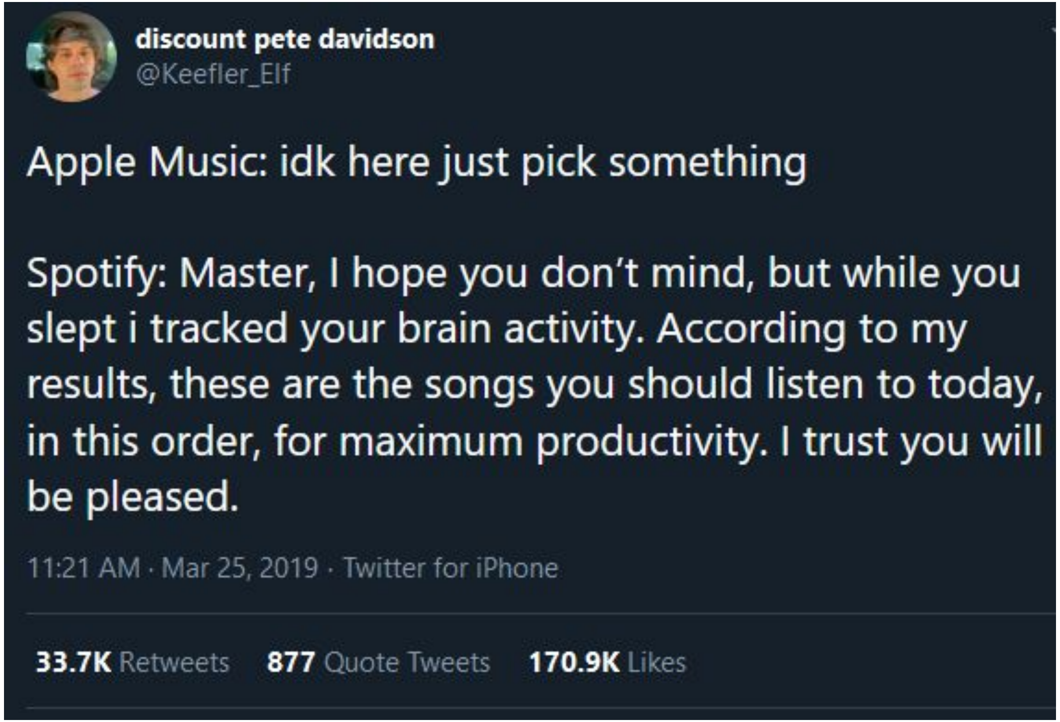
Clustering Music

Unsupervised Learning Based on Spotify Audio Analysis Data

Context

- Brand new music streaming service
- No user data
- Just a library of songs
- Need some kind of recommendation method



discount pete davidson
@Keefler_Elf

Apple Music: idk here just pick something

Spotify: Master, I hope you don't mind, but while you slept i tracked your brain activity. According to my results, these are the songs you should listen to today, in this order, for maximum productivity. I trust you will be pleased.

11:21 AM · Mar 25, 2019 · Twitter for iPhone

33.7K Retweets 877 Quote Tweets 170.9K Likes

How can we make song recommendations with only audio analysis data?

Roadmap

- The Data
    - Collection
    - Processing
- Clusters
    - Dimensionality Reduction
    - Model Selection
- Interpretation
    - Genre Density
    - Artist vs Song Similarity
- Conclusions
    - Limitations
    - Steps Forward

The Data

- >13,000 unique song ids
- >9,000 unique artist ids
- >2,000 different genres
- 90 initial analysis features
  - >4 hours of code runtime
- 10 high-level interpretation features
  - Not used in clustering

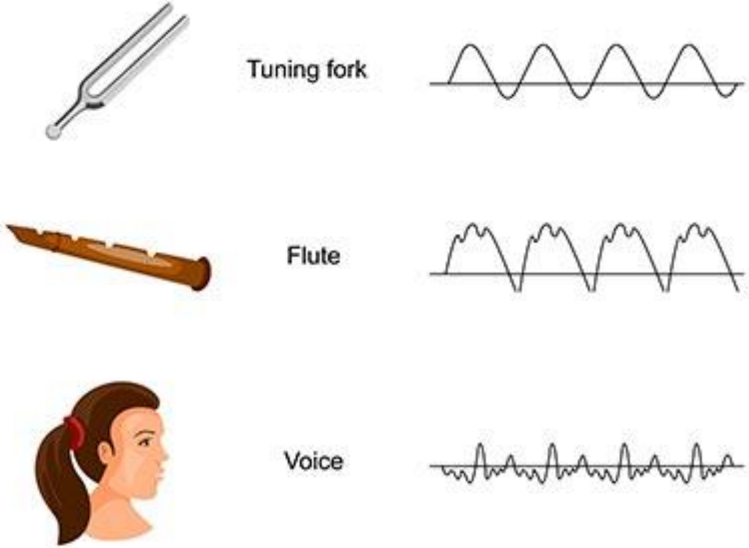
Collection

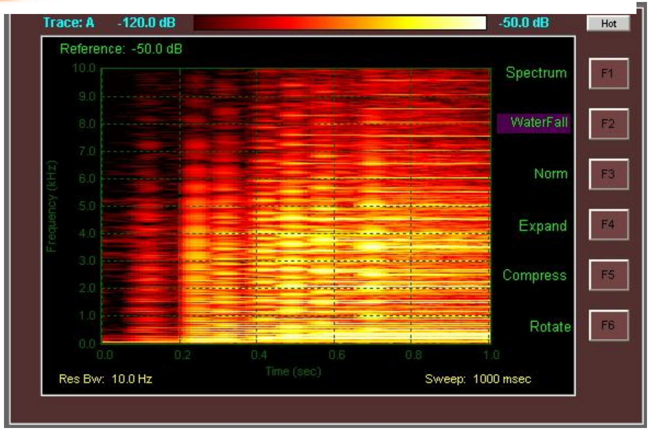
- Spotify Web API
- “Audio Features”
    - High level / processed
    - Acousticness, valence, energy, etc.
- “Audio Analysis”
    - Low level / technical
    - Pitch vectors, timbre vectors, amplitude, etc.

Timbre

API documentation:

- Timbre is the quality of a musical note or sound that distinguishes different types of musical instruments, or voices.
- Timbre vectors are best used in comparison with each other.



