Debiasing Music Recommendations

A Project Overview

Aryan Chopra (u20230086)
Sanyam (u20230141)
Lakhwinder Singh(u20230117)

Group Number 43

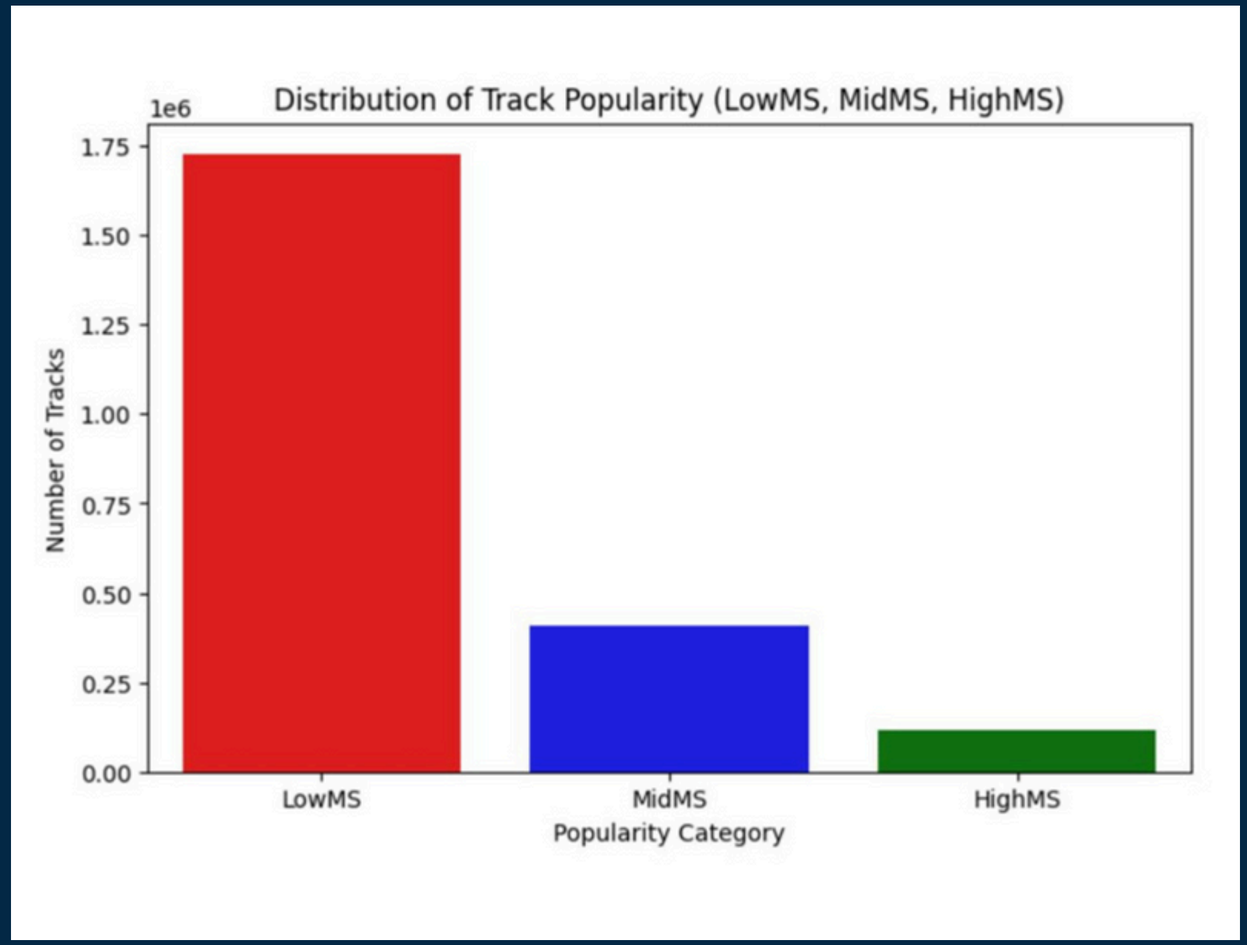
Problem Statement

Core Issue

- Popularity Bias in Music Recommendation
- Underrepresentation of Low-Stream Artists

Potential Applications

- Fair Music Recommendation Systems
- Artist Discovery Platforms
- Personalized Playlists for Niche Interest



