SPOTIFY PLAYLIST EXTENSION WITH PATTERN MINING AND CLUSTERING

Automatic Playlist Continuation Using Association Rules, Clustering, and Hybrid Recommendation Models

Adarsh Singh

CSCI 6443: Data Mining

Fall 2025

Project Repo: spotify-playlist-mining

Dataset Overview: The "Long Tail"

1.0M
PLAYLISTS

66.3M
TRACKS

2.26M
UNIQUE TRACKS

66.3
AVG LENGTH

Track Frequency Distribution

1.07M
1 Playlist
(47% Sparsity)

370K
10+ Playlists

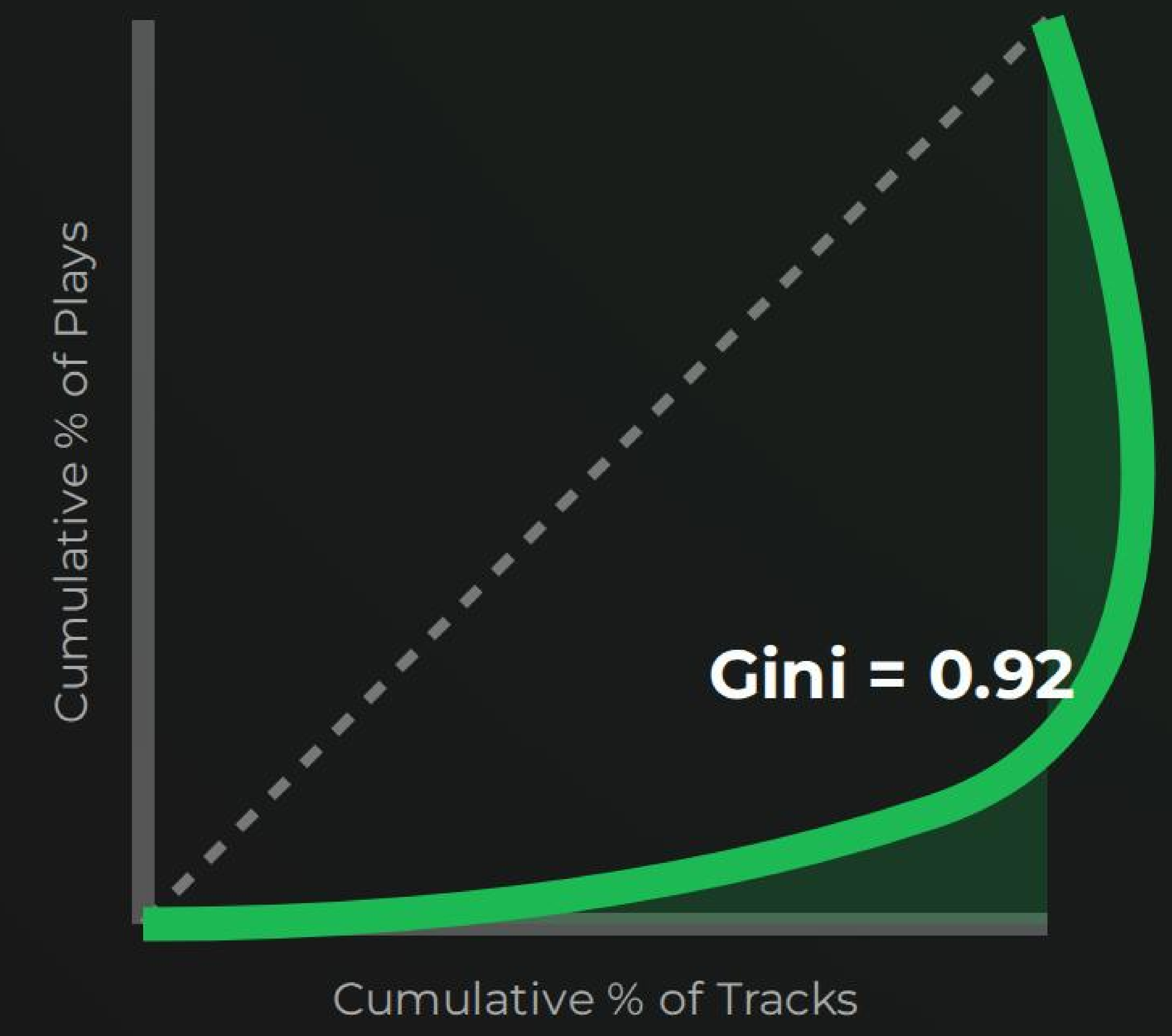
70K
100+ Playlists

10K
1000+ Playlists

The Challenge: Sparsity & Inequality

Key Insights

- **Extreme Sparsity:** 47% of tracks appear in only 1 playlist. This creates a massive "Cold Start" problem.
- **Inequality:** A Gini Coefficient of **0.921** indicates an extreme popularity bias.
- **Oligarchy:** A tiny fraction of "superstar" songs dominate the dataset.