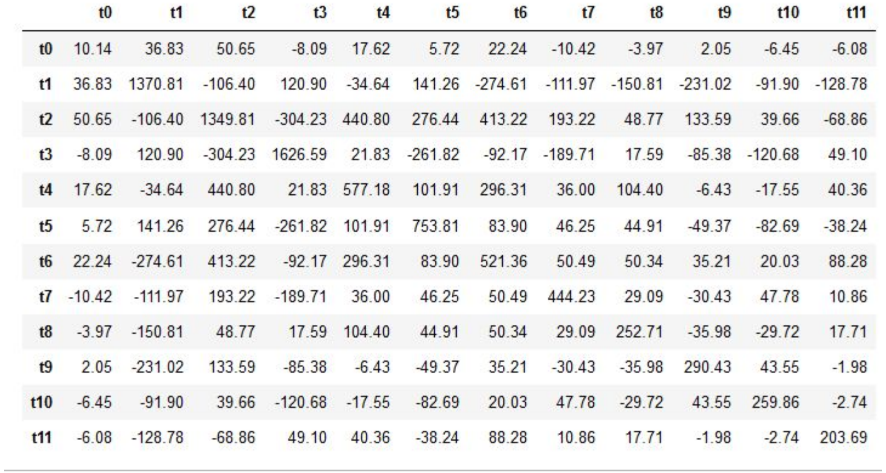


Processing

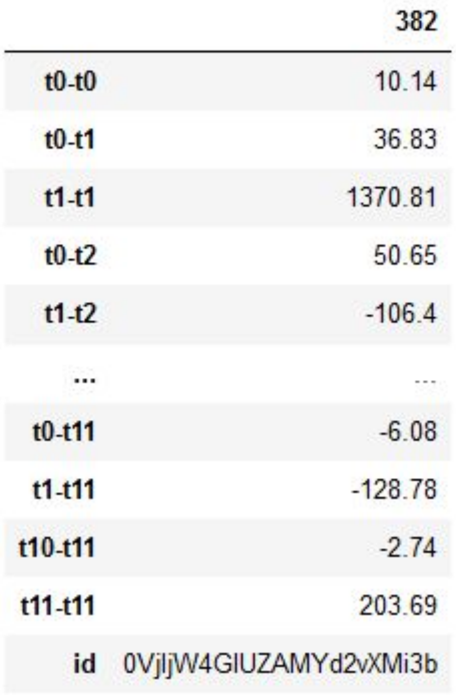
Arrays of varying size

Mean Vectors and Covariance Matrix

1-D pairwise relationships

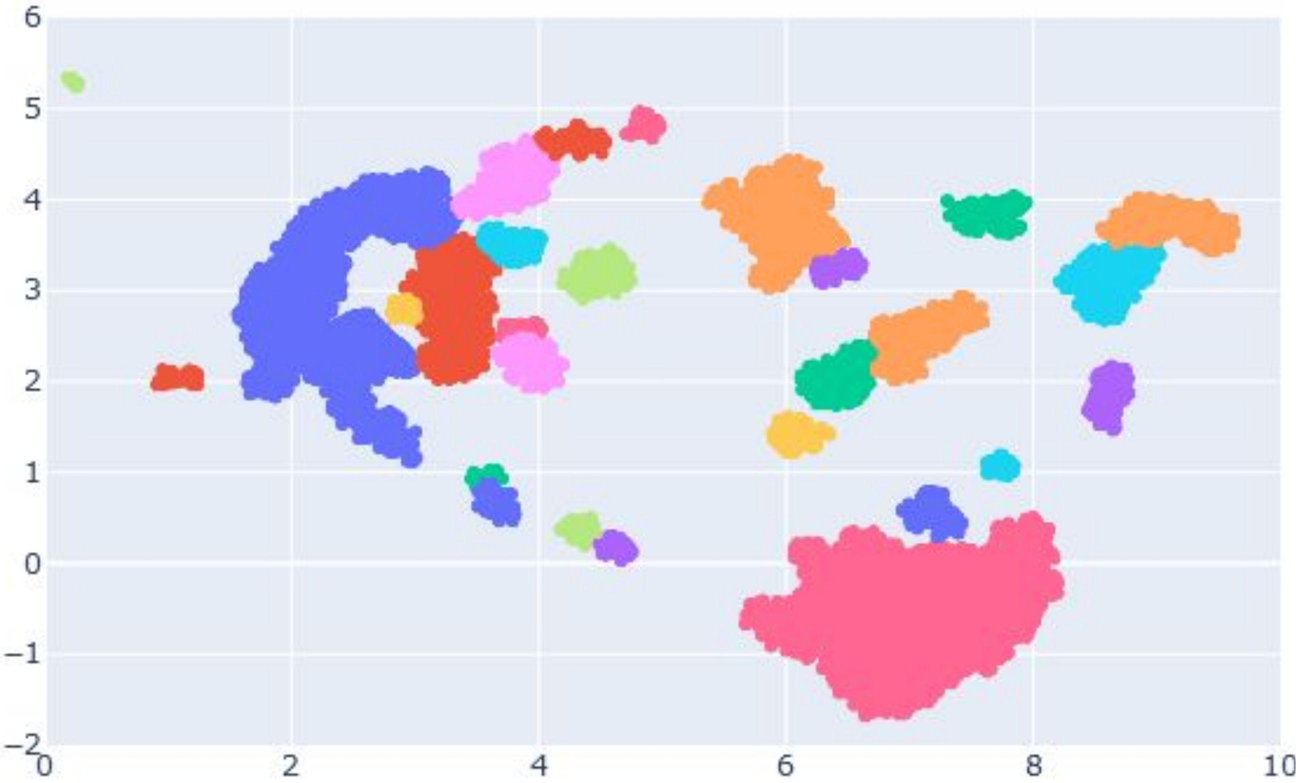


	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
t0	10.14	36.83	50.65	-8.09	17.62	5.72	22.24	-10.42	-3.97	2.05	-6.45	-6.08
t1	36.83	1370.81	-106.40	120.90	-34.64	141.26	-274.61	-111.97	-150.81	-231.02	-91.90	-128.78
t2	50.65	-106.40	1349.81	-304.23	440.80	276.44	413.22	193.22	48.77	133.59	39.66	-68.86
t3	-8.09	120.90	-304.23	1626.59	21.83	-261.82	-92.17	-189.71	17.59	-85.38	-120.68	49.10
t4	17.62	-34.64	440.80	21.83	577.18	101.91	296.31	36.00	104.40	-6.43	-17.55	40.36
t5	5.72	141.26	276.44	-261.82	101.91	753.81	83.90	46.25	44.91	-49.37	-82.69	-38.24
t6	22.24	-274.61	413.22	-92.17	296.31	83.90	521.36	50.49	50.34	35.21	20.03	88.28
t7	-10.42	-111.97	193.22	-189.71	36.00	46.25	50.49	444.23	29.09	-30.43	47.78	10.86
t8	-3.97	-150.81	48.77	17.59	104.40	44.91	50.34	29.09	252.71	-35.98	-29.72	17.71
t9	2.05	-231.02	133.59	-85.38	-6.43	-49.37	35.21	-30.43	-35.98	290.43	43.55	-1.98
t10	-6.45	-91.90	39.66	-120.68	-17.55	-82.69	20.03	47.78	-29.72	43.55	259.86	-2.74
t11	-6.08	-128.78	-68.86	49.10	40.36	-38.24	88.28	10.86	17.71	-1.98	-2.74	203.69



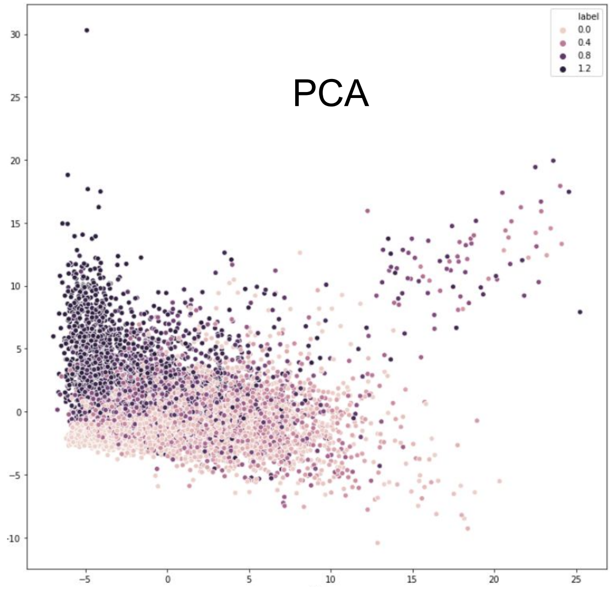
	382
t0-t0	10.14
t0-t1	36.83
t1-t1	1370.81
t0-t2	50.65
t1-t2	-106.4
...	...
t0-t11	-6.08
t1-t11	-128.78
t10-t11	-2.74
t11-t11	203.69
id	0VjljW4GlUZAMYd2vXMi3b

Clusters

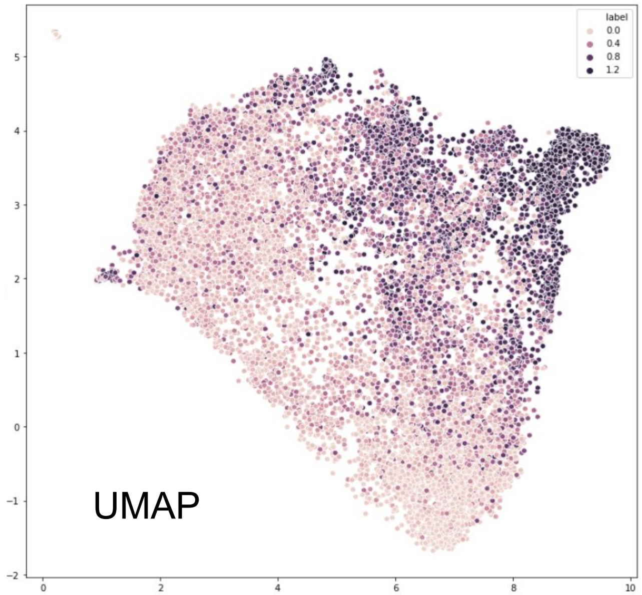


Dimensionality Reduction

- Balanced multicollinearity
- PCA - 2 components only 25% of variance
- UMAP - Semi-Global approach