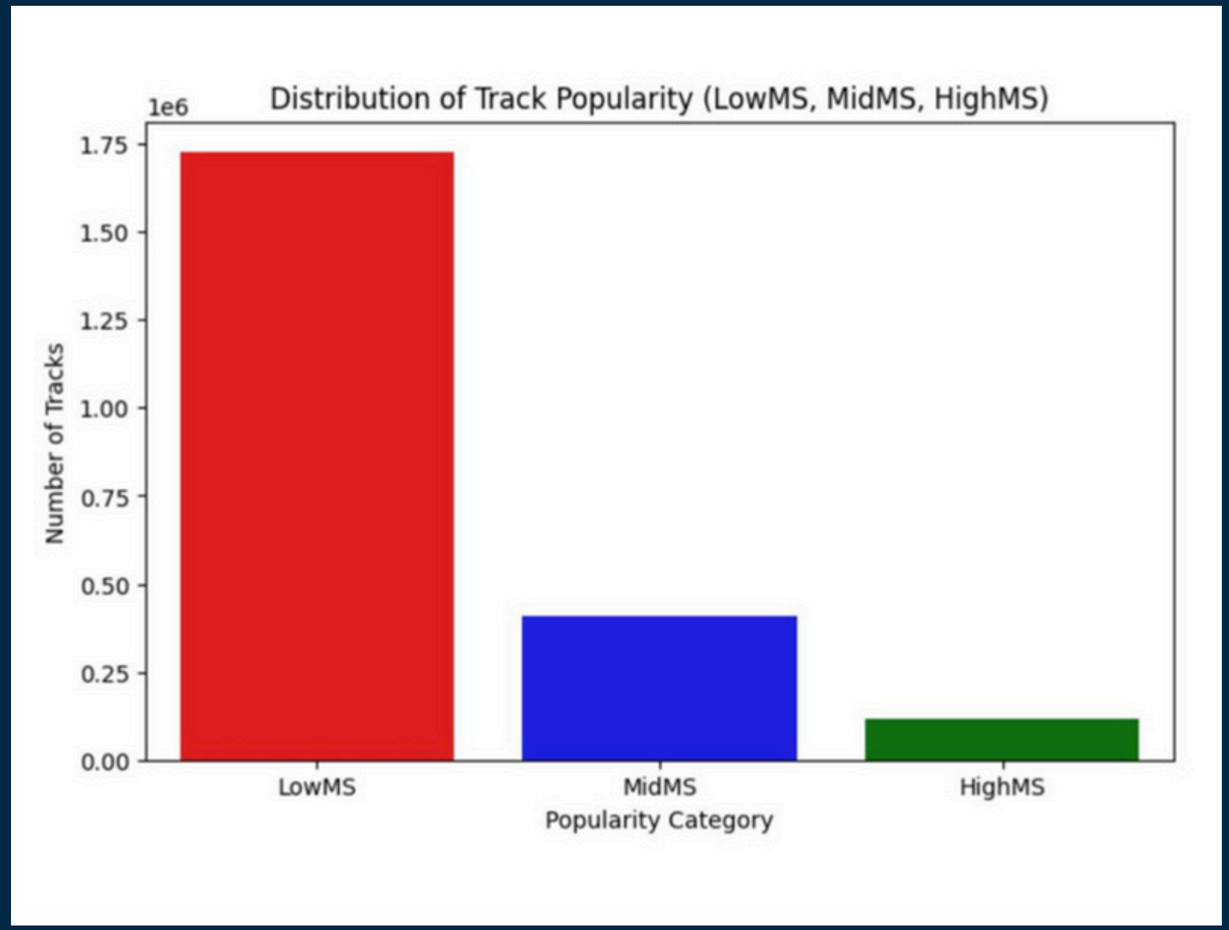
Problem Statement

Why Important?

- Limits Diversity
- Hinders Discovery of New Talent
- Reinforces “Rich-get-richer” Effect

Potential Impact

- Increased Visibility for Emerging Artists
- Enhanced User Experience
- Cultural Diversity in Digital Media



Existing Solutions Overview

Common Approaches

- Collaborative Filtering (CF)
- Content-Based Filtering (CBF)
- Hybrid Models (CF + CBF)
- Non-Negative Matrix Factorization (NMF)

Popularity Bias

- → Favors mainstream artists
- → Limits diversity & discovery

Limitations

- Overfocus on accuracy
- Persistent popularity bias
- Lack of user-specific fairness

