Group Playlist Recommender

By Christopher Shaffer

Motivation & Goal

- Apps like Spotify create personalized recommendations
- How can this be extended to groups of people?

Dataset

Echo Nest Taste Profiles

- 1,019,318 unique users
- 384,546 unique songs
- 48,373,586 entries
  - user – song ID – play counts

	user_id	track_id	plays
0	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAKIMP12A8C130995	1
1	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAPDEY12A81C210A9	1
2	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBBMDR12A8C13253B	2
3	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBFNSP12AF72A0E22	1
4	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBFOVM12A58A7D494	1

Song ID table

- song id – song title – artist

Source: **Million Song Dataset**

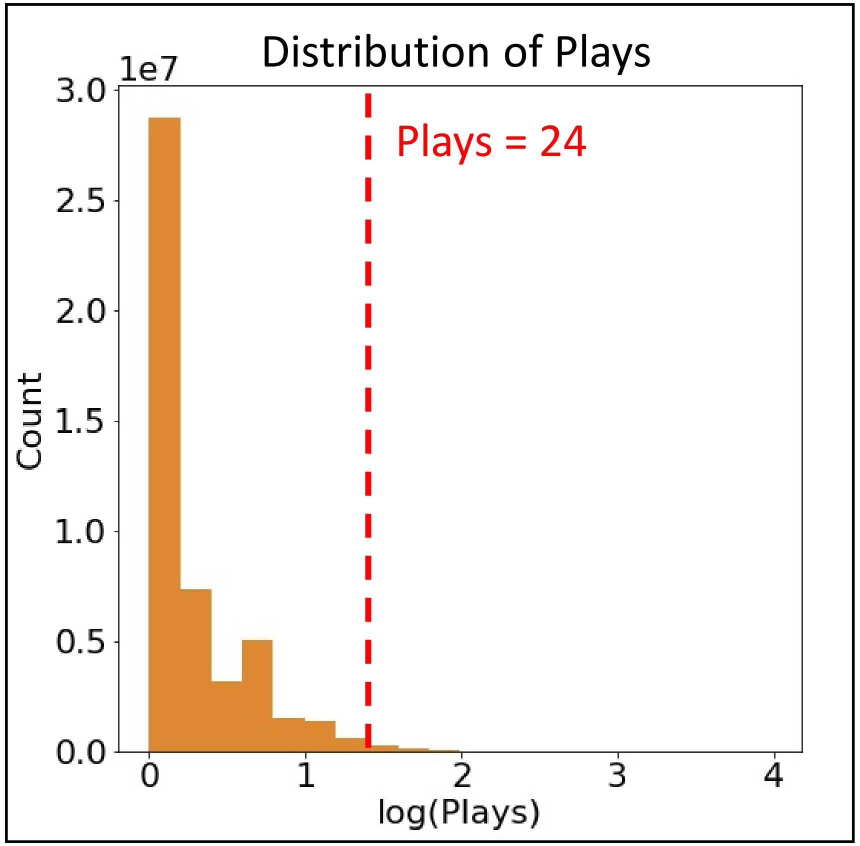
Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The Million Song Dataset. In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011.

