
Spotify
wRapped
Charlie.Engineer

The dataset

Taken from users who published their “#nowplaying” in tweets




Users

15.847 unique spotify users

Playlists

167.523 unique playlists

Artists

285.909 unique artists

Songs

2.786.106 unique songs

And me: *28 playlists with 1.604 songs scraped from spotify*

M. Pichl, E. Zangerle and G. Specht, "Towards a Context-Aware Music Recommendation Approach: What is Hidden in the Playlist Name?," *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, Atlantic City, NJ, USA, 2015, pp. 1360-1365, doi: 10.1109/ICDMW.2015.145.

Can we....

- see how users are linked by their playlists at a macro scale?
- use that to analyse the quality of their musical taste?
- sensibly make big playlists smaller?

The user network

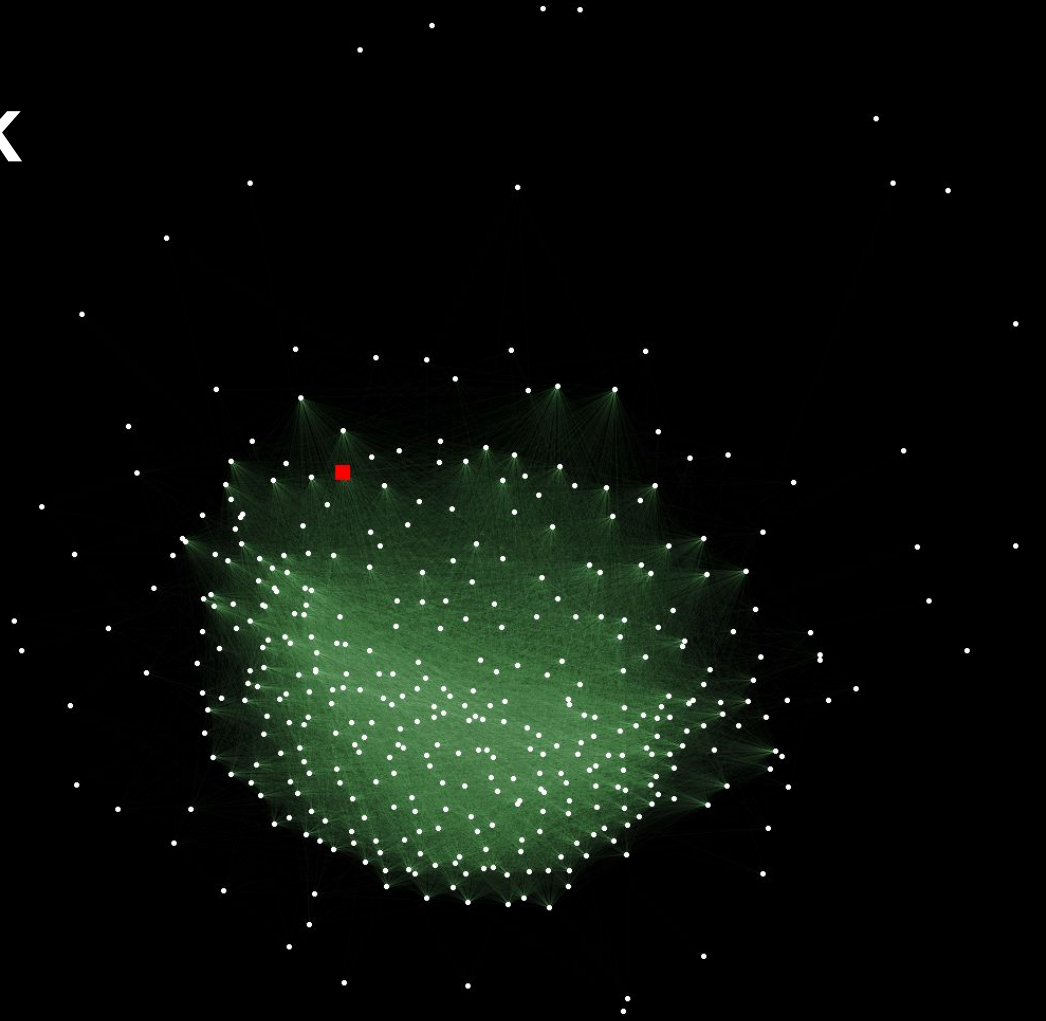
- Nodes: Users
- Edges: Songs in common

The network subset displayed:

- Transitivity: 0.71
- Diameter: 13
- Mean distance (excluding disconnected nodes): 1.97

