Exploring Spotify Dataset Using Clustering and MIPS

Allan Bishop
adb262@cornell.edu

Rashmi Sinha
rs2584@cornell.edu

Abstract

It has long been a human task to choose the next song in a playlist, but modern algorithms have reached a new level of sophistication in this domain. In this paper we explore a practical real life dataset of Spotify songs. We analyzed the relationships and the patterns occurring between various songs by applying numerous clustering techniques such as k-Means, OPTICS and DBSCAN. In the first part of our experiments we cluster songs based on their predefined attributes with the naive assumption that genres should cluster together. Next, we implement MIPS and ScaNN algorithms to generate a playlist from a list of seed songs. Although modern recommendation systems use machine learning and complex analysis of song attributes, as well as user history, we find that ScaNN and MIPS are able to successfully (although primitively) suggest music of similar taste.

1 Introduction

With the world being engulfed by the COVID-19 pandemic, content consumption has been on the rise. As music has been an even more important part of people's lives throughout the pandemic, we decided to delve deeper into the Spotify dataset [5] and analyse the various patterns present within the data. This dataset has been created using the Spotify API and consists of a total of 232,725 tracks. The original dataset used for this project was a different dataset [6]. After a short period of working with this, we decided to expand to a more feature extensive and better labelled dataset [5]. With this new set, we are able to make our naive assumptions of clustering based on genres. We quickly learned that although genres are intended to define the attributes of a song, they can often have too much variation to be considered clusterable as a whole. k-Means, OPTICS and DBSCAN were all employed in an attempt to accurately cluster our songs. Figure 1 demonstrates the vagueness of the genre classification and the natural error that is present with these algorithms.

2 Data Preprocessing

We used a practical dataset consisting of songs by thousands of artists belonging to different genres. The features present in this dataset are genre, artist name, track name, track id, popularity, acousticness, danceability, duration, energy, instrumentalness, key, liveliness, loudness, mode, speechiness, tempo, time signature and valence. As we are interested in clustering and making recommendations only based on the song's audio features, we limited our analysis only to columns with relevant numeric features. The final columns chosen for our analysis are - popularity, acousticness, danceability, energy, instrumentalness, liveness, loudness, speechiness, tempo, valence. The observed dataset had a total of 26 unique genres.

On analyzing the dataset further, as displayed in Figure 1, we observed that a lot of songs had been repeated throughout the dataset under different genres. For example Major Lazer's Lean On song is categorized under Rap, Pop and Dance genres and hence repeated thrice throughout the dataset. We wanted to analyze this overlap throughout the dataset and see how many of these instances have occurred across genres. We see that the overlap between the genres ranges from 0% to 50%. The overlap indicates that the 2 genres are correlated as a number of the songs from that genre are recurring in the other genre. On observing the scale of the data, we decided on a threshold of 15% as the appropriate value to indicate whether 2 genres are well correlated. Hence if the overlap of songs between 2 genres was more than 15%, we merged them into a single parent genre. We used Depth First Search to find the connected components to determine which clusters were to be merged in a single parent cluster. We also removed the duplicate songs from the dataset, to reduce the data overhead. For the songs which belonged to genres with less than 15% overlap, we randomly chose one of the genre labels for the song. Additionally we removed the movie genre as we believed that this genre can have unrelated songs having any numeric feature and would be irrelevant for the clustering algorithms. We did use it for the recommendation algorithms though.

CS 5112 Final Report. Fall 2020.

The final preprocessing step involved a scaling of the data. Since some features ranged from 0...100 while some ranged from 0...1, we had to normalize the data. This is especially important when applying distance based clustering methods such as k-Means. If we do not process the data, some features will be much more important for the clustering attribution than others. We used Sklearn's MinMaxScaler to get all of our features between 0 and 1.

