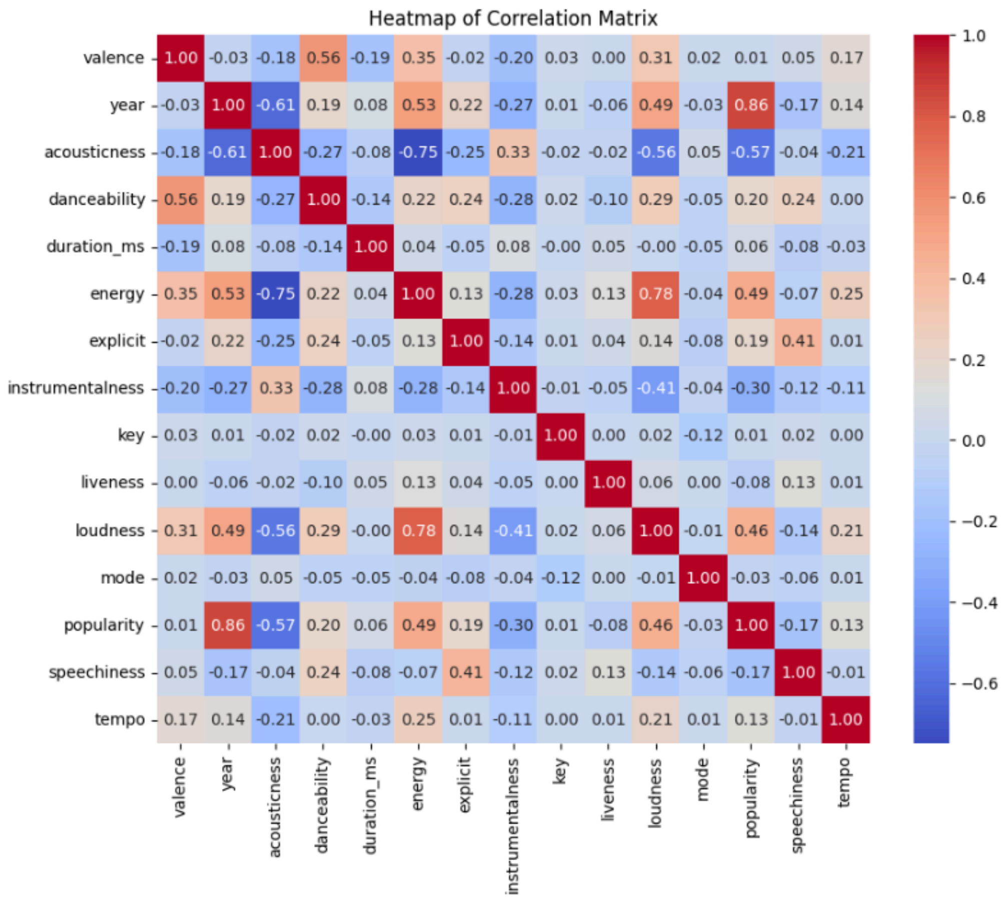
The data used for this project consists of 4 datasets. The four datasets are broken down by artist, genre, year, and one combined with everything. The combined data consists of 170k rows and 19 columns, the genre data consists of almost 3k rows and 14 columns, the artist data consists of almost 29k rows and 15 columns, and the year data consists of 100 entries with 14 columns (starting at year 1921 and ending 2020).

Several key variables are used to characterize and recommend music. *Valence* measures the musical positiveness conveyed by a track, with high valence tracks sounding more happy and cheerful, while low valence tracks sound more sad and angry. *Danceability* describes how suitable a track is for dancing, based on elements like tempo, rhythm stability, and beat strength. *Energy* indicates the intensity and activity level of a track, with higher energy tracks feeling fast and loud, and lower energy tracks feeling slower and quieter. *Acousticness* measures the extent to which a track is acoustic, with higher scores indicating more acoustic tracks. *Instrumentalness* predicts whether a track contains no vocals, with higher values suggesting a higher likelihood of the track being instrumental. *Liveness* detects the presence of an audience in the recording, with higher scores indicating a higher probability of a live performance. *Speechiness* measures the presence of spoken words in a track, with higher scores indicating more spoken content, ranging from podcasts to rap music. Finally, *Tempo* refers to the speed or pace of a track, measured in beats per minute (BPM), with higher tempos indicating faster-paced tracks.