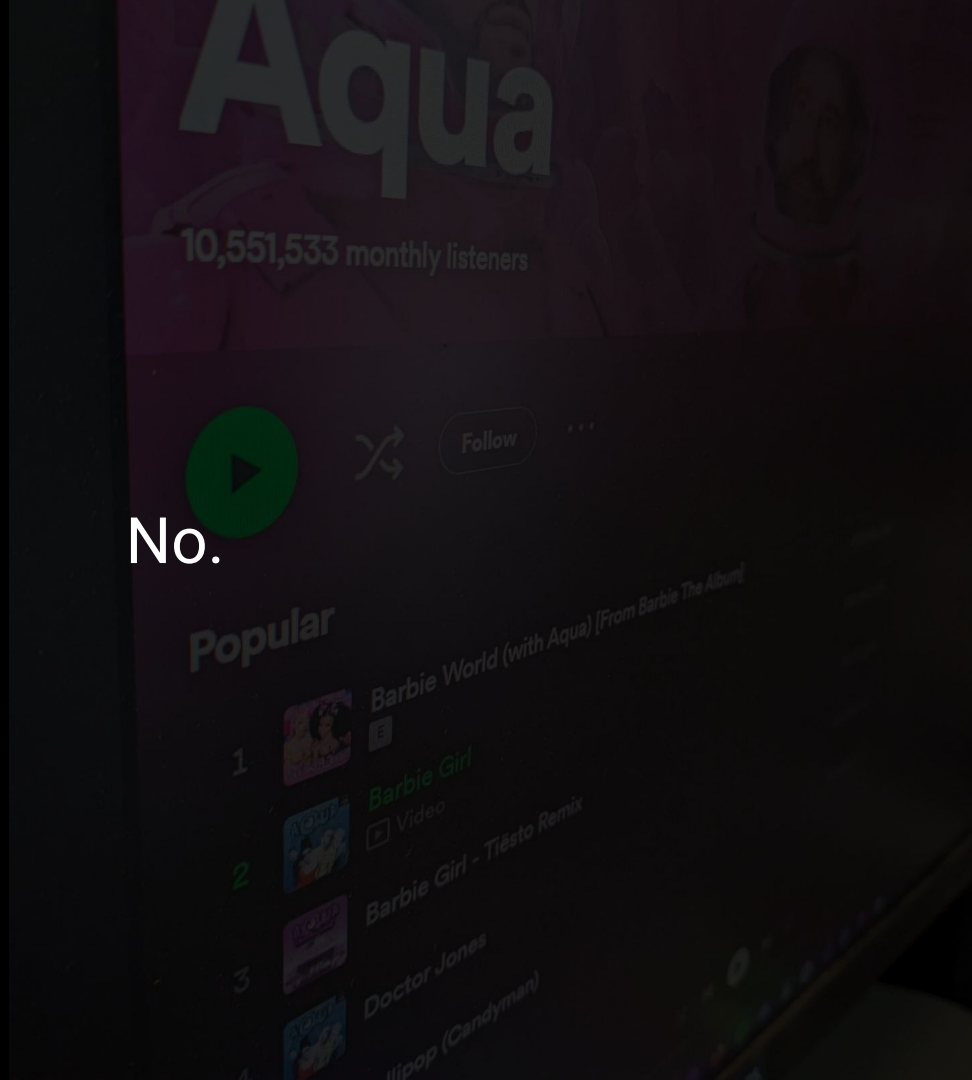
The full network:

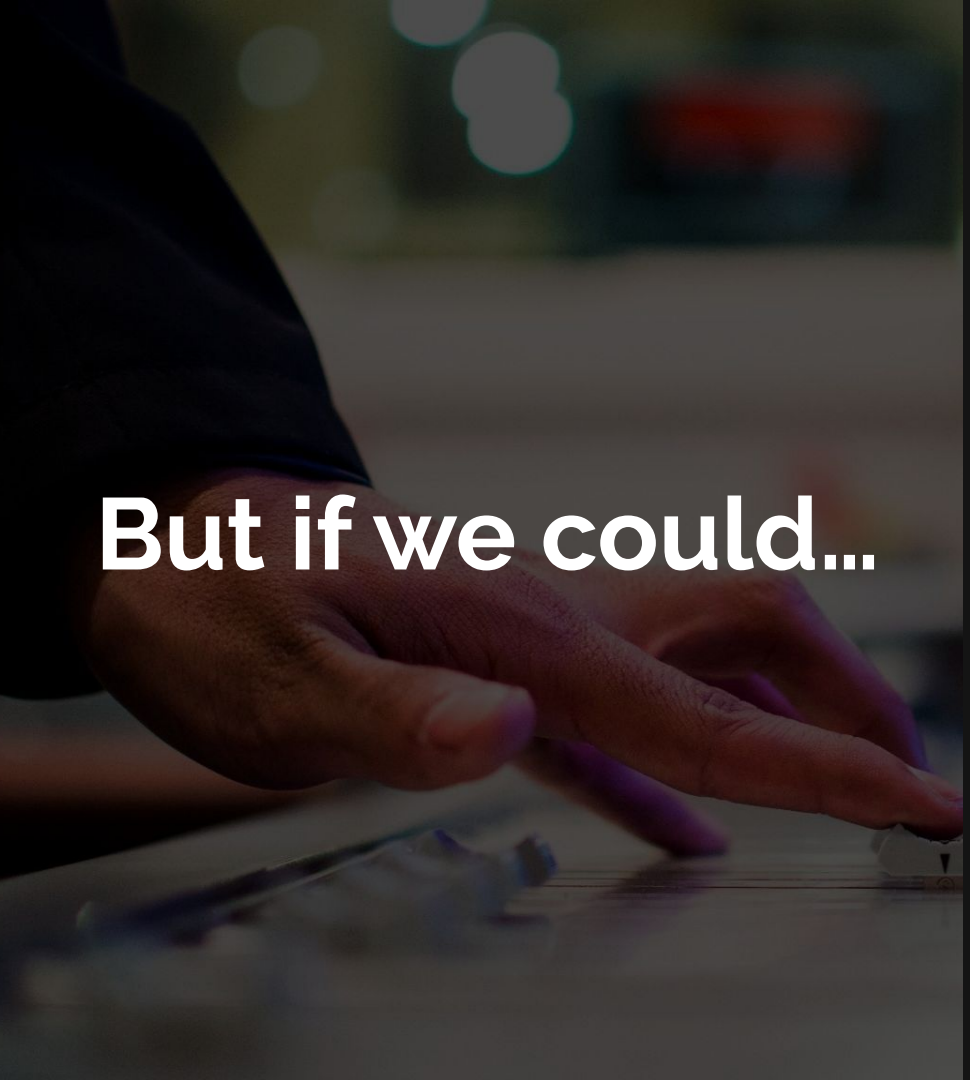
- Edge density: 0.41
- Median edges: 6929
- Degree centralization: 0.50



Can we define good taste in music?

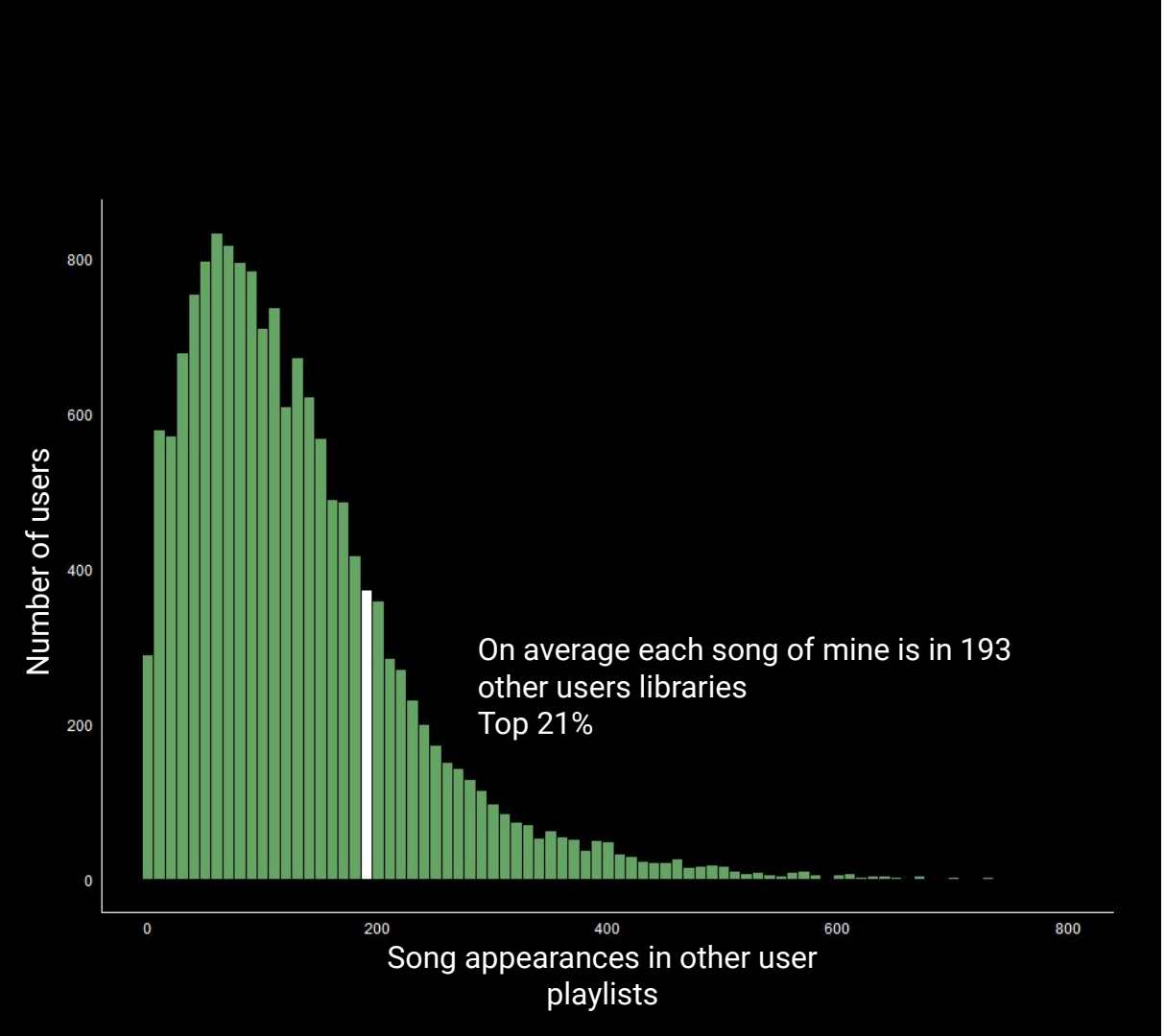
No.