http://millionsongdataset.com/tasteprofile/

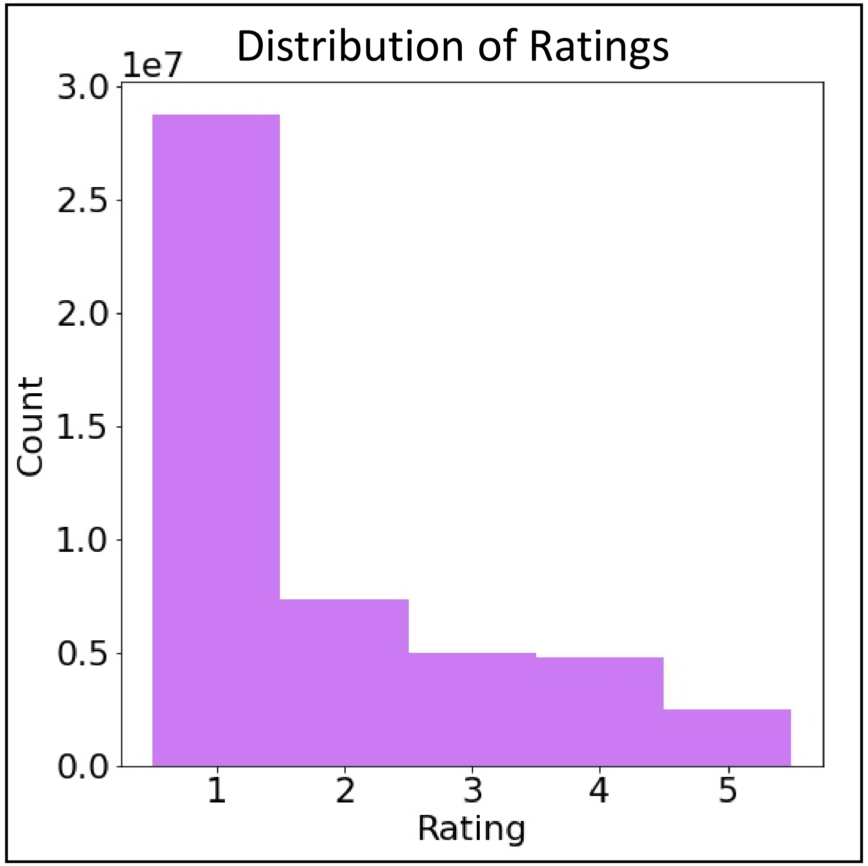
Rating Scale

- 99% of song/user pairs have ≤24 plays
  - Set max plays to 24

- $\text{Rating} = \log_{10}\left(\frac{plays}{24}\right) \cdot 5 + 1$