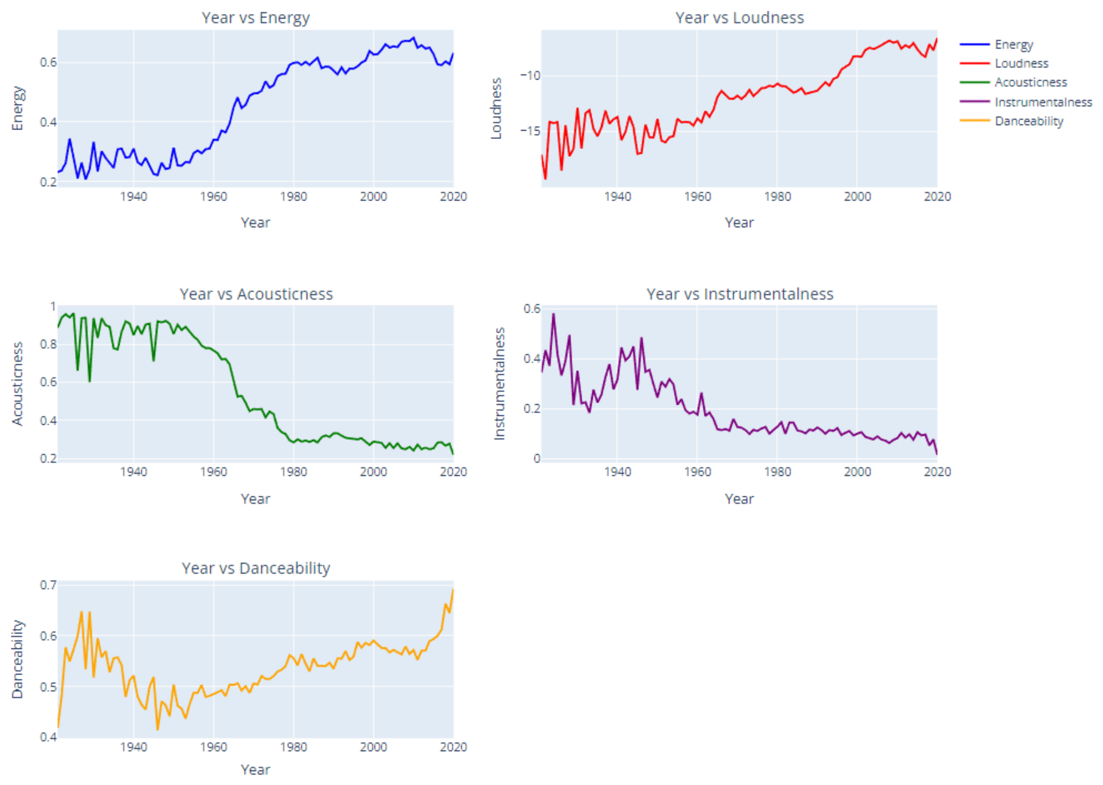
Methodology

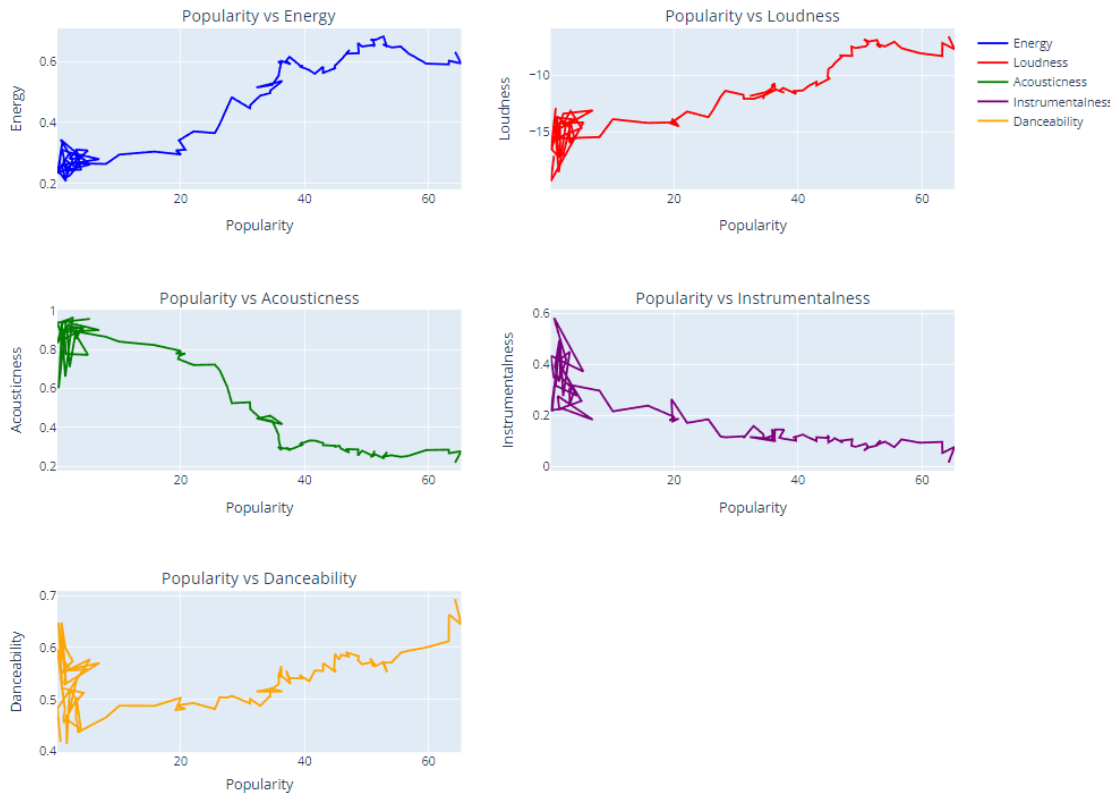
Initially, we attempted to create this API from scratch. To use the API, we would need to gather URLs of all the songs and extract data based on these URLs. However, that process proved to be too tedious, so we decided to use the Kaggle data instead.

We then performed exploratory data analysis on the data, including examining a correlation matrix to see how the music features relate to each other. This step was crucial to check for multi-collinearity among the features.