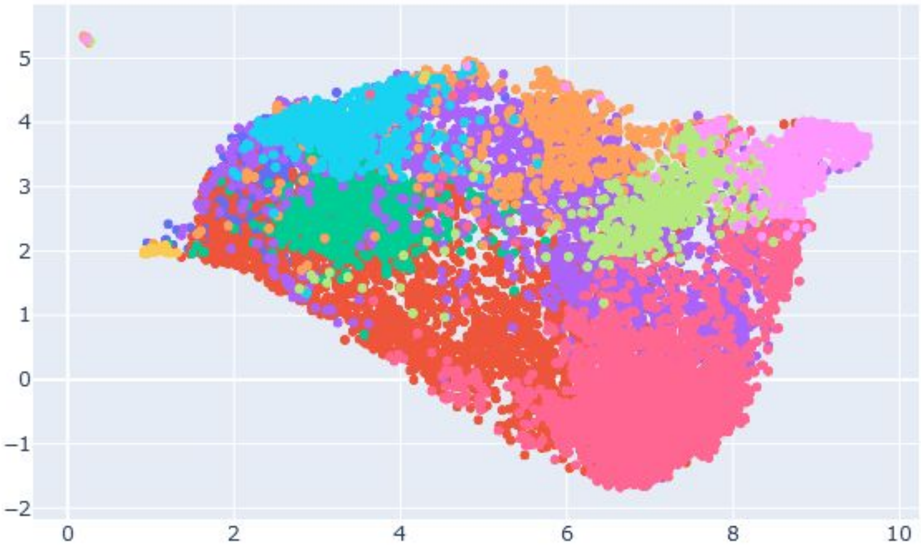


Colored by “acousticness”



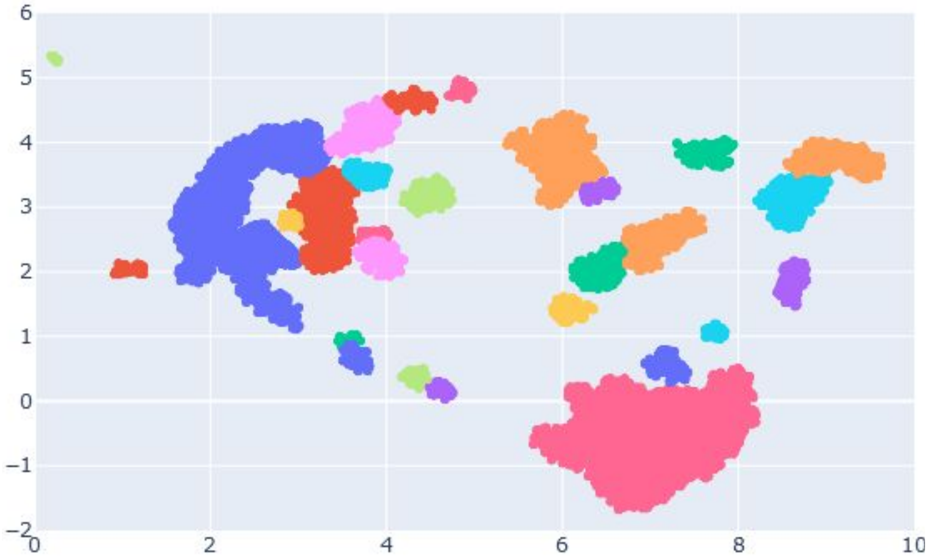
Model Selection

K-Means before UMAP

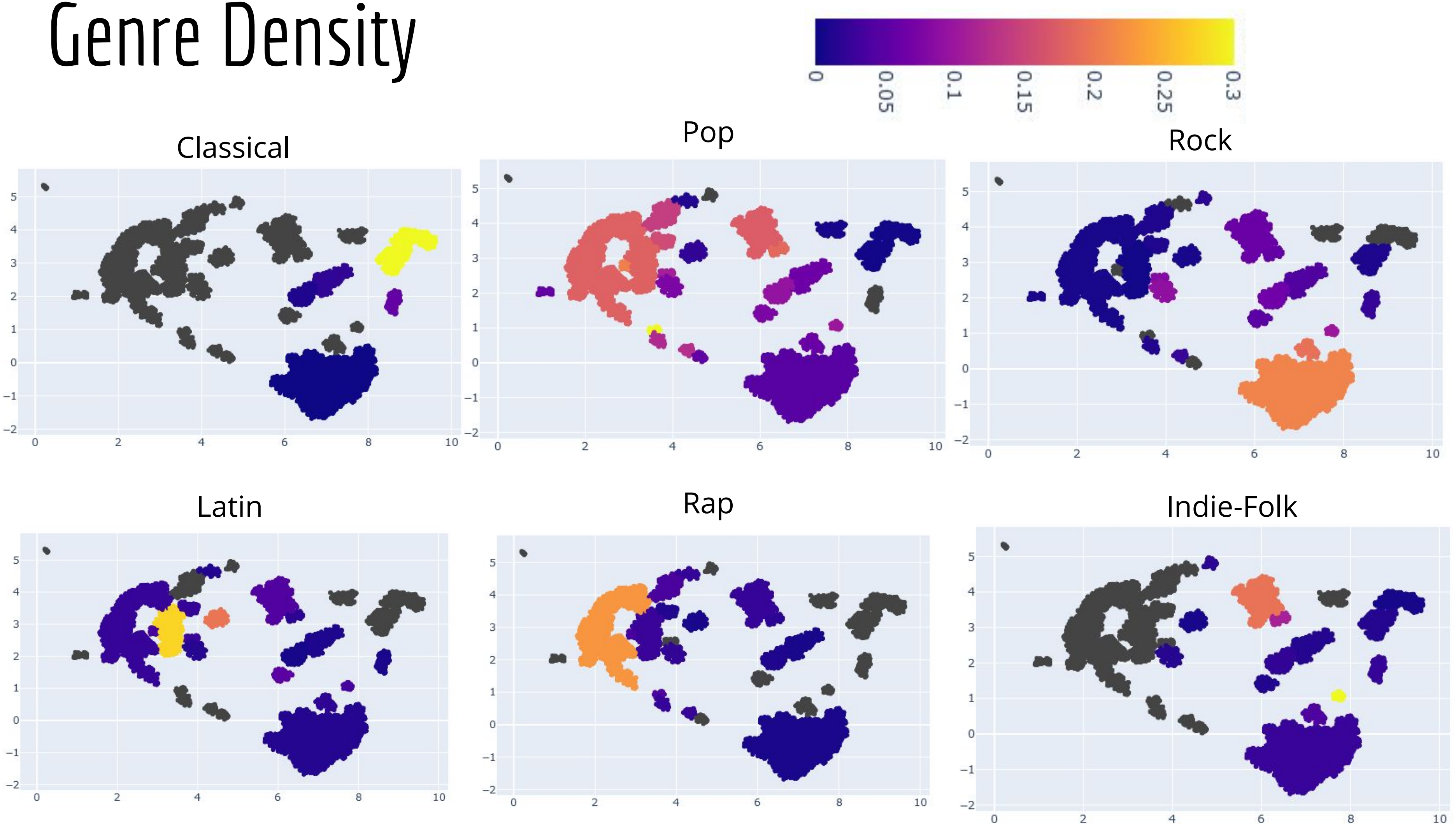


DBSCAN after UMAP

- 1/4th of the data is ‘noise’



