Hybrid Music Recommender System

Pranshu Trivedi ptrivedi31@gatech.edu
Sung Jae Hong shong313@gatech.edu
Wyndham Hudson whudson9@gatech.edu

Motivation

Problem Description

- How do entertainment companies like Spotify recommend songs a user might like?
- Develop a robust music recommendation engine using the Million Songs Dataset

Importance

- Spotify has over 40 million tracks. Music listeners often have particular interests and tastes in music. Most music listeners want to discover and find new music that fits their tastes.

Approaches - Algorithms and Visualizations

Users \ Items	1	2	...	i	...	m
1	5	3		1	2	
2		2				4
:			5			
u	3	4		2	1	
:					4	
n			3	2		

Example User-Item Matrix

Baseline Approach: Collaborative Filtering

- How it Works:
  - Using a User-Item matrix, find the closest songs that a user is predicted to like based on their past preferences.
  - Singular Value Decomposition (matrix factorization)
  - K-Nearest Neighbors

Approaches - Algorithms and Visualizations

Our Approach: Content-Based Filtering

- **How it Works:** K-Nearest Neighbors Algorithm
  - Using relevant audio features of user’s songs, find K closest songs
  - Evaluate the returned songs by hiding some songs and finding Average Recall

- Why it Works:
  - Maximizing Recall helps increase likeliness the User will enjoy the songs returned

Approaches - Algorithms and Visualizations

Our Approach: Hybrid Combination of Collaborative Filtering and Content Based Filtering

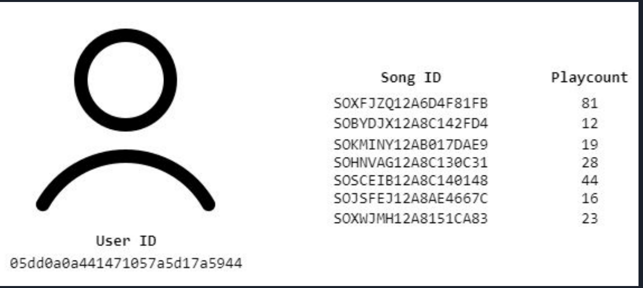
- **Novelty:** no previous hybrid recommender developed for the Million Songs Dataset.

- **How it works**: based on the number of songs listened to by the input user, a combined list of recommendations from a content-based recommender and collaborative filtering recommender are given.
    - The ratio of content-based recommendations to collaborative filtering recommendations in the output is set linearly according to number of songs the user has listened to.

- **Why it works**: Avoids the cold start problem (content-based filtering) and avoids recommending similar songs only (collaborative filtering)

Data

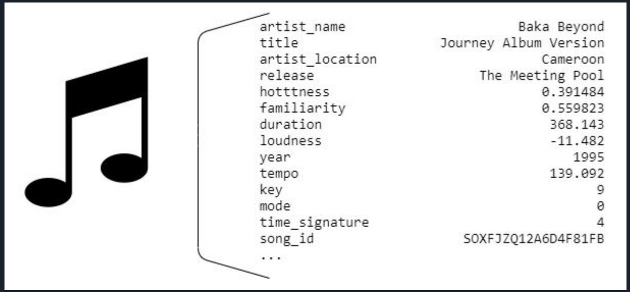
- Two datasets: Taste Profile User Dataset and MSD Features dataset
- How was it obtained?
  - The user dataset was directly downloaded from MSD website.
  - The features dataset was obtained by running a script on an EC2 instance with access to the whole 245GB of the MSD features dataset.
- *Dataset 1:* User-Song Dataset
  - Includes ‘user-id’, ‘song-id’, and ‘playcount’
  - Used for **Collaborative-Filtering** recommendation
  - Included ‘playcount’ of each song for rating of each user
- Properties of Dataset 1
  - 1,019,318 unique users
  - 384,546 unique songs
  - 48.23 songs per user



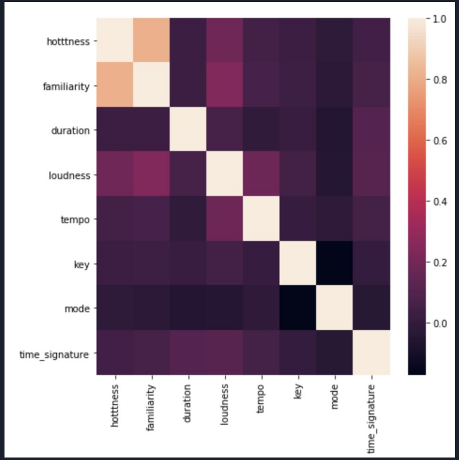
Example Entry from the User-Song Dataset

