Music Recommendation System

Kojen Wang @ MIT Applied Data Science Final Presentation

Agenda

- Problem Definition
- Data Computation Flow
- Potential Models
- Solution Design
- Evaluating the Final Model
- Limitations & Recommendations

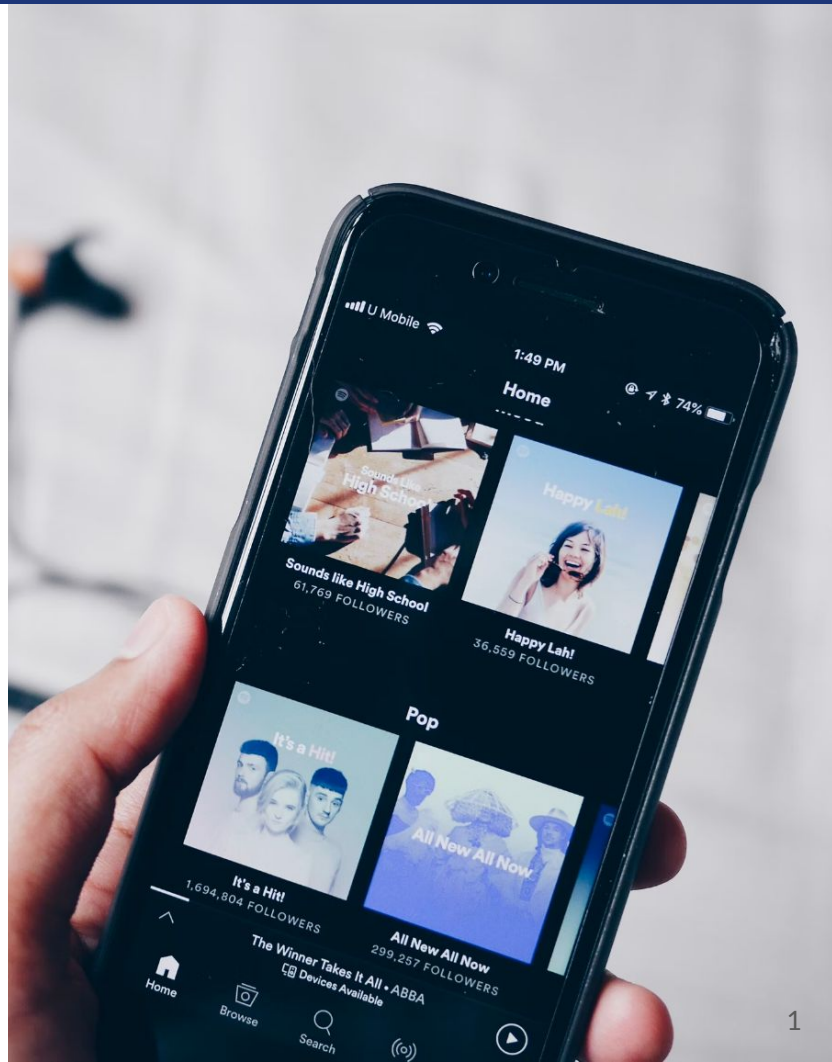
Problem Definition

Why do we need music recommender system?

- Increase the company's revenue
- Increase user satisfaction
- Increase user usage and retention
- Provide engaging music discovery experience
- Promote new artists, benefiting the record labels
- Help music professionals understand what listeners prefer and how to compete in the growing market