Genre Density



Classical

Pop

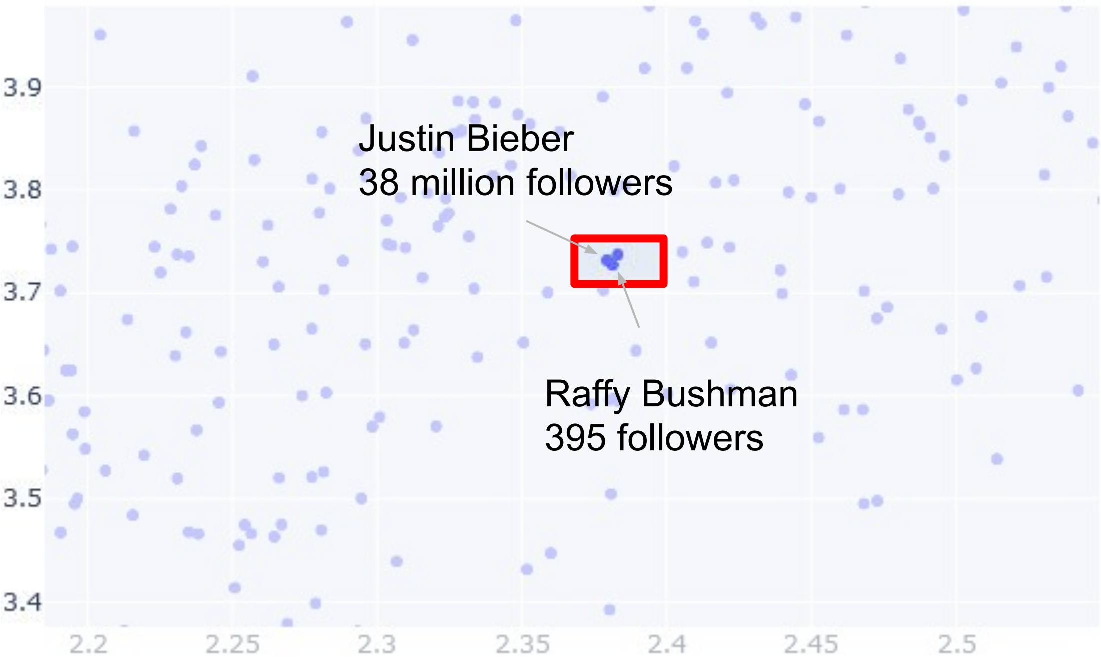
Rock

Latin

Rap

Indie-Folk

Song Similarity

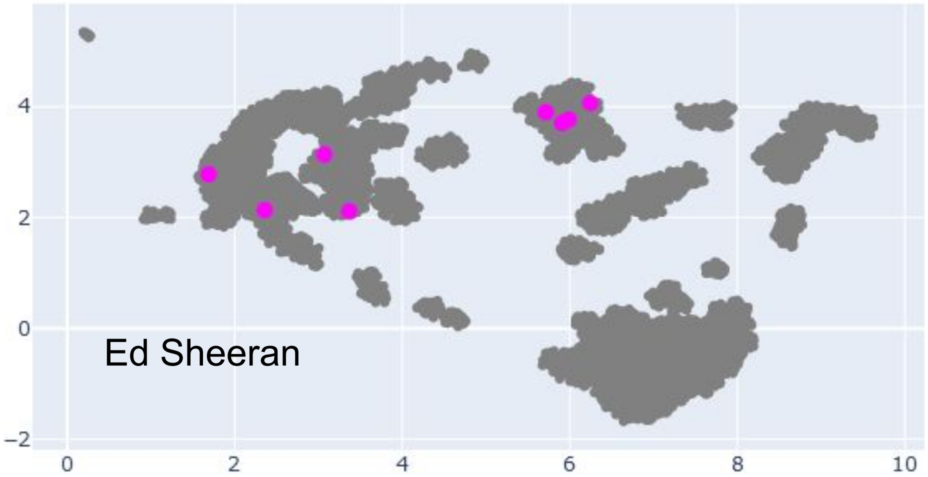


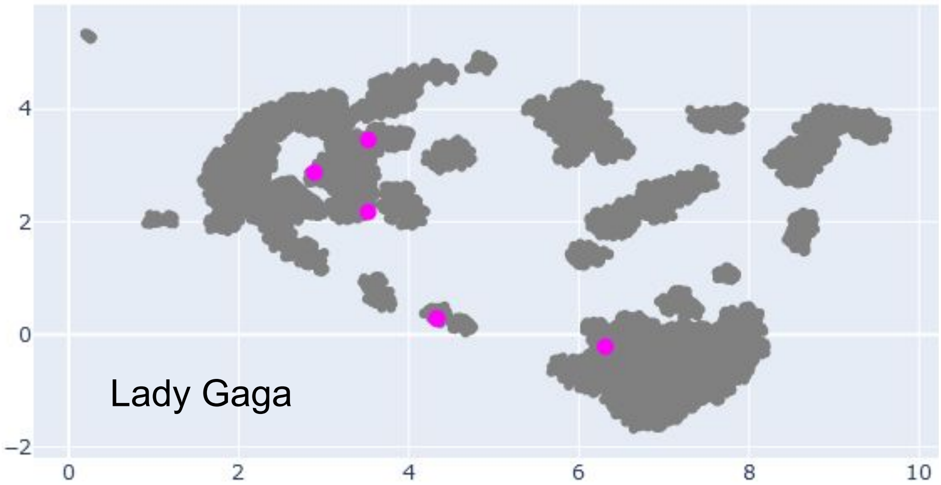
Justin Bieber
38 million followers

Raffy Bushman
395 followers

Intra-Artist Similarity

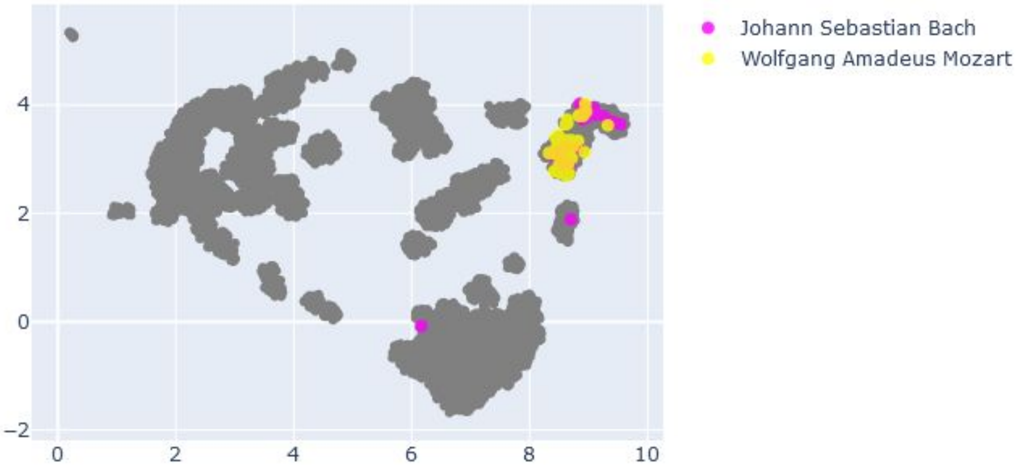
- Proximity implies similarity
- Cluster group can imply song type
- Not strong enough to individually classify song genre

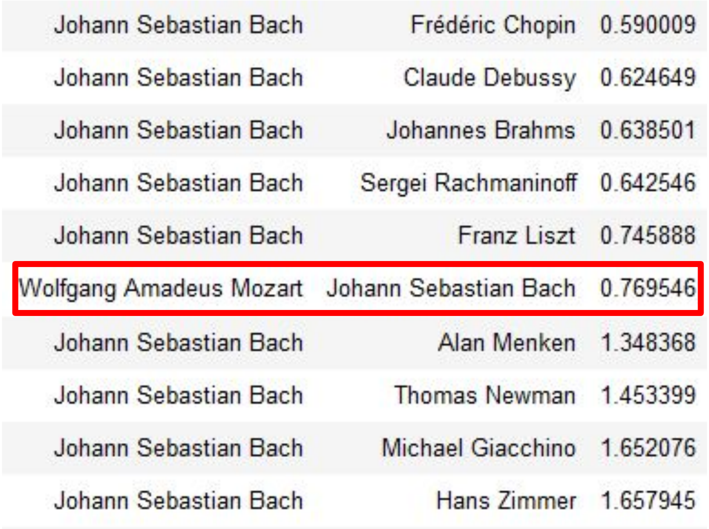




Inter-Artist Similarity

- Average euclidean distance between songs
- Semi-reliable to recommend similar artists

