Advanced Marketing

Network Analysis

Final Project

Gal Gritzerstein 316064773

& Daniel Mandelbaum 316014570

Lecturer: DR. Ofrit Lesser

Table of Content

Defining Research Topic

Spotify is a Swedish audio streaming and media services provider founded in 2006 by Daniel Ek and Martin Lorentzon. **It is the world's largest music streaming service provider**, with over 381 million monthly active users, including 172 million paying subscribers, as of September 2021.[1]

Spotify made itself unique by leveraging personal mobile app data for personalizing the user experience and discovering new music. From its machine-learning, artificial intelligence, and data filtering technology, Spotify analyses its users' listening habits and builds out customized recommendations. The recommendations include playlists and music suggestions based on the genres and artists users listen to.

As employees in the Spotify company, we would like to increase the use time of our users on our platform. **We would like to explore how we recommend new artists to our users and optimize this process to give better recommendations.**

Beyond making music accessible to users we want to be able to recommend them new music that they will like based on their past streaming plays. In doing so, we increase our advantage over competitors, retain users and increase engagement.

Choosing this project

Spotify is our main platform to stream music and we believe the current recommendation algorithm to help users discover new music is lacking. We wish to explore the techniques used by Spotify and optimize them to discover our users with new music and artists.

[1] https://en.wikipedia.org/wiki/Spotify

Main challenges

Analyze different users:

The recommendation tool we are developing is intended to suit all Spotify users. With that in mind, we want to analyze the artists' networks of different users. In order to meet the time and size restrictions (handling multiple streams at once), we will only focus on one student's streaming history. This might lead to biased results. As a way to meet this challenge, we will use tools that we believe will be appropriate for every user (as generic as possible). Our network measurement tools will be elaborated in our network analysis.

Measuring our success:

Originaly we wanted to measure our performance by performing link prediction. Link Prediction requires computational power beyond what we have (Transforming edges and nodes into vectors and analysing them). After getting into technical problems and **consulting with Ofrit** we decided to take a different approach. We will recommend new artists based on the top streamed artists of the user. After making recommendations, ,we will check if the recommended artist was in the original user's streaming list, but not one of the top streamed artists.

Research Approach

Our research aims to explore how we recommend new artists to our users. To accomplish this goal, we will analyze the streaming history of a single user - Yoni Cohen (a student from the BA and data analytics class).

We will create a music streaming data set for a single user to understand his musical taste and preference - artists he heard, genres he focused on, songs, playlists, etc. We will do this by creating a dataset combined from the user's

streaming history and Spotify's API. After doing so, we will conduct an exploratory analysis to understand the data.

Analyzing the data will allow us to create the artists' network. For each top streamed artist we will find similar artists using Spotify's algorithm, calculate a "similarity score" that we will define, between every pair of artists and explore the links of our network them to find new artists that match the user's musical taste.

We will define the similarity score to find connected artists while establishing the definition of "the strength of connection" or "the possibility of the user to like the artist's songs" with the network analysis tools we learned in class.

Working Process

Creating The Dataset

In order to create our dataset we used 2 main sources:

1. Yoni's Spotify data (in JSON files)

2. Spotify API

Using these two sources, we were able to bring together all the tracks that the user has **streamed in the past year** and their attributes - artist name, release date ,audio features, duration of each track and more.

Firstly we filtered out streams **played for less than 30 seconds.** This threshold was chosen because we believe that if the song lasted less than 30 seconds he did not really listen to it.