Data - Continued

- *Dataset 2:* Song-Feature Dataset
  - Songs chosen from User-Song Dataset
  - 384,456 songs with unique audio and metadata features
    - Audio Features: i.e. ‘hotttness’, ‘familiarity’, ‘loudness’
    - Metadata Features: i.e. ‘artist_name’, ‘title’, ‘ year’
  - Features were gathered from the Million Song Database using ‘song_id’ assigned in the User-Song Dataset
  - Used for **Content-Based** recommendation



Example Song from the Song Feature Dataset



Correlation Matrix for the Attributes in the Million Songs Dataset

Experiments and Evaluations - Collaborative Filtering

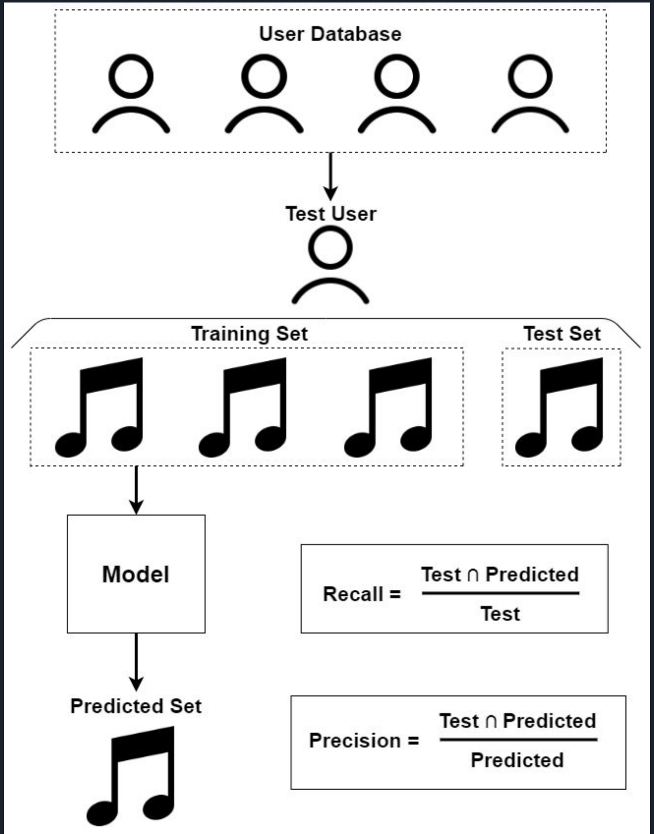
- Experimentation Setup
  - **Model:** CF Algorithms
    - Singular Value Decomposition, K-Nearest Neighbors, Slope One, Co-Clustering, Non-negative Matrix Factorization
  - **Input:** User-Song Dataset
  - **Output:** K Songs

- Evaluation
  - 5-Fold cross validation and RMSE was used to assess prediction error

Experiments and Evaluations - Content Based

- Experimentation Setup
  - **Model:** K-Nearest Neighbors from User-Song Dataset
  - **Input:** 75% of User's Songs
  - **Output:** K Songs
  - K-Nearest Neighbors was used on different feature sets
    - i.e. ['hotttness', 'familiarity', 'loudness', 'tempo', 'key', 'mode']
  - K-Nearest Neighbors used on different distance metrics
    - I.e. 'Manhattan' , 'Euclidean', 'Chebyshev', 'Minkowski (p=5)'

- Evaluation
  - Average Precision
  - Average Recall



Experimental Setup for the Content-Based model

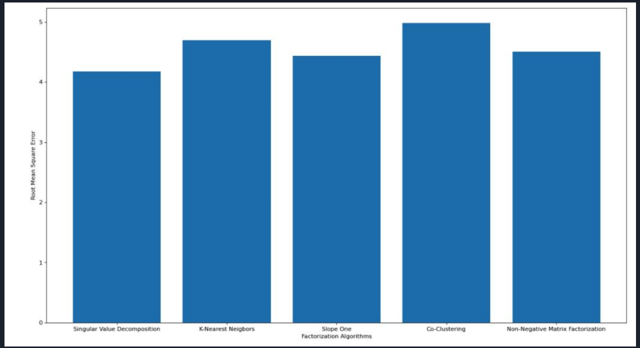
Experiments and Evaluations - Hybrid Filtering

- Setup: 5-fold CV
  - User dataset is divided into 75% known and 25% unknown users.
  - Hybrid recommender is queried for a list of song recommendations for all unknown users.
  - The RMSE is calculated from the list of song recommendations given against the actual user history in the whole user dataset.
- Evaluation
  - The RMSE is used to compare the Hybrid recommender against the baseline approaches.

