Clustering on a Music Artists Network

Dylan Lewis [†] **William Torous** [†]
{drlewis, wtorousa}@mit.edu

Abstract

Due to the rising popularity of music streaming services and organized efforts by industry leaders and researchers for large-scale data aggregation, the music industry is no exception to the big data revolution. In this work, we construct a network from music artists features, streaming features, and computed similarity measures between the artists in the network. We investigate various featurization schemes and clustering algorithms in order to uncover clusters in the data and see how the clustering results compare between the clustering algorithms and their ability to cluster on artists by genre.[1]

1. Introduction

We examine interaction networks for music artists whose work appears on streaming services such as Spotify. More specifically, we seek to uncover a clustering of artists by genre using a graph with artist-specific features as node attributes and computed features of similarity between artists as edge attributes.

The data we will use to construct these graphs and answer these questions comes from the Million Song Dataset[2] (MSD), a large-scale dataset of music metadata provided by Columbia. It assigns a unique ID to one millions songs which appear in streaming and classification datasets provided by industry partners, allowing the data to be merged and for the derivation of other features.

We hypothesize that the nodes of the artist network will gather into densely connected clusters based on genre. There will therefore be sparse interconnections across genres using similarity between artists as edge features, as described in detail in the Datasets section. We expect artists who are not defined by one single genre in particular but multiple genres simultaneously to form the sparse links between genre clusters manifesting the inter-genre nature of their discography within the artist graph.

2. Related Work

2.1. Finding Clusters in Networks

In large networks such as social-media or phone networks, the number of nodes can be on the order of millions to billions (Blondel et al., 2008). In our case, we have constructed a small but dense graph with 97 nodes and $\binom{97}{2}$ edges consisting of 97 music artists. In order to find tightly coupled subunits of nodes in a large network, or clusters, there are some options for clustering algorithms to apply to the network. One such method is the Louvain method, which is a superior algorithm to other graph clustering algorithms in terms of computational efficiency and it yields comparably good results to the state-of-the-art graph clustering methods (Blondel et al., 2008).

A measure used to identify the quality of the clusters founds in networks is called modularity. Intuitively, this provides a measure in the range $[-\frac{1}{2}, 1]$ indicating how densely connected nodes within a cluster are compared to the the average edge density on a randomly generated graph with the same nodes and edge weights. This measure works in both the unweighted and weighted cases. It is defined as:

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{i,j} - P_{i,j}\right] \delta(c_i, c_j)$$

Here, $A_{i,j}$ is the weight of the edge between nodes i

[1]The code for this project can be found here: https://github.com/dyllew/18.065-fp

[2]http://millionsongdataset.com/

and j, m is the sum of weighted edges in the graph or $m = \sum_{i,j} A_{i,j}$, $P_{i,j}$ is the expected edge density in the null model or $P_{i,j} = \frac{k_i k_j}{2m}$, $k_i = \sum_j A_{i,j}$ or the sum of the weights of edges connected to node i, c_i is the community node i is assigned to $\delta(u,v)$ is 1 if $u = v$ and 0 otherwise. (Blondel et al., 2008)

Intuitively, the Louvain method begins by putting every node of the network into its own cluster or community. It then continues to put each node i into the cluster j, which leads to largest increase in modularity. The clusters found by the algorithm are then replaced by supernodes which are connected by weighted edges whose weight represents the sum of the edge weights between the nodes in each respective community or supernode. This process continues iteratively combining smaller supernodes into larger supernodes until the modularity cannot be improved. The advantages of this network clustering method over others is that it is very efficient (runs in $\mathcal{O}(n \log n)$, where n is the number of nodes in the graph) and provides clusters at different levels of granularity because of the inherent hierarchical clustering nature of the algorithm. We apply the Louvain method to the artist graph and use accuracy obtained for all artists by predicting genre on the basis of the plurality genre of each cluster found by Louvain for evaluating this method against K-means clustering.

2.2. Clustering Music Data

Analysis of music-information graphs is a popular topic for both class projects and academic research. The closest example to our own graphical analysis comes from Stanford graduate students in CS221 (Agrim Gupta, 2017). Using some of the same datasets as us, this group employed hard and soft clustering algorithms such as k-means and Gaussian mixture models to classify songs into clusters. Unlike our project, however, the clusters intentionally spanned multiple genres and were then used to train a recommender system. This system had 70% prediction accuracy, which speaks favorably to the descriptive power of our data. Additionally, this paper lead us to use mean song feature vectors from across an artist's discography, as described in 3.2.

In academia, music data more often appears as an application for technical graph theory results. The papers which focused exclusively on music graphs extended beyond the scope of our project, such as clustering with constraints on the Fourier Coefficients of a song (Wei Peng, 2007) and discovering how clusters of listeners influence each other and evolve over time (Schlitter & Falkowski, 2009).

3. Datasets

3.1. Data collection

Data for our analysis comes from the Million Song Dataset (MSD) (Bertin-Mahieux et al., 2011) and Spotify's Application Programming Interface (API) (Spotify API, 2021). Broadly, our features about user listening activity come from the MSD while features describing specific songs come from Spotify.