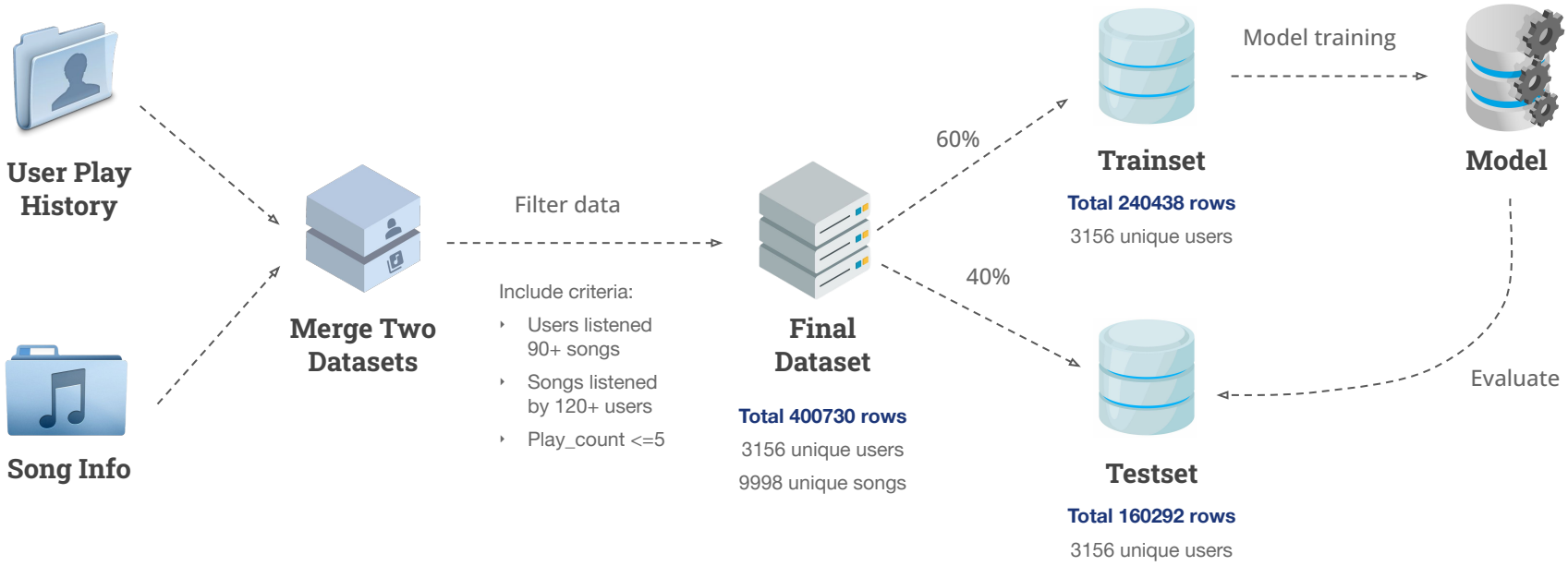
Problem to Solve

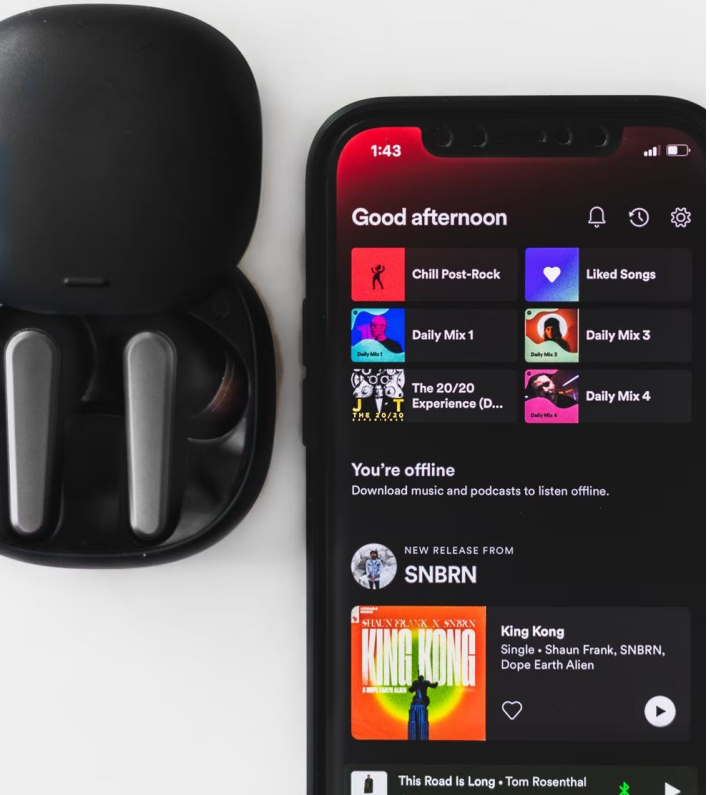
How can we build a music recommendation system that proposes the top 10 songs for each user based on the likelihood of listening to those songs?



Data Computation Flow

How do we solve this problem?





Potential Models

What types of models have we tried so far?

Using These Algorithms to Build Models

- User-user similarity-based collaborative learning
- Item-item similarity-based collaborative learning
- Model-based collaborative filtering
- Cluster-based learning

Success Metrics

- Precision@K
- Recall@K
- F1 Score@K

Performance Evaluation

Which algorithm is performing better?

Algorithms / Metrics	User-user Baseline	User-user Tuned	Item-item Baseline	Item-item Tuned	Model Based	Model Based Tuned	Cluster Based	Cluster Tuned
Run Time	20.9 s		**21.5 s**		19.5 s		**8.19 s**	
Precision	0.404	0.448	0.404	0.464	0.42	0.419	0.411	0.406
Recall	0.263	0.29	0.263	0.286	0.25	0.247	0.245	0.234
F-1 Score	0.319	0.352	0.319	**0.354**	0.313	0.311	0.307	**0.297**

- **Tuned item-item model** has the highest F1 score. It is performing relatively **better**.
- **Tuned cluster based model** has the lowest F1 score. It is performing relatively **worse** but it has the shortest run time.

Overall, these models' performance are very similar.