	popularity	release_date	artist_name	end_time	ms_played	track_name	track_id	energy	liveness	tempo	...	weekday	time
0	65	2020-12-11	Kid Cudi	13/12/2020 12:27	242266	Solo Dolo, Pt. III	27oVCAziETRbNuo5A8LNpg	0.727	0.164	152.058	...	6	12:27:00
1	65	2020-12-11	Kid Cudi	13/12/2020 13:02	242266	Solo Dolo, Pt. III	27oVCAziETRbNuo5A8LNpg	0.727	0.164	152.058	...	6	13:02:00
2	65	2020-12-11	Kid Cudi	13/12/2020 16:31	161417	Solo Dolo, Pt. III	27oVCAziETRbNuo5A8LNpg	0.727	0.164	152.058	...	6	16:31:00
3	65	2020-12-11	Kid Cudi	14/12/2020 9:51	118117	Solo Dolo, Pt. III	27oVCAziETRbNuo5A8LNpg	0.727	0.164	152.058	...	0	09:51:00
4	65	2020-12-11	Kid Cudi	14/12/2020 22:26	242266	Solo Dolo, Pt. III	27oVCAziETRbNuo5A8LNpg	0.727	0.164	152.058	...	0	22:26:00
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5239	62	2011-09-27	J. Cole	13/12/2021 16:53	208160	Lights Please	6sZv0ECT8cEW3oZtoCuVg4	0.563	0.125	83.009	...	0	16:53:00

Data preparation

All of the data preparation process is elaborated in the notebook files. Below is a summary of this stage:

- Combine the user's streaming history with Spotipy's API
- Join tables to create the final steaming history dataframe
- Extract time series data, year, month, day etc
- Normelized track's audio features to be all between 0-1

Explenatory Data Analysis

As part of our EDA, after creating our complete dataset we had to understand the user's streaming habits in order to find patterns and decide how to build the network.

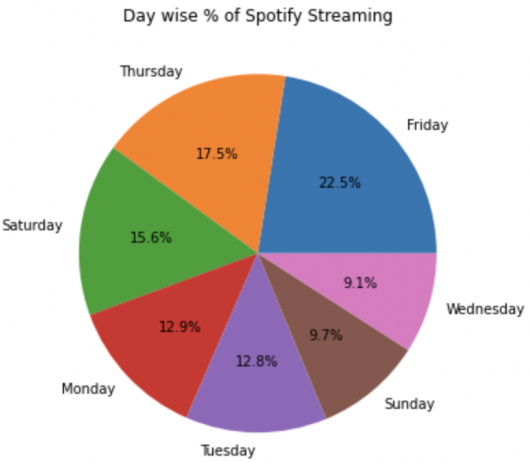
For our network building efforts, we wanted to focus on four main topics:

- Artists
- Genres
- Tracks Audio Features
- Usage Time

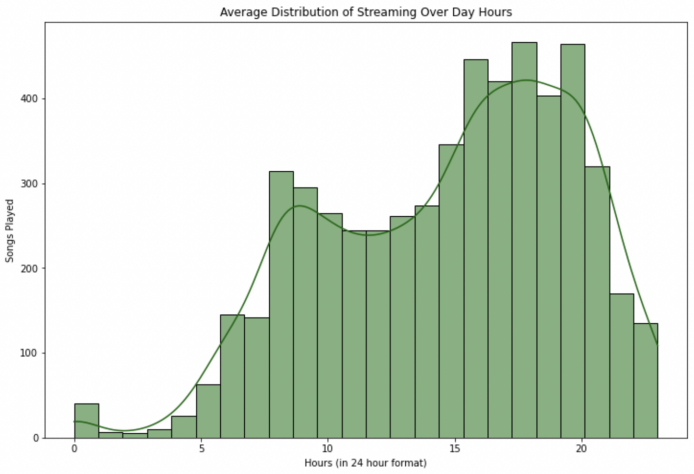
The EDA is attached in the python file

Our main conclusions are:

- Song features have great potential to use as weights in our network.
- Looking at most streamed artists by number of times streamed and total listening time in hours we see Kanye West is leading significantly in both measures. This is due to the fact that Kanye has released his album “Donda” this year after 4 years with no releases.
- Comparing streaming times by day of week we see that most of the streams accrued on Friday.