Gaps

Fairness often secondary

Limited personalization of diversity

Need for better evaluation metrics

Solutions & Our Focus

Post-hoc reranking (e.g., Smooth XQuAD)

Fairness-aware learning

Psychological & domain insights

Solutions & Our Focus

Fairness integrated in model

NMF + popularity-aware reranking

Empower low-stream artists

Balance accuracy & diversity

Literature Survey

S. No	Title	Year	Journal/ Conference	Methods
1.	Managing Popularity Bias in Recommender Systems with Personalized Re-ranking	2019	AAAI FLAIRS Conference	Personalized diversification re-ranking (xQuAD, Smooth xQuAD), post-processing
2.	Fairness and accuracy in recommender systems	2022	ACM Transactions on Recommender Systems	Survey/Review of fairness-aware algorithms, evaluation frameworks
3.	Algorithms for non-negative matrix factorization	2001	NIPS 13 (Neural Information Processing Systems)	Non-negative Matrix Factorization (NMF) algorithms

Paper 1

Title:
Managing Popularity Bias in Recommender Systems with Personalized Re-ranking

Source:
AAAI FLAIRS Conference / arXiv:1901.07555

Authors:
Mehdi Abdollahpouri, Robin Burke, Bamshad Mobasher

Methodology:
Personalized Re-ranking:
Uses xQuAD and Smooth xQuAD algorithms to balance accuracy and long-tail (niche) item exposure.
Post-processing step after standard collaborative filtering (CF) or matrix factorization (MF) recommendations.

Key Results:

Long-tail Coverage:
Significant increase in exposure for less popular (long-tail) items.

Trade-off:
Some reduction in accuracy for higher diversity, but overall user satisfaction and catalog coverage improved.

Limitations:
Focuses on item-side fairness (long-tail), not user-side or provider fairness.
Re-ranking is a post-processing step, not integrated into model training.
May require tuning to balance between accuracy and diversity for different application needs.
Let me know if you need this in even more compact form for a slide, or want a table version!
Answer from Perplexity: pplx.ai/share

Paper 2

Title:
Fairness and Accuracy in Recommender Systems

Source:
ACM Transactions on Recommender Systems

Authors:
M.D. Ekstrand, H. Abdollahpouri, R. Burke, C. Cramer, B. Mobasher

Methodology:

- Comprehensive Survey:
  - Reviews definitions and measurements of fairness in recommender systems (user-side, item-side, provider-side).
  - Analyzes algorithmic approaches: post-processing, in-processing, and pre-processing debiasing methods.
  - Discusses trade-offs between fairness and accuracy, and the impact of different fairness metrics (e.g., demographic parity, equalized odds).
- Evaluation:
  - Presents frameworks and metrics for assessing both fairness and accuracy in various real-world scenarios.

Key Results / Insights:

- No universal definition of fairness; context and stakeholder priorities matter.
- Improving fairness often reduces accuracy, and vice versa-trade-offs are inevitable.
- Calls for context-sensitive, multi-stakeholder evaluation and transparent reporting of fairness impacts.

Limitations:

- Survey/review paper-does not propose or empirically test a new algorithm.
- Highlights complexity and subjectivity in defining and achieving fairness.
- Stresses need for further research on adaptive, context-aware fairness strategies.

Paper 3

Title:
Algorithms for Non-negative Matrix Factorization

Source:
NIPS 13 (Neural Information Processing Systems), 2001

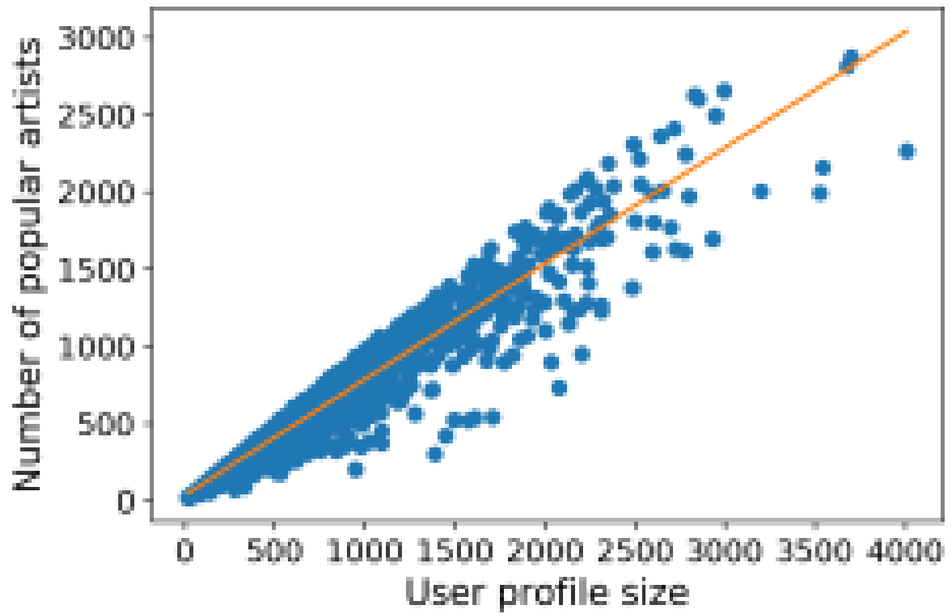
Authors:
Daniel D. Lee, H. Sebastian Seung