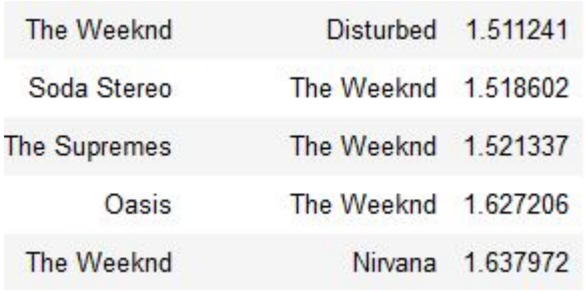




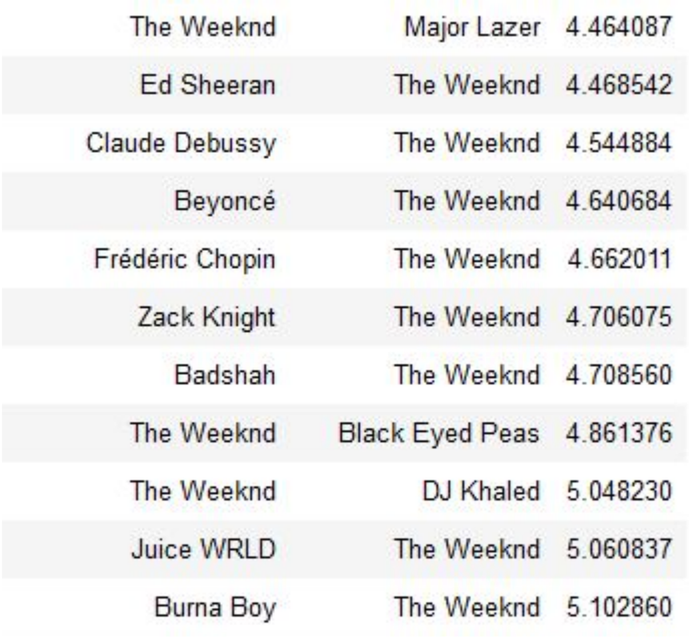
Johann Sebastian Bach	Frédéric Chopin	0.590009
Johann Sebastian Bach	Claude Debussy	0.624649
Johann Sebastian Bach	Johannes Brahms	0.638501
Johann Sebastian Bach	Sergei Rachmaninoff	0.642546
Johann Sebastian Bach	Franz Liszt	0.745888
Wolfgang Amadeus Mozart	Johann Sebastian Bach	0.769546
Johann Sebastian Bach	Alan Menken	1.348368
Johann Sebastian Bach	Thomas Newman	1.453399
Johann Sebastian Bach	Michael Giacchino	1.652076
Johann Sebastian Bach	Hans Zimmer	1.657945

Inter-Artist Similarity

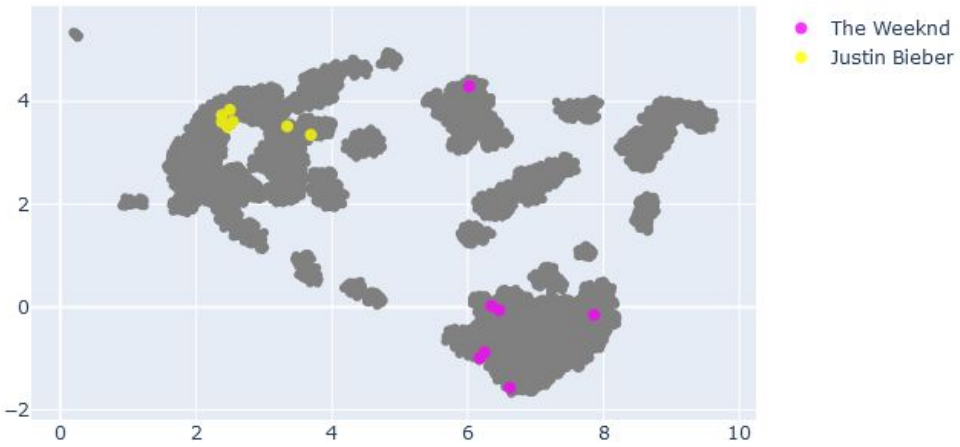
- *Semi*-reliable



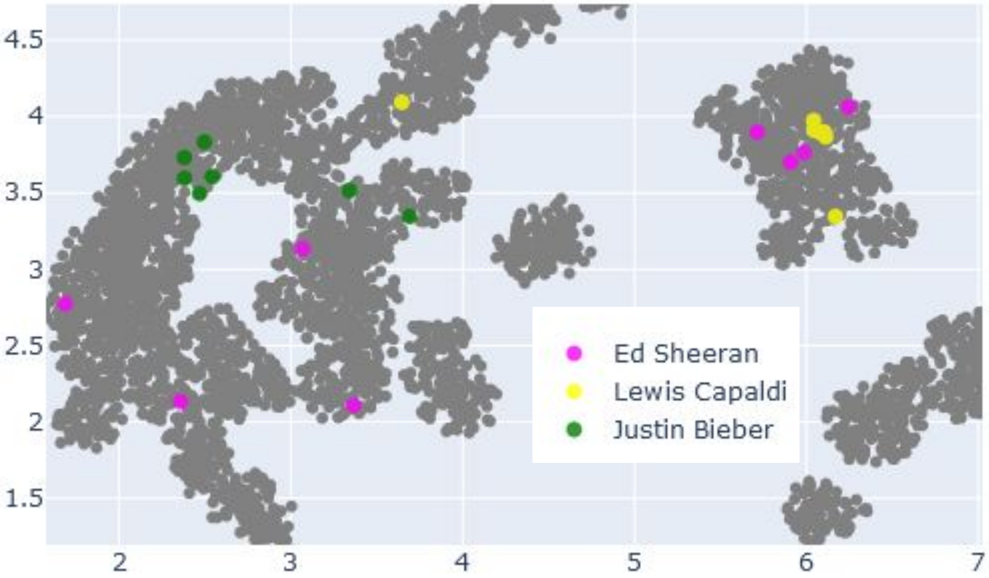
The Weeknd	Disturbed	1.511241
Soda Stereo	The Weeknd	1.518602
The Supremes	The Weeknd	1.521337
Oasis	The Weeknd	1.627206
The Weeknd	Nirvana	1.637972



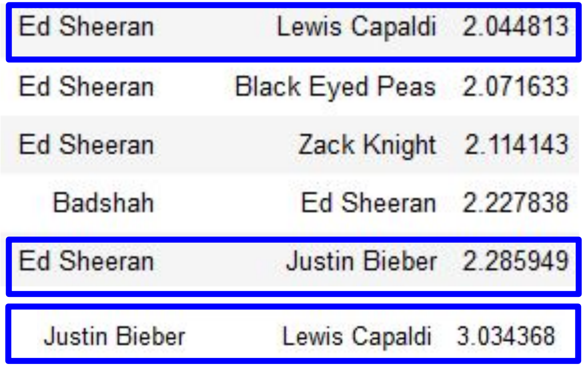
The Weeknd	Major Lazer	4.464087
Ed Sheeran	The Weeknd	4.468542
Claude Debussy	The Weeknd	4.544884
Beyoncé	The Weeknd	4.640684
Frédéric Chopin	The Weeknd	4.662011
Zack Knight	The Weeknd	4.706075
Badshah	The Weeknd	4.708560
The Weeknd	Black Eyed Peas	4.861376
The Weeknd	DJ Khaled	5.048230
Juice WRLD	The Weeknd	5.060837
Burna Boy	The Weeknd	5.102860



