Recommender System (MSBA 7027)

Zhengli Wang

Faculty of Business and Economics
The University of Hong Kong
2023


NETFLIX
Netflix Prize
Home
Rules
Leaderboard
Register
Update
Submit
Download
Welcome!
The Netflix Prize seeks to substantially improve the accuracy of predictions about how much someone is going to love a movie based on their movie preferences. Improve it enough and you win one (or more) Prizes. Winning the Netflix Prize improves our ability to connect people to the movies they love.
Read the Rules to see what is required to win the Prizes. If you are interested in joining the quest, you should register a team.
You should also read the frequently-asked questions about the Prize. And check out how various teams are doing on the Leaderboard.
Good luck and thanks for helping!

Recommender Systems

- Overview
- Content-based
- Collaborative filtering
  - User-based Collaborative filtering
  - Item-based Collaborative filtering
- Latent-factor model

Recommender Systems: Overview

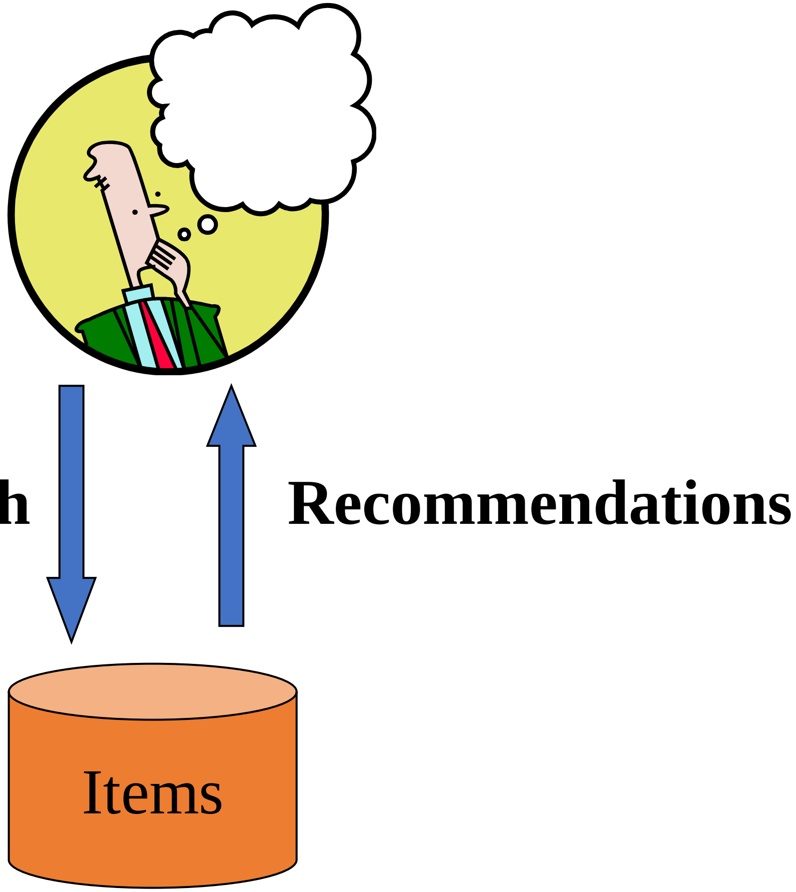
- Imagine a situation: user interacts with large catalog of items

Recommender Systems: Overview

- Imagine a situation: user interacts with large catalog of items

- Key feature: #items large, e.g. millions/billions
- Moreover, user doesn't know exactly what he/she wants

- Two ways user interacts with platform
  - Search
  - Recommendation



Recommender Systems: Overview

- **Why recommendation becomes so important in the last 10 years**
  - From an era of scarcity to an era of abundance

- Past: Era of scarcity
  - E.g. Shelf space is a scarce commodity for traditional retailers (like Walmart)

- Now: Era of abundance
  - E.g. Amazon/Taobao
  - Web enables near-zero-cost dissemination of information about products