Methodology:
- Two Multiplicative Update Algorithms:
  - One minimizes squared error (Frobenius norm)
  - One minimizes generalized Kullback-Leibler divergence
- Both use iterative multiplicative updates for factors W and H
- Monotonic convergence proven using auxiliary functions (like EM algorithm)
- Algorithms are simple to implement and guarantee local optimum

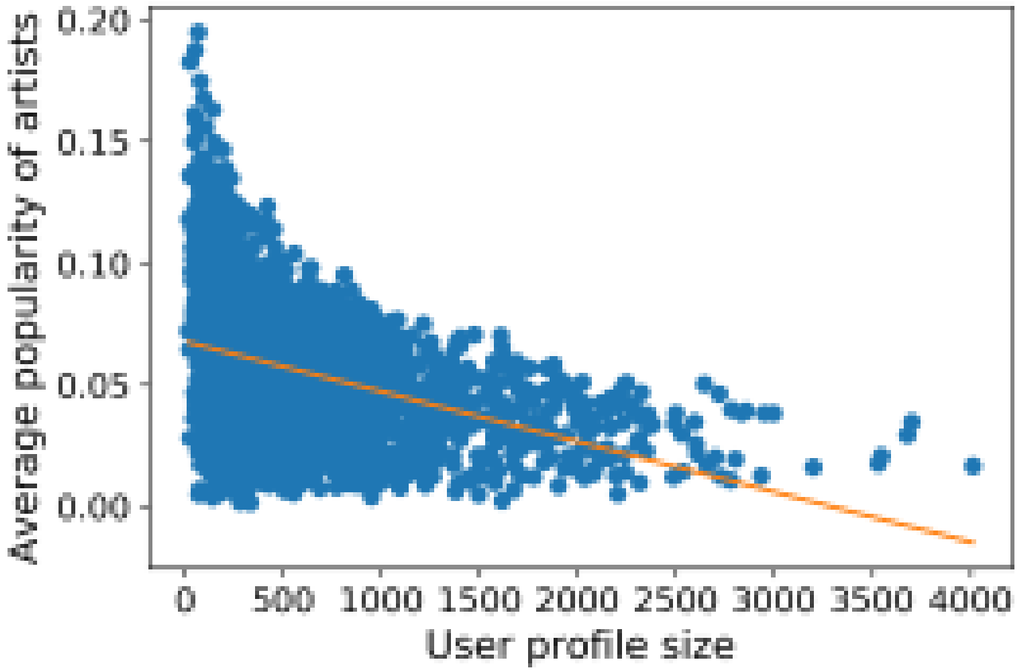
Key Results:
- NMF produces parts-based, interpretable decompositions (unlike PCA)
- Widely applicable for clustering, text mining, and image analysis
- Demonstrated stable and monotonic convergence in experiments

Limitations:
- Converges to local minima (not guaranteed global optimum)
- Sensitive to initialization
- Does not address regularization or sparsity (extensions required for those)