But if we could...

- Wide appeal - could we put on a song everybody loves?
- Interestingness - could we put on a song nobody’s heard of before?

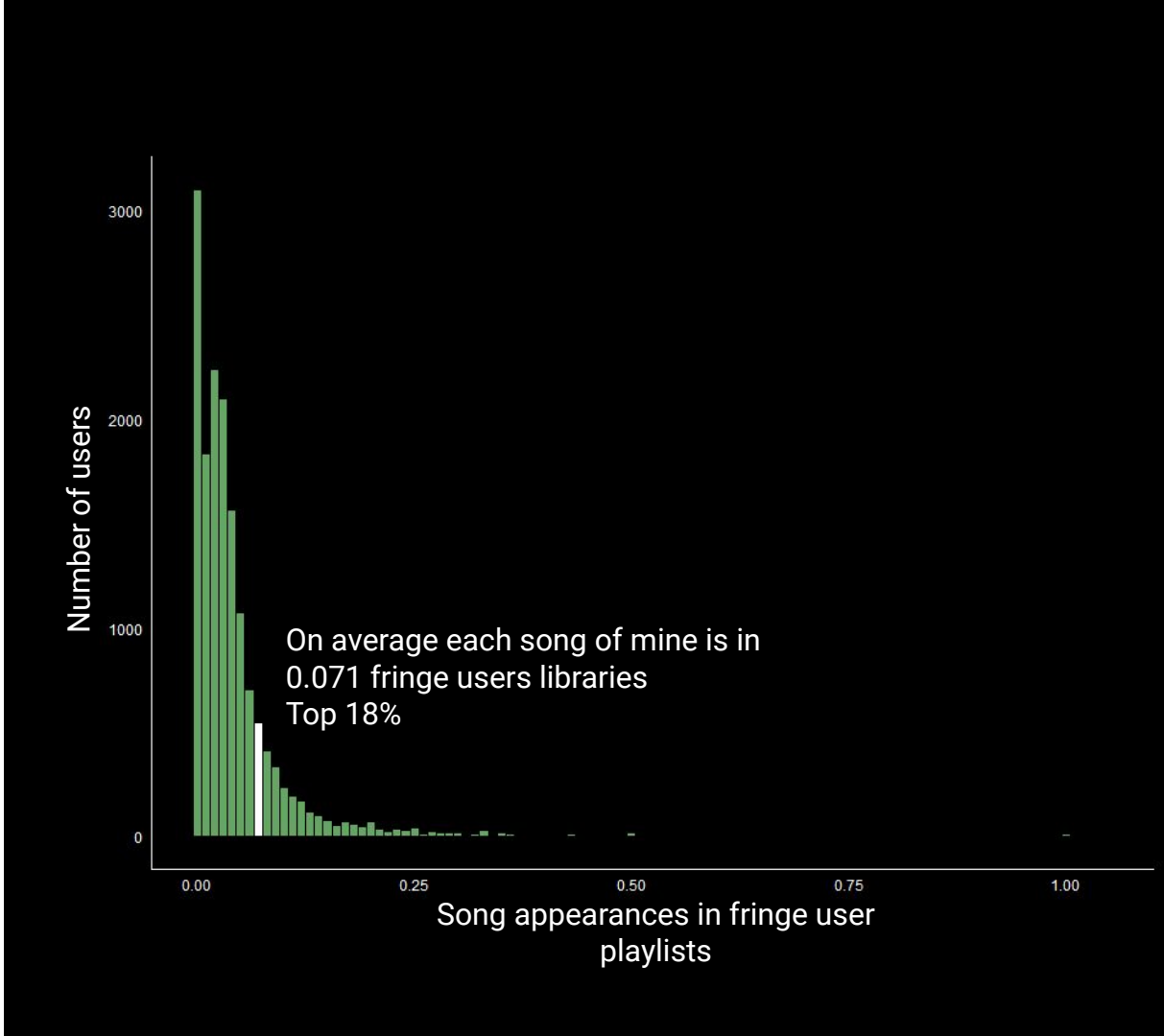


Wide appeal

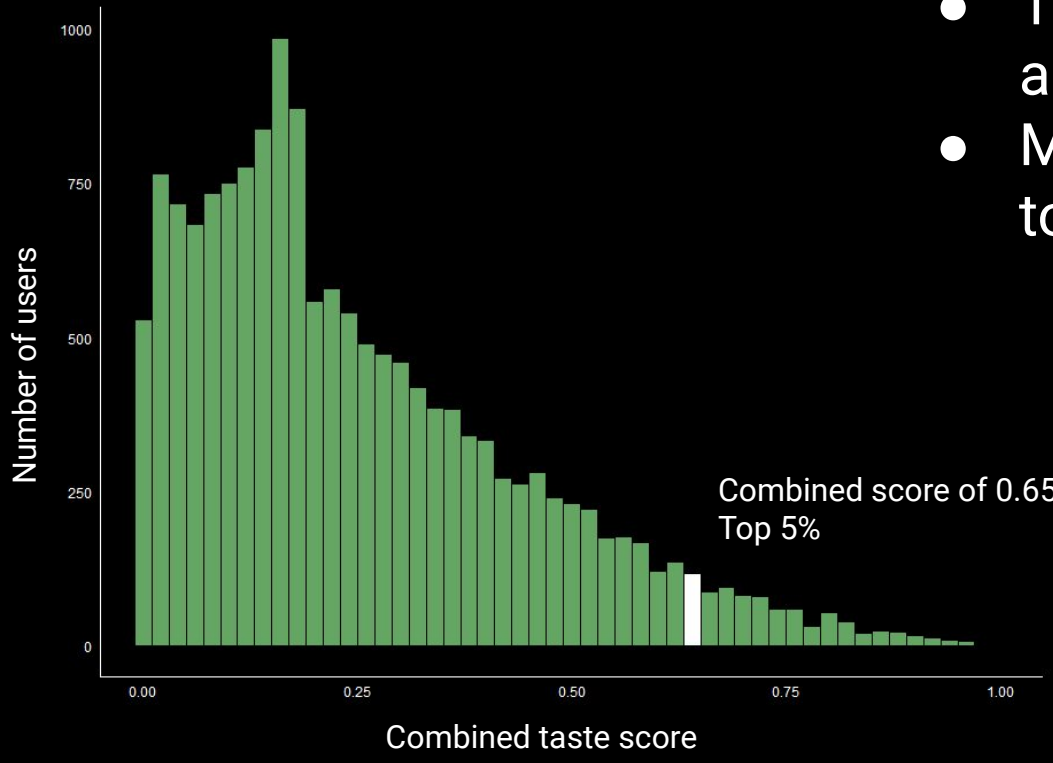
- For each user, how many times, on average, does each song appear in somebody else's playlists?

Interestingness

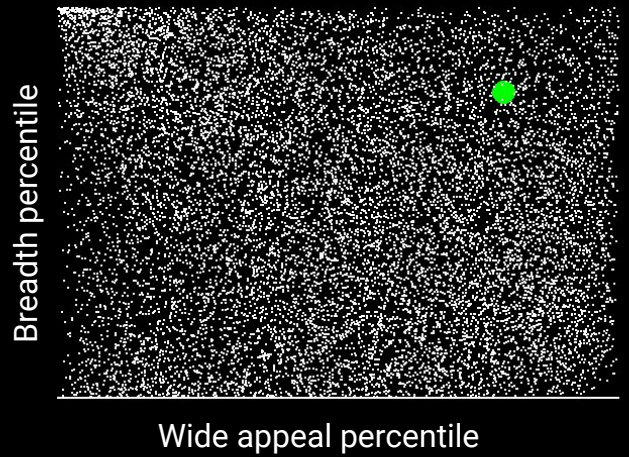
- Do we like interesting music, music on the fringes?
- How many times, on average, does each of our songs appear in a fringe user's playlist?