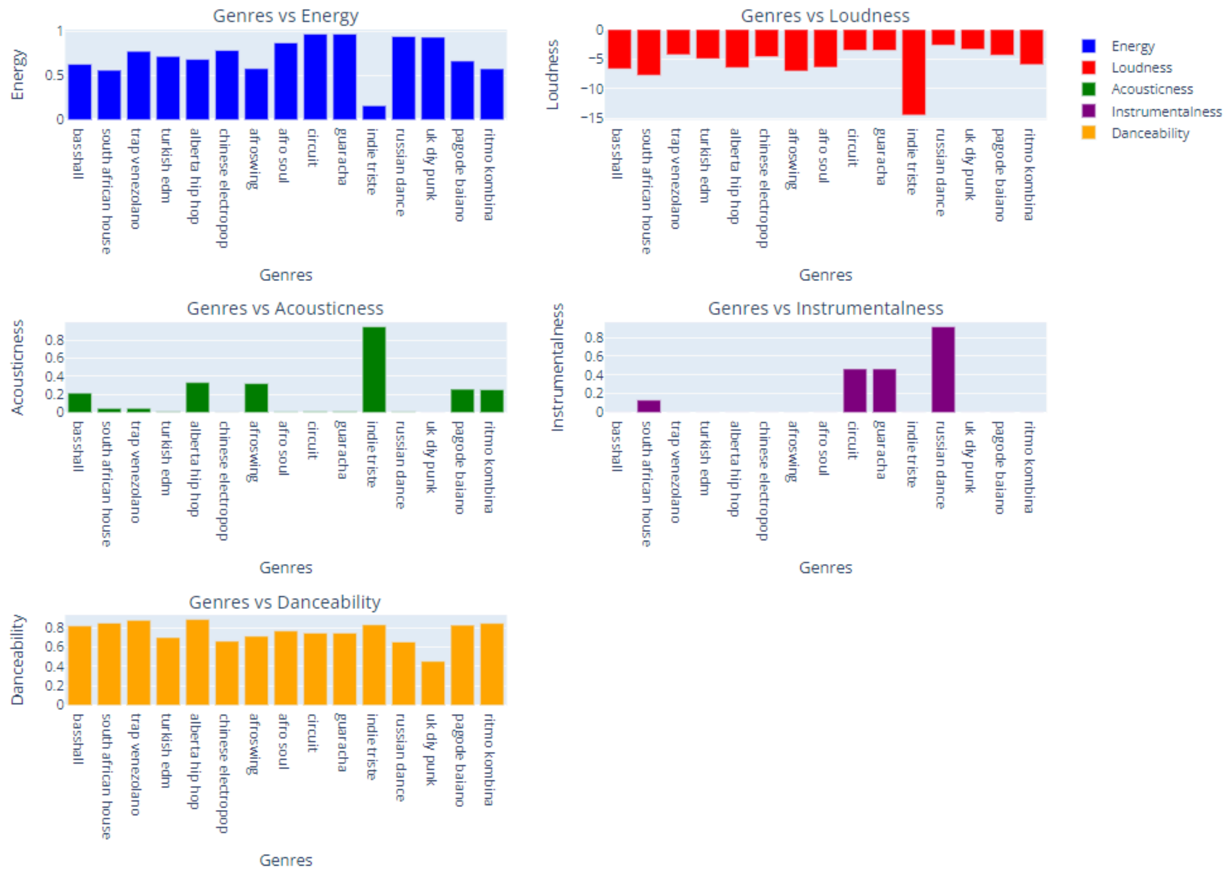


We also examined the trends of some of these features over the years to observe how musical tastes have evolved. Over time, there has been an increase in music that is higher in energy, loudness, and danceability. Conversely, there has been a decrease in acoustic and instrumental music. In the modern age, songs that are loud, high in energy, and highly danceable appear to be the most popular.

