Slide 1: Title Slide

"Good Evening, everyone. My name is Adarsh Singh, and today I'm presenting my work on **Spotify Playlist Extension**.

The core premise of this project is simple: **Music taste is learnable.** By analyzing patterns in millions of playlists, we can predict what a user wants to hear next—even without knowing who they are. I'll walk you through how I used Association Rule Mining and Clustering to solve the 'Automatic Playlist Continuation' problem."

Slide 2: Dataset Overview (The Long Tail)

"We started with the **Spotify Million Playlist Dataset**. The numbers are massive: 1 million playlists and over 66 million track entries.

However, the raw numbers hide a difficult truth: this data is incredibly **sparse**. If you look at the chart on the right, you'll see the problem. **47% of all tracks** (over 1 million songs) appear in only **one single playlist**. This creates a massive 'Long Tail' distribution, making it very hard for algorithms to learn patterns for half the music in the world."

Slide 3: The Challenge (Sparsity & Inequality)

"This leads us to our main technical challenge: **Inequality.**

We calculated the **Gini Coefficient** of the dataset and got **0.92**. For context, a Gini of 0 is perfect equality, and 1 is perfect inequality. A score of 0.92 means the musical world is an **oligarchy**. A tiny fraction of 'Superstar' songs get all the plays, while the vast majority get none.

This creates the 'Cold Start' problem: How do we recommend a song that has almost no historical data?"

Slide 4: Phase 1 - Data Processing

"Before we could model, we had to process this massive dataset. Dealing with 66 million rows usually requires a cloud cluster, but I wanted to prove this could be done on local silicon.

Using an **M4 MacBook Pro**, I optimized the pipeline using **Parquet files** for compression and fast I/O, and vectorized Pandas operations to manage memory.

The pipeline (on the right) involved loading the raw JSON, extracting features like URIs and Artist names, building co-occurrence matrices, and finally splitting the data into Train/Test sets."

Slide 5: Phase 2 - Mining & Clustering

"For the methodology, I took a two-pronged approach:

First, Phase 2A: Association Rules. We treated playlists like 'shopping baskets' at a grocery store. If you buy bread, you buy butter. Similarly, if you add *Song A*, do you add *Song B*? We used the **FP-Growth algorithm** to find these connections efficiently.

Second, Phase 2B: Clustering. We used **K-Means** to group playlists based on their content vectors. The goal was to automatically discover 'types' of playlists—like 'Workout' or 'Study'—without needing human labels."

Slide 6: Phase 3 - The Four Models

"We then built and tested four different recommendation models to see which approach works best:

1. **Popularity Baseline:** Simply recommending the most-played songs globally. (The 'Wisdom of Crowds').
2. **Co-occurrence:** Using the rules we mined in Phase 2.
3. **SVD:** A Matrix Factorization approach (Collaborative Filtering).
4. **Neural Network:** Using Deep Learning/PCA on the playlist embeddings."

Slide 7: Association Mining Results

"The results from the Association Mining were staggering.

We generated 10,000 rules with a Lift greater than 2.0.

As you can see in the center, the Max Lift was 10,960x. This means that for certain pairs of songs, they are 10,000 times more likely to appear together than random chance would predict. These aren't just coincidences; they are mathematically inseparable pairs."

Slide 8: The Network Structure

"When we visualize these connections, we see a **'Hub-and-Spoke'** network structure.

The graph has a high density of **0.74**, but it relies heavily on 'Superstar Hubs.' A single hub node can connect to over **3.5 million paths**.

This structure explains why the Popularity model works so well—these hubs act as bridges that connect otherwise unrelated clusters of songs."

Slide 9: Clustering Results

"In the Clustering phase, the K-Means algorithm automatically identified **5 distinct listener archetypes** without any genre labels.

- **Clusters 0 & 4** represented 'Mainstream Fans'—people who listen to Top 40 hits.
- **Clusters 1 & 3** were 'Album Listeners'—users who play an entire album in order and reject random singles.

This proves that we can tailor recommendations. We shouldn't recommend a random single to an 'Album Listener' type."

Slide 10: Model Evaluation (The Surprise)

"Now for the main result. Which model won?

Surprisingly, the **Popularity Baseline** (the simplest model) won with a Precision of **14.61%**. The Complex Neural Network actually performed the worst at 1.04%.

Why? It goes back to that Gini Coefficient of 0.92. The Neural Network tried to learn patterns in the sparse 'Long Tail' and overfitted. The Popularity model just bet on the 'Superstars'—and in an unequal world, betting on the winner is a statistically safe strategy."

Slide 11: Diversity Analysis

"But accuracy isn't everything. A DJ who only plays the Top 10 hits is boring.

While our Mining approach lost on precision, it crushed the baseline on **Diversity**.

- **Album Diversity:** 79%
- **Artist Diversity:** 64%

Our model spans **14+ genres** and finds the 'hidden gems.' It prevents the 'Filter Bubble' effect where users never hear anything new."

Slide 12: Genre-Wise Performance

"We also analyzed performance by context. As you can see, **'Chill'** and **'Party'** playlists dominate the dataset.

This implies that 'Context' is just as important as the song itself. A recommendation that works for a 'Party' playlist will fail completely in a 'Chill' playlist, even if the user is the same. Our clustering approach helps solve this by identifying the context first."

Slide 13: Key Findings

"To summarize our key findings:

1. **Patterns are Predictive:** User behavior isn't random. Max lifts of 10,000x prove strong underlying rules.
2. **Clustering Works:** We can automatically detect context (Party vs. Study) without labels.
3. **Simplicity is Strong:** On sparse data, simple models often beat complex deep learning models because they are more robust to noise."

Slide 14: Conclusion & Future Work

"In conclusion, this project demonstrates that **systematic data mining** can unlock powerful insights even from sparse data.

For **Future Work**, I would look into:

- **Sequential Analysis:** Treating playlists as a timeline (Song A *then* Song B), not just a bucket.
- **Graph Neural Networks:** To better model the hub structures we found.
- **Ensembles:** Combining the Popularity model (for precision) with our Mining model (for diversity) to get the best of both worlds."

Slide 15: Q&A

"Thank you for your time. The full code and project details are available on the GitHub repo listed here. I’m happy to take any questions."