- Comparing streaming times by hour of day we see two picks, in the morning and afternoon, could be on working time,



- Genres can be a great feature to use in our network, but they are too generic as a measure of weights by themselves.

- In order to create the network, the "similar artists" function of the Spotify algorithm will be used.
- Playlists may be a useful tool to create communities of track in a network (we won't use this conclusion in our network).

Building our Network

For the network we filtered out artists that Yoni listened to **less than 40 times** in order to focus on artists he definitely liked. After filtering we remained with 44 relevant artists for our network.

With the 44 artists we wanted to create a weighted network for each artist on our list with his related artists. The network for each artist is built in the following stages:

1. Using Spotify's related artist algorithm we retrieve ~20 related artists to our core artists (the 44 artists mentiond above).

How does the spotify related artist algorithm work?

It relies primarily on two factors. The first one is shared fans— "The more fans two artists have in common, and the larger the share of each artist's total fans those shared fans represent, the more similar we consider them".

The second one involves shared descriptions—using not just the content on Spotify but also on blogs, in magazines, and at other places where music is being discussed.[2]

We will define the types of artists lists:

- Core artists - the 44 artists that were filtered from the dataset.
- Related artists - An artist related to a core artist, using the Spotify algorithm.

[2] https://artists.spotify.com/blog/how-fans-also-like-works

2. Each pair (core artist and related artist) is assigned a similarity score, which will be used as the edge's weight between the two artists. A similarity score is calculated using a formula we have created and Spotify's API as follows:
    a. For every genre both artists have in common, we increase the weight score.
    b. Normalize the genre score by dividing it with the total number of different genres the core artist has.
    c. Retrieve for each artist's top 20 streamed songs
    d. For every top song of two artists we increase the weight if there is a similarity in the song's features scores:
        i. **Danceability**: Describes how suitable a track is for dancing
        ii. **Energy**: Represents energetic tracks that feel fast, loud, and noisy
        iii. **Valence**: Describing the musical positiveness conveyed by a track
        iv. **Mode**: Major or Minor
    e. Normalize the score
    f. The final similarity score is combined by the two scores: genre score and top songs score (with normalization between 0-1).
3. Repeat the process for each core artist and his related artists
4. Create a data frame of an edges list with 3 columns:
    a. **Source** - the core artist (that the user listened to more than 40 times and played more than 30 seconds).
    b. **Target** - an artist related to him
    c. **Weight -** of the edge (the similarity score explained above)
5. Create a network using this edges list.

See the python file for further details on the network creation and similarity score calculation

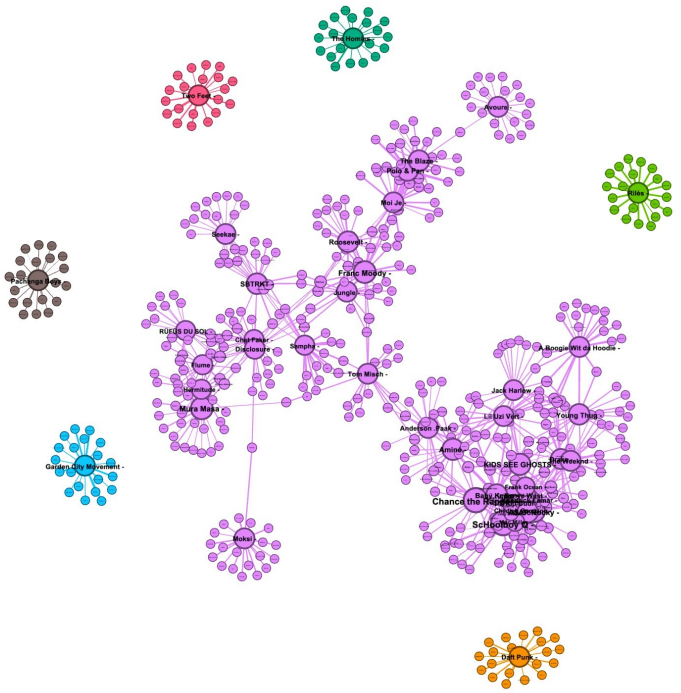
$$Genre\ score = \frac{number\ of\ common\ genres}{number\ of\ genres\ of\ core\ artist}$$