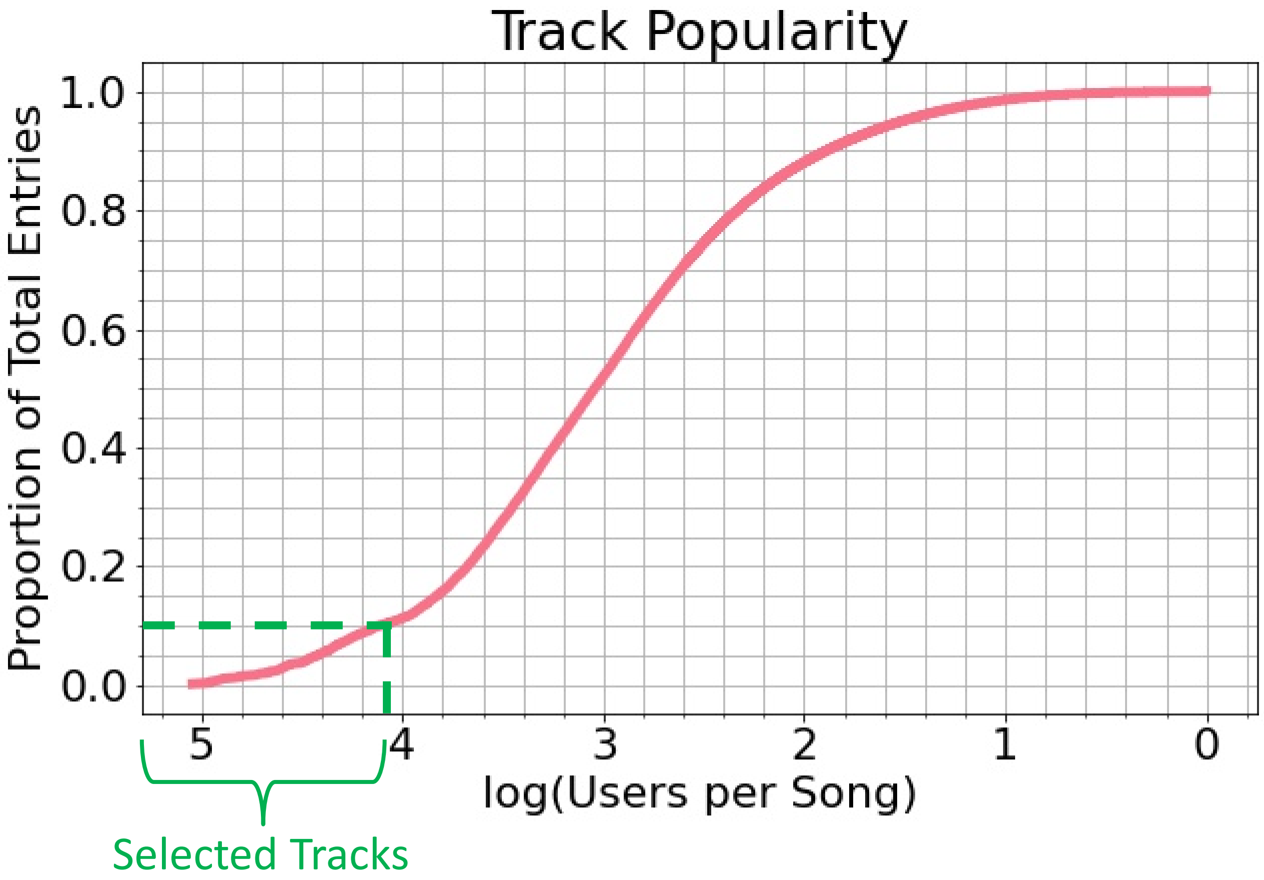


Percentile	Plays
50	1
75	3
90	6
95	10
99	**24**



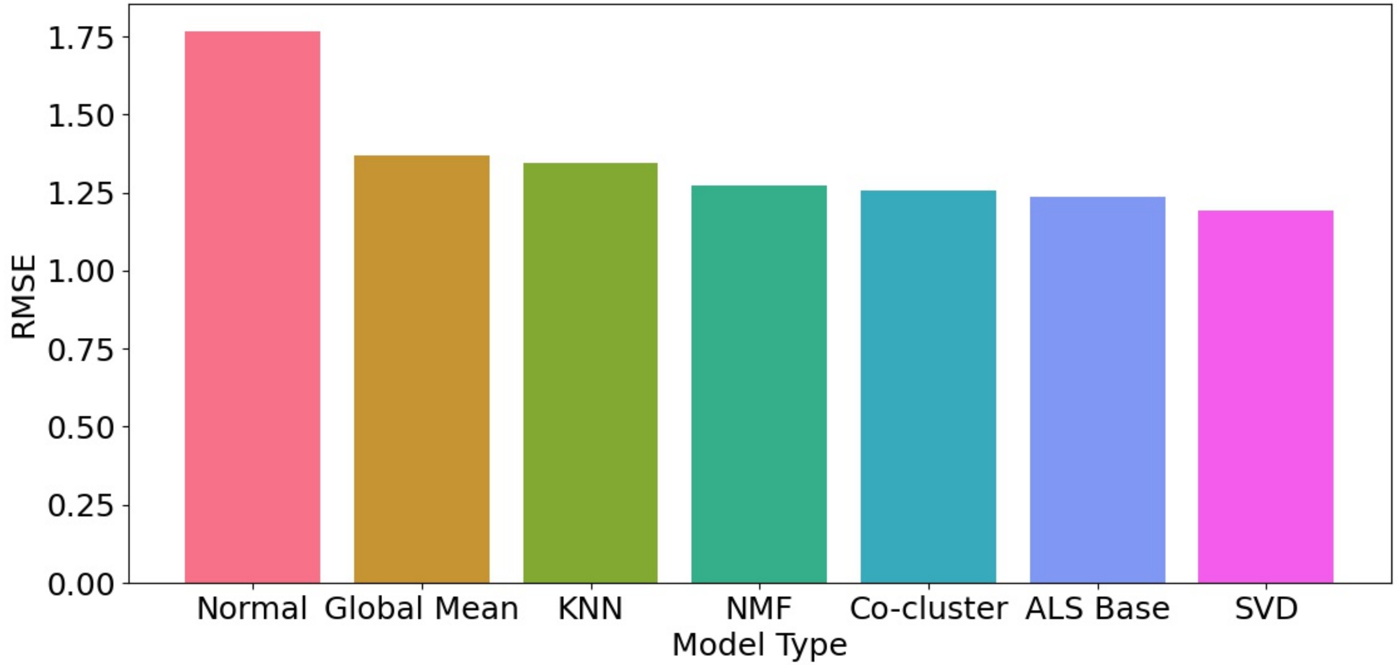
Sparsity

- Average user played ~50 tracks
  - Sparsity is the percentage of all possible user-song combinations with data
  - Sparsity = 99.99%
- For 250 most popular songs:
  - Sparsity = 97%
  - 11% of total dataset



Model Testing

- Train/validation split of 80%, 20%
- Predicting ratings for validation data
- Metric - RMSE of ratings



Model	RMSE
Normal	1.77
Global Mean	1.37
KNN*	1.35
NMF	1.27
Co-cluster	1.26
ALS Base	1.24
SVD	1.19

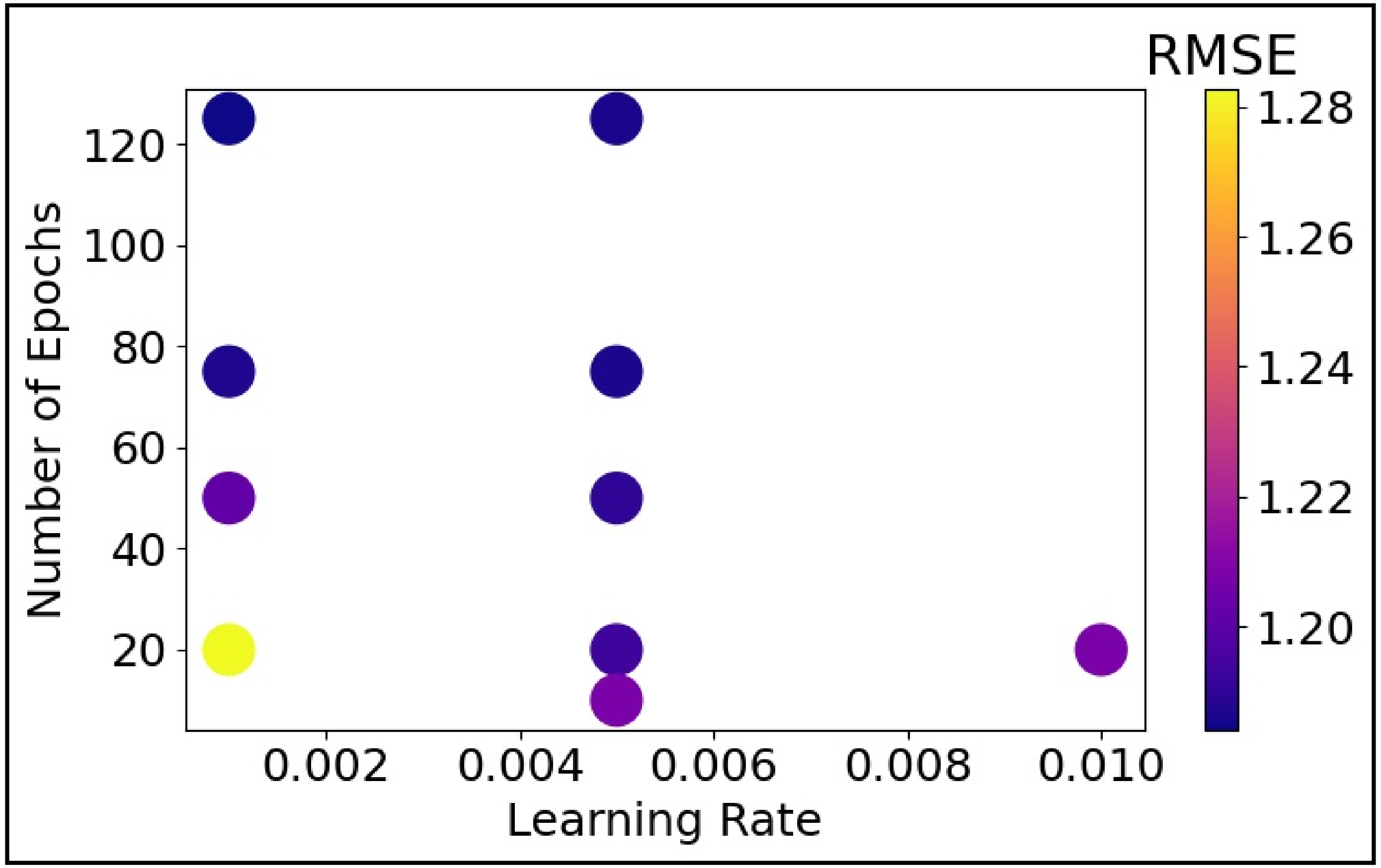
*Only used 0.5% of training data due to lack of RAM

All models are from the **surprise** library

Hyperparameter Optimization

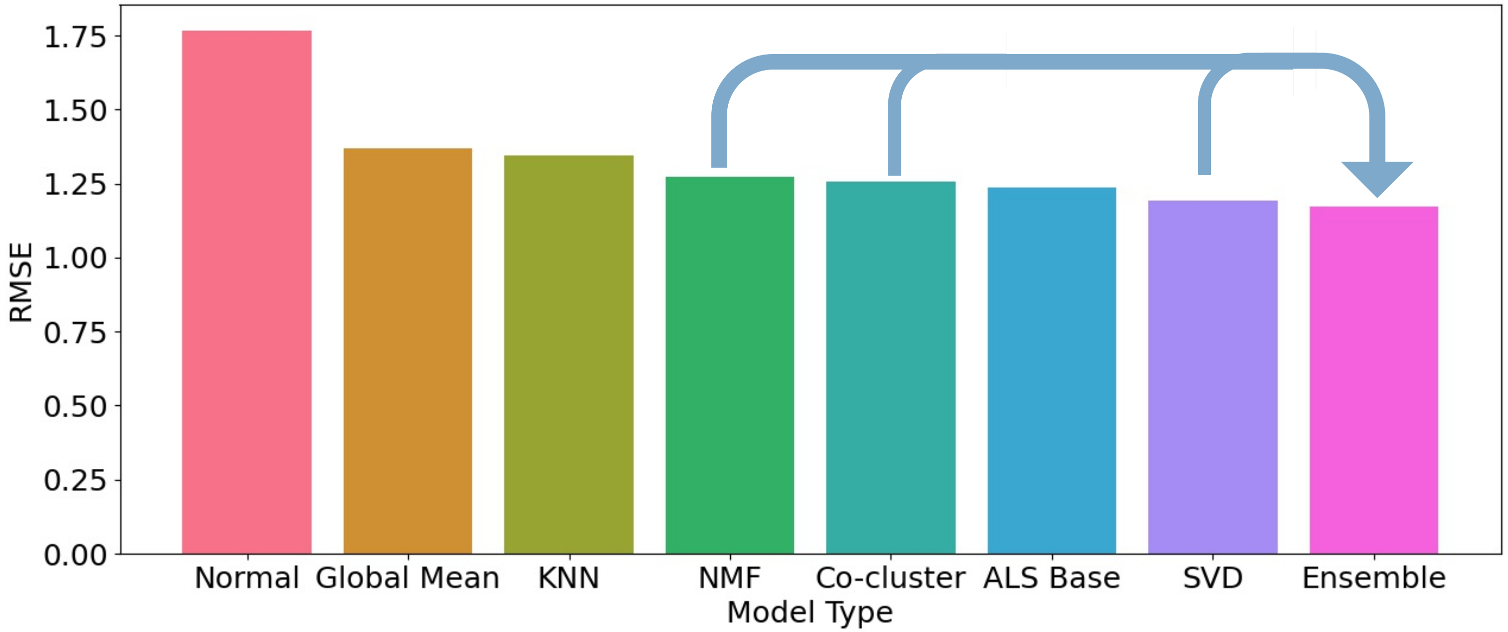
- SVD model

	Learning Rate	Epochs	RMSE
	0.001	20	1.283
	0.01	20	1.208
	0.005	10	1.208
	0.001	50	1.202
Default	0.005	20	1.193
	0.005	50	1.190
	0.001	75	1.187
	0.005	75	1.186
	0.005	125	1.186
	0.001	**125**	**1.184**



Ensemble Model

- Weighted ensemble of SVD, Co-cluster, and NMF models
- Has the best performance, but is the slowest



Model	RMSE
Normal	1.77
Global Mean	1.37
KNN*	1.35
NMF	1.27
Co-cluster	1.26
ALS Base	1.24
SVD (optimized)	1.18
Ensemble	**1.17**

Group Playlist Recommender

- For each user in group:
  - Imputes known ratings from listening history
  - Predicts ratings for remainder of 250 songs
  - Ranks all songs per user by rating
  - Recommends based on ranks using Average Strategy

Artist	Song	Rank 1	Rank 2	Rank 3	Avg Rank
Dwight Yoakam	You're The One	6	4	5	5.0
Barry Tuckwell/Academy of St M...	Horn Concerto No. 4 in E flat ...	2	18	8	9.3
OneRepublic	Secrets	19	3	11	11.0
Base Ball Bear	Sayonara-Nostalgia	1	45	12	19.3
Florence + The Machine	Dog Days Are Over (Radio Edit)	59	1	9	23.0

Other Group Ranking Strategies

Least misery strategy: Worst (highest) of users' rankings

Artist	Song	Rank 1	Rank 2	Rank 3	Worst Rank
Dwight Yoakam	You're The One	6	4	5	6
Barry Tuckwell/Academy of St M...	Horn Concerto No. 4 in E flat ...	2	18	8	18
OneRepublic	Secrets	19	3	11	19
Five Iron Frenzy	Canada	29	33	7	33
Tub Ring	Invalid	33	7	35	35

Most pleasure strategy: Best (lowest) of users' rankings

Artist	Song	Rank 1	Rank 2	Rank 3	Best Rank
Kings Of Leon	Revelry	0	34	81	0
Harmonia	Sehr kosmisch	83	0	1	0
Frumpies	Fuck Kitty	12	224	0	0
Florence + The Machine	Dog Days Are Over (Radio Edit)	59	1	9	1
Base Ball Bear	Sayonara-Nostalgia	1	45	12	1