Overall score



- Take into account both wide appeal and interestingness
- Multiply the percentiles of each to get a combined “taste score”



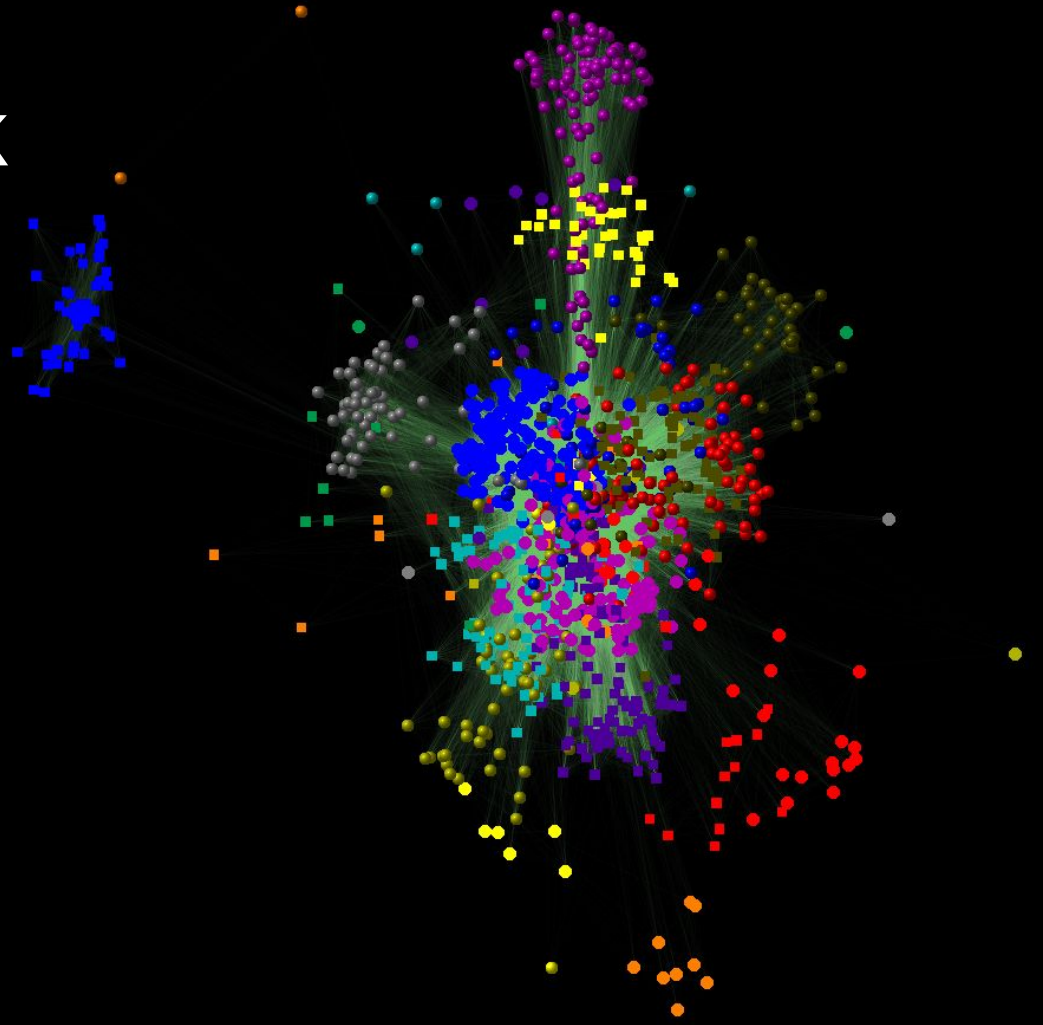
Changing playlists from monolithic to manageable

Using the wisdom of the masses
to make large and varied playlists
into multiple smaller and
focused playlists