- Gives rise to the “Long tail” phenomenon

Long Tail

Imagine a graph

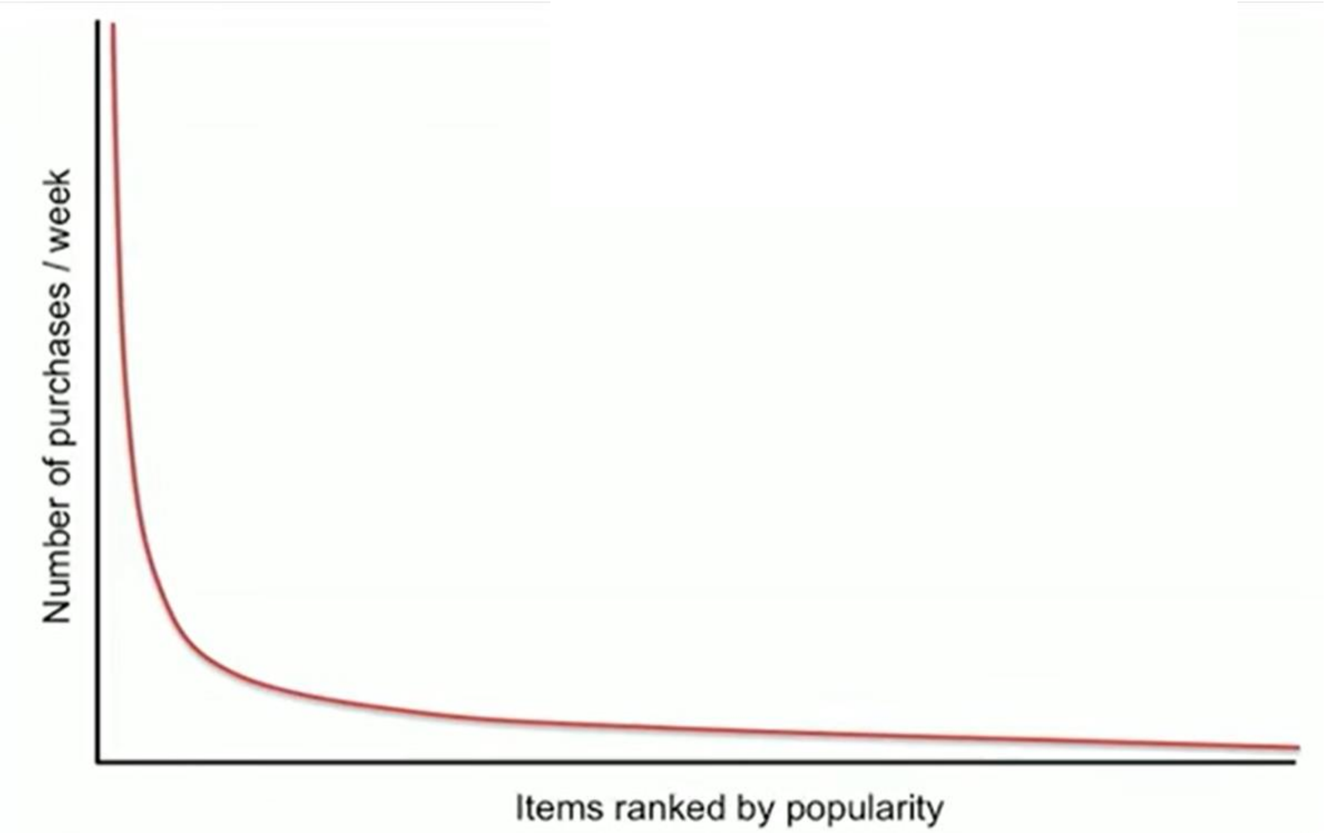
- X: items from a catalog (ranked by popularity)
  - Decreasing levels of popularity (left: more popular)
- Y: Actual popularity

Curve

- steep fall initially
  - Very few super-popular items
- Then slope becomes less steep
  - Never really reach the x-axis

Certain item

- purchase frequency not very high, but also not very low
  - e.g. once a week
- Not worthwhile for physical retailer to stock this item



(Left: more popular)

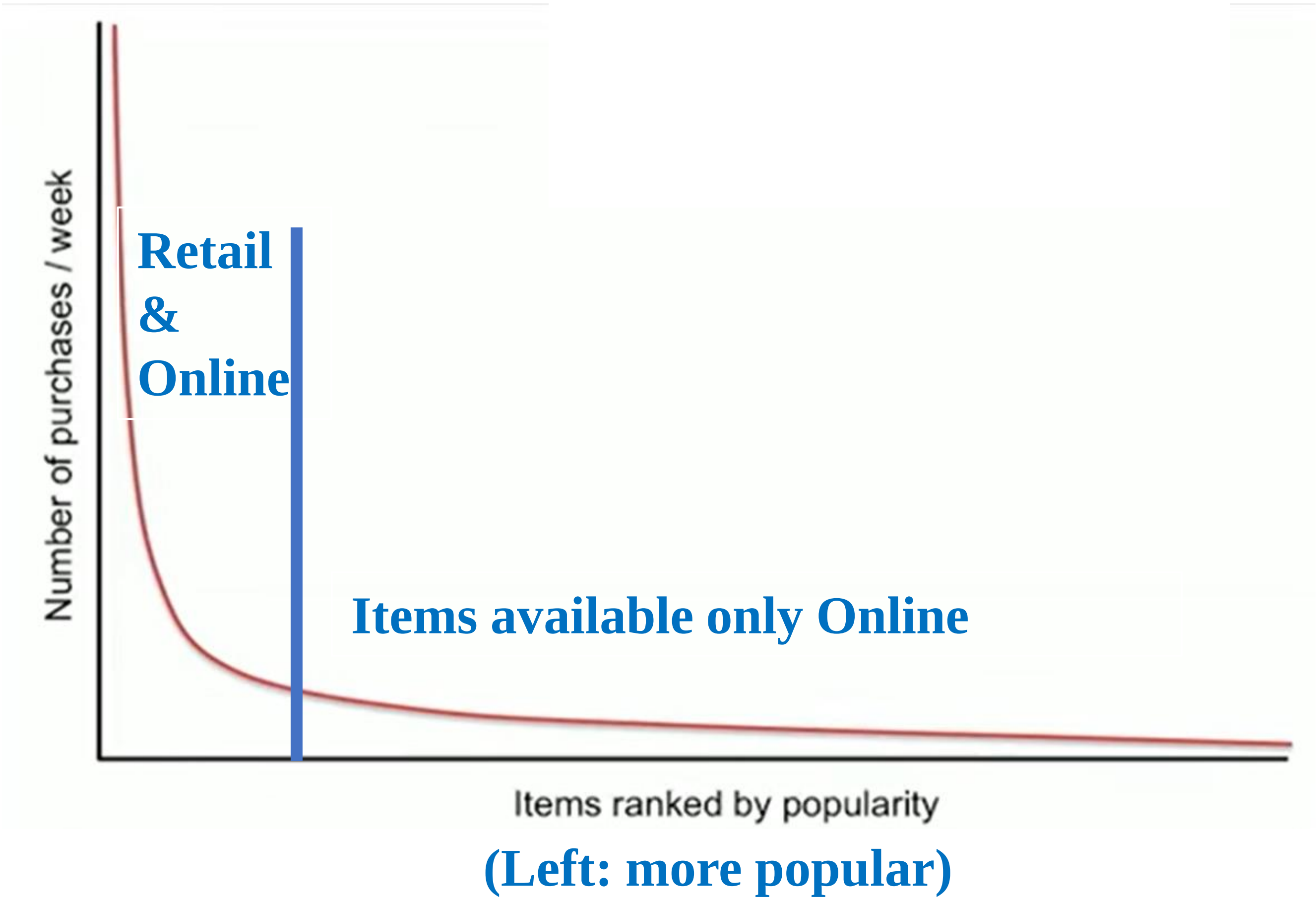
Long Tail

- **Cutoff point**
  - Left: retail & online
  - Right: only online

- **Phenomenon see in**
  - Movies, music, products, etc

- **Area under the curve** to the right may be larger than to the left

- **Problem:** too many items, how to let users know them



Long Tail

- **Problem:** too many items, how to let users know them

- Need a **better way** to let users find them easily
  - Recommendation systems

- **Examples**
  - Books, movies, music
  - News articles
  - People (friend recommendations on Facebook, LinkedIn, and Twitter)

Types of Recommendations

- **Manually selected**
  - List of favorites
  - Lists of “essential” items

- **Simple aggregates**
  - Top 10, Most Popular, Recent Uploads

- **Tailored to individual users**
  - E.g. Amazon, Netflix, Pandora
  - Our focus here

Formal Model

- ***X*** = set of **Customers**
- ***I*** = set of **Items**
- **Utility function *u*: *X* × *I* → *R***
  - ***R***: rating
  - ***R*** ∈ an ordered set
    - e.g., **1-5** stars

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Key Problems

- **1) Gathering known ratings for matrix**
  - Different ways of collecting the data in the utility matrix

- **2) Infer unknown ratings from the known ones**
  - Different ways of extrapolating unknown ratings for a user

Let’s talk about each in detail

Key Problems

- **1) Gathering Known Ratings**
- **Explicit**
  - Ask people to rate items
  - Given incentives for people to rate items

- **Implicit**
  - Learn ratings from user actions
    - E.g., purchase implies high rating
  - Ways to learn about low ratings

Key Problems

- **2) Infer Unknown Ratings**
- Suppose we have gathered enough ratings, how to extrapolate

- Popular approaches to recommender systems:
  - Content-based
  - Collaborative Filtering
    - User-based
    - Item-based
  - Latent-factor based

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Content-Based

- **Main Idea:** Recommend items to customer x similar to previous items rated highly by x

- **Examples:**
- Movie recommendations
  - Recommend movies with same actor(s), director, genre, …
- Book recommendations
  - Recommend books with same author(s), topic, …
- News recommendations
  - Recommend other news with “similar” content

Content-Based

- For each item, create an **item profile**

- Item profile involves a set (vector) of features
- For example
  - Movies: movie types, actor(s), title, director,…
  - Text: Set of “important” words in document

- How to pick important features?
  - Common techniques from text mining: TF-IDF (Term freq * Inverse Doc Freq)

Content-Based

- TF-IDF (for text mining, Optional)
  - TF: count of term in the doc / document length
  - IDF: log (#documents / #doc that contains the term)
  - TF-IDF score: TF * IDF

E.g.

- She is pretty.
- He is handsome.
- Both he and she are experts in machine-learning.

	d1	d2	d3
handsome	0	(1/3)*log(3/1)	0
She	(1/3)*log(3/2)	0	(1/8)*log(3/2)

Content-Based

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Content-Based

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1
Features							
x_1	0.8	1	0.9	0.1	0.9	0.8	0.01
x_2	0.1	0.01	0.02	1	0.3	0.5	0.7

e.g. x_1: romance,
x_2: action
/
x_1: seriousness,
x_2: female-oriented

- For user *i* & movie *j*, predict rating as: $\theta^{(i)\prime} x^{(j)}$
- $\theta^{(i)}$ will be learnt

e.g. User 1 (with para $\theta^{(1)}$) & movie 2 ($x^{(2)} = (1, 0.01)'$)

Content-Based

Decision variable: $\{\theta^{(i)}\}, i = 1, 2, 3, \ldots$

$$\min \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{i} \sum_{k} \left(\theta_k^{(i)}\right)^2$$

$\theta^{(i)}$: user *i*'s para
$x^{(j)}$: movie *j*'s para
$w^{(i,j)}$: value = 1 if user *i* has rated movie *j*
$r^{(i,j)}$: rating value (movie *j* rated by user *i*)

This is just a linear regression problem!

Content-Based

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Features							
x_1	0.8	1	0.9	0.1	0.9	0.8	0.01
x_2	0.1	0.01	0.02	1	0.3	0.5	0.7

Decision variable: $\theta^{(i)}$: user i's para

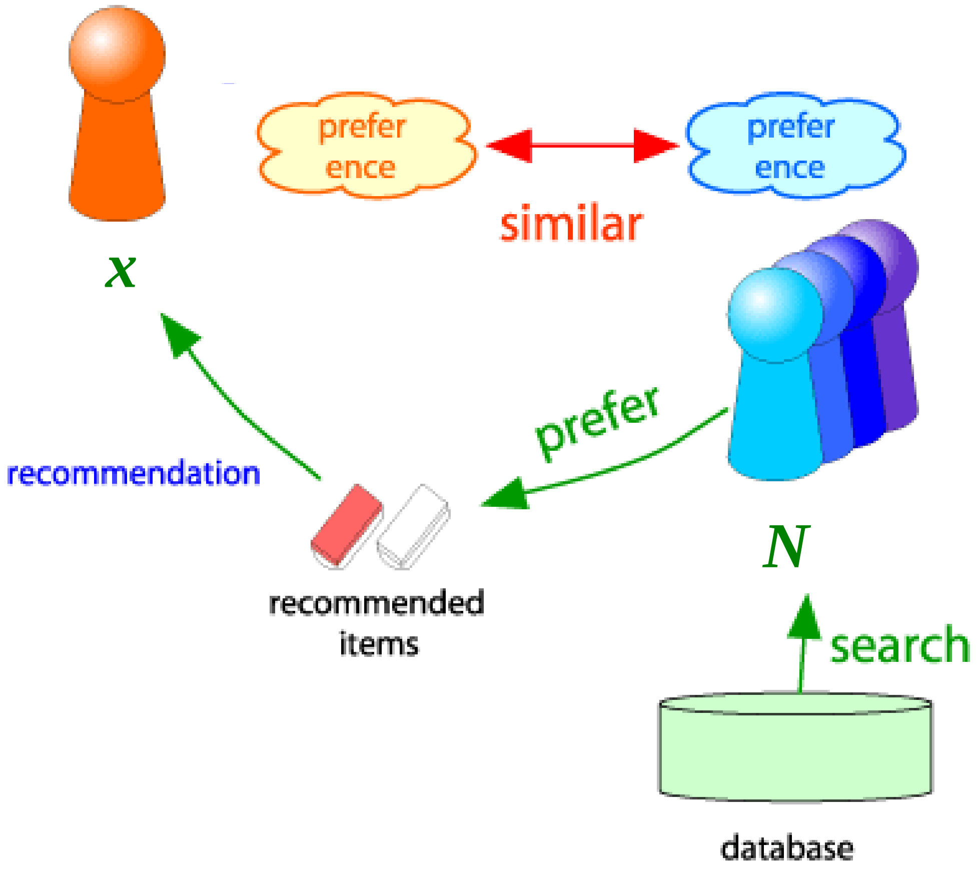
User 1: $\min\ [4 - \left(\theta_1^{(1)}, \theta_2^{(1)}\right) \cdot (0.8, 0.1)]^2 + [5 - \left(\theta_1^{(1)}, \theta_2^{(1)}\right) \cdot (0.1, 1)]^2 + [1 - \left(\theta_1^{(1)}, \theta_2^{(1)}\right) \cdot (0.9, 0.3)]^2$

User 2: $\min\ [5 - \left(\theta_1^{(2)}, \theta_2^{(2)}\right) \cdot (0.8, 0.1)]^2 + [5 - \left(\theta_1^{(2)}, \theta_2^{(2)}\right) \cdot (1, 0.01)]^2 + [4 - \left(\theta_1^{(2)}, \theta_2^{(2)}\right) \cdot (0.9, 0.02)]^2$

…

Collaborative Filtering

- Consider user ***x***
- Find set ***N*** of other users whose ratings are "**similar**" to ***x***'s ratings
- Estimate ***x***'s ratings based on ratings of users in ***N***



Motivating Example

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

- Consider user ***x*** and ***y*** with rating vectors r_x and r_y

- Need similarity metric s(x,y)
  - A & B seem more similar (M1)
  - A & C seem more dissimilar (M4 & M5)

- Want similarity metric to capture the intuition: s(A,B)>s(A,C)