Phase 1: Data Processing

Infrastructure

Processing 66M+ entries required careful optimization on local hardware.

- **Hardware:** Local processing on M4 MacBook (32GB RAM).
- **Optimization:** Optimized Pandas operations for memory efficiency.
- **Format:** Used **Parquet** format for fast I/O and compression.

Pipeline Flow

1. Load MPD JSON Format

↓

2. Extract Track Features (URIs, Artists, Genres)

↓

3. Build Co-occurrence Matrices & Embeddings

↓

4. Create Playlist Vectors (66M entries → Vectors)

↓

5. Split into Train/Test Sets

Phase 2: Mining & Clustering

Phase 2A: Association Rules

Algorithm: FP-Growth (Fast Pattern Growth)

Input: 1M Playlists as Transactions

Metrics:

- **Support:** How many playlists have A & B?
- **Confidence:** If A appears, how likely is B?
- **Lift:** How much more likely than random chance?

Phase 2B: Clustering

Algorithm: K-Means on Track Features

Input: Playlist Embeddings

Purpose: Group similar playlists together.

Result: 5 distinct cluster groups representing playlist "types".

Phase 3: Build & Test Four Models

1. Popularity Baseline

Rec: Most-played global songs.
Reasoning: "Wisdom of Crowds". If many like it, you probably will too.

2. Co-occurrence Based

Rec: Using association rules.
Reasoning: Find songs that frequently appear with your tracks (Lift > 2.0).

3. SVD (Collab Filtering)

Rec: Matrix factorization.
Reasoning: Find playlists mathematically similar to yours and suggest their songs.

4. Neural Network (PCA)

Rec: Deep learning on embeddings.
Reasoning: Learn complex non-linear patterns from playlist vectors.

Association Mining Results

We discovered **10,000** high-confidence patterns.

1,282x

AVERAGE LIFT

10,960x

MAX LIFT

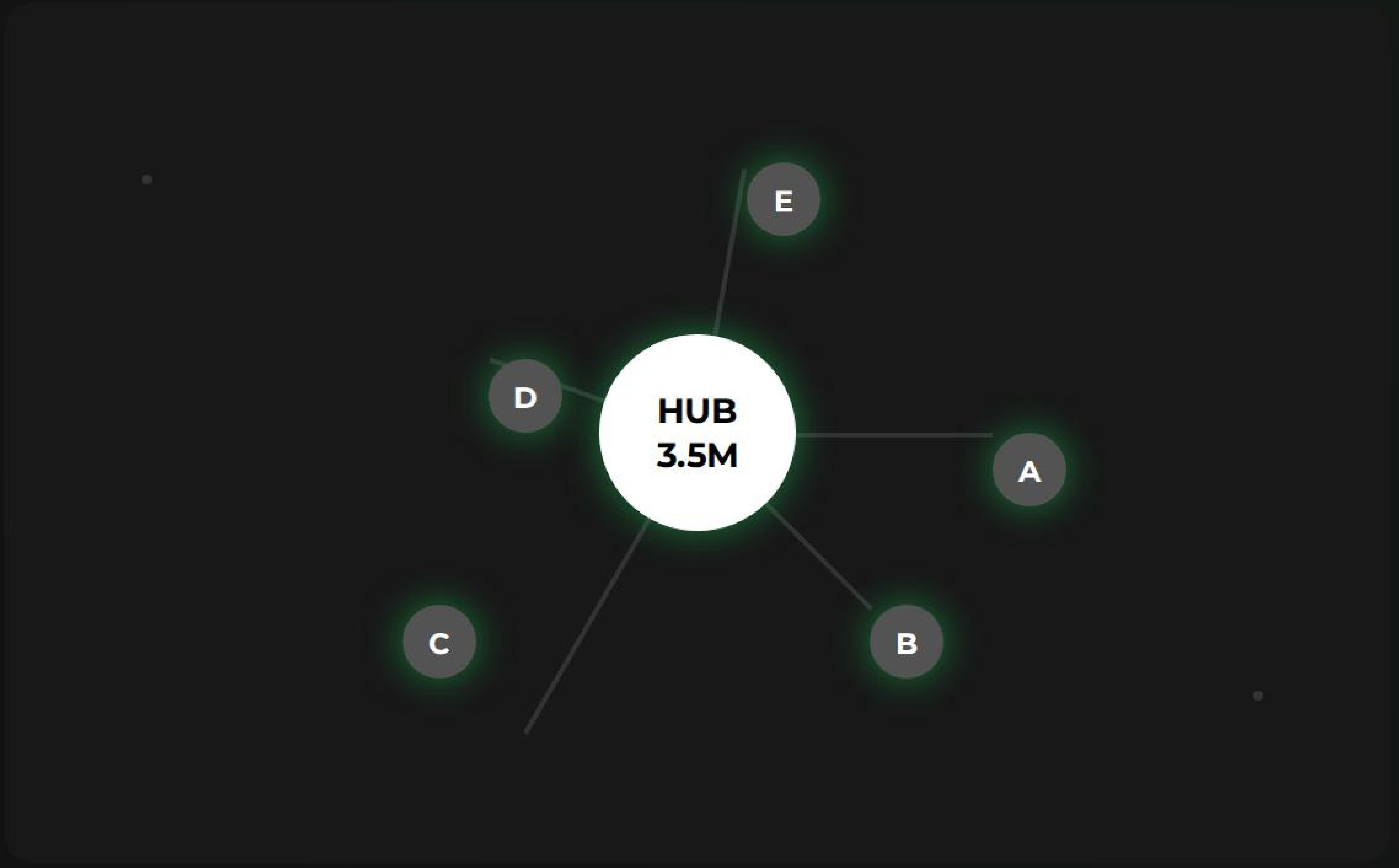
Songs with this lift are mathematically inseparable.