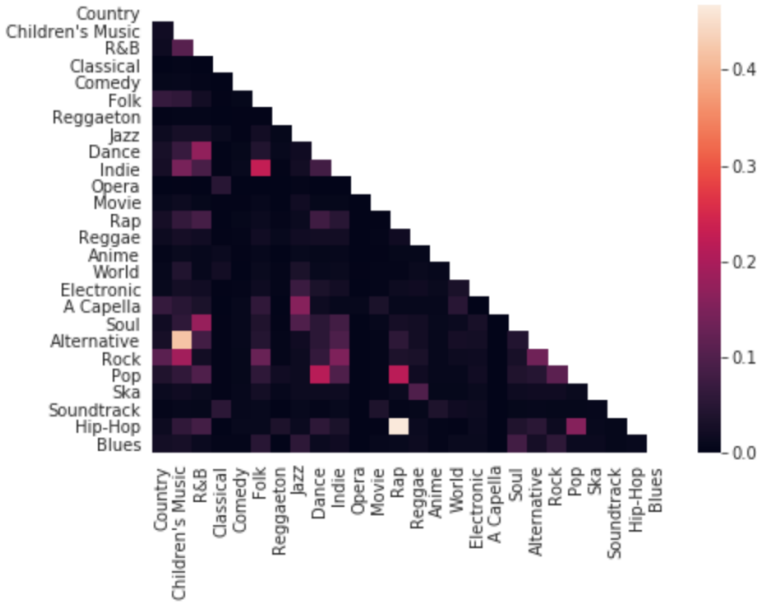


Figure 1: percent of crosslisted songs by genre

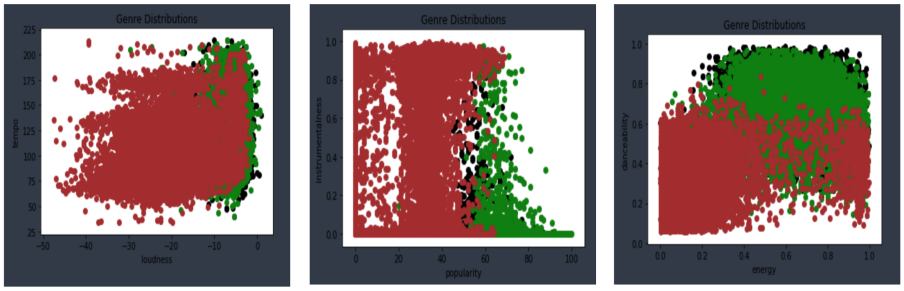


Figure 2: Pop, Hip-Hop, Classical attribute distributions

3 Experiments

3.1 Clustering

We conducted experiments using multiple clustering algorithms to observe which one performed the best. We used k-Means, DBSCAN and OPTICS. Throughout the clustering algorithms, we made the assumption that songs belonging to the same genres are similar and hence should be clustered together. We used the genres obtained in the Data Preprocessing step as the baseline clusters. We then compared the clustering results of the below algorithms with those baseline clusters using the rand score metric. All of these clustering algorithms were run using the Sklearn library implementations.

k-Means k-Means is a distance based clustering algorithm that is guaranteed to converge on any data set, although not always with the optimal clustering results. The process in itself is fairly simple - (1) randomly assign centroids, (2) assign points to the centroid they are closest to (by Euclidean distance), (3) calculate the mean of the clusters and assign them as new centroids. Steps 2 and 3 are repeated until the centroids no longer change from iteration to iteration. Since k-Means allows us to deterministically assign the number of clusters, (n_clusters) is intuitive. When analyzing n different genres we assigned the initial number of clusters to be n.

DBSCAN Density Based Spatial Clustering of Applications with Noise (DBSCAN) is exactly as the name declares - a density based algorithm to determine clusters [1]. DBSCAN was created to solve clustering where regions of high density are the same class. There are two key parameters in DBSCAN that determine the clusters - Epsilon and MinPts. Epsilon represents the maximum distance in which two points could be considered the same cluster while MinPts is the minimum number of points which are required to declare a grouping a 'cluster', otherwise the points are labelled as outliers. DBSCAN has proven to be poor at handling distributed clusters and groupings of varying densities.

OPTICS Ordering Points To Identify the Clustering Structure (OPTICS) is a clustering algorithm to find density based spatial clusterings. It is extremely similar to the DBSCAN algorithm save for a single improvement. OPTICS actually allows us to detect clusters of varying density distributions within our data [2]. Like DBSCAN, OPTICS uses Epsilon and MinPts as it's two parameters to determine cluster assignments.

3.1.1 Experimental Results

The data is extremely dense and thus creates certain logistical problems for algorithms like OPTICS and DBSCAN (density based algorithms). As a result, in order to test the effectiveness of these three clustering algorithms on our data, we worked with genre specific subsets to try to identify clusters. The first subset we tested was Folk, Pop and Jazz, three seemingly unrelated genres by sound. When we use the full feature vector for k-Means with k=3, we get an adjusted rand score of 0.138. However, we want to analyze the individual feature correlations and how they form clusters. So, we ran k-Means (k=3), OPTICS and DBSCAN (MinPts = [50, 100, 200, 400, 800, 1000]) on all unique permutations of 2d feature combinations. k-Means was not only the most efficient, but was also the only algorithm that yielded rand scores above 0.1!

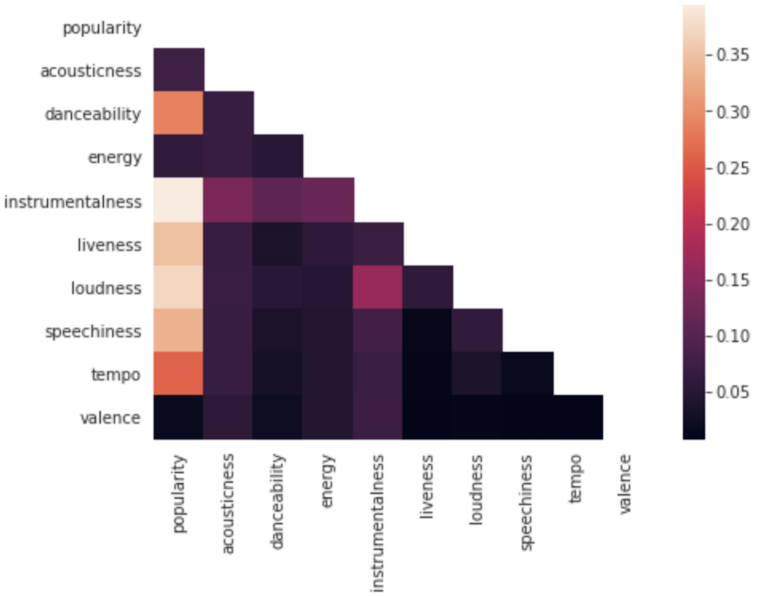


Figure 3: Pop, Folk, Jazz top 2d feature pairings by rand score

Figure 3 shows the relationship each feature held with the k-Means rand score. Popularity and instrumentalness yielded the highest rand score of 0.4. We assume this combination of features to be near optimal for our clustering. Figure 4. shows these genres plotted on this 2d plane. As you can see, these genres are naturally clustered fairly tightly, even for the most differentiating features! Let's try a different combination of genres.... Hip-Hop, Children's Music, and Opera. These should be differentiable, right? k-Means achieves the best rand score when using the whole feature set with a score of 0.52. We investigate further to find the most important features (Figure 5).

As we dig deeper into the underlying features, we see that k-Means again yielded popularity as one of the most differentiating features. Popularity vs danceability achieves the highest performance on k-Means (k = 3) with a rand score of 0.57. The highest rand scores for OPTICS and DBSCAN (MinPts = [50, 100, 200, 400, 800, 1000]) were 0.24 and 0.11 respectively with MinPts = 50. k-Means has proven to be far and away the most effective clustering technique for this specific task, both in high and low dimensions. DBSCAN and OPTICS both struggled to classify points as anything other than outliers. DBSCAN received a rand score of < 0.1 on every single MinPts and feature combination. OPTICS did slightly better, although it was not much worth noting. Figure 7 shows the diminishing rand score the OPTICS algorithm received as MinPts was scaled (epsilon = 0.1). Note -

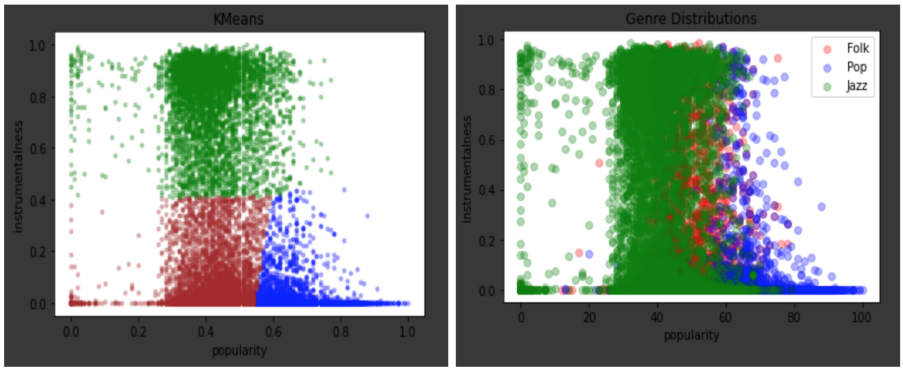


Figure 4: Pop, Folk, Jazz | k-Means (k=3) result (left) vs actual (right)

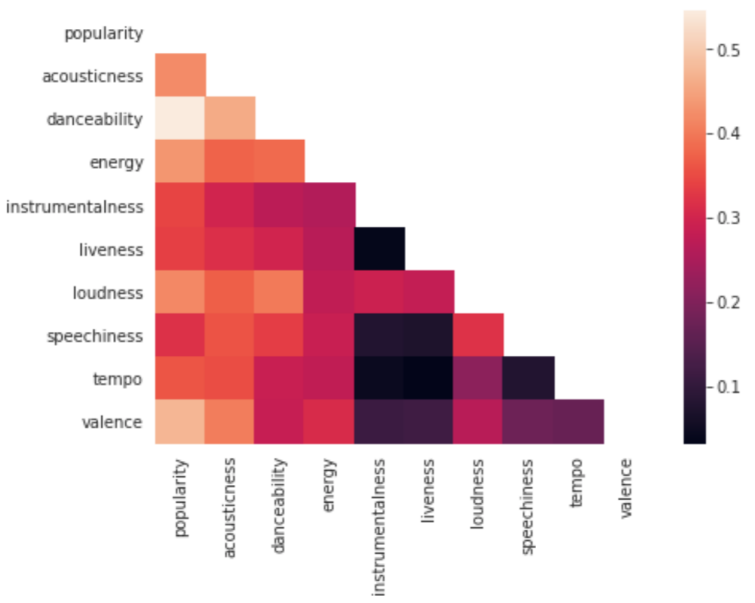


Figure 5: Hip-Hop, Children's Music, Opera top 2d feature pairings by rand score

Epsilon could be scaled further to optimize the OPTICS algorithm but, regardless, it will not have 3 clusters. Figure 6 displays the density of the groupings and the inseparability of the data by genre, even on the optimal 2d plane. So far we have been trying to separate 3+ genres, but what happens when we limit it to two genres? The best results are illustrated by vastly different genres - say Hip-Hop and Opera. Using the entire feature set, the genres are easily differentiable. k-Means (k=2) achieves a rand score of 0.9901, while OPTICS and DBSCAN struggle to find the clusters. When we view the data on a 2d plane something changes... Density based algorithms excel at this task. Figure 8. shows the results of the optimal parameters for each algorithm on the two genre 2d data. OPTICS and DBSCAN, when tuned correctly, blow k-Means away with rand scores of 0.992 and 0.991, respectively. k-Means' best 2d rand score is only 0.935, showing cracks in the distance based clustering method after all.

3.2 Recommendation Algorithms

After clustering our dataset, we wanted to implement the recommendation algorithms we learned in class. We observed how the recommended songs were relevant to the initial seed song with respect to numeric and qualitative metrics. We also wanted to use the algorithms to generate a recommended playlist for a user.

MIPS Maximum Inner product Search (MIPS) is identified as an important task pertinent to recommendation systems and high dimensional similarity search. One of the main issues with MIPS is that it takes a long time for high dimensional data. For our scenario, we implemented the MIPS algorithm from the SITQ library [4], as it is a fast algorithm for approximate Maximum Inner Product Search. It maximizes the inner product against a query in sublinear time. This implementation of MIPS combines

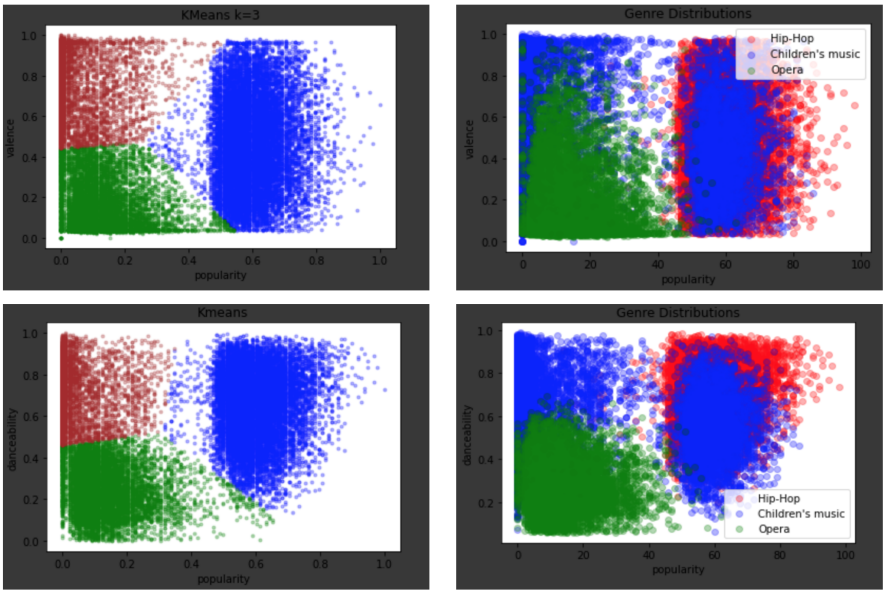


Figure 6: Hip-Hop, Children's Music, Opera | k-Means (k=3) result (left) vs actual (right)

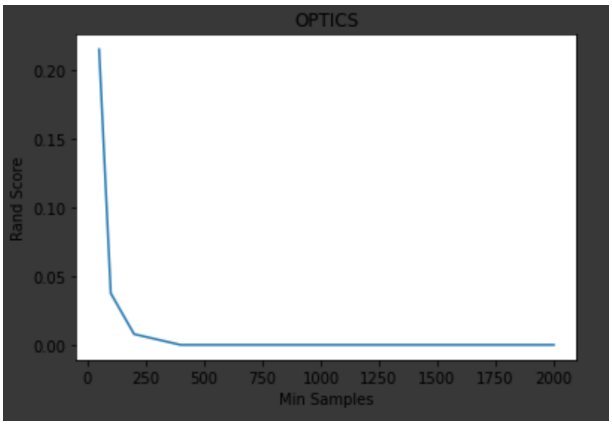


Figure 7: Hip-Hop, Children's Music, Opera [popularity vs danceability] OPTICS diminishing rand score

Simple-Locality Sensitive Hashing and Iterative Quantization. The Simple LSH utilizes ordinary LSH which is for cosine similarity. The LSH algorithm computes the signature through a transformation matrix which is fixed. The Iterative Quantization learns the transformation matrix from the item vectors for better hashing. For our algorithm, we chose the signature size of 4. The issue we observed on using this was that the recommendations generated were almost the same. The results improved significantly upon normalizing the data using MinMaxScalar.

ScaNN ScaNN is an efficient vector similarity search algorithm with a new quantization approach for MIPS. It utilizes an anisotropic vector quantization approach and has the ability to trade increased quantization error of lower inner products in exchange for superior accuracy for high inner products [3]. This algorithm is known to result in better performance gains. We applied this algorithm on our dataset and it resulted in having relevant recommendations for various seed songs.

3.2.1 Experimental Results

We ran our recommendation algorithms for 10 popular songs to evaluate various metrics. As observed in the Table1, we see that MIPS takes a lot more time to fetch the recommendations than the ScaNN algorithm.

We also compared the numeric metrics of popular songs with their corresponding recommendations. We compared the metrics of the inputted query song with the average of the metrics of the songs recommended by the MIPS and ScaNN algorithms

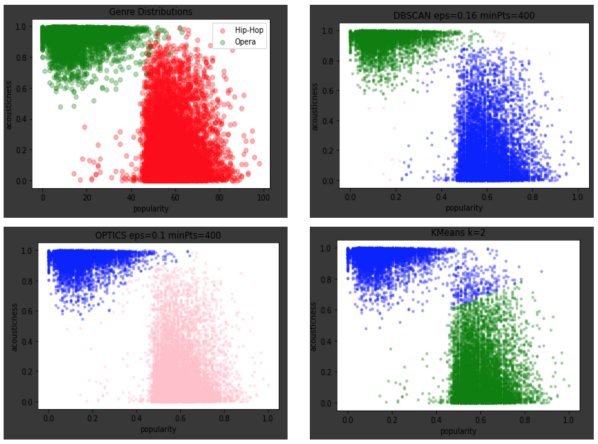


Figure 8: Hip-Hop, Opera 2d clustering results

Artist-Song	MIPS (ms)	ScaNN (ms)
Luis Fonsi - Despacito	6.178855895996094	0.7758140563964844
Major Lazer - Lean On	5.621910095214844	1.5647411346435547
Ed Sheeran - Shape of You	3.6122798919677734	0.8497238159179688
Krewella - Alive	6.412029266357422	1.9831657409667969
Sia - Chandelier	5.192756652832031	1.1336803436279297
Katy Perry - Firework	3.5359859466552734	1.0194778442382812
Kanye west - Power	6.510257720947266	1.4319419860839844
Nicki Minaj - Only	6.522893905639648	0.9167194366455078
SOB X RBE - Paramedic	5.819082260131836	2.8107166290283203
Kanye West - Amazing	5.456209182739258	1.7673969268798828

Table 1: Table to compare latencies of MIPS and ScaNN algorithms on popular songs

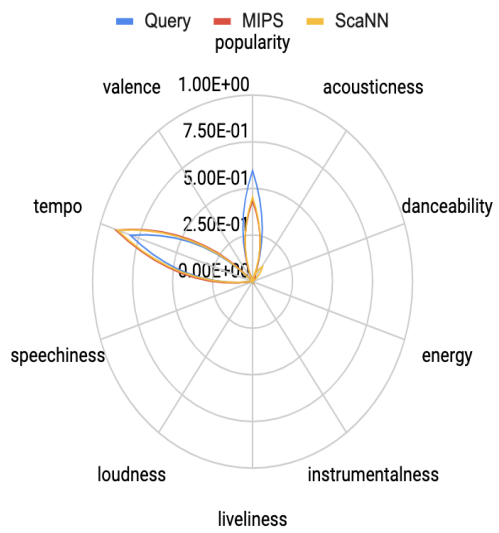
respectively. As observed in Figure 9b, the MIPS and ScaNN algorithms have both resulted in very close recommendations to the original songs in terms of numeric metrics. In the rest of the cases, ScaNN results in slightly better results as compared to MIPS but not significantly better. We also decided to do the qualitative analysis on our recommendations to evaluate whether we as a consumer would be satisfied if we received those results. Hence both of us picked up 5 songs each from our personal favourite songs and evaluated the recommendations quality served to us by the algorithms. We marked the recommendation as green if we liked the it, and red if we did not like it. Figure 10 shows a sample of the qualitative analysis done by Rashmi for her recommendations. Figure 11 shows a sample of the qualitative analysis done by Allan for his recommendations.

As a part of this project, we wanted to generate a playlist based on the seed favourite songs. Hence we each chose our seed favourite songs to generate the playlist. For every seed song, the MIPS and ScaNN generated recommendations. We picked the top 2 recommended songs by the score provided by the MIPS and ScaNN algorithms for each of the seed song. Finally we removed duplicate songs across the recommendations. Table 2 is the playlists generated for each of us by MIPS. Table 3 is the playlists generated for each of us by ScaNN.

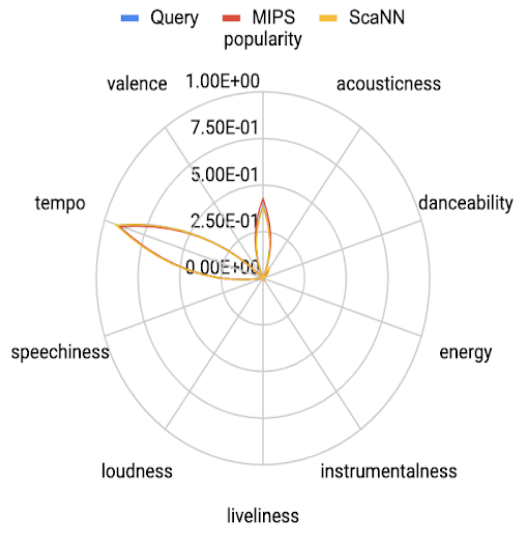
4 Discussion

4.0.1 Clustering

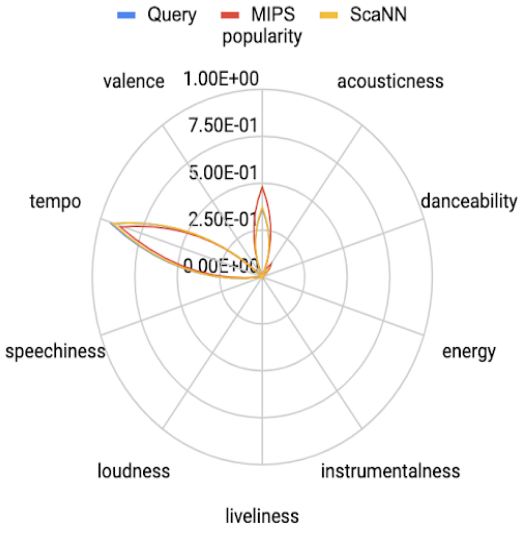
k-Means proved to be far and away the best clustering algorithm for multi-genre (3+) song feature analysis. It is not only the fastest algorithm to converge, but it also delivered the highest rand score by comparison. Although the groupings in 2d and high dimensional space were extremely blurred, k-Means was able to deliver a decent suggestion for where the genre boundaries should be. The k in k-Means certainly differentiates this algorithm because it gives us the ability to tell the method how many clusters there should be. In comparison to OPTICS and DBSCAN, k-Means is guaranteed to have the exact number of clusters necessary for the genre distribution. However, when we compare only two genres directly in a 2d space, density based algorithms



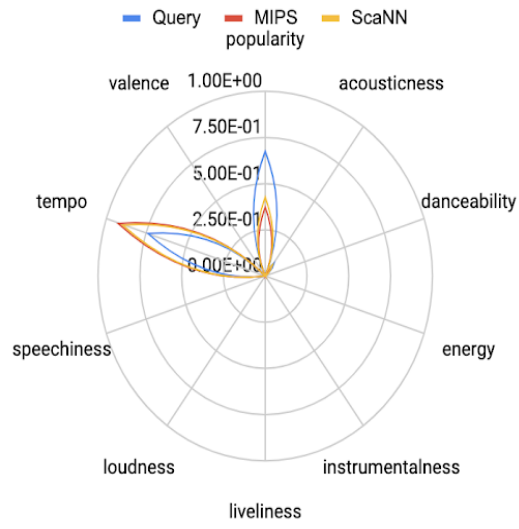
(a) Major Lazer - Lean On



(b) Krewella - Alive



(c) Nicki Minaj - Only



(d) Kanye West - Amazing

Figure 9: Radar Graphs of original query features versus average of recommendations from MIPS and ScaNN algorithms

Rashmi's recommended playlist	Allan's recommended playlist
Carly Simon - Anticipation	D-Enyel - Decisiones
Atlantic Five Jazz Band - The Things We Do For Love	Atlantic Five Jazz Band - The Things We Do For Love
Benny Sings - Duplicate	Andrew Belle - Black Clouds
Chris Brown - Questions	Harry Styles - Ever Since New York
Rubén Blades - Amor y Control - Live	The National - All the Wine
BADBADNOTGOOD - Street Knowledge	Foy Vance - Into the Fire

Table 2: Recommended playlists generated by MIPS algorithm

like OPTICS and DBSCAN outperform the simple Euclidean distance based k-means technique. They yield a method for clearly differentiating genres in a low dimensional space.

Query Song - Shape of You - Ed Sheeran [Pop]

MIPS

Atlantic Five Jazz Band - The Things We Do For Love [Jazz]
Benny Sings - Duplicate [Jazz]
D-Enyel - Decisiones [Reggaeton]
Litany - Call On Me [Indie, Soul]
Oceanvs Orientalis - Television [Electronic]

ScaNN

Wolfgang Amadeus Mozart - Concerto No. 2 in E-Flat Major, K.417: I. Allegro [Classical]
M.I.A. - Paper Planes [Dance, Pop]
BTS - Lie [Pop]
Scatman Crothers - I'm Gonna Rock & Roll [Movie]
Giacomo Meyerbeer - Les patineurs: Pas de deux (arr. C. Lambert on music from Le prophète and L'etoile du nord) [Opéra]

Query Song - Krewella Alive [Dance]

MIPS

Andrew Belle - Black Clouds [Folk]
Chris Brown - Questions [Dance, Pop, Rap, R&B]
Drake - Redemption [Hip-Hop, Pop, Rap]
Rihanna, Hate That I Love You [R&B, Dance, Pop]
Beyoncé - Countdown [R&B, Dance, Pop]

ScaNN

Shakey Graves - Tomorrow [Folk, Indie, Pop]
Demi Lovato - Heart Attack [Dance, Pop, R&B]
Felix Mendelssohn - Violin Concerto in E Minor, Op. 64, MWV O 14: I. Allegro molto appassionato [Classical]
No Use For A Name - Friends of the Enemy [Ska]
Rival Sons - Feral Roots [Alternative, Blues, Children's Music]

Figure 10: Qualitative analysis on recommendation results by Rashmi

4.0.2 Recommendation Algorithms

We have observed in the experimental results section that ScaNN algorithm is a lot more quicker at getting recommendations than the MIPS algorithm. Also, with respect to the numerical metric analysis, ScaNN yields results which are numerically slightly closer to the query song as compared to MIPS. The qualitative analysis results vary depending on the person as we observed that Rashmi preferred recommendations served to her by MIPS (Figure 10) while Allan seemed to prefer recommendations served to him by ScaNN (Figure 11).

5 Conclusion

In this paper we have presented and analyzed existing clustering and recommendation techniques and their respective applications in the music industry. We have seen how dense and variable song features are, and that a genre is not necessarily indicative of a songs features. Genres merely provide vague boundaries for specific features from which many songs diverge and still hold the classification of the genre. A telltale sign of the extensive diversity of songs within a genre is the number of songs with multiple labelled genres. This indicates that a song falls under many different classifications with a single data point. This can only be true if there is significant overlap in the features of different genres.