The song network

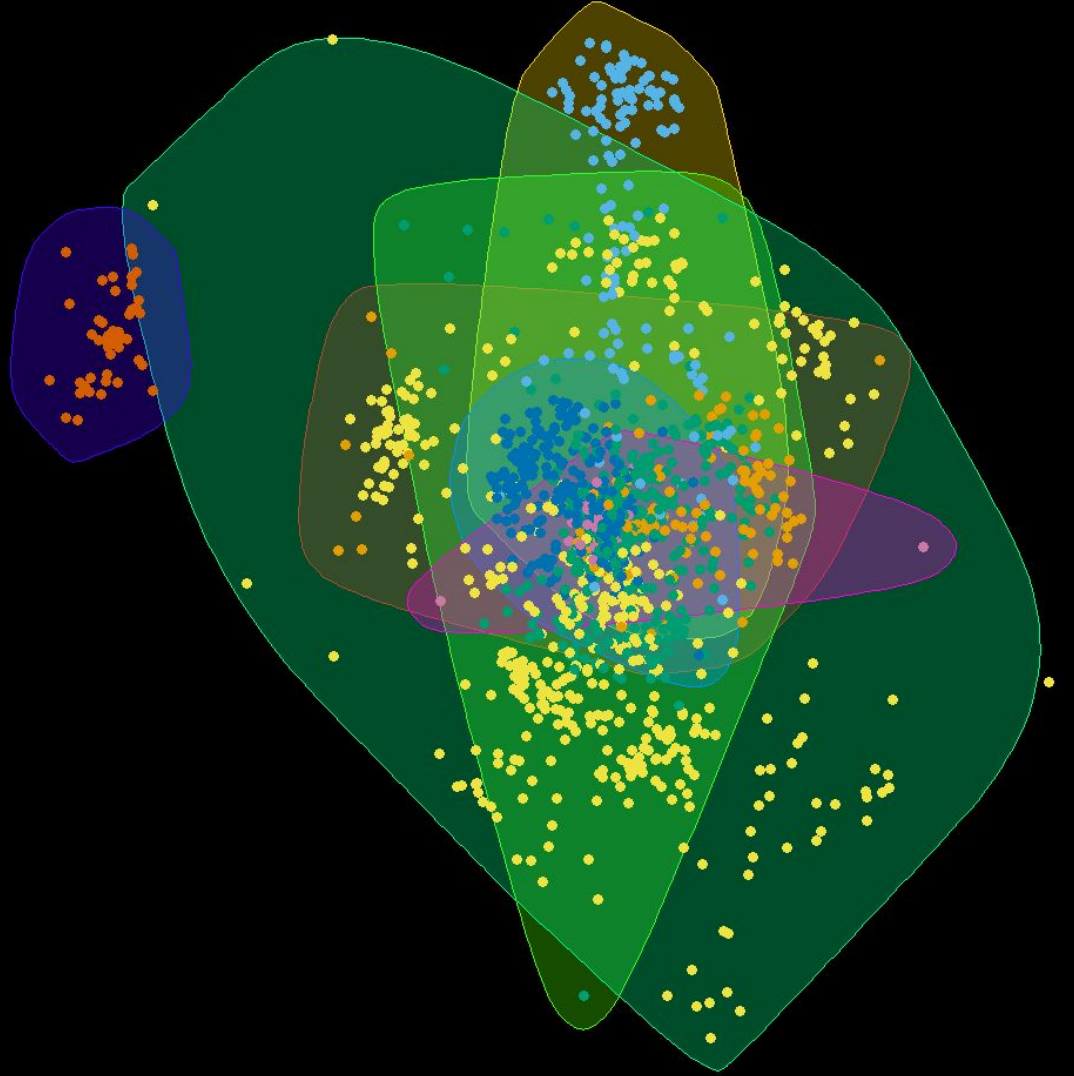
- Nodes: Songs (exclusively from my library, identified by my assigned playlist)
- Edges: Playlists