$$\begin{aligned} Audio\ Fetures\ score = \quad & (1 - (artist1\ \boldsymbol{valence}\ score - artist2\ \boldsymbol{valance}\ score)) + \\ & (1 - (artist1\ \boldsymbol{dancability}\ score - artist2\ \boldsymbol{dancability} score)) + \\ & (1 - (artist1\ \boldsymbol{energy}\ score - artist2\ \boldsymbol{energy}\ score)) + \\ & (1 - (artist1\ \ \boldsymbol{mode} - artist2\ \boldsymbol{mode})) \end{aligned}$$

$$Similarity\ score = \frac{\left(Genre\ score + \frac{Audio\ Fetures\ score}{4}\right)}{2}$$

The Network

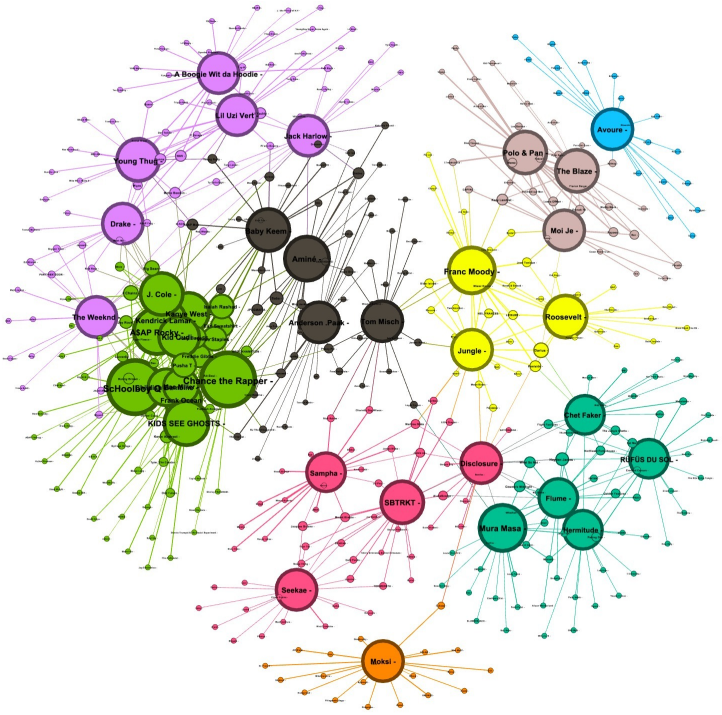
The network structure is built from core artists and related artists as nodes and the formula above defines the edges based on the weight. On the graph below we can see the hole network colored by the different components.



- We chose to define the network as undirected
- 7 components in total.
- 1 significant giant component.
- The network is not connected.
- Our highest degree ranks belong to the core artists.

Focus on the giant component

To analyse the network we decided to focus on the giant component. This is because all other 6 components are disconnected “small networks”, and each one of them only contains a core artist and 20 artists related to him. All centrality measures for these related artists will be low and we will not recommend them to the user. We can look at these 6 disconnected components of core artists as sort of “outliers” in the user’s music taste. They are artists he liked and listed to a lot but they are not related or similar to the majority of his musical taste - the giant component.



- Size of nodes by degree.
- Colors by modularity (measures the strength of division of a network into clusters).

Approche To Predict New Artists

For each artist node we calculated 3 centrality scores to be used as predictors for new artists the user will like.

- **Degree Centrality** - Measures how many core artists recommended an artist.
- **Betweenness centrality** - Captures how much a given node is in-between others. This could point to “bridge” artists connecting clusters.
- **Eigenvector centrality** - Measures an artist’s importance while considering the importance of its neighbors. He might have a lower degree but has neighbors with many recommendations.