In 2d space, it is near impossible to directly discern multiple genres from each other. However, when there were boundaries to be found, k-Means, OPTICS, and DBSCAN can be tuned to fit the task. As expected, k-Means significantly outperformed the density based clustering algorithms in high-dimensional space because of the vast density and distribution of the data. However, the latter two algorithms were able to outperform k-Means on certain distributions of data. Hence we realized that it is extremely important to analyze the nature and essence of the data to perform clustering algorithms.

Nicki Minaj - Only [Hip-Hop]

MIPS

Harry Styles - Ever Since New York [Dance, Pop]
The National - All the Wine [Folk, Indie]
Pyotr Ilyich Tchaikovsky - Swan Lake, Op. 20, Act III No. 17: Act III In the Castle of Prince Siegfried: A Ball at the Castle: No. 17. Arrival of the guests and waltz Allegro - Tempo di valse [Classical]
Ashley Tisdale - Love Me & Let Me Go [Dance, Pop]
Nelly Furtado - I'm Like A Bird [Dance, Pop, Rap, Rock

ScaNN

Tyler, The Creator - She (Featuring Frank Ocean) [Hip-Hop, Pop, Rap]
Big Mama Thornton - Sweet Little Angel [Blues]
James Gang - The Bomber A: Closet Queen B: Bolero C: Cast Your Fate To The Wind [Blues]
Blackbear - idfc - Tarro Remix [Pop, Rap]

Kanye West - Amazing [Pop]

MIPS

Andrew Belle - Black Clouds [Folk]
Nitti Gritti - Blame on Me [Electronic]
Maná - Oye Mi Amor [Pop, Rock]
Beyoncé - Countdown [R&B, Dance, Pop]
Chris Brown - Questions [Dance, Pop, Rap, R&B]

ScaNN

Flo Rida - Whistle [Dance, Hip-Hop, Pop, Rap]
Machine Gun Kelly - LIVEFASTDIEYOUNG [Hip-Hop, Rap]
Ellie Goulding - On My Mind [R&B, Dance, Pop]
Adolphe Adam - Giselle / Act 1: No. 8a Final [Opera]

Figure 11: Qualitative analysis on recommendation results by Allan

On the recommendations algorithms front, we observed that we need to normalize the data to use the MIPS algorithm from the SITQ [4] library to achieve relevant results. We also found that ScaNN was much faster than MIPS. Hence, for time sensitive applications where instant response is crucial, it would be a better to use ScaNN algorithm over MIPS due to the tremendous difference. Also, as observed through the qualitative and quantitative results analysis, although ScaNN resulted in slightly superior numerical results, it does not imply that the recommendations would actually be relevant for the person. There are more important features required apart from just the numerical metrics of the songs such as user history, other people's preferences who listen to similar music, user geography, etc. Such information can help us predict more accurate and relevant songs for the users. This is why many modern recommendation systems use supervised learning and massive datasets to strengthen their results.

Rashmi's recommended playlist
Mudvayne - Forget to Remember
Run The Jewels - Banana Clipper
KALEO - Way Down We Go - Recorded at Spotify Studios NYC
Fugazi - I'm So Tired
Wolfgang Amadeus Mozart - Concerto No. 2 in E-Flat Major, K.417: I. Allegro
M.I.A. - Paper Planes
Shakey Graves - Tomorrow
Demi Lovato - Heart Attack
Jim Carmichael - When I See An Elephant Fly - From "Dumbo"/Soundtrack Version
Craig Ferguson - I Don't Do Drugs

Allan's recommended playlist
Travis Scott - goosebumps
Juice WRLD - I'm Still
Tyler, The Creator - She (Featuring Frank Ocean)
Big Mama Thornton - Sweet Little Angel
Mike WiLL Made-It - Midnight (with Tessa Thompson & Gunna)
All - Gnutheme
Flo Rida - Whistle
Chorus - Sleeping Beauty - Final (La Belle au Bois Dormant)
Machine Gun Kelly - LIVEFASTDIEYOUNG

Table 3: Recommended playlists generated by ScaNN algorithm

References

[1] Ester et al. DBSCAN - https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[2] Ankerst et al. OPTICS - https://www.dbs.ifi.lmu.de/Publikationen/Papers/OPTICS.pdf

[3] ScaNN- https://ai.googleblog.com/2020/07/announcing-scann-efficient-vector.html

[4] MIPS with SITQ - https://pypi.org/project/sitq/

[5] Primary Dataset - https://www.kaggle.com/zaheenhamidani/ultimate-spotify-tracks-db

[6] Previous Dataset - https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks

[7] Github Repo - https://github.com/adb262/5112_final