Naive Recommendation Algorithm

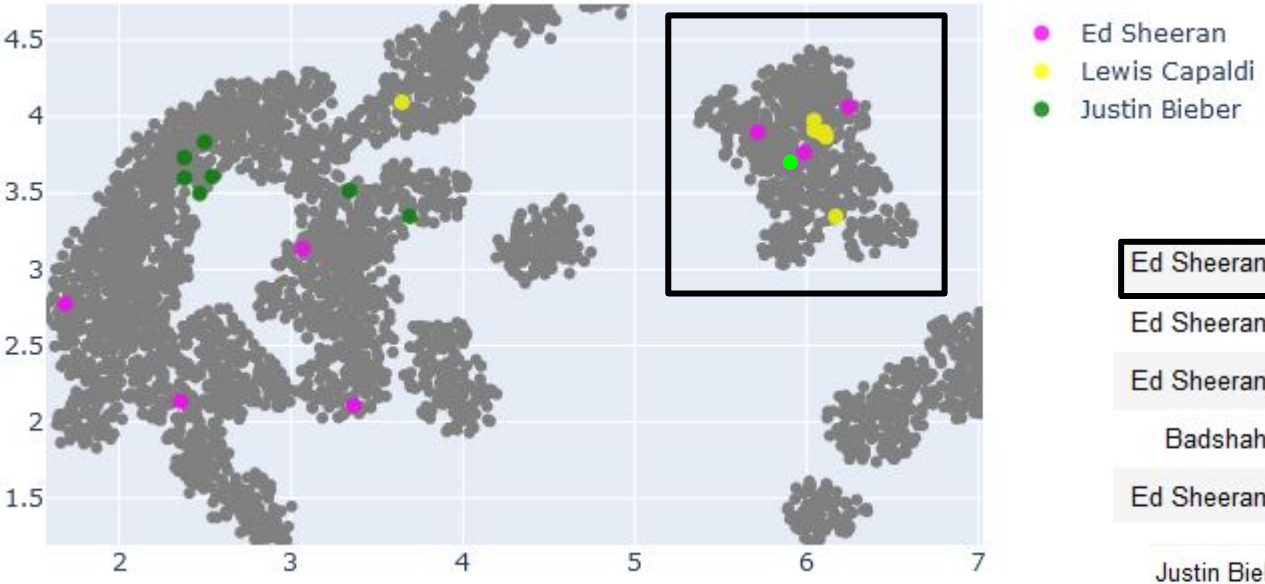


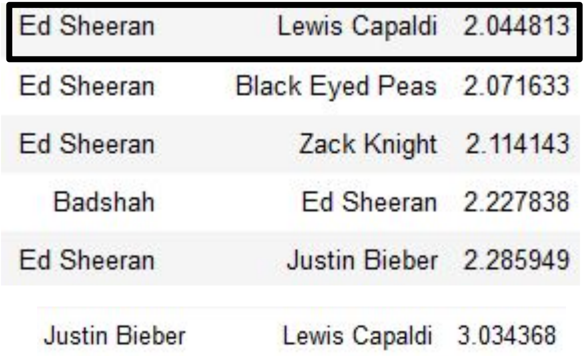
1. Filter songs by overall artist similarity
2. Find the closest proximity song to the one currently playing
3. Prioritize the artist that's closest overall



Ed Sheeran	Lewis Capaldi	2.044813
Ed Sheeran	Black Eyed Peas	2.071633
Ed Sheeran	Zack Knight	2.114143
Badshah	Ed Sheeran	2.227838
Ed Sheeran	Justin Bieber	2.285949
Justin Bieber	Lewis Capaldi	3.034368

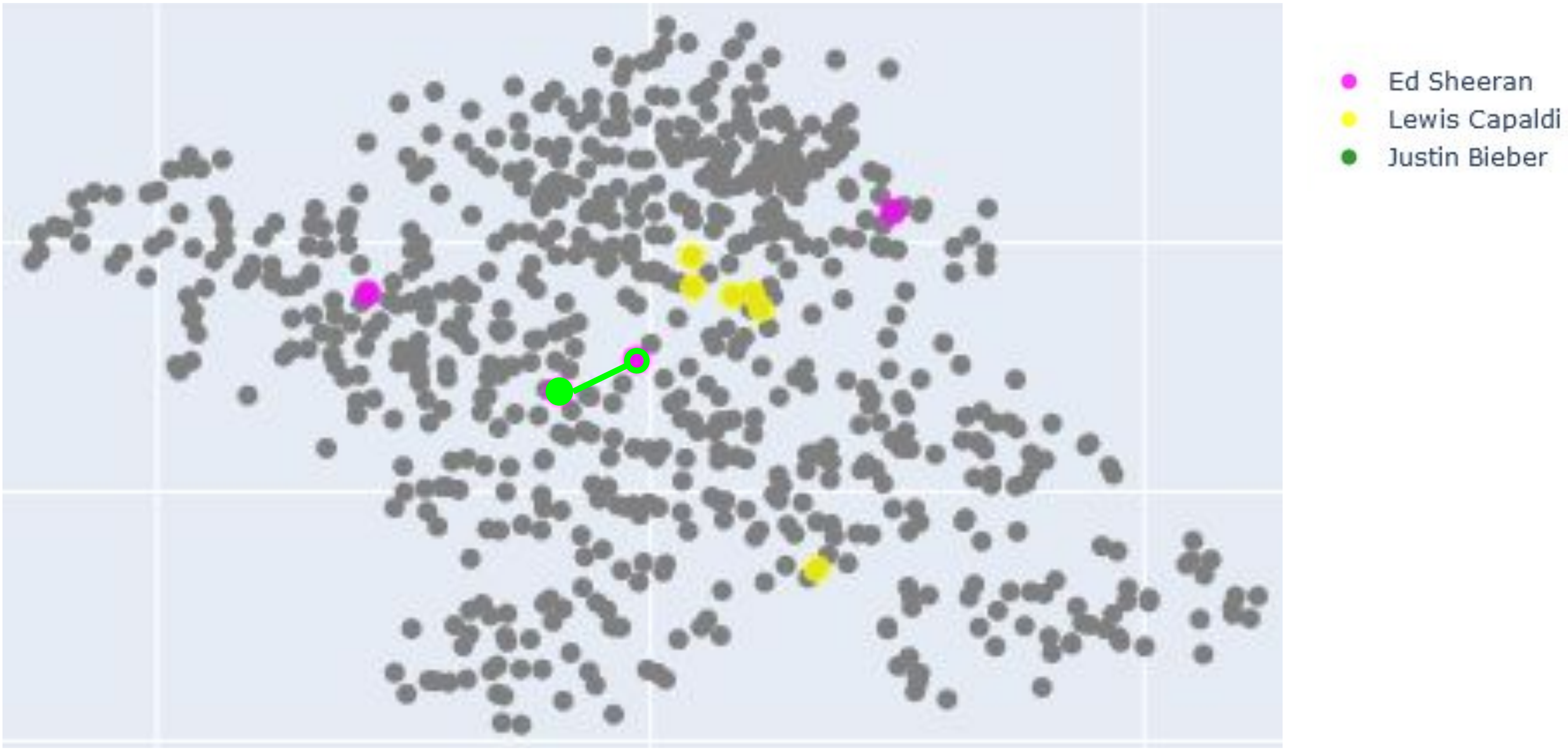
Naive Recommendation Algorithm Example 1