The MSD is a rich dataset of popular music prepared by researchers at Columbia. In a sense the MSD is a meta-dataset because it unifies data provided by several industry partners. As the name suggests, the million song dataset contains information about one million songs; each song is given a unique identifier `(MSD id)` which can be used to join the various data sources. Data shared to the MSD by The Echo Nest, a music data platform since acquired by Spotify, is integral to our analysis. The Echo Nest provides 48 million triples of the form (`user id, MSD id, number plays`) collected from an anonymous music streaming platform. Grouping these triples by `user id` allows us to recover the listening activity for individual users over an unspecified period of time. Each `MSD id` is also associated with a unique `artist id` in the MSD that allows us to determine how frequently and by whom an artist is streamed.

Although the MSD provides some qualitative features about individual songs such as genre or release year and quantitative features such as similarity with other songs, we chose to integrate Spotify's much richer API data. Through this API we associate each song with a list of genre tags and several scalar variables quantifying a song such as tempo, danceability, and energy. A full list of our song features is provided below. We utilized a mapping from songs' `MSD id` to Spotify API id provided by AcousticBrainz (AcousticBrainz, 2016).

Working with the full MSD is challenging because of the sheer size (300 GB) and logistics of requesting access. Instead, we chose to analyze a directly downloadable subset of the data with only 10,000 songs. According to the MSD, these 10,000 songs were selected at random from the full sample. Selecting on `MSD ids` in this subset reduced the number of streaming triples from 48 million to 700,000.

3.2. Music Artist Network

We construct a undirected graph consisting of a artists which are the nodes in the graph. This graph is a complete graph in which every artist shares an edge with the other $a - 1$ artists. This yields a complete graph consisting of a nodes and $\frac{a(a-1)}{2}$ edges.

Each artist has the following features:

- Artist Node ID
- Artist Name (string)
- Total number of song plays by users (int)
- Number of unique listeners (int)
- Total Plays / Number of Users a.k.a. Play Ratio (float)
- Track Frequency (Song ID: Total Number of Plays) (map)
- Top Listeners (User ID: Number of Plays of the artist by the user) (map)
- Mean Spotify song features (vector of floats)
- Genres (array – ordering has no significance)

The Spotify API provides 11 numeric features for each song in our dataset. These features are particularly useful because they assign numeric values to the qualitative features of a song. Eight of the features are floats in the range $[0, 1]$ which capture `danceability`, `energy`, `speechiness`, `acousticness`, `instrumentalness`, `valence`, and `liveness`. `speechiness` measures the amount of spoken word in a track, `instrumentalness` is higher for tracks without any lyrics, `valence` quantifies the happiness in a song, and `liveness` describes the likelihood a track was recorded before a live audience. Spotify also gives information about the `tempo` of a track, measured in beats per minute, the `time signature` in beats per bar, and the `key` using the pitch class mapping e.g. a song in any octave of D maps to 2.

As artists in our dataset do not have an equal number of streamed songs, we decided to consider the mean Spotify feature vector across all their tracks. At the cost of information about how each feature varies within an artist's discography, this method ensures that the features for each artist are of the same dimension. A Stanford class project which clustered artists from the MSD with Spotify data reported strong accuracy using only mean feature vectors, which inspired us to try this approach (Agrim Gupta, 2017).

To ensure the artists in our graph have a meaningful set of streaming features such as overlap in listeners, we only considered artists who have been streamed over 1000 times by at least 50 unique listeners. This reduced our dataset to 207 artists. Only 128 of these artists had Spotify features for all of their tracks, further reducing the number of nodes in our graph. Through Spotify we also associated each artist with a list of genres. These genres tend to be quite specific, such as "Argentine rock" or "melodic metalcore", so we assigned each artist to a broad genre by hand. These hand classified genres are the "true" clusters we hope to reveal through analysis of the graph. The 14 genre tags we assigned included `rock`, `pop`, `hip hop`, `soul`, `celtic`, `jazz`; however, we only considered tags with at least 10 artists which were `rock`, `pop`, `hip hop`, `metal`, `electronic`. This reduced our dataset to the final size: 97 artists in 5 clusters.

The weights on the edges between artists are discussed in Section 5

4. Feature Analysis

Before running clustering algorithms, we analyzed the statistical properties of our features and true clusters. Our main goal was to check that our selected features were rich enough to differentiate between genres. A secondary goal was to discover which methods of data dimensionality reduction best suited our features, as some of our graph algorithms require scalar edge weights.

4.1. Spotify Features

In the artist graph, the “true” hand-labeled clusters are known. Fixing a single Spotify feature such `valence`, we generated box and whisker plots to compare our 5 clusters. For each cluster we have a list of artists and a list of songs performed by those artists. To gain insight into the effect of associating each artist with a single mean feature vector instead of a vector for each song, we ran this box and whisker analysis twice. First with every song in the genre contributing to the plot and then with each artist in the genre contributing their mean vector of song features.

The plots shared below have boxes between the first and third quartile, a green line at the median, and whiskers which extend to no more than 1.5 times the interquartile range (IQR); outliers are shown if necessary.