Analysis of Popularity Bias - Key Graphs and Insights

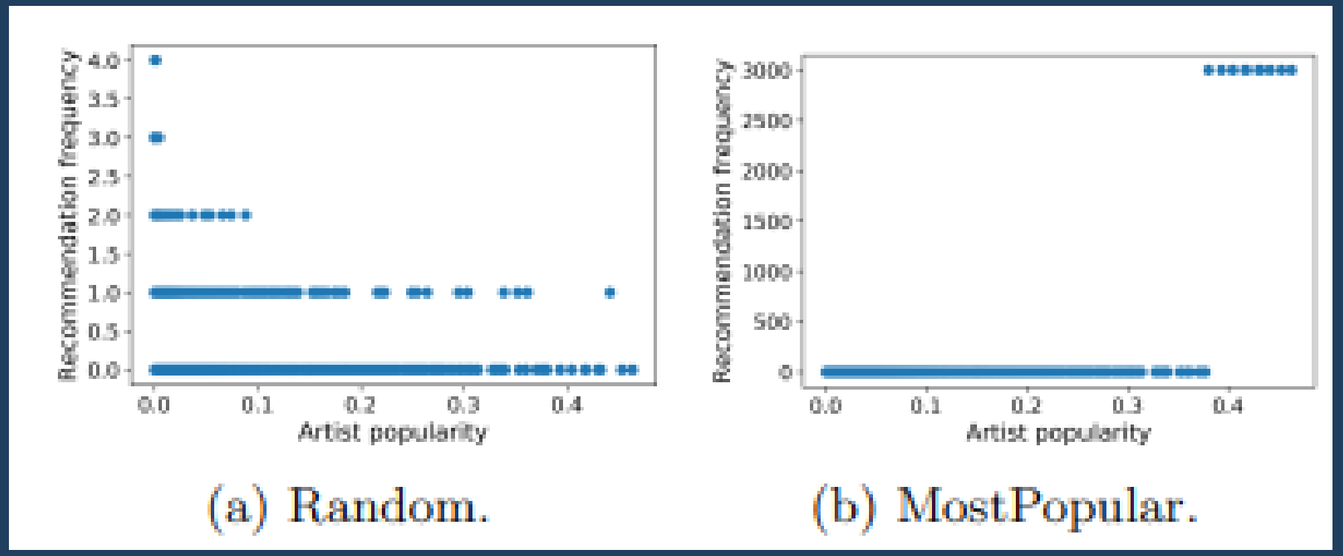


(a) Number of popular artists.



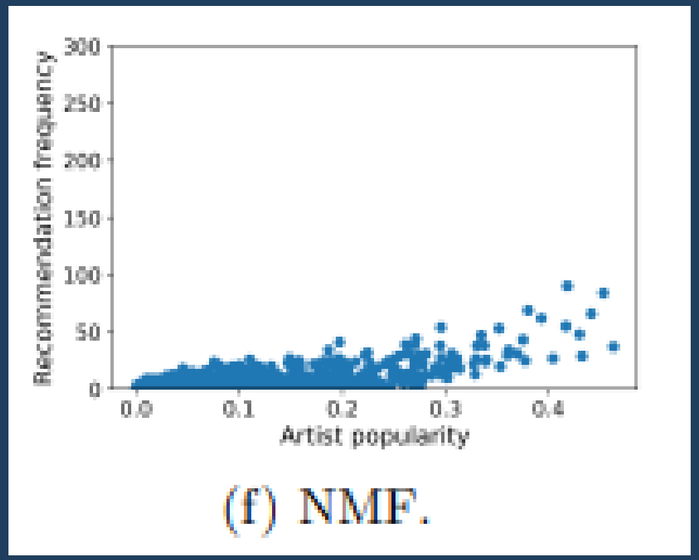
(b) Average popularity of artists.