In order to avoid reccomending our core artists we filtered them out, remaining only with related artists.

We calculated the means and standard deviations for each centrality measure and kept only artists whose score was three SD's or more from the mean, meaning they were “outliers” and had a much higher centrality score than the other artists.

Centrality measures recommendation results:

- Degree Centrality - 8 artists
- Betweenness centrality - 6 artists
- Eigenvector centrality - 7 artists

Resarch Results

We then joined the recommendations of all measures and dropped duplicates. Some of the artists were recommended both based on Degree centrality and Eigenvector centrality. This is because eigenvector centrality is an extension of degree centrality. A node with high Eigenvector centrality is not necessarily a node with high degree (the node might have few but

important links).

After dropping duplicates we are left with 15 artists to recommend.:

name	eigen_vector_cantrality	degree_cantrality	degree_caount	betweenness_centrality
NxWorries	4.986027e-04	0.004739	2	0.087897
Rejjie Snow	3.355597e-03	0.007109	3	0.153820
Phony Ppl	1.133376e-02	0.007109	3	0.247312
Loyle Carner	8.361584e-05	0.004739	2	0.207101
Darius	1.816216e-05	0.014218	6	0.188087
Tchami	2.838353e-07	0.004739	2	0.090509
Kidnap	2.463799e-06	0.004739	2	0.090509
Isaiah Rashad	1.836553e-01	0.026066	11	0.001250
Vince Staples	1.871860e-01	0.026066	11	0.000000
Earl Sweatshirt	1.574049e-01	0.023697	10	0.000687
Pusha T	2.090039e-01	0.026066	11	0.000000
Joey BadaSS	1.789283e-01	0.023697	10	0.000000
Freddie Gibbs	1.779095e-01	0.023697	10	0.000000
Jay Rock	1.206108e-01	0.011848	5	0.000000
2 Chainz	1.096706e-01	0.014218	6	0.000000

Verifying the Results

In order to evaluate our performance, we analyzed the user's streaming history without the core artists, that is artists that the user listened to but **less than 40 times.**

We then looked if the artists from our final recommendations are in this streaming list. Comparing the **full** user's streaming history with our list, he had heard 9 out of 15 artists from our list! This means 60% of our recommendations were relevant.

Conclusion and Recommendations for Further Analysis

Based on our findings, we believe the recommendation algorithm can be upgraded to determine related artists based on similarity between the audio features and genres of the artist's tracks as we have defined in this project.

Our recommendation for future work on this model:

- Use more audio features to calculate similarity score between artists
- Use Date and time attributes to recommend artists for a specific week day or time of day (e.g. Relaxing artists for the morning)
- Use user's playlists data as attributes to improve our recommendations
- Deploy our model on more users to obtain more results
- Upgrade the model to recommend not only new artists but also new tracks

Refrences that inspired the project:

1. https://www.kaggle.com/mohitkr05/spotify-data-visualization
2. https://medium.com/geekculture/spotify-data-visualization-and-analysis-using-python-4af81c5531a7
3. https://towardsdatascience.com/viz-your-music-with-spotify-api-and-plotly-eaa65f652191
4. https://www.tableau.com/about/blog/2019/7/how-visualize-spotify-music-trends-tableau
5. https://medium.com/m2mtechconnect/spotify-data-exploration-with-python-74dcc292031d
6. https://medium.com/@Infogram/how-to-visualize-data-using-the-spotify-api-and-infogram-7430f5e8525b
7. https://towardsdatascience.com/a-music-taste-analysis-using-spotify-api-and-python-e52d186db5fc
8. https://towardsdatascience.com/finding-your-next-beloved-artists-37596d48f32c
9. https://towardsdatascience.com/predicting-spotify-track-skips-49cf4a48b2a5

Thank you for tuning in and sorry for being so complicated :)