We observe that the distribution of a single Spotify feature varies across genres, but there is significant overlap in the IQR. The attached plot of valence is representative of what we observe for most features. There are some features for which a single genre has a much larger median value, such as `speechiness` for `hip hop` and `instrumentalness` for `electronic`. These observations make intuitive sense considering, for example, that electronic music is unlikely to contain much singing. Using the mean artist features instead of all songs in a genre somewhat compressed the IQR and whiskers and did not change the medians in most genres. This suggests visually that this data reduction strategy does not sacrifice too much information. Furthermore, these plots revealed nearly every song in our dataset as a `time signature` of 4, making that feature redundant.

We also tested the power of each feature to differentiate between genres. For each feature, we generated the 5×5 correlation matrix between genres. The Pearson correlation coefficient requires paired data, which we did not have because the number of songs and artists within each genre varies. To estimate the correlation between the observed features in two genres, we instead sampled with replacement m samples of a single feature from each genre, where m is the minimum artists in either genre. Next we calculated the Pearson correlation between these random samples, repeated this process 100 times, and averaged the results. We

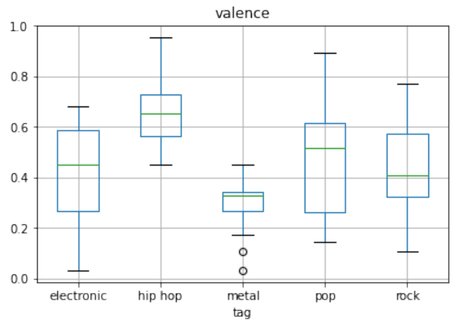


Figure 1. Computed with mean song vector for each artist

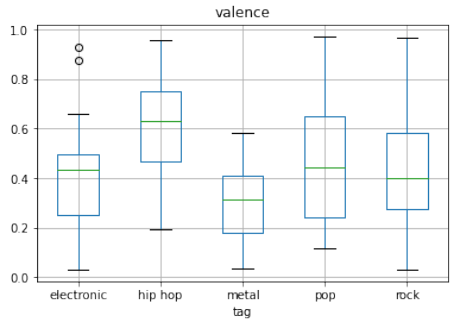


Figure 2. Computed with all songs in a genre. Note the wider IQR and whiskers in some genres

also considered the cross-correlation as a metric without resampling, but this metric appears more suitable for time-series data.

These correlation matrices reiterated the results we observed through the box plots. Namely, the correlation is not consistently highest along the diagonal, so any single Spotify feature is not enough information to determine genre. However some features are more highly correlated with a particular genre such as `speechiness` and `hip hop`. However, the correlation’s magnitude is less than 0.1 for all features and genre pairs which suggests no single feature has enough information to predict a cluster well.

Finally, we used confusion matrices to test whether pairs of artists within the same genre have the more similar feature vectors than pairs across genres. For a variety of distance metrics described below, we calculated the pairwise distances between every song in each of our $\binom{5}{2}$ cluster pairs. Considering the pairwise distances as a vector (without repetition), as opposed to a matrix, we measured the mean. When graphing this information in a confusion matrix, we hoped to see the smallest mean distances along the diagonal; this

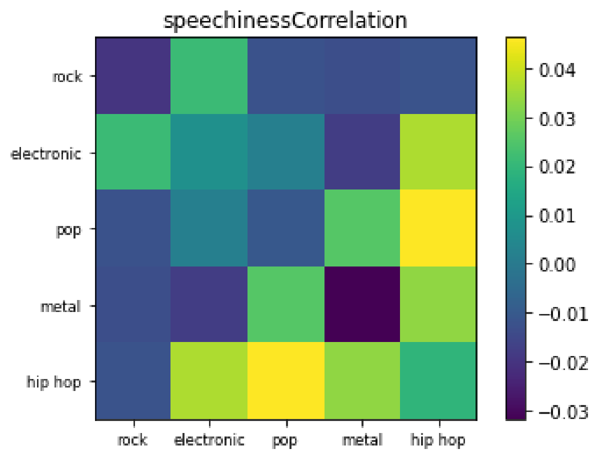


Figure 3. Speechiness is more highly correlated with `hip hop`

would indicate that the mean distance of song pairs within a genre is smaller than across genres.

Using the raw Spotify features, which have different scales, the diagonal mean pattern we hoped for only appeared using the standardized Euclidean and Mahalanobis distances; note that the standardized Euclidean distance is a special case of Mahalanobis with a diagonal covariance matrix. This result suggested we should standardize our features. We tested two strategies: standardizing each feature to zero mean and unit variance and the min-max scaler. For feature x from a set X of observations across all units, $x_{standardized} = \frac{x-\mu(X)}{\sigma^2(X)}$ and $x_{min/max} = \frac{x-x_{min}}{x_{max}-x_{min}}$ where $\min$ and $\max$ are taken across all units.

For each of these two rescalings, we generated confusion matrices for a variety of distances using SciPy's `cdist` function. These distances include the p-Minkowski distance, vector correlation, cosine distance, and Chebyshev distance. We consistently found that using the mean-variance standardization method produced a smaller intracluster mean than min-max scaling. This result makes sense as the key distinction of standardized Euclidean and Mahalanobis distances is rescaling the variance. We noted the most pronounced diagonal pattern with Euclidean and Chebyshev distances and decided to test our clustering methods with both distance metrics.