Algorithmic Influence and Mitigating Popularity Bias

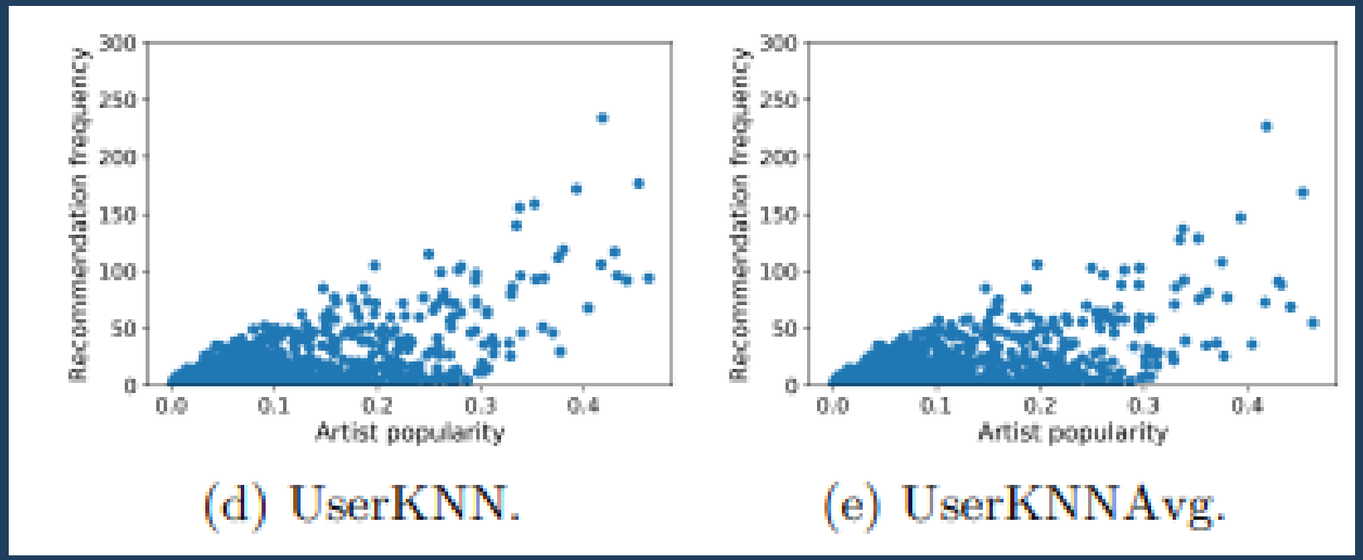


(a) Random.

(b) MostPopular.



(f) NMF.



(d) UserKNN.

(e) UserKNNAvg.

Dataset Overview

-We utilized a combination of publicly available datasets and data we collected ourselves.

Public Datasets:

Million Playlist Dataset (MPD)

We did not collect this dataset ourselves. It was collected by Spotify Research from publicly available Spotify playlists created by US users. The data consists of anonymized playlist information, including playlist titles, track lists (with Spotify URIs, track names, and artist names), and some basic playlist metadata. Ethical considerations included the fact that this data was derived from public playlists, meaning users had implicitly agreed to some level of public visibility. The dataset is massive, containing 1 million playlists and over 2 million unique tracks, resulting in millions of user-item interactions (implicit feedback).

spotify-tracks(Kaggle)

We utilized a public dataset referred to in the code as the Spotify Tracks Dataset, loaded from /kaggle/input/-spotify-tracks-dataset/dataset.csv. This dataset serves as the source for explicit item features, providing detailed characteristics for a large collection of music tracks. It contains metadata such as track ID, track name, and artist name, along with various audio features computed by Spotify. As seen in our code's processing and output, key features extracted and used from this dataset include 'popularity', 'duration_ms', 'danceability', 'energy', 'loudness', 'speechiness', 'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo', 'mode', and 'explicit'. This rich set of 13 features is crucial for the Hybrid NCF model to understand item characteristics and improve recommendations, particularly for songs where extensive user interaction data might be limited.

Music Dataset: 1950 to 2019 (Kaggle):

We also did not collect this dataset. It was compiled by its authors (available on Kaggle) likely by gathering data from various music sources or APIs. The nature of the dataset is primarily track-level metadata and audio features (like danceability, energy, loudness, etc.), along with track names and artist names. Ethical concerns would relate to the original source of the data and ensuring proper licensing for research use, which is typically covered by public datasets on platforms like Kaggle. The dataset contains over 28,000 tracks with around 30 features per track.

	pos	artist_name	track_uri
0	0	Beyoncé	spotify:track:1uXbwHHfgsXcUKfSZw5ZJ0
1	1	Beyoncé	spotify:track:1z6WtY7X4HQJvzxC4UgkSf
2	2	Missy Elliott	spotify:track:3jagJCUbdqhDSPuxP8cAqF
3	3	The Sugarhill Gang	spotify:track:0tm6gsXe0LSm9zeSspyMQu
4	4	The Sugarhill Gang	spotify:track:5mMxriCaSYyZopQDPYkyqT

	artist_uri	track_name
0	spotify:artist:6vWDO969PvNqNYHIOW5v0m	Run the World (Girls)
1	spotify:artist:6vWDO969PvNqNYHIOW5v0m	Love On Top
2	spotify:artist:2wIVse2owClT7go1WT98tk	Work It
3	spotify:artist:7zliF6Q946WznVk3ZMYhZX	Rapper's Delight - Single Version
4	spotify:artist:7zliF6Q946WznVk3ZMYhZX	Apache (Jump On It)

	album_uri	duration_ms
0	spotify:album:1gIC63gC3B7o7FfpPACZQJ	236093
1	spotify:album:1gIC63gC3B7o7FfpPACZQJ	267413
2	spotify:album:6DeU398qrJ1bLuryetSmup	263226
3	spotify:album:1oQef7548sivCtMQwK8Wgo	235840
4	spotify:album:6ao0po28zzbNEl6MoUigrQ	374760

	album_name	pid
0	4	647000
1	4	647000
2	Under Construction	647000
3	The Sugarhill Gang - 30th Anniversary Edition	647000
4	Rapper's Delight	647000