The Network Structure

"Superstar" Hubs

The recommendation graph relies heavily on central hubs.

- **Graph Density:** 0.74 (Very High connectivity).
- **Hub Nodes:** A single song can connect to 3.56 Million other paths.
- These hubs act as "bridges" between different clusters, crucial for the Popularity model.



Clustering Results: Listener Archetypes

We found **5 Distinct Patterns** automatically without genre labels. Users naturally gravitate toward these "styles".

Cluster 0	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Mainstream Fans	Album Listeners	Mixed/Eclectic	Album Listeners	Mainstream Fans
High Artist Pop	High Consistency	No Strong Bias	High Consistency	High Artist Pop

How Clustering Improves Recommendations:

1. Identify which cluster the input playlist belongs to.
2. Only suggest songs from similar playlists (Preserve Context).
3. Tradeoff: Better thematic consistency vs. potential reduction in diversity.

Model Evaluation: Precision@10

Comparison of 4 recommendation approaches. Surprisingly, **simplicity wins** on sparse data.

Why? The Neural Network overfits on the sparse long tail. The Popularity Baseline leverages the Gini inequality (the "Wisdom of Crowds") to maximize precision.

But Precision Isn't Everything

The Diversity Gap

Popularity models have high precision but **zero serendipity**. Our mining approach excels at discovery.

Album Diversity	79.3%
Artist Diversity	64.3%

Discovery Score

Our model recommends across **14+ distinct genres**, whereas popularity models restrict users to the "Top 50".

✓ Prevents Filter Bubbles

Genre-Wise Performance

Context determines the algorithm. "Chill" and "Party" playlists dominate usage.

Key Findings

Strong Patterns Exist & Are Predictive

Max lift of **10,960x** proves that user listening habits follow strong, predictable rules, not random chance.

Playlists Naturally Cluster

5 distinct groups identified automatically. This enables **Context-Sensitive Recommendations**.

Simple Models Are Hard to Beat

Popularity baseline achieved **14.61%** precision. Complex models underperform on sparse data without ensemble methods.

Bottom Line: The co-occurrence patterns + clustering + diversity together create a stronger recommendation system than any single approach alone.

Conclusion & Future Work

Future Directions

- **Sequential Analysis:** Analyzing order (A → B) rather than just co-occurrence.
- **Graph Networks:** Advanced graph neural networks (GNNs).
- **Hybrid Ensembles:** Weighted average of Popularity (Precision) + Mining (Diversity).

"Music taste is learnable."

This project proves that even with sparse data, **pattern mining** combined with **clustering** can successfully automate playlist continuation, improving user retention and discovery.

Questions?

Adarsh Singh

github.com/adarsh-singh/spotify-playlist-mining