Recall that for a pair of vectors u and v in $\mathbb{R}^d$, $d_{Chebyshev} = \max_i \|u_i - v_i\|$, which takes the L_∞-norm of the vectors' absolute difference. The Euclidean distance is given by $\sqrt{(u-v)^T(u-v)}$

The Louvain method requires scalar edge weights between nodes in our artist graph. This experiment also helped us determine which scalar distance comparing vectors in $\mathbb{R}^{11}$ preserves the most information.

4.2. Streaming Features

Recall that from the MSD we also have features describing how frequently an artist was played and by whom.

The numeric streaming features include the artist's total number of streams, number of unique listeners, and the ratio of listens coming from well-listened streamers, users with at least 100 total streams. We used the same analysis methods as the previous section, box and whiskers plots as well as confusion matrices of pairwise distance between genres. An artist's number of total streams varies across genres with metal and hip hop artists receiving less than 400 median streams and rockers above 750. Including the other streaming features did not meaningfully reduce the mean intracluster pairwise distance, measured with both Euclidean and Chebyshev, so we opted not to include them in our analysis. This omission was confirmed by the tests described in the next section.

4.3. Final Features

Combining the useful Spotify and streaming data, each artist is associated with a feature vector in $\mathbb{R}^{11}$. These features and some summary statistics are listed below:

```
       num_listeners      tempo  danceability     energy        key  \
count      97.000000  97.000000     97.000000  97.000000  97.000000
mean        0.004386   0.232879     -0.094211   0.416531  -0.027106
std         1.015022   0.810810      0.820343   0.775940   0.768111
min        -0.378703  -1.336916     -1.953280  -1.577251  -1.475275
25%        -0.319143  -0.448009     -0.743324  -0.056799  -0.530684
50%        -0.253097   0.194609     -0.154697   0.511949  -0.058389
75%        -0.037562   0.721488      0.510233   1.027613   0.508365
max         8.485060   2.644465      1.420425   1.543278   1.358496

        loudness  speechiness  acousticness  instrumentalness    valence  \
count  97.000000    97.000000     97.000000         97.000000  97.000000
mean    0.570779    -0.005087     -0.454575         -0.186965  -0.261312
std     0.599210     0.687373      0.664587          0.709794   0.740002
min    -1.494094    -0.509037     -1.046763         -0.575716  -1.815493
25%     0.254239    -0.446180     -1.019682         -0.575706  -0.737251
50%     0.632547    -0.246335     -0.633123         -0.573598  -0.303283
75%     1.045076     0.088836     -0.094237         -0.120927   0.378268
max     1.661091     3.945122      1.399469          1.933250   1.628917

        liveness
count  97.000000
mean    0.056214
std     0.776874
min    -0.821078
25%    -0.484973
50%    -0.240156
75%     0.574288
max     3.383233
```

Three features were omitted from this final collection: `time signature,total plays, play ratio`. We confirmed that adding any combination of these features back into the graph increased the mean pairwise distance in most of the true clusters, for both the Euclidean and Chebyshev distances.

5. Network Structure & Node Centrality Analysis

After constructing the artist graph with L_∞-norm as the edge weights determined in the section above, we conduct network analysis on the resulting artist graph to gain a deeper understanding of the structure of the network as well important nodes in the network.

5.1. Degree Distribution

Our network analysis begins with a coarse overview of the degree distribution of the network, namely, the degree distribution of the artists. We observe in Figure 4 that the majority of artists share small edge weights with most other artists in the graph. This is seen by the right-skewed nature of the degree distribution plots. We analyze degree centrality in the following sections to gain an understanding of important nodes that exist in this network.

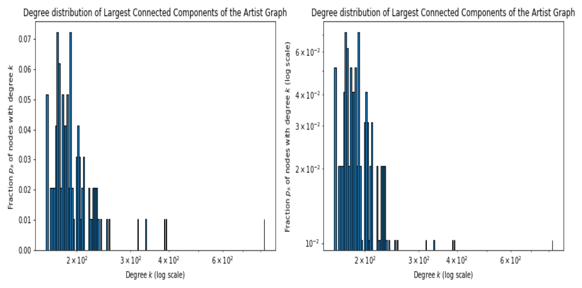


Figure 4. Degree Distribution of Artist Network

5.2. Degree Centrality

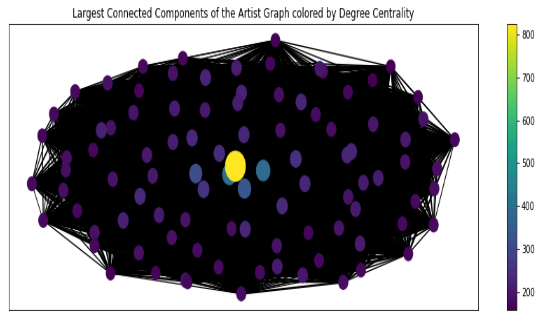
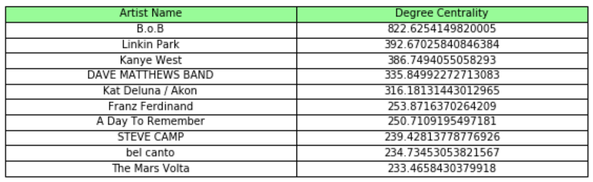