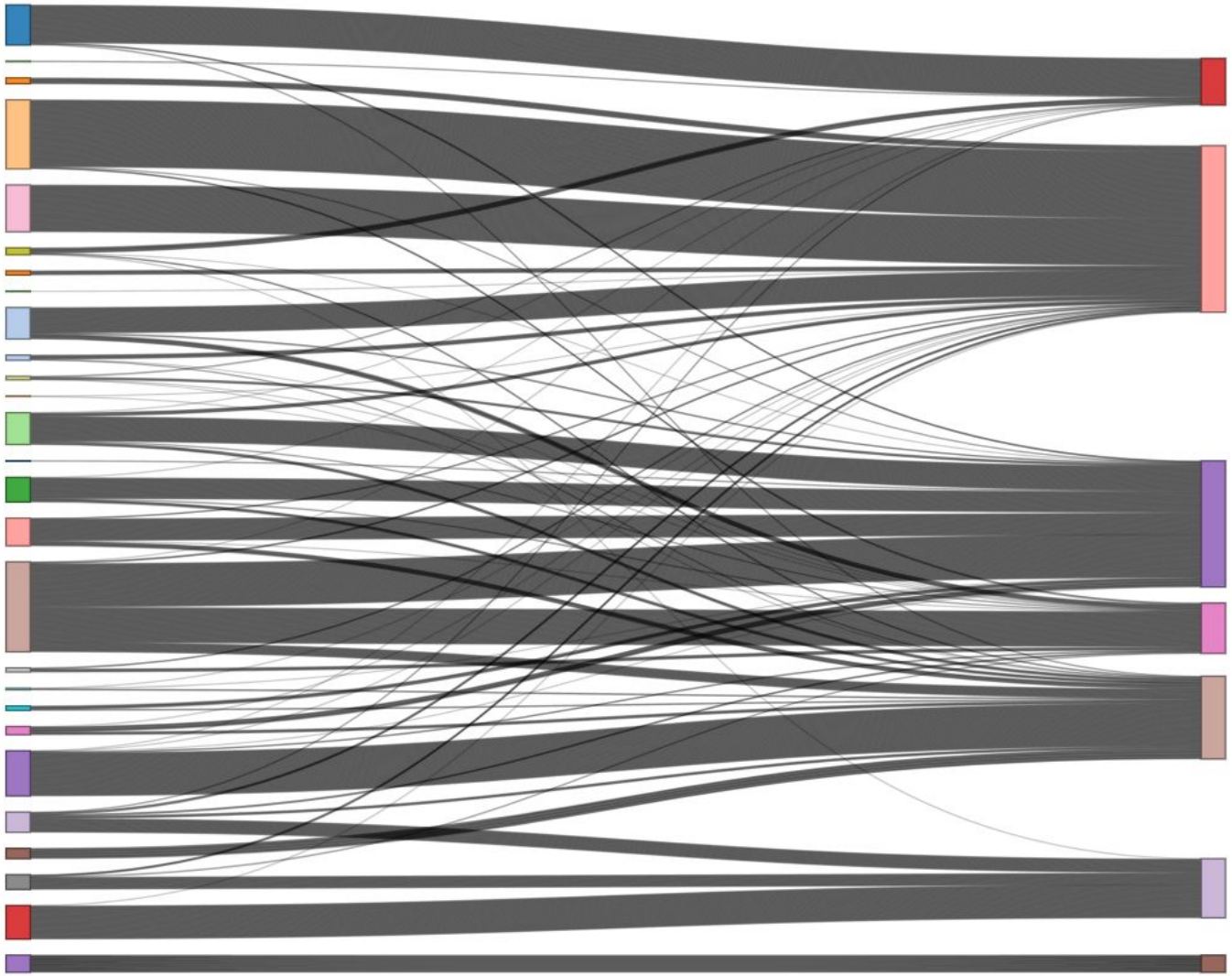
- Edge density: 0.23
- Transitivity: 0.68
- Diameter: 4
- Mean distance (excluding disconnected nodes): 1.9



Grouping by community analysis

- Nodes: Songs (coloured by Louvain assigned community)
- Coloured blobs: External limits of the different communities





Objectively, how does it do?

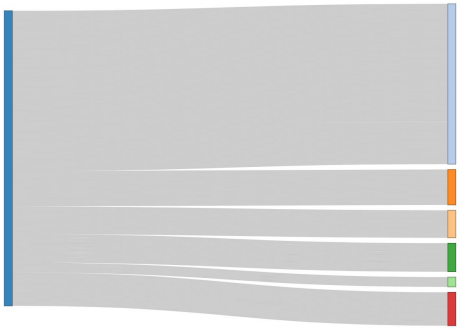
On average, 83% of each source playlist's songs end up in an output community together

Testing the process

Playlist to be split:

Millennial guitar music

- 311 songs
- 20 hours
- General theme:
  - Varied guitar music
  - 1990s-2010s
  - Indie, grunge, punk, hard rock, stoner rock, lo-fi, britpop…



Subjectively, how did it do?

Playlist 2:
Alice In Chains
Alien Ant Farm
Bloodhound Gang
Foo Fighters
Hole
Lenny Kravitz
New Radicals
Nirvana
Pearl Jam
Radiohead
Red Hot Chili Peppers
Soundgarden
Tenacious D
The Breeders
The Cranberries
The Offspring
Velvet Revolver

Playlist 3:
alt-J
Arcade Fire
Cold War Kids
Courtney Barnett
Django Django
Feist
Florence + The Machine
Foster The People
Imagine Dragons
Jack White
Lana Del Rey
Metronomy
Milky Chance
Passenger
Tame Impala
The Black Keys
The Heavy
The Kills
The War On Drugs
The xx

Playlist 4:
Amy Winehouse
Electric Six
Franz Ferdinand
Gorillaz
Jet
Kaiser Chiefs
Keane
Maroon 5
MGMT
Modest Mouse
OK Go
The Hives
The Killers
The Strokes
The Subways
The White Stripes
Vampire Weekend
Wolfmother

Thank you

Github:

Top 5% musical taste: