# MUSIC RECOMMENDER SYSTEM

Shefali Garg
Fangyan Sun

# An everyday Problem

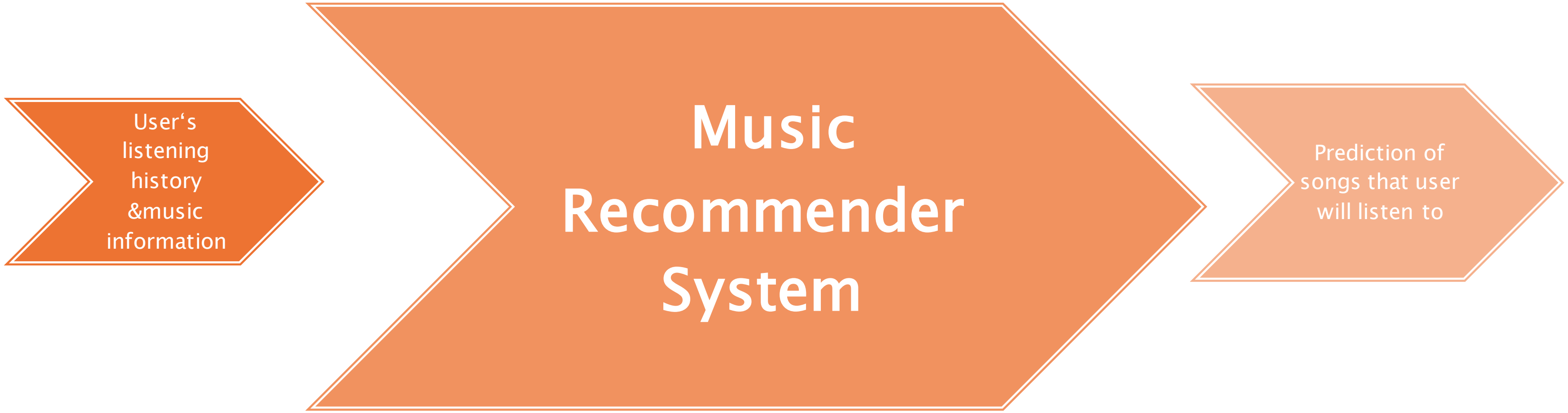
## Why a recommender system?

Music dataset is too big while life is short!!!! You need someone to teach you how to manage and give you **wise** suggestions according to **your taste**!

Try our recommender system!!!

Music service providers need a more efficient system to attraction their clients!

## Our task


User's listening history &music information
Music Recommender System
Prediction of songs that user will listen to

Our system: off-line system

## Dataset

Provided by Kaggle's Million Song Dataset Challenge

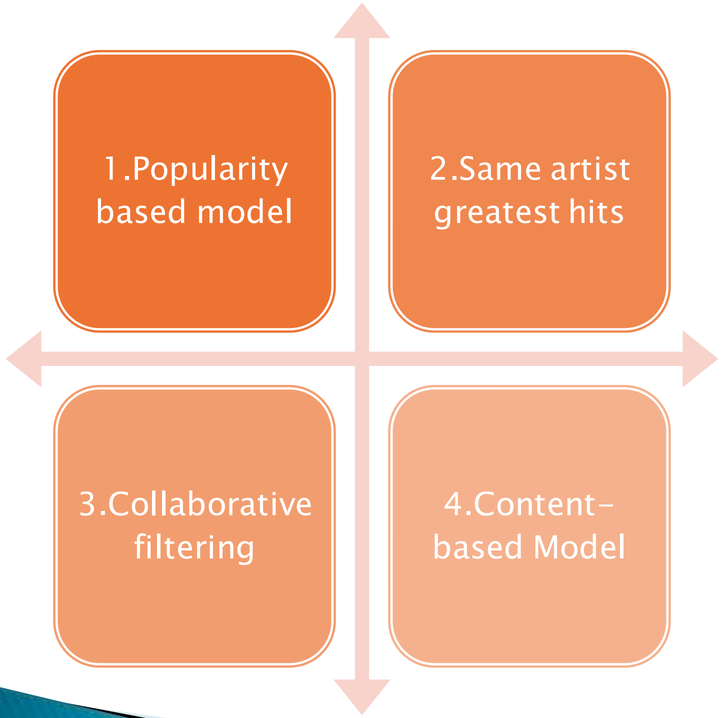
- **Features:**
  - Large-scale: 1 000 000 users
    15000 000 songs
  - Open
  - Implicit feedback
- **Content:**
  - Triplets (user, song, count)
  - Meta-data, content-analysis
  - No users' demographic information, timestamp

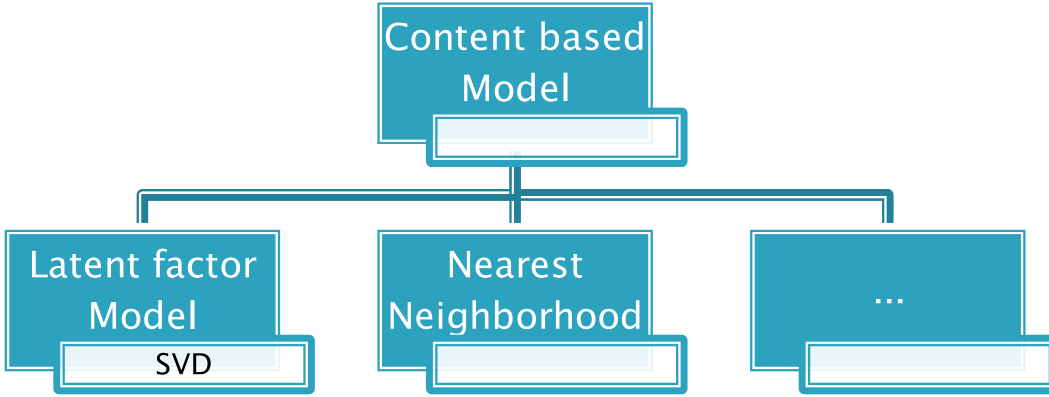
**Features & Content**

- **Too big dataset:**
  - Difficult to implement the whole dataset, so need to create a small dataset by ourselves
- **Format of the Dataset:**
  - Hdf5 files
  - Need to be opened by a Python Wrapper

**Difficulties linked to the Data**

# Four ideas


1.Popularity based model
2.Same artist greatest hits
3.Collaborative filtering
4.Content-based Model


Content based Model
Latent factor Model
SVD
Nearest Neighborhood
…

## 1.Popularity based model

- Idea

1. Sort songs by popularity in a decreasing order
2. For each user, recommend the songs in order of popularity, except those already in the user's profile

- Pros:

- Idea is simple
- Easy to implement
- Served as **baseline**

- Cons:

- Not personalized (users and songs' information is not taken into account)
- Some songs will never be listend

Idea & Steps

Pros & Cons

## 2.Same artist greatest hits

- Idea

1. Sort songs by popularity in a decreasing order
2. For each user, the ranking of songs is re-ordered to place songs by artists
3. recommend the songs in the new order, except those already in the user's profile

- Pros:
  - Idea is simple
  - Easy to implement
  - Minimum personalized

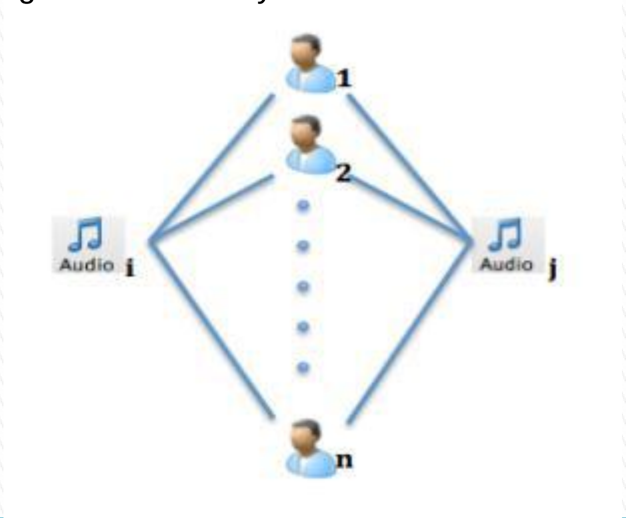
- Cons:
  - Only single-meta-data is used
  - Maximally conservative: doesn't explore the space byond songs with which the user is likely already familiar

Idea & Steps

Pros & Cons

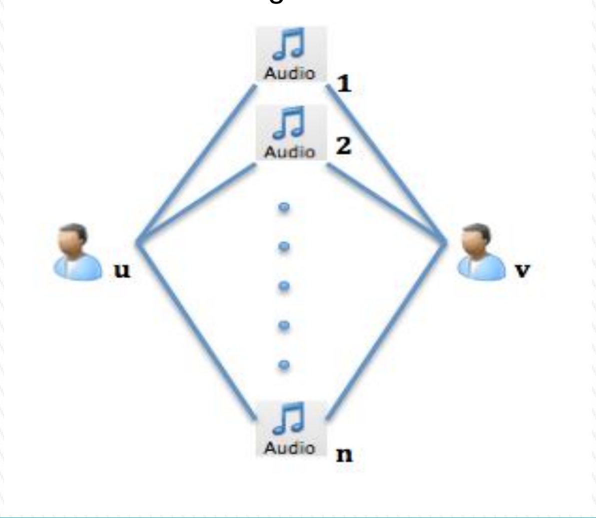
# 3. Collaborative Filtering

Idea: songs that are often listened by the same user tend to be similar and are more likely to be listened together in future by some other user.



Item-based

Idea: users who listen to the same songs in the past tend to have similar interests and will probably listen to the same songs in future.