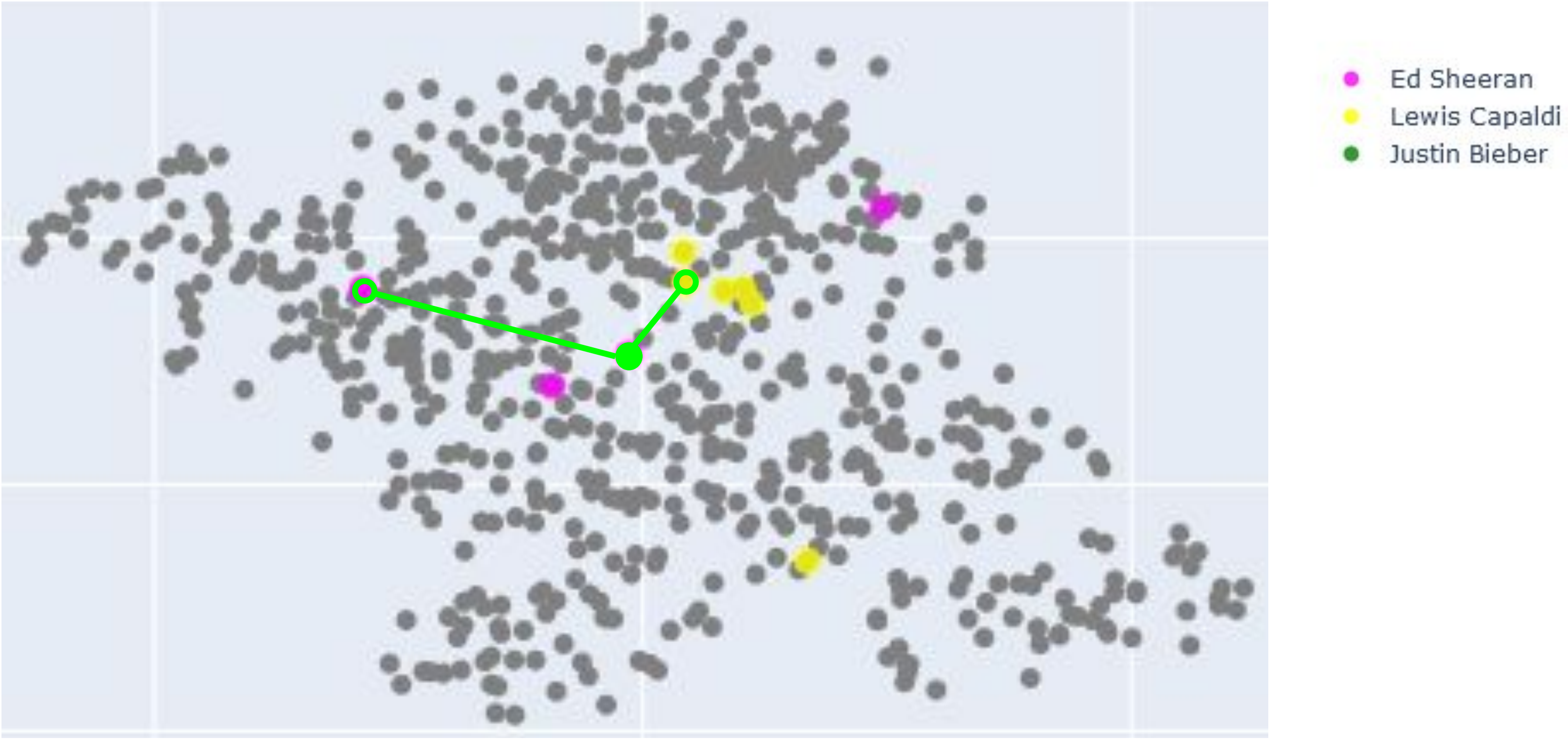


Ed Sheeran	Lewis Capaldi	2.044813
Ed Sheeran	Black Eyed Peas	2.071633
Ed Sheeran	Zack Knight	2.114143
Badshah	Ed Sheeran	2.227838
Ed Sheeran	Justin Bieber	2.285949
Justin Bieber	Lewis Capaldi	3.034368

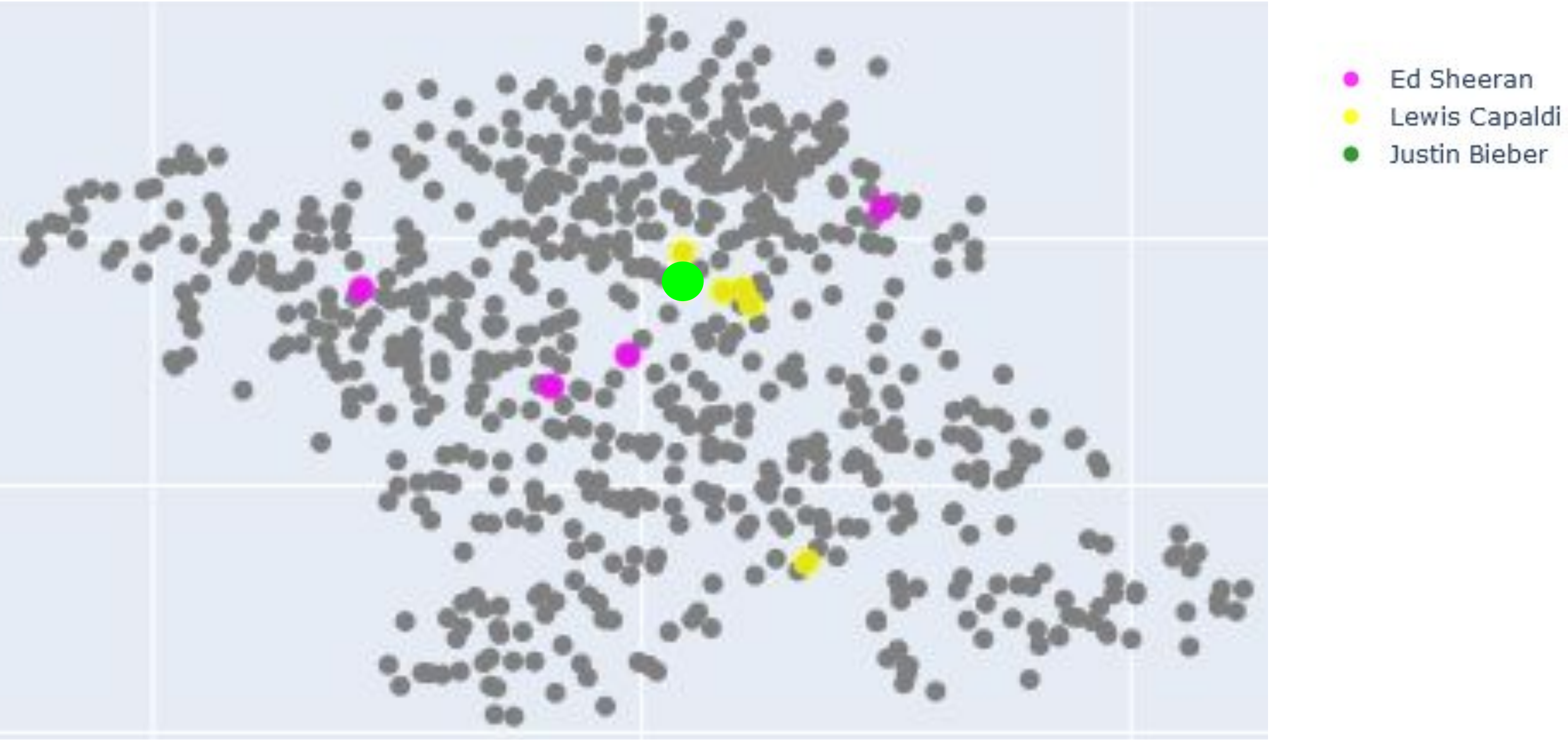
Naive Recommendation Algorithm Example 1



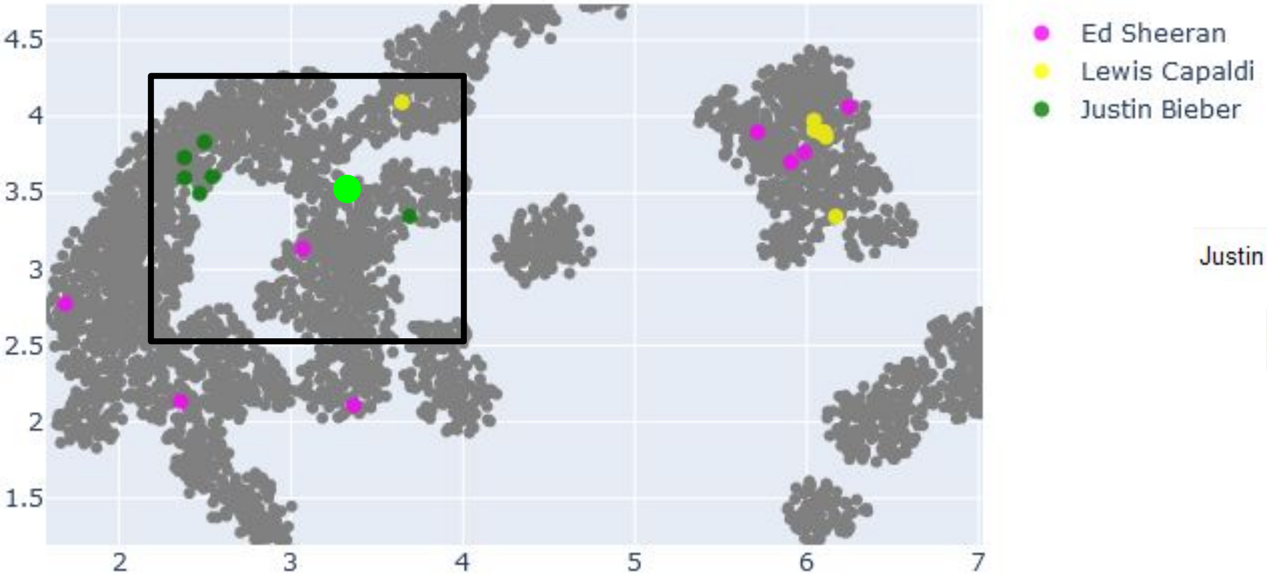
Naive Recommendation Algorithm Example 1