Largest Connected Components of the Artist Graph colored by Degree Centrality
800
700
600
500
400
300
200

Figure 5. Degree Centrality of Artist Network



Artist Name	Degree Centrality
B.o.B	822.6254149820005
Linkin Park	392.67025840846384
Kanye West	386.7494055058293
DAVE MATTHEWS BAND	335.84992272713083
Kat Deluna / Akon	316.18131443012965
Franz Ferdinand	253.8716370264209
A Day To Remember	250.7109195497181
STEVE CAMP	239.42813778776926
bel canto	234.73453053821567
The Mars Volta	233.4658430379918

Figure 6. Degree Centrality of Artist Network

From Figures 5 & 6, we see the top 10 most important nodes by degree centrality within the artist network. These artists include B.o.B, Linkin Park, and Kanye West. Now that we have a coarse understanding of the structure of our network, we move on to applying clustering methods on the artists.

6. Clustering Results

6.1. Louvain Clustering using L_∞-norm Edge Weights

We begin our graph clustering comparisons by using the Louvain method on the artist graph. We note that a drawback of this algorithm is that it operates on edge weights which are represented by a singular measure rather than leveraging the information contained across multiple edge attributes or features in computing the clustering. This is why we investigate K-means as an alternative clustering algorithm to Louvain on the artists data, so that we can make use of the original features rather than a singular measure computed from the original features.

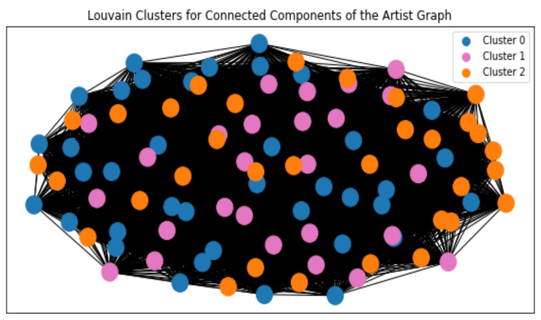
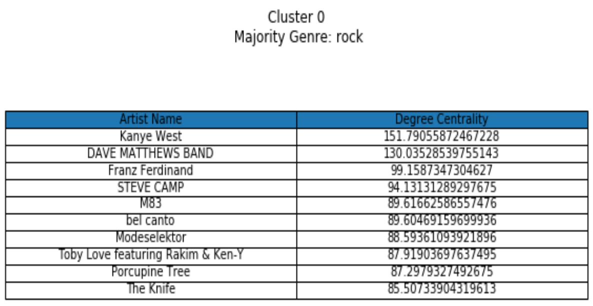

Louvain Clusters for Connected Components of the Artist Graph
● Cluster 0
● Cluster 1
● Cluster 2

Figure 7. Louvain Clusters found on Artist Graph with Optimal Modularity

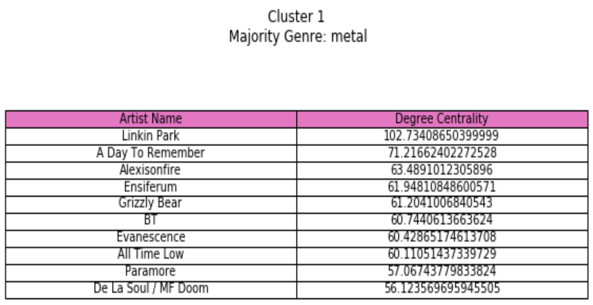
After applying Louvain, the algorithm finds an optimal clustering at 3 clusters, specifically Cluster 0 (Blue), Cluster 1 (Pink), and Cluster 2 (Orange). The optimal modularity found by Louvain at 3 clusters is 0.006, which is indicates the clustering is marginally better than what we would expect from a random graph generated from the same nodes and edges.

In Tables 8 & 9 & 10, we list the top 10 artists ranked by degree centrality within each cluster respectively. These nodes provide a representative pool of artists representing the clustering uncovered by the Louvain method. We observe that the nodes which had highest degree centrality for the network overall, namely, B.o.B, Linkin Park, and Kanye West, are the most central nodes in the three clusters found by Louvain: B.o.B (Cluster 2), Linkin Park (Cluster 1), and Kanye West (Cluster 0)


Cluster 0
Majority Genre: rock

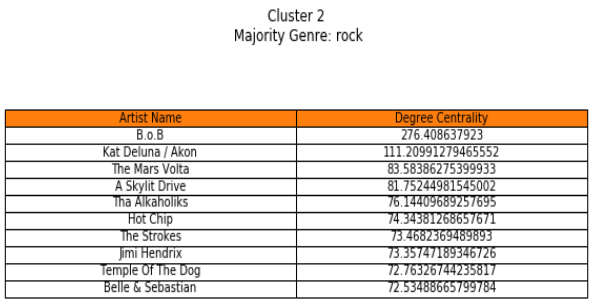
Artist Name	Degree Centrality
Kanye West	151.79055872467228
DAVE MATTHEWS BAND	130.03528539755143
Franz Ferdinand	99.1587347304627
STEVE CAMP	94.13131289297675
M83	89.61662586557476
bel canto	89.60469159699936
Modeselektor	88.59361093921896
Toby Love featuring Rakim & Ken-Y	87.91903697637495
Porcupine Tree	87.2979327492675
The Knife	85.50733904319613

Figure 8. Cluster 0 Top 10 Artists by Degree Centrality


Cluster 1
Majority Genre: metal

Artist Name	Degree Centrality
Linkin Park	102.73408650399999
A Day To Remember	71.21662402272528
Alexisonfire	63.4891012305896
Ensiferum	61.94810848600571
Grizzly Bear	61.2041006840543
BT	60.7440613663624
Evanescence	60.42865174613708
All Time Low	60.11051437339729
Paramore	57.06743779833824
De La Soul / MF Doom	56.123569695945505

Figure 9. Cluster 1 Top 10 Artists by Degree Centrality


Cluster 2
Majority Genre: rock

Artist Name	Degree Centrality
B.o.B	276.408637923
Kat Deluna / Akon	111.20991279465552
The Mars Volta	83.58386275399933
A Skylit Drive	81.75244981545002
Tha Alkaholiks	76.14409689257695
Hot Chip	74.34381268657671
The Strokes	73.4682369489893
Jimi Hendrix	73.35747189346726
Temple Of The Dog	72.76326744235817
Belle & Sebastian	72.53488665799784

Figure 10. Cluster 2 Top 10 Artists by Degree Centrality

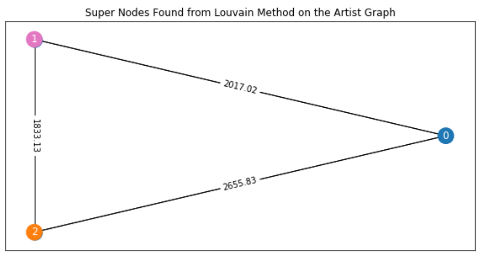
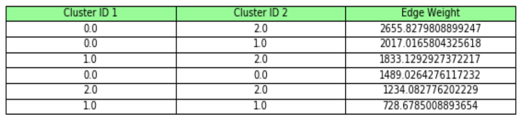


Figure 11. Louvain Cluster Super Nodes



Cluster ID 1	Cluster ID 2	Edge Weight
0.0	2.0	2655.8279808899247
0.0	1.0	2017.0165804325618
1.0	2.0	1833.1292927372217
0.0	0.0	1489.0264276117232
2.0	2.0	1234.082776202229
1.0	1.0	728.6785008893654

Figure 12. Louvain Cluster Super Nodes Table

In Figure 11, we see all of the artist nodes condensed into 3 supernodes based on their respective Louvain clusters. Ideally, when the clusters found by Louvain are largely dissimilar, the super nodes share small edge weights between each other and have self-loops with large edge weight. However, in the case of the artist graph with L_∞-norm, we observe that the weight of the self-loops on the supernodes are all smaller than the weights on the edges between the supernodes. This is indicative of clustering that is marginally better than random, and hence why we see a small modularity measure on the resultant clusters found by Louvain.

Finally, for each cluster, we assign it a plurality genre label, i.e. the genre that is most present in the cluster based on the artists placed in that cluster by Louvain. The plurality genre label for each cluster are as follows:

- Cluster 0: **rock**
- Cluster 1: **metal**
- Cluster 2: **rock**

Thus the majority of artists are classified as being in the rock genre. We then predict all artists within a cluster

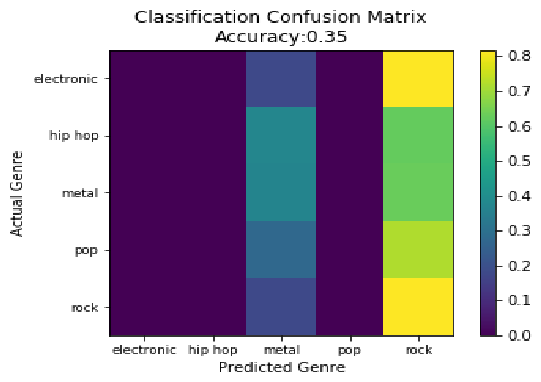


Figure 13. Confusion Matrix of Louvain Cluster Plurality Genre Prediction

as being a part of the plurality genre for that cluster. The results of this cluster plurality prediction on the artists are depicted in the confusion matrix in Figure 13. This results in a classification accuracy for artist genre of 35%. This is the same as the baseline model of predicting all artists as being in the rock genre, the plurality genre in the entire dataset, 35%. We now compare the Louvain clustering results to K-means clustering on the whole feature set, not just a singular measure as was done for the Louvain method.

6.2. Clustering using K-Means

K-means is a classical hard clustering approach, which means that each artist is deterministically assigned to one of k groups. k is a hyperparameter for the method, but because we created 5 true genre tags, we hypothesized strong results with a k near 5.

As described in class, k-means is an iterative approach. k centroids are randomly initialized and artists are clustered to the centroid nearest them, with respect to the Euclidean distance. An algorithm such as Lloyd's repeatedly updates the centroids, aiming to reduce the sum of intracluster variances. Due to a fundamental connection with the Expectation-Maximizing (EM) algorithm, k-means will decrease that objective with each update but potentially get stuck at a local optima. Formally the objective k-means attempts to minimize is: $\arg\min_C \sum_{i=1}^{k} \sum_{x \in C_i} (x-\mu_i)^T (x-\mu_i)$ where C is the feasible set of all possible centroids.

Unlike the Louvain method, k-means considers features in their original dimension. For this reason and the ability to select the number of clusters, we hypothesized and observed more predictive accuracy from this approach. After the k-means algorithm, implemented with SKLearn converged to a set of centroids, we labeled each cluster with a genre by taking a plurality-vote. Our outcome of interest is classification accuracy, or the fraction of artists placed in the correct cluster.

To help tune the hyperparameter k, we consider a visual aid: the "elbow plot." This graphs two outcomes as the number of clusters increases: classification accuracy and sum of intracluster squared errors, an algebraic manipulation of the objective.

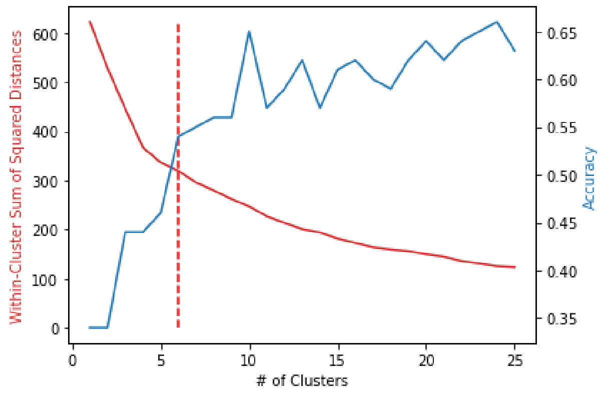


Figure 14. Elbow plot for k-means. The vertical dashed line occurs at $k = 6$.

From this plot we decided on $k = 6$ as the optimal hyperparameter; this is remarkably close to the true value of $k = 5$. The classification accuracy improves by over 10% from $k = 5$ to 6, and quadrupling the number of clusters to 24 is required to gain an additional 10%. The curve of within cluster sum of squared distance decreases monotonically, which makes intuitive sense because adding more clusters allows the most spread out ones to be split. Moreover, $k = 6$ occurs after the curve's portion of steepest descent, a selection heuristic which gives the "elbow plot" its name.

As a form of validation, we regenerated this plot with each covariate removed, one at a time. The classification accuracy at $k = 6$ lowered with every removed feature, suggesting each contributes meaningful information. The limitation of this strategy is that it neglects any higher order interactions.

As 16 indicates the k-means approach achieves 54% classification accuracy. The accuracy is highest along

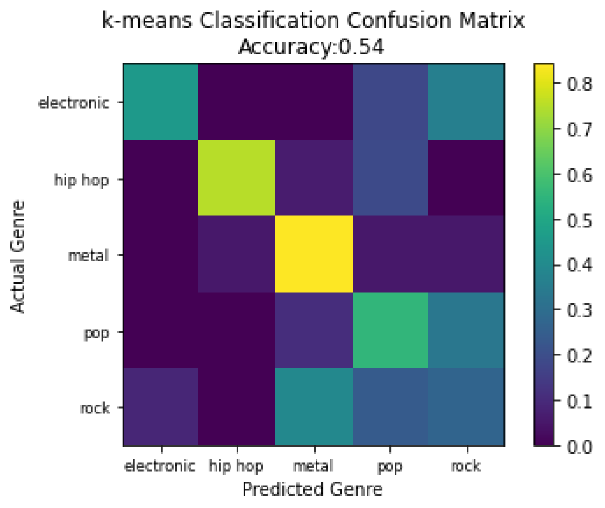


Figure 15. Confusion Matrix of k-means Majority Genre Classification for k=6. Note that each row sums to 1.

the confusion matrix's diagonal except for rock, which demonstrates the most likely clustering from this algorithm is a correct one. The algorithm has particular success at classifying metal bands, with over 80% accuracy.

To gain insight into the clusters k-means revealed, we provide a table on the last page which contains the 5 artists closest to each centroid. The clusters are labeled with `metal, rock, hip hop (x2), pop, electronic`, representing each of the true genres. However, the artists closest to each centroid do not match the predicted genre very well. For instance The Chieftains, a traditional Irish band, are the second closest band to the metal centroid while rock band blink-182 is listed under hip hop. This is surprising because cluster labels are assigned by plurality, so many of the distant artists in each cluster must have to the same true genre. Unfortunately, $\mathbb{R}^{11}$ is impossible to visualize and plotting on the plane through t-SNE did not give any intuition as to why this might happen.

7. Discussion

We compared two clustering methods, Louvain & k-means, using a featurization of artist features shown to be best at highlighting similarity between artists in the same genre and dissimilarity between artists in different genres.

Applying the Louvain method and evaluating the clustering on the basis of genre resulted in an accuracy score which was the same as the baseline model of selecting all artists to be in the rock genre, the plurality genre label of the entire dataset. One of the clusters found by Louvain had plurality label metal and thus predicted all of the artist within that cluster as metal, but most of the rock artists were correctly placed inside the rock clusters, so these predictions contribute most to the accuracy score found by this method along with the correctly predicted metal artists. Thus, Louvain with L_∞-norm edge weights does not cluster the artists on the any better than the random graph generated from the same nodes and edges and does not find clusters which appear to have any relation to genres.

The k-means algorithm performed reasonably well on our dataset. At the optimal $k = 6$, removing each covariate lowered classification accuracy, which suggests the method utilized the rich feature vectors well. Unfortunately, the difficulty of interpreting results in $\mathbb{R}^{11}$ limits the power of this method in practice. We remain puzzled as to why the artists closest to each centroid often do not match the genre associated with that cluster.

Given our time, memory, and compute constraints, we were limited to a significantly smaller dataset than we would have ideally liked to have used. With a much larger dataset, consisting of more artists data, streaming data, etc. our featurization using these data would be richer, yielding features with better differentiation capabilities between the artists. For instance, the song features for an artist usually took the mean over 5 or fewer songs instead of the tens which make up an average discography. Having more complete streaming data would have potentially allowed us to use user preferences to determine clusters. We would expect the clustering to have higher between cluster variance (dissimilar artists between clusters) and lower within group variance (similar within clusters).

8. Future Work

The easiest extension of our project would be to repeat the process on the full 300GB MSD database. As mentioned in 7, we believe the lack of detailed streaming features and handful of songs per artist weakened the power of our analysis. The streaming data provided by the MSD is particularly sparse and lacks useful information about when streams occurred and in relation to what other songs.

Another interesting use of the MSD would be to explore the connections amongst users of a streaming service, the listeners. We had initially planned to look at this and formulated the question: Are there cliques of users that emerge from the graph of user listening preferences?

Similar to 3.2, we would construct an undirected graph consisting of u nodes representing users. Each user shares an edge with the other $u - 1$ users. This yields a complete graph with u nodes and $\frac{u(u-1)}{2}$ edges.

We observed that many users in the subset streamed music only a handful of times, making the data an unreliable measure of their listening tastes. To improve the quality of our dataset, we would only consider users who streamed to at least 100 songs (repeated listens included). Selecting the songs played by these "high-frequency" streamers gives a final subset of data. This includes 321 unique users, 304 unique artists, and 395 unique songs. Of the unique songs 63 are listened to by at least 4 distinct users and 44 are listened to by at least 5, demonstrating an overlap in listening tastes. Although this potential user dataset has the sparsity issues of our artist one, the positive results of k-means suggest it is worth investigating.

9. Conclusion

In this project, we endeavored to classify artists by the genre. For a collection of 97 artists spanning 5 broad genres, we built features about song characteristics from Spotify and streaming from the MSD. Two clustering methods were employed, the Louvain method applied to a graphical representation of this data and k-means. The Louvain method had limited success, having the same classification accuracy achieved by plurality-vote, 35%, but k-means was more successful and achieved 56% accuracy. Furthermore, that approach had its best characteristics with 6 clusters, very close to the 5 true ones.

References

AcousticBrainz. Million song dataset echo nest mapping archive, 2016. URL https://labs.acousticbrainz.org/million-song-dataset-echonest-archive/.

Agrim Gupta, Karan Rai, N. B. Community detection based on music listening habits. 2017. URL http://web.stanford.edu/~agrim/pdfs/cs221-project-final.pdf.

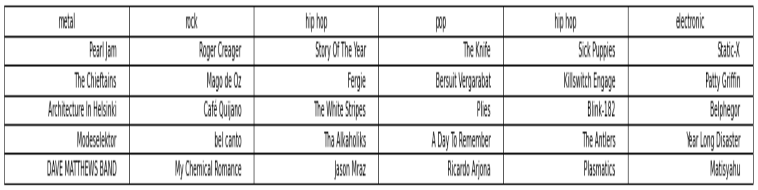
Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011. URL http://millionsongdataset.com/.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, Oct 2008. ISSN 1742-5468. doi: 10.1088/1742-5468/2008/10/p10008. URL http://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

Schlitter, N. and Falkowski, T. Mining the dynamics of music preferences from a social networking site. In *2009 International Conference on Advances in Social Network Analysis and Mining*, pp. 243–248, 2009. doi: 10.1109/ASONAM.2009.26.

Spotify API. Spotify web api, 2021. URL https://developer.spotify.com/documentation/web-api/.

Wei Peng, Tao Li, M. O. Music clustering with constraints. 2007. URL https://www.researchgate.net/profile/Mitsunori-Ogihara/publication/220723183_Music_Clustering_with_Constraints/links/54295f030cf2e4ce940d388a/Music-Clustering-with-Constraints.pdf.



metal	rock	hip hop	pop	hip hop	electronic
Pearl Jam	Roger Creager	Story Of The Year	The Knife	Sick Puppies	Static-X
The Chieftains	Mago de Oz	Fergie	Bersuit Vergarabat	Killswitch Engage	Patty Griffin
Architecture In Helsinki	Café Quijano	The White Stripes	Plies	Blink-182	Belphegor
Modeselektor	bel canto	Tha Alkaholiks	A Day To Remember	The Antlers	Year Long Disaster
DAVE MATTHEWS BAND	My Chemical Romance	Jason Mraz	Ricardo Arjona	Plasmatics	Matisyahu

Figure 16. The 5 artists closest to each centroid of our k-means clustering