Spotify® - Featuring Artists Analysis

github.com/carattj/spotify_featuring_artists_analysis

1. Introduction

Music is a universal language that transcends borders, cultures, and beliefs. It speaks to our emotions and brings us together, regardless of our differences. Spotify, one of the world's largest music streaming platforms, has captured the essence of this shared experience by providing a platform that connects artists and listeners from around the world.
In this project, we analyze the collaborations between artists on Spotify to uncover hidden patterns in the music industry. Using community detection algorithms, we identify clusters of artists and analyze the genres within each group. We also examine influential nodes, connections between clusters, and their peculiar musical shades. Our analysis provides insights into how music is created and evolves through collaboration and offers a better understanding of how collaborations shape music.

2. Data

2.1 Data Description

We're analyzing the Artists Featurings Network dataset[1] from Kaggle, which includes over 156,000 artists, considered as nodes, and 300,000 collaborations, viewed as edges. The data is authentic and reliable, sourced directly from the Spotify API, and - additionally to the featurings between artists - it comprehends the following information about each artist.

spotify_id	name	followers
Unique id of the artist	Name of the artist	Number of followers
popularity	**genres**	**chart_hits**
Popularity score	List of genres	List of #chart hits per country

Table 1 – The attributes of each artist in the dataset.

2.2 Data Reduction

We decided to simplify the dataset due to the large number of nodes and edges, which would result in excessively long computation times: the clustering's complexity is in fact heavily dependent on that. We only kept artists with a number of collaborations between 22 and 101, reducing the nodes to 4,326 and collaborations to 25,511, a more manageable amount. Artists with less than 22 collaborations were a lot, but their contribution to the overall collaborations was negligible. On the other side, artists with more than 101 collaborations were only a few, but they increased the complexity of the dataset. Removing them allowed us to streamline our analysis.

2.3 Data Enrichment

We found the Tracks dataset[2] on Kaggle, which includes over 500,000 songs with author data linking them to their respective artists. The dataset also includes additional information that we used to uncover patterns related to groups of songs. All data was again sourced from the official Spotify API.

3. Network Analysis

3.1 Community Detection

We implemented the Louvain clustering algorithm to detect communities. The implementation consists of four steps, which are repeated until convergence. To test the correctness of our implementation, we used an example graph provided in the *simulation_network.py* Python file, and confirmed the correctness of our algorithm.

[1] https://www.kaggle.com/datasets/jfreyberg/spotify-artist-feature-collaboration-network?select=edges.csv
[2] https://www.kaggle.com/datasets/lehaknarnauli/spotify-datasets?select=tracks.csv

Initially, we employed a list of lists structure to represent the communities, which allowed us to manipulate clusters with a complexity of O(1) for adding and removing elements. However, this approach is not efficient for search operations, since locating an element in the list requires looping through all single elements in each community, resulting in a slow double for loop. To address this limitation, we improved the implementation by incorporating the hash table *ht*. In different circumstances, the algorithm needs to loop through the current partitions stored in the list to find out where the neighbors of the node are located and use those communities to calculate the modularity gain. In essence, the hash table helps us to take the correct community of which we want to find the modularity gain by just looking for the current position of the neighbor in the hash table. This modification resulted in a significant performance improvement, making our implementation about 35-40% faster than the initial approach. Consequently, we were able to eliminate many of the for loops and search operations for nodes and neighbors in specific communities stored in the "partitions" list, which can be done in O(1) time complexity. In Table 2, we present a comparison between NetworkX's built-in Louvain implementation and ours.

Louvain Implementation	NetworkX	Scratch
Runtime	0.5 seconds	314.04 seconds
Total communities	46	193
Most frequent communities (#nodes, #communities)	(2, 10), (3, 3), (4, 4)	(2, 74), (3, 35), (6, 11)
Biggest community (#nodes, #communities)	(820, 1)	(775, 1)
State	Randomized	Deterministic
Normalized Mutual Information Score (NMI)	0.810	

Table 2 – Performance comparison between our Louvain and NetworkX's Louvain implementations.

As we expected, our implementation is slower, since we probably didn't follow the best strategies to build communities and other data structures might be more performant.

3.2 Communities Distributions Comparison

Figure 1 shows that the obtained communities are different compared to the built-in ones.

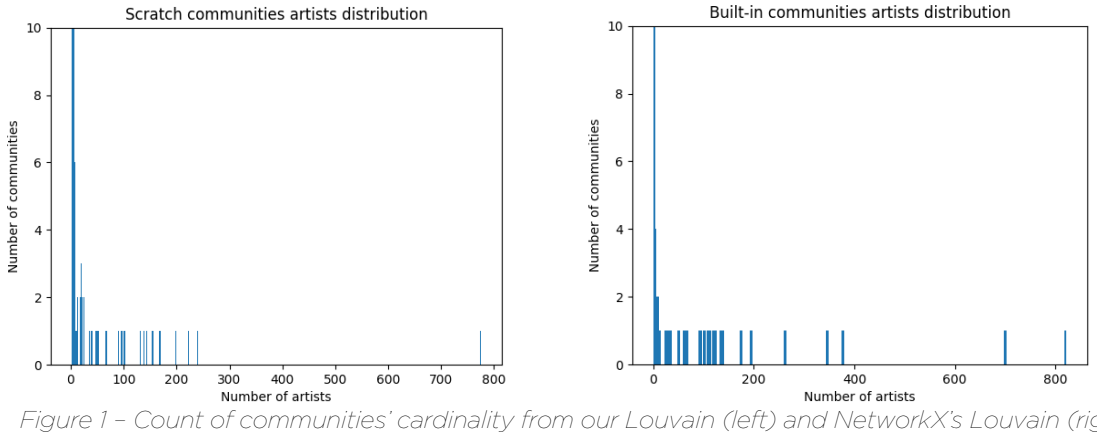

Figure 1 – Count of communities' cardinality from our Louvain (left) and NetworkX's Louvain (right).

However, it is also evident that both trends are similar in the fact that they follow an exponential distribution. To bring the plots in the same scale and make them better comparable, we set the Y axes limit to 10. On the left histogram, this causes to clip the counts of the 3 biggest communities - 74 with 2 artists, 35 with 3 artists, 11 with 4 and 6 artists - to 10 artists.

The disparity between the distributions is normal and further substantiated by the NMI score, which we computed by utilizing the built-in Louvain outcomes as the ground truth for the detected communities. The resulting NMI score of 0.810 implies that the two distributions are not the same, but very similar, as a score approaching 1 denotes a high degree of similarity. Therefore, it can be inferred that the two algorithms share similar community structures.

The observed discrepancies between the two implementations may arise from contrasting formulations employed to calculate the modularity and modularity gain. Our implementation adhered to the approach elucidated in the lecture, while the built-in implementation employed different formulas. Regrettably, the choices behind these variations remain unclear to us. Another possible contributing factor could be the numerous normalization steps incorporated in the built-in version, which likely facilitate a more equitable distribution of artists across all communities. Also, the randomicity involved in some steps of the built-in implementation may cause some differences. Conversely, our algorithm is deterministic and always produces the same result.

3.3 Community Genre

To differentiate clusters, we assign labels based on the most prominent genre of music produced by the artists in each cluster. To avoid assigning the same label to multiple clusters, we designed our algorithm to identify the n-most frequent genres, but ultimately chose n=1 because this issue did not arise often in our analysis.

3.4 Communities Correlation

The obtained results indicate that the clustering technique applied to the collaborations of artists led to the formation of coherent and sensible clusters. This suggests that artists who create similar music are likely to collaborate with each other, and that this pattern is reflected in the resulting clusters: an example is *hip hop* and *rap*. By considering the collaboration matrix and the size of each cluster, we have also been able to produce a visualization of the whole graph (Figure 2). Each bubble corresponds to a cluster, its size reflects the number of artists it groups, and the closeness between bubbles represents the tendency to collaborate.

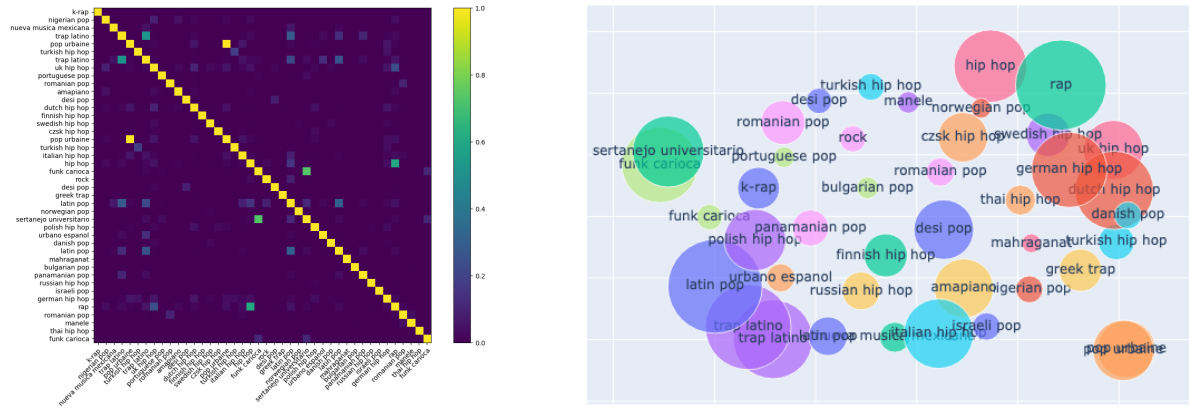

Figure 2 – Communities collaboration matrix (left) and communities feature network visualization (right).

Our analysis revealed that the titles of songs produced by artists within a cluster are consistent with our finding that artists with similar backgrounds tend to collaborate together. In particular, if a cluster is labeled as belonging to a particular geographical region, we found that the most frequently occurring words in the titles of its songs are written in the language commonly used in that region. This additional analysis provides further evidence of the effectiveness of the clustering approach in identifying meaningful patterns in the music industry, as presented in one of the next sections. The overall visualization of the clusters offers a captivating perspective on their size and proximity based on the number of collaborations. It is truly fascinating to observe how artists who share similar genres, musical styles, geographical locations, and languages tend to cluster together, creating a rich tapestry of musical collaborations, each with its own unique character and identity.

3.5 Most Influential Community Artists

To further analyze the artist communities, we incorporated measures of artist influence within these groups. An influential artist is considered an artist with many collaborations or more prone to collaborate in the future. To identify these artists, we applied to each community-genre the centrality measures that suited the best our dataset: degree, closeness, and betweenness.

A high degree centrality indicates an artist with a lot of collaborations within its community-genre, which may potentially open opportunities for future featurings. We interpret an artist with high betweenness centrality as an intermediary between a lot of different artists, which may - thanks to him/her - collaborate in the future. Finally, a high closeness centrality corresponds to an artist which already has a good number of featurings and a lot of potential for future collaborations.

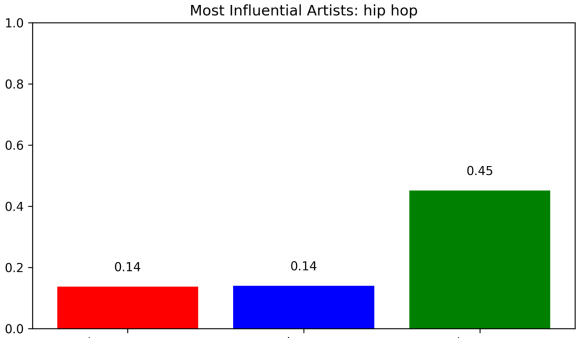

Figure 3 – Most influential artist of hip hop community

As depicted in Figure 3, the most influential artists in *hip hop* community are Missy Elliott and Pharrell Williams. Given the results, we can conclude that PW collaborated a lot in the past (degree), and has a lot of potential future collaborations (closeness) given his advanced social connections. At the same time, ME opens a lot of possible featurings between different artists whose collaborations pass through her.

3.6 Community Most Used Words

This analysis allows us to uncover the linguistic patterns of different genres, gaining insights into the creativity of each community. For each of them, as shown in Figure 4, we generated a word-cloud with the most used words in the titles of songs produced by its artists. For a more proper analysis, drawing of various sources we generated a file with stopwords to be removed.

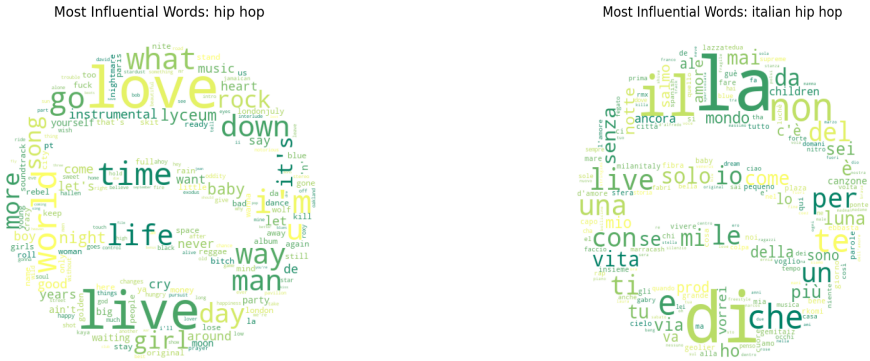

Figure 4 - Word-clouds with communities' most influential words for hip hop (left) and italian hip hop (right).

However, we encountered language-related obstacles due to song titles being written in different languages. Although we removed English functional words, functional words from foreign languages remained. This made it difficult to apply a standardized approach to data cleaning, posing a challenge to our analysis.

3.7 Community Music Features

For each community, we extracted the most interesting information about songs produced by its artists into 2 indexes:

Properties	Songs' perceptual features
danceability	dancing suitability based on factors like tempo, rhythm, and beat strength
energy	intensity or activity, characterized by a fast tempo, powerful vocals, and beats
loudness	overall volume, louder songs being perceived as more intense and energetic
valence	emotional tone or mood, ranging from negative or sad to positive or happy

Qualities	Songs' composition aspects
speechiness	spoken word content
acousticness	degree to which there are acoustic instruments or sounds
instrumentalness	degree to which it is instrumental

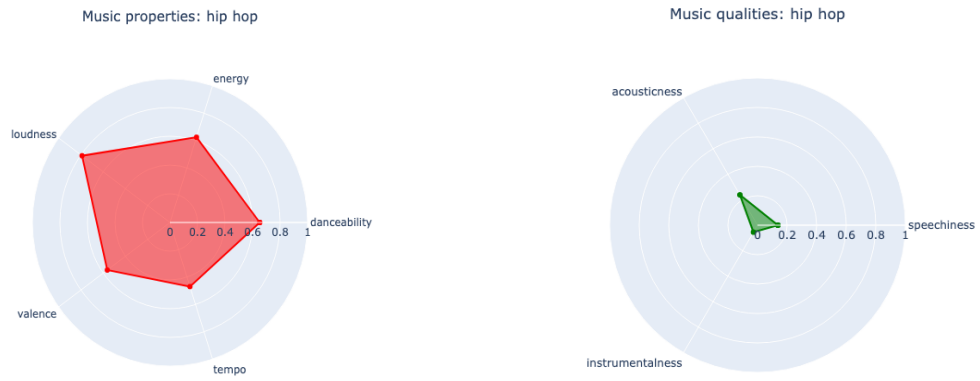

Figure 5 – Properties (left) and Qualities (right) of songs produced by artists in hip hop cluster.

To ensure consistency and comparability across all songs, values have been firstly normalized to a 0-1 range over the whole collection, and only then average values for each community-genre have been computed.

Properties graphs of different community-genres are quite different, and they really represent distinctive features of each community. For instance, Figure 5 shows that *hip hop* is pretty danceable and loud, with a medium level of energy. Different are the qualities graphs, which are very similar among communities. We lead this back to outliers, which may have compromised the results' accuracy, particularly during the initial min-max normalization step. For instance, a song with exceptionally high speechiness compared to the others could have distorted the analysis. Hence, it is worth filtering out outliers.

4. Data Persistence

We initially thought about using Neo4j for this task. However, we then concluded that data persistence offers no substantial benefit in terms of gathering further insights into our dataset, given the abundance of analytical and exploratory measures we already have.
Furthermore, time constraints and obstacles that arose within our team shifted our priorities towards other aspects of the project and took away the time needed to further investigate this task.

5. Future Works

One of the limitations behind our project is the complexity of running our Louvain on the entire dataset due to runtime constraints. Further optimization of the implementation would certainly lead to performance benefits, allowing to provide analysis based on the entire data set. Additionally, although similarity measures could have been a valuable addition to our project, we struggled to find a way to use them effectively and obtain meaningful results. An initial idea that we didn't pursue was that of measures of similarity between artists from different communities. We have not found a way to make sense of this measure, however, deeper analysis could give it deeper meanings.