Results - Collaborative Filtering

- Results
  - Singular Value Decomposition held the lowest error with 4.15 in RMSE on average

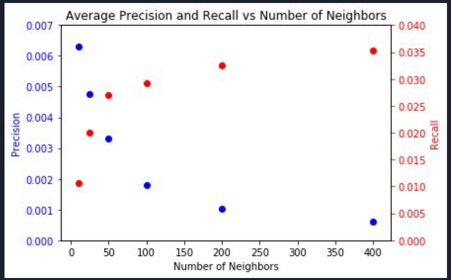
- Baseline Comparison
  - K-Nearest Neighbors had 4.59 in RMSE on average
  - Singular Value Decomposition had 4.15 in RMSE on average



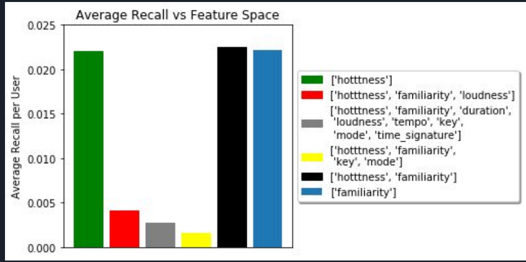
RMSE vs Different Collaborative Filtering Algorithms

Results - Content Based

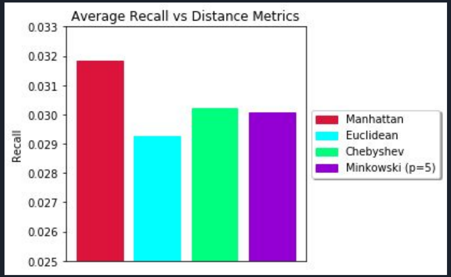
- Songs Returned
  - As the number of songs returned increases, average recall increases and precision decreases
- Feature Space
  - Using more features **reduced** Average Recall
  - Best feature space: [‘hotttness’, ‘familiarity’]
- Distance Metric
  - Limited Improvement
  - Best metric: Manhattan distance



Precision and Recall vs Songs Returned



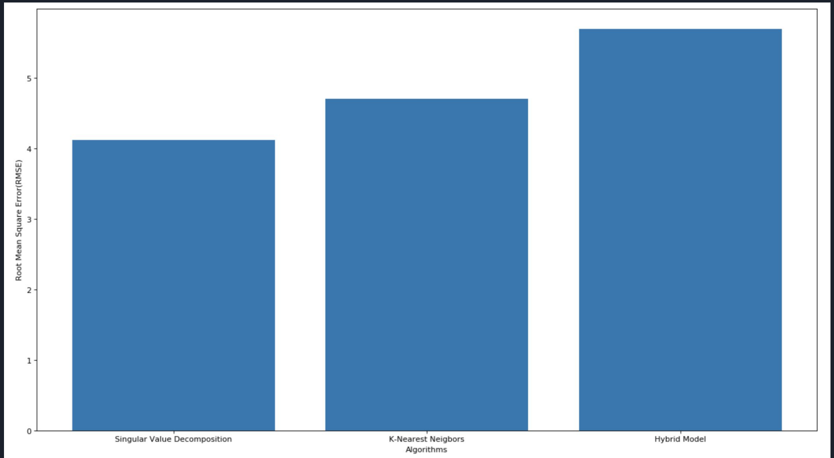
Average Recall vs Feature Space



Average Recall vs Distance Metric

Results - Hybrid Model

- 5-fold CV RMSE for SVD: **4.15**
- 5-fold CV RMSE for KNN: **4.59**
- 5-fold CV RMSE for Hybrid: **5.79**



RMSE of Baselines compared to Hybrid Approach

Future Work

- Integrate custom features generated from audio data to make the system more robust
- Use of state-of-the-art techniques such as deep learning to build a hybrid recommender system (DNNRec)