Explore Our Solution

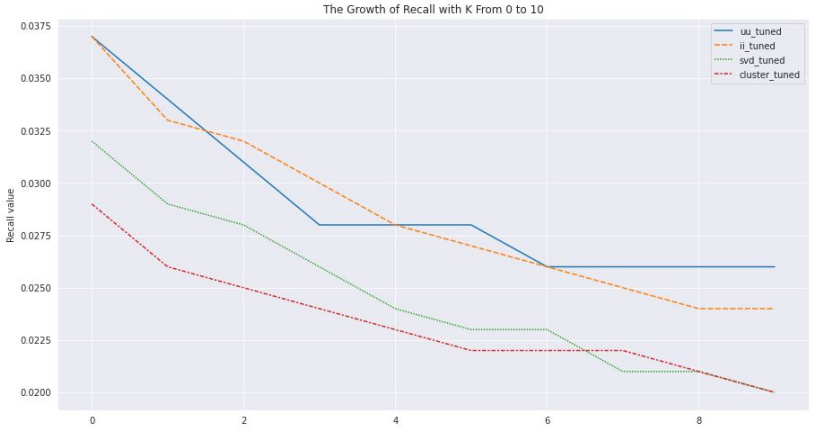
Could we combine different algorithms?

Define the **Intersection Rate** to measure how similar the recommendations are provided by two different algorithms.

Algorithms / Algorithms	User-user Baseline	Item-item Baseline	Model Based	Cluster Based
User-user	1	0.03	**0.11**	**0.00**
Item-item	0.03	1	**0.00**	**0.00**
Model-based	**0.11**	**0.00**	1	0.02
Cluster-based	**0.00**	**0.00**	0.02	1

In general, different algorithms provide different lists of recommendations.

Check the growth of Recall for all models, which has decreased as the K value increases.



The Kth prediction power is decreasing when we provide more recommendations.

This shows the potential that we can combine different algorithms to optimize the performance of recommender systems.

Solution Design

What is our final proposed solution?

We see the opportunity to use a hybrid model because …

- There is no dominant model as these models have similar model performance. We are unable to choose a model to be our solution.
- Different algorithms provide different lists of recommendations (i.e., low intersection rate)
- The Kth prediction power is decreasing when we provide more recommendations. So in our solution, we can select the top subset of recommendations from different models to boost the model performance.

Strengths of using the hybrid model

- Provide various types of songs to users based on the nature of different algorithms
- Capable of satisfying different types of users, including new and returning users (i.e., prevent cold-start problem)

Evaluating the Hybrid Model

Would combining different models yield a better performance?

Algorithms / Metrics	User-user Tuned	Item-item Tuned	SVD Tuned	Cluster Tuned	Hybrid (User-User Tuned@5 + Item-item Tuned@5)
Precision	0.448	0.464	0.419	0.406	**0.467**
Recall	0.29	0.286	0.247	0.234	**0.302**
F1 Score	**0.352**	**0.354**	0.311	0.297	**0.367**

- Combine the **tuned user-user model** and the **tuned item-item model** to create a hybrid model.
- The hybrid model's Precision, Recall and F1 Score have been the **highest**.
- The hybrid model's performance has greatly improved in comparison to other types of models.
- The hybrid solution is very effective in recommending the top 10 songs that are relevant to users.

We are excited to propose this hybrid model as our final solution.

Limitations & Recommendations

What are our recommendations for stakeholders?

Limitations of Missing Features

- Without timestamps, it is hard to identify the new and returning users
- Without music genre, it is hard to develop content-based recommendation model

Recommendations

- Fine-tuning the hybrid model by combining different models with different K values. This may help find another good performing model if any
- Incorporate additional data points/features to help with further analysis (e.g. experiment and improve the model)
- With timestamps, try experimenting and segmenting the user type before implementing the solution
- Design the machine learning pipeline to retrain models based on the new observed data

Thanks for listening!

APPENDIX: Recall@K of Different Models

Do we see patterns of recall values as K increases?

We have analyzed the Recall@K of different models with K from 1 to 10. It is shown that the Recall of all these models slightly increase as the value of K increases.

K / Algos	1	2	3	4	5	6	7	8	9	10
User-user Tuned	0.037	0.071	0.102	0.13	0.158	0.186	0.212	0.238	0.264	0.29
Item-item Tuned	0.037	0.07	0.102	0.132	0.16	0.187	0.213	0.238	0.262	0.286
SVD Tuned	0.032	0.061	0.089	0.115	0.139	0.162	0.185	0.206	0.227	0.247
Cluster Tuned	0.029	0.055	0.08	0.104	0.127	0.149	0.171	0.193	0.214	0.234

Note: The values are the Recall@K of different algorithms.

Performance Evaluation

Which algorithm is performing better?

Algorithms / Metrics	User-user Baseline	User-user Tuned	Item-item Baseline	Item-item Tuned	Model Based	Model Based Tuned	Cluster Based	Cluster Tuned
Run Time	20.9 s		**21.5 s**		19.5 s		**8.19 s**	
RMSE	1.0919	1.0546	1.0919	1.0222	1.0295	1.0177	1.0611	1.0911
Precision	0.404	0.448	0.404	0.464	0.42	0.419	0.411	0.406
Recall	0.263	0.29	0.263	0.286	0.25	0.247	0.245	0.234
F-1 Score	0.319	0.352	0.319	**0.354**	0.313	0.311	0.307	**0.297**

- **Tuned item-item model** has the highest F1 score. It is performing relatively **better**.
- **Tuned cluster based model** has the lowest F1 score. It is performing relatively **worse** but it has the shortest run time.

Overall, these models' performance are very similar.