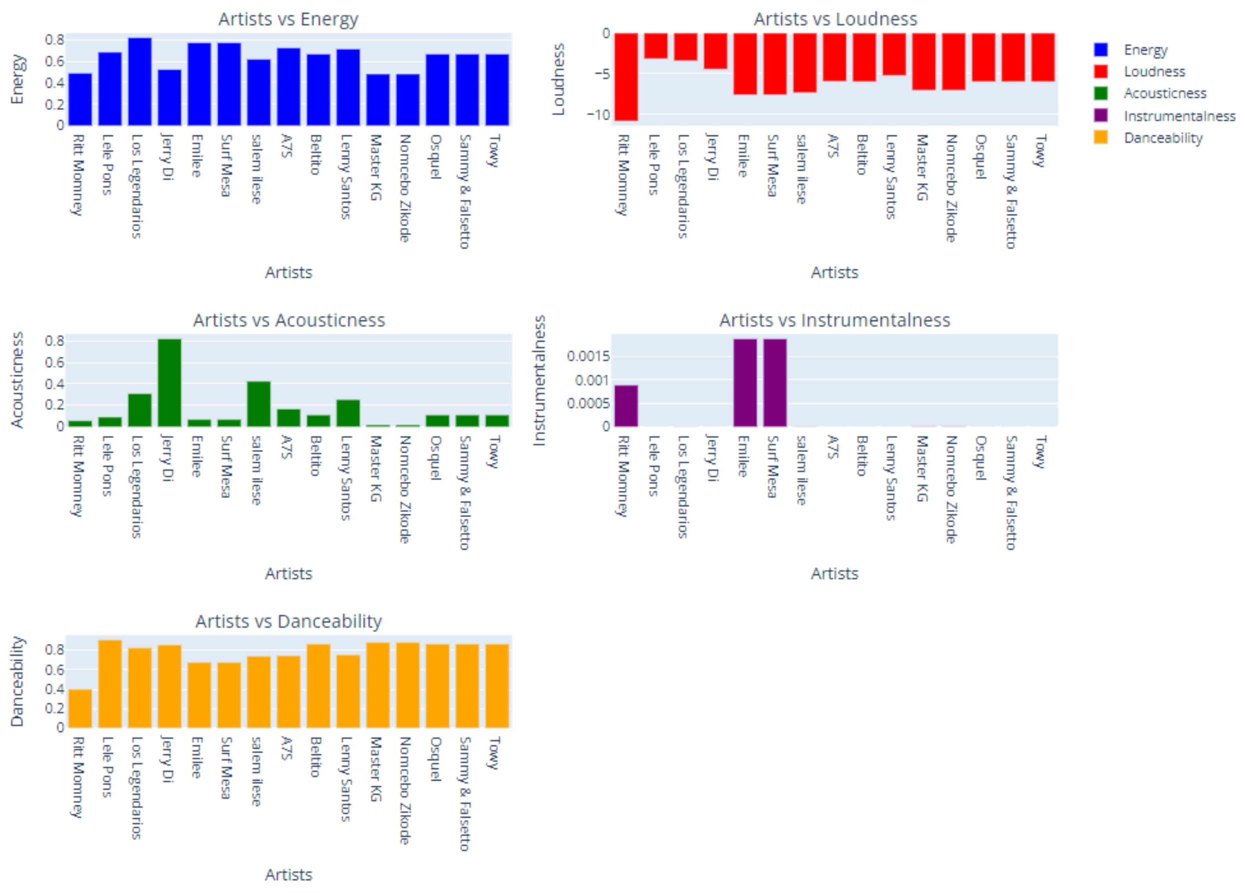

Year vs Energy
Energy
Year
Year vs Loudness
Loudness
Year
Year vs Acousticness
Acousticness
Year
Year vs Instrumentalness
Instrumentalness
Year
Year vs Danceability
Danceability
Year
1940
1960
1980
2000
2020
Energy
Loudness
Acousticness
Instrumentalness
Danceability


Popularity vs Energy
Energy
Popularity
Popularity vs Loudness
Loudness
Popularity
Popularity vs Acousticness
Acousticness
Popularity
Popularity vs Instrumentalness
Instrumentalness
Popularity
Popularity vs Danceability
Danceability
Popularity
20
40
60
Energy
Loudness
Acousticness
Instrumentalness
Danceability

Additionally, we analyzed the features based on the top 15 most popular genres. We observed that most of these popular genres exhibit high danceability and energy. Conversely, few of them utilize acoustic and instrumental features extensively.



Next, we examined the top 15 most popular artists to identify the features they incorporate into their songs. Given that Spotify, launched in 2006, has a user base skewed towards younger generations (Millennials, Gen Z, and Gen Alpha), there is a stronger preference for artists who fit within those generations' taste. However, for all of these artists, their songs typically exhibit high danceability and energy as shown in the bar plots visualized below.

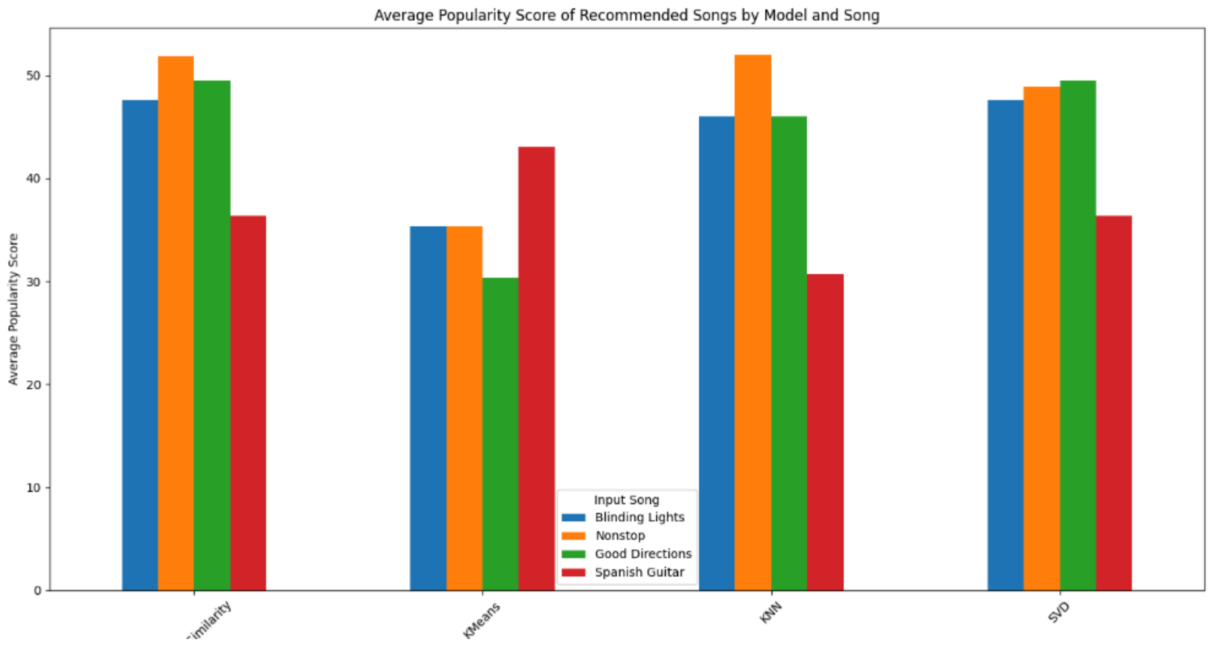


Machine Learning Algorithms

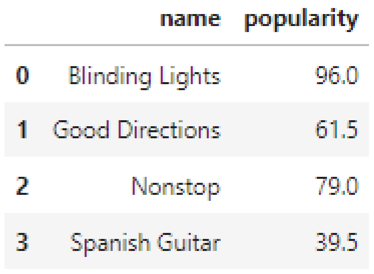
We evaluated four different machine learning algorithms for the recommendations: Cosine similarity, K-means clustering, KNN, and singular value decomposition (SVD). Cosine similarity measures the cosine of the angle between two non-zero vectors, determining their directional similarity regardless of magnitude, commonly used in text analysis and recommendation systems. K-means clustering partitions a dataset into K clusters by assigning each data point to the nearest centroid and iteratively updating centroids to minimize within-cluster variance, suitable for grouping similar data points. K-nearest neighbors is a non-parametric algorithm that classifies a data point based on the majority class among its k-nearest neighbors, determined by a distance metric such as Euclidean distance. Singular value decomposition decomposes a matrix into three component matrices, capturing essential data patterns in a reduced-dimensional form, often used in dimensionality reduction and noise reduction. We applied these algorithms to four songs: Blinding Lights, Nonstop, Good Directions, and Spanish Guitar. We selected these four songs because they are distinctly different from each other, representing a variety of genres. Choosing songs from similar genres could introduce bias, as they would likely share similar feature scores. By selecting diverse songs, we ensured a more balanced and comprehensive evaluation of our recommendation algorithms. For each algorithm, we selected the top five recommended songs and used their popularity as an evaluation metric to assess the success of our recommendations. Given the subjective nature of music preferences, determining the effectiveness of recommendations can be challenging. Therefore, we chose popularity as a fair and objective evaluation criterion.

Evaluation of Results

Based on the average popularity score of the recommended songs, we observed that Nonstop consistently has the highest popularity score, while Spanish Guitar has the lowest, on average. However, when examining the original songs' popularity scores, Blinding Lights holds the highest popularity score, and Spanish Guitar the lowest. This suggests that the recommended songs should align closely with the original songs' popularity scores to ensure the recommendations are effective and relevant.

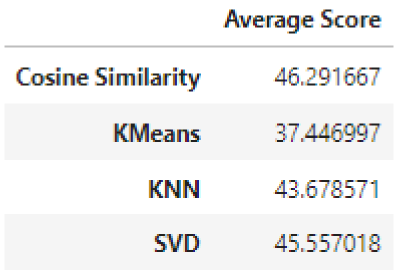


However, to determine the most effective algorithm for our recommendations, we calculated the average popularity scores for the recommended songs across the four algorithms. The dataframe below lists the algorithms and their corresponding average popularity scores. Based on these results, cosine similarity achieved the highest average score, while K-means clustering had the lowest. It is important to note that the cosine similarity measure typically has low bias, but also high variance. This is similar for the singular value decomposition, while KNN's variance and bias level is based on the number of nearest neighbors we set the algorithm to. K-means clustering would have higher variance due to initial centroid placement and data sensitivity, but it would also have moderate to high bias.



	name	popularity
0	Blinding Lights	96.0
1	Good Directions	61.5
2	Nonstop	79.0
3	Spanish Guitar	39.5

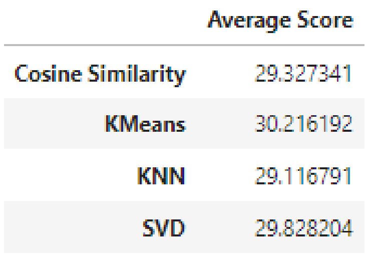
Original Popularity score for the songs



	Average Score
Cosine Similarity	46.291667
KMeans	37.446997
KNN	43.678571
SVD	45.557018

Average score of the 4 songs that were picked initially

When we tested our data on a sample of 100 songs (set seed = 12 for reproducibility), we found that the average scores were very similar. In this case, K-means produced the best results, while KNN performed the worst. The choice of algorithm depends on the project's direction. If the goal is to prioritize explainability, using an algorithm like cosine similarity would be ideal. However, for user satisfaction, selecting an algorithm that favors more popular songs is crucial, making K-means the most effective choice in this scenario.



	Average Score
Cosine Similarity	29.327341
KMeans	30.216192
KNN	29.116791
SVD	29.828204

Average score of the 100 songs that were chosen randomly (seed = 12)

Challenges and Future Recommendation

Formulating a robust music recommender posed many challenges ranging from data-related issues to algorithmic considerations and metrics of evaluation. High quality data is of paramount importance for this task. While the Spotify API serves as a powerful tool for accessing a vast library of music metadata, there were many limitations that came with its use. Rate limits, metadata inconsistencies, and the extremely manual and inefficient process of building queries to gather enough data proved the data sourcing method to be inappropriate given the scope of this project. A large proportion of our time and efforts were dedicated towards locating public datasets with rich data and all the auditory features we needed.

Once we obtained the data that met our standards, we ran into the challenge of selecting the right type of model that could efficiently handle large volumes of data. We attempted building many of the algorithms that were covered in this class, making sure to optimize each model with hyperparameter tuning for optimal complexity and best results.

Since our generated algorithms use unsupervised machine learning, choosing the proper evaluation metrics was challenging. Without actual user data such as ratings, there was no way to compute the relevancy of the recommended songs to test the accuracy of the models. We considered collecting user feedback for each recommender, but determined that to be outside the scope of this assignment. Instead, popularity was used as a proxy for assessing the quality of the recommended songs.

Given the limitations of our approach, we believe it serves as a good starting point for a machine learning music recommender. In the future, we recommend finding a way to fully utilize the Spotify API and implementing user studies to build a much more powerful music recommender.