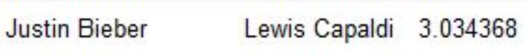


Naive Recommendation Algorithm Example 1

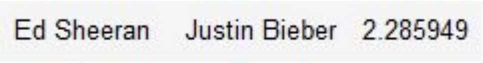


Naive Recommendation Algorithm Example 2



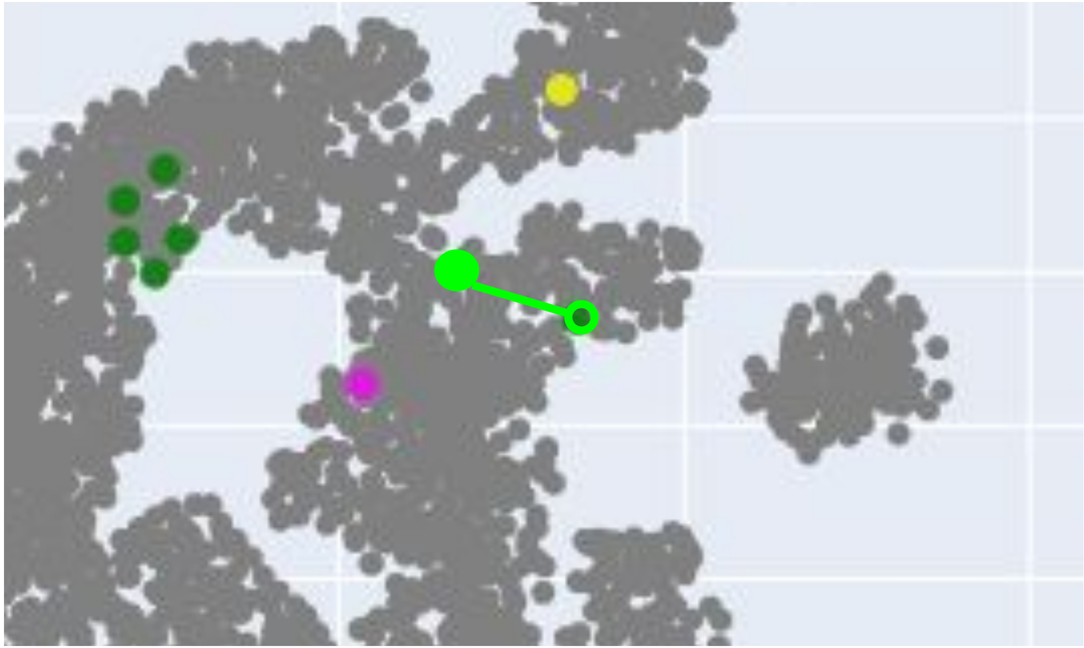


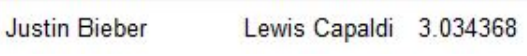
Justin Bieber	Lewis Capaldi	3.034368



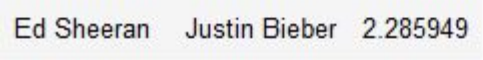
Ed Sheeran	Justin Bieber	2.285949

Naive Recommendation Algorithm Example 2



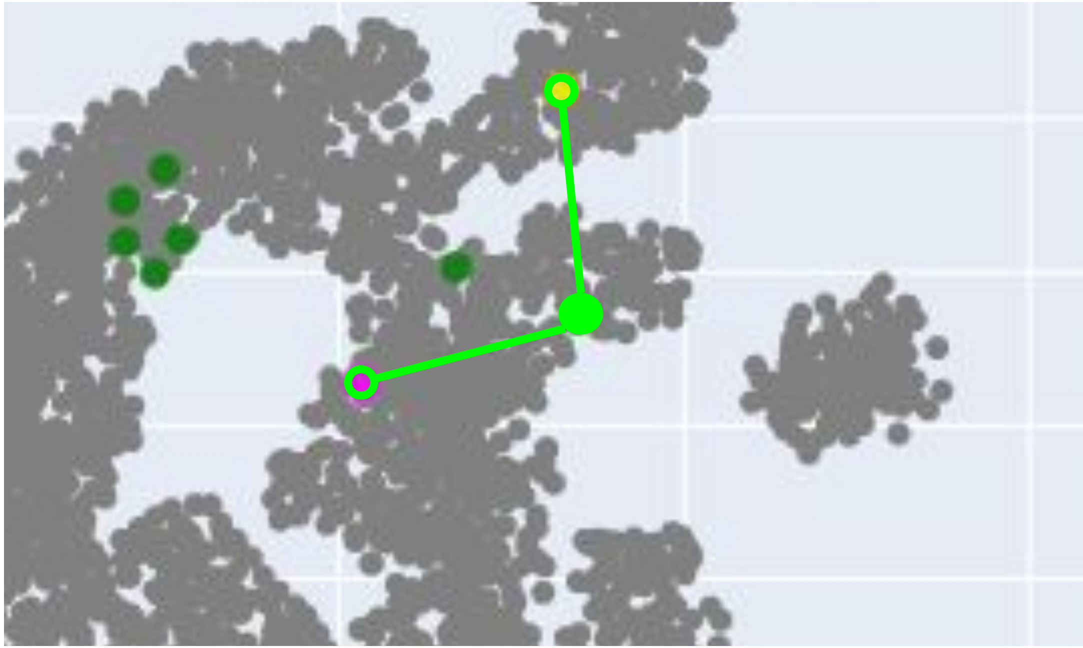


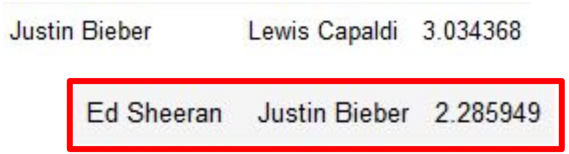
Justin Bieber	Lewis Capaldi	3.034368



Ed Sheeran	Justin Bieber	2.285949

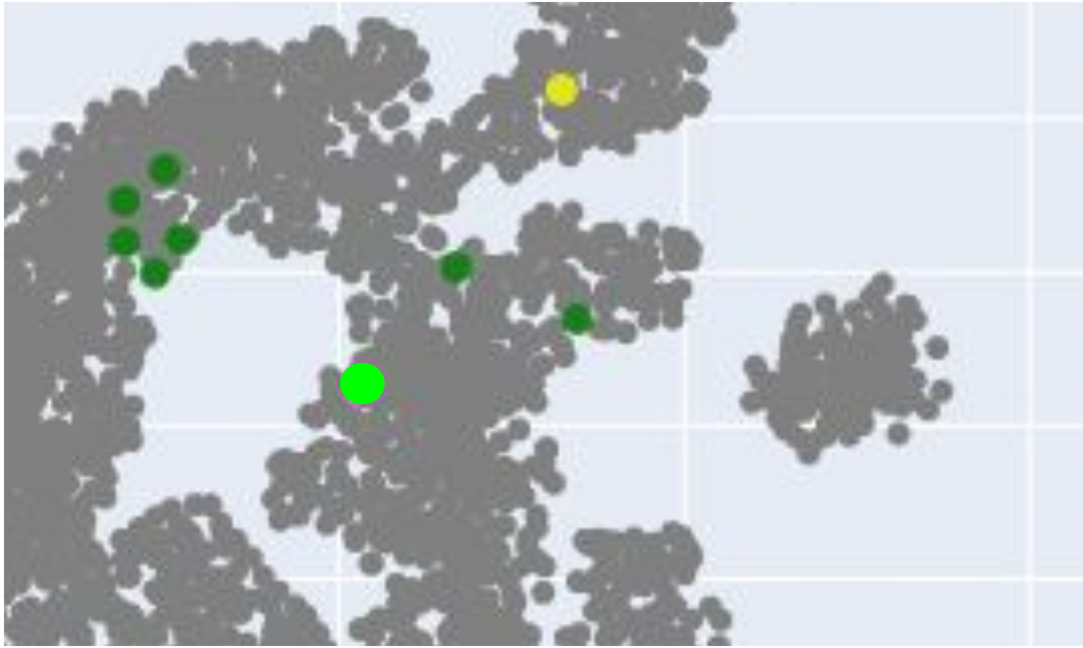
Naive Recommendation Algorithm Example 2

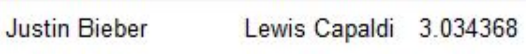




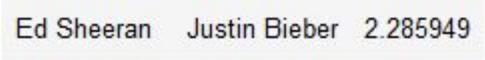
```
Justin Bieber      Lewis Capaldi   3.034368

Ed Sheeran    Justin Bieber   2.285949
```

Naive Recommendation Algorithm Example 2





Justin Bieber	Lewis Capaldi	3.034368



Ed Sheeran	Justin Bieber	2.285949

Conclusions

- UMAP most helpful for interpretation
- Clustering out the noise gives us a strong base to start similarity comparison
- Proof of a naive recommendation based purely on audio analysis
- Need more data

Steps Forward

- Use this purely as a filtering method
- Begin clustering in higher dimensions
- Subset the music to get better artist similarity
- Find other ways of representing timbre over time

Questions?

Thank You!