DATASET OVERVIEW

Our User Study Data

user_id	day	item_type	rank	song_name	artist_name	popularity	feedback
user_study_participant_1	1	Listened	1	whatever cost	giant panda guerilla dub squad	66	
user_study_participant_1	1	Listened	2	prairie rose	roxy music	91	
user_study_participant_1	1	Listened	3	you'll always find me in the kitchen at parties	jona lewie	88	
user_study_participant_1	1	Listened	4	get'cha head in the game	troy	45	
user_study_participant_1	1	Listened	5	the night goes on	kenny rogers	51	
user_study_participant_1	1	Listened	6	i never cared for you	willie nelson	89	
user_study_participant_1	1	Listened	7	freakin' at the freakers' ball	dr. hook	68	
user_study_participant_1	1	Listened	8	una ocacion	bubaseta	84	
user_study_participant_1	1	Listened	9	my way to you	jamey johnson	37	
user_study_participant_1	1	Listened	10	grind	phish	54	

This dataset was collected by us from real participants. We recruited 10 individuals to participate in a 7-day study. Each day, our system provided them with 10 song recommendations. We collected their explicit feedback on each recommended song (whether they liked or disliked it).

SELF COLLECTED DATA SET

Dataset Size

This dataset contains approximately 10 users * 7 days * 10 recommendations/day = 700 data points (recommendation events), each with associated feedback.

user_study_participant_1	1	Recommended	1	get real	wailing souls	3	1,0
user_study_participant_1	1	Recommended	2	stay	astrud gilberto	2	0,0
user_study_participant_1	1	Recommended	3	the devil don't sleep	brantley gilbert	0	0,0
user_study_participant_1	1	Recommended	4	i believe (in everything)	jj grey & mofro	0	0,0
user_study_participant_1	1	Recommended	5	thinking of you	sister sledge	2	0,0
user_study_participant_1	1	Recommended	6	plastic fantastic lover	jefferson airplane	0	0,0
user_study_participant_1	1	Recommended	7	on the run	pink floyd	2	0,0
user_study_participant_1	1	Recommended	8	take the moment	tony bennett	0	0,0
user_study_participant_1	1	Recommended	9	cold cold world	stephen stills	0	0,0
user_study_participant_1	1	Recommended	10	scatterbrain	radiohead	0	0,0

Features Preprocessing

MPD (from Kaggle dataset)

We loaded the JSON slice files, extracted user (playlist) IDs and item (track) URIs. We then created a mapping from original IDs/URIs to contiguous internal integer IDs for both users and items. Finally, we constructed a sparse interaction matrix (CSR format) where entries represent a user interacting with an item.

track-spotify-dataset(from Kaggle Dataset)

We loaded the CSV file. We identified the relevant feature columns (track_name, artist_name, danceability, energy, etc.). We handled missing values by dropping rows with missing critical identifiers (track_name, artist_name) and imputing numerical feature NaNs with 0 (a simple strategy for this project). We performed scaling on the numerical features using StandardScaler to bring them to a similar range, which is important for neural networks. For categorical features (like 'mode'), we mapped them to integer IDs to be used in embedding layers within the NCF model. We also created a unique item_uri for each item based on its name and artist to align with the MPD data.

Feature Importance

We used a RandomForestClassifier to analyze feature importance. We trained this classifier on a dataset of user-item pairs (sampled positive interactions and sampled negative non-interactions) combined with the item features. The importance scores from the RandomForest model indicated which features were most predictive of an interaction. This analysis was primarily for understanding which features were most correlated with engagement, rather than for dimensionality reduction in the main NCF model. It confirmed that features like 'popularity' (simulated) and audio features had varying degrees of predictive power.