User-based

## e.g. user-based

Conditional probability measure of similarity between two users:

$$\boldsymbol{W_{u,v}} = \mathbf{P(v|u)}^{\boldsymbol{\alpha}}.\mathbf{P(u|v)}^{\mathbf{1-}\boldsymbol{\alpha}} \text{ with } \alpha \epsilon \, [0,1]$$

## Social recommendation:

e.g. If two users are friends, rescale $W_{u,v}$ to give more weight to the similarity

## Locality of scoring function:

It aggregates the scores obtained using individual users or items

$$\boldsymbol{f(w) = w^{q} \; with \; q \in N}$$

Exponential, so smaller weights drop to zero, higher ones emphasized

Stochastic aggregation of item-based and user-based ranker

# 4.Content-based Model

1. Based on item's description and user's preference profile
2. Not based on choices of other users with similar interests
3. We make recommendations by looking for music whose features are very similar to the tastes of the user

What's content-based model?

- Similarity != recommendation (no notion of personalization)
- Majority of songs have too few listeners, so difficult to "collaborate"

And Why?

## Nearest-neighbor technique

- 1. Create a space of songs according to songs features. We find out neighborhood of each song.
- 2. We look at each user’s profile and suggest songs which are neighbors to the songs that he listens to

# Latent factor model

- Idea: SVD
- Listening histories are influenced by a set of factors specific to the domain (e.g. Genre, artist).
- These factors are in general not obvious and we need to infer those so called latent factors from the data.
- Users and songs are characterized by latent factors.

Idea

- Personalized
- Meta-data is fully used, all the information is well explored
- It works well in many tested cases

# Basic notations

- Matrix M, a user-song play count matrix

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | … | | | |
| 1 | 1 | 0 | 0 | 0 | 0 | | | | |
| 0 | 1 | | | | | | | | |
| … | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

$M \in \{0,1\}^{m*n}$ where m is the number of users and n the number of songs

$M_{u,i} = 1 \; if \; user \; u \; listens \; to \; item \; i$

Otherwise 0

## SVD (singular value decomposition)

- Decomposes M into a latent feature space that relates users to songs

$$M = U^T . V$$

with M $\in R^{m*n}$ , $U \in R^{k*m}$ and V $\in R^{k*n}$

- U is a low-rank representation of m users and k features containing the so-called ‘user factors’. Each row of U, represents user’s degree of interest in each topic.
- V represents the n songs containing the so-called ‘song factors’. Each row of V represents the relevance of the song v to each topic.
- Personalized recommendations are calculated for a user u by ranking each item i in descending order of the predicted feedback:

$$W_i = U_u^T . V_i$$

- perform stochastic gradient descent on objective function

## Evaluation Metric

- Off-line evaluation
- Truncated mAP (mean Average Precision)

$precision-at-k$: $proportion\ of\ correct\ recommendations\ within\ the\ top-k\ of\ the\ predicted\ ranking$: $\boldsymbol{P_k(u,y) = \frac{1}{k}\sum_{j=1}^{k} M_{u,y}(j)\ ,\forall k \leq t}$

$for\ each\ user, the\ average\ precision\ at\ each\ recall\ point$:

$$\boldsymbol{AP\ (u,y) = \frac{1}{n_u}\sum_{j=1}^{t} P_k(u,y)\ .\ M_{u,y}(k)}$$

mean average precision: $\mathbf{mAP} = \boldsymbol{\frac{1}{m}\sum AP(u, y_u)}$

# Challenge1 Can we believe in Matrix M? Problem of implicit feedback

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | ... | | | |
| 1 | 1 | 0 | 0 | 0 | 0 | | | | |
| 0 | 1 | | | | | | | | |
| ... | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

1. Haven’t listend to a song != dislike it. The « 0 » gives a lot confusion and little confidence.

2. We use weighted matrix factorization

3. Each entry is weighted by a confidence function so as to put more confidence on non-zero entries

## Challenge2 Non user-specific parameterization

- Represent users as a combination of item features
- User rating matrix R is estimated by the factorization:

$$R = U . \sum . Q^T$$

$$P = U . \sum$$

with $P_u$ represents the user factors vector while $Q_i$ represents the item factors vector

$$P = R . Q$$

as U and Q have orthonormal columns

## Challenge3 Dimensionality for SVD

- First latent factors capture properties of the most popular items, while the additional latent factors represent more refined features related to unpopular items.
- Number of latent factors influences the quality of long-tail items differently than head items.

# Reference

- [1] McFee, B., BertinMahieux,T., Ellis, D. P., Lanckriet, G. R. (2012, April). *The million song dataset challenge*. In Proceedings of the 21st international conference companion on World Wide Web (pp. 909916).ACM.

- [2] Aiolli, F. (2012). *A preliminary study on a recommender system for the million songs dataset challenge*. PREFERENCE LEARNING: PROBLEMS AND APPLICATIONS IN AI

- [3] Koren, Yehuda. *"Recommender system utilizing collaborative filtering combining explicit and implicit feedback with both neighborhood and latent factor models."* U.S. Patent No. 8,037,080. 11 Oct. 2011.

- [4] Cremonesi, Paolo, Yehuda Koren, and Roberto Turrin. "**Performance of recommender algorithms on top-n recommendation tasks**." *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010

# Thank you!

Any questions or suggestions?