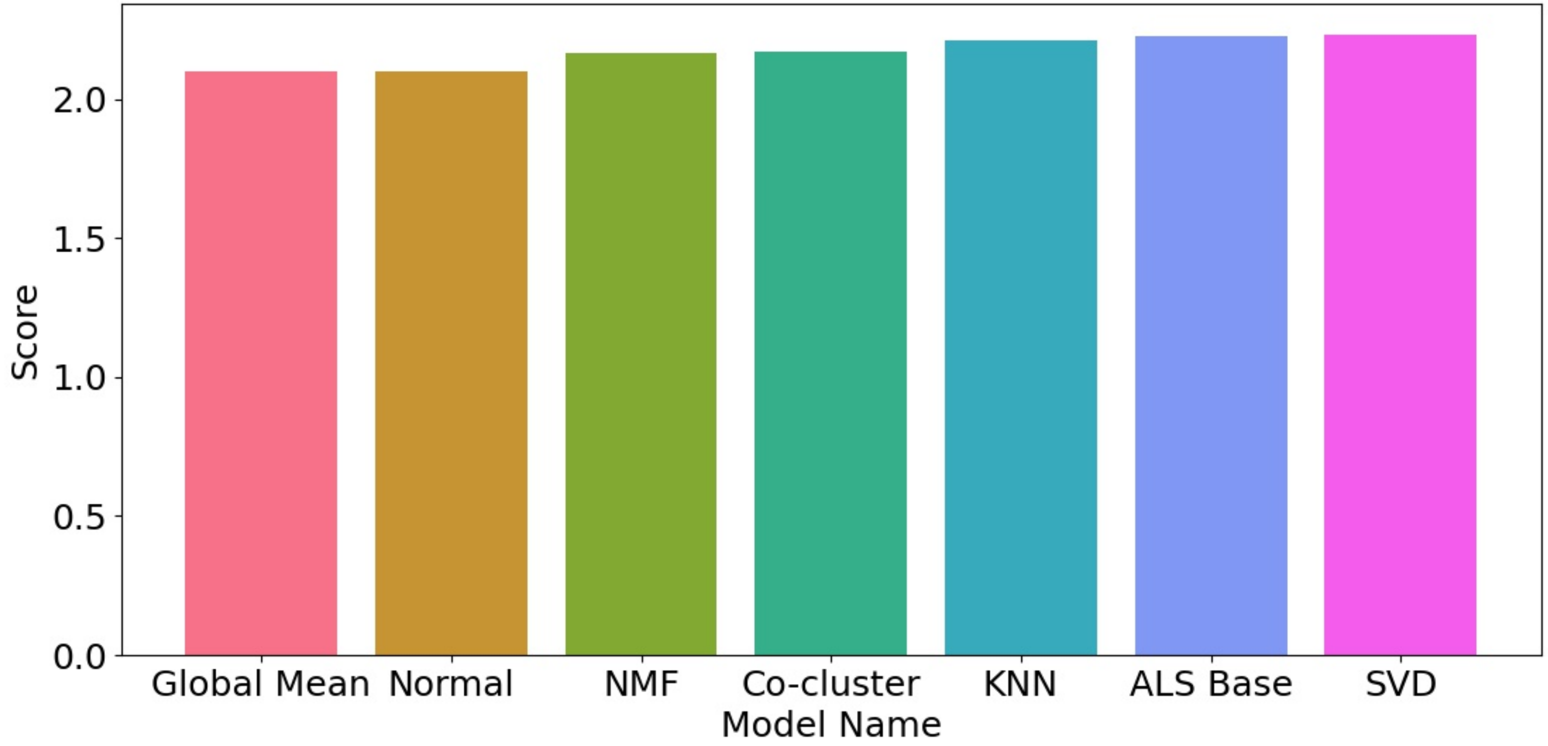
Future Work

- Incorporate >50 categories of song metadata for item-item similarity
  - ~300 GB of metadata from Million Song Dataset
- Expand to much larger song catalog
- Incorporate more complex criteria
  - Diversity
  - Harmony
  - Serendipity

Flask App Demonstration

Appendix: Alternate Metric

- Score using mean of top 5% of songs from each user



Appendix: SVD Model

- The Surprise SVD model calculates the prediction using matrix factorization:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

- Then it uses SGD to define baseline values μ, b_u, b_i to minimize the loss:

$$\sum_{r_{ui} \in R_{train}} \left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda\left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2\right)$$

- This process repeats until stopping criteria is reached