Methodology

Core Approach

Hybrid NCF (Neural Collaborative Filtering)

- Combines user-item interactions + item features (audio, metadata)
- Trained with Bayesian Personalized Ranking (BPR) loss
- Aims to balance relevance & diversity in recommendations

Models Used

- NMF as base for learning latent user preferences
- Reranking module to boost niche content
- Explored TensorFlow implementation for alternate modeling
- Used for initial recommendation generation

Methodology

Challenges Faced

- Data Sparsity: Limited user-item interactions
- Cold Start: Recommending new or niche content
- Algorithmic Complexity: Balancing fairness & accuracy
- Compute intensive: Matrix factorization and embedding calculations

Solutions Implemented

- Negative sampling for BPR training
- Popularity-aware reranking to promote niche artists
- Scalable NMF and TensorFlow for model design

Performance Metrics(During Training)

Results on Main MPD Test Set:

Hybrid NCF Model:

Reranking: None

```
Reranking: None
  Precision@10: 0.0008
  Recall@10: 0.0007
  NDCG@10: 0.0009
  Average Diversity (Inverse Popularity): 0.1599
```

Takeaway

- Very Low accuracy and ranking metrics
- Significant boost in diversity

Performance Metrics(During Training)

Results on Main MPD Test Set:

Hybrid NCF Model:

Reranking: Smooth Xquad

```
Reranking: Smooth Xquad
  Precision@10: 0.0012
  Recall@10: 0.0010
  NDCG@10: 0.0012
  Average Diversity (Inverse Popularity): 0.2062
```

```
Evaluating Classifier Model for Feature Importance Analysis...
Accuracy: 0.9533

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.95      8526
           1       0.95      0.96      0.95      8644

    accuracy                           0.95     17170
   macro avg       0.95      0.95      0.95     17170
weighted avg       0.95      0.95      0.95     17170

ROC AUC Score: 0.9810
Average Precision Score: 0.9768

Confusion Matrix:
[[8075  451]
 [ 350 8294]]
```

```
--- Hybrid NCF Reranking Comparison (Smooth XQuAD vs None) on Main Test Set (%) ---
  Precision@10: +51.28% change
  Recall@10: +38.26% change
  NDCG@10: +33.88% change
  Average Diversity (Inverse Popularity): +28.94% change
```

Real-World Model Evaluation

-We utilized a combination of publicly available datasets and data we collected ourselves.

Self-Collected Dataset

Our User Study Data

This dataset was collected by us from real participants. We recruited 10 individuals to participate in a 7-day study. Each day, our system provided them with 10 song recommendations. We collected their explicit feedback on each recommended song (whether they liked or disliked it).

Data Collection Considerations

We focused on getting genuine reactions to the recommendations. Ethical concerns were paramount: we obtained informed consent from all participants, clearly explaining the purpose of the study and how their data would be used. All collected data was anonymized to protect participant privacy. The data collected included the user ID (anonymized), the recommended song (URI, name, artist), and their binary feedback (liked/disliked).

Dataset Size

This dataset contains approximately 10 users * 7 days * 10 recommendations/day = 700 data points (recommendation events), each with associated feedback.

Performance Metrics(On Student Data)

Results on Main MPD Test Set:

Hybrid NCF Model:

Reranking: Smooth Xquad

Metrics from Study	
Precision@10	0.35
Recall@10	0.30
NDCG@10	0.32
Average Diversity (Inverse Popularity)	0.25
Hit Rate@10	0.38

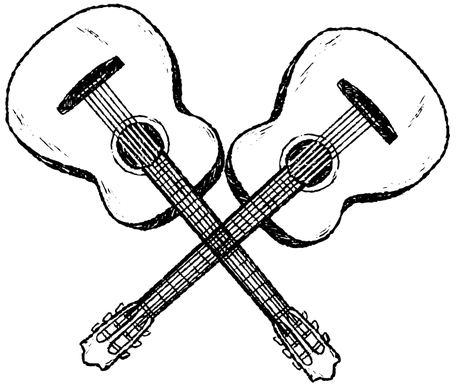
Deployability

- Can be integrated with Plaksha's music or learning platforms
- Personalized recommendations for each user

Ready for Deployment

- Large datasets: need for efficient computation
- Real-time performance for many users
- Continuous updates for new songs/users

Scalability Challenges



- Package as API or web app
- Connect to Plaksha's user data
- Real-time recommendation delivery

Deployment Steps

Thank You!

We value your feedback and insights!