Motivating Example: 1st Approach (Jaccard)

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

- $s_J(x, y) = |r_x \cap r_y| / |r_x \cup r_y|$
- $s_J(A, B) = 1/5$; $s_J(A, C) = 2/4$
  - $s_J(A, B) < s_J(A, C)$
- **Problem**: ignores rating values

Motivating Example: 2nd Approach (Cosine)

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

- $s_C(x, y) = \frac{r_x \cdot r_y}{|r_x||r_y|}$
- $s_C(A, B) = \frac{4 \cdot 5}{\sqrt{4^2+5^2+1^2}\sqrt{5^2+5^2+4^2}} = \frac{20}{\sqrt{42}\sqrt{66}} = 0.38$
- $s_C(A, C) = \frac{5 \cdot 2 + 1 \cdot 4}{\sqrt{4^2+5^2+1^2}\sqrt{2^2+4^2+5^2}} = \frac{14}{\sqrt{42}\sqrt{45}} = 0.32$
  - $s_C(A, B) > s_C(A, C)$, but we can do better

Motivating Example: 3^{rd} Approach (Centered-Cosine)

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Standardize each user's ratings

User/Movie	1	2	3	4	5	6	7
A	2/3			5/3	- 7/3		
B	1/3	1/3	- 2/3				
C				- 5/3	1/3	4/3	
D		1					- 1

Motivating Example: 3rd Approach (Centered-Cosine)

User/Movie	1	2	3	4	5	6	7
A	2/3			5/3	- 7/3		
B	1/3	1/3	- 2/3				
C				- 5/3	1/3	4/3	
D		1					- 1

- $s_{CC}(x, y) = \frac{r_x \cdot r_y}{|r_x||r_y|}$, where r_x, r_y are standardized ratings
- $s_{CC}(A, B) = 0.09$
- $s_{CC}(A, C) = -0.56$
  - $s_{CC}(A, B) > s_{CC}(A, C)$
  - Captures intuition better
    - Missing ratings treated as “average”
    - Remove bias for users who consistently rate high/low

Predicting ratings

- Goal: given user $\boldsymbol{x}$ and movie i, predict rating r_{xi}

- Two Popular Approaches:
  - User-based Collaborative Filtering
  - Item-based Collaborative Filtering

Predicting ratings

User-based collaborative filtering

- Let r_x be the vector of user *x*'s ratings
- Let *N* be the set of *k* users most similar to *x* who have rated movie *i*
- Prediction for user *x and* movie *i* :
  - $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$
  - where s_{xy} is the similarity between user *x* and user *y*

Predicting ratings

Item-based collaborative filtering (Dual Approach)

- Let r_x be the vector of user *x*'s ratings
- Let *M* be the set of *k* movies (which user *x* has rated) most similar to *i*
- Prediction for user *x and* movie *i* :
  - $r_{xi} = \frac{\sum_{j \in M} s_{ij} \cdot r_{xj}}{\sum_{j \in M} s_{ij}}$
  - where s_{ij} is the similarity between movie *i* and movie *j*

Predicting ratings: Example

User/Movie	1	2	3	4	5	6
1	5	5	5		3	2
2		2	3	5		2
3	1	5	3	4	2	
4	2			3		4
5	4		5	5	4	2
6	3	2		4	3	
7	4	1	?	4	1	
8			1		2	3

Predicting ratings: Example

User/Movie	1	2	3	4	5	6
1	1	1	1		- 1	- 2
2		- 1	0	2		- 1
3	- 2	2	0	1	- 1	
4	- 1			0		1
5	0		1	-1	0	- 2
6	0	-1		1	0	
7	1.5	- 1.5	?	1.5	- 1.5	
8			- 1		0	1

Predicting ratings: Example

User-Based Collaborative Filtering

User/Movie	1	2	3	4	5	6
1	1	1	1		- 1	- 2
2		- 1	0	2		- 1
3	- 2	2	0	1	- 1	
4	- 1			0		1
5	0		1	-1	0	- 2
6	0	-1		1	0	
7	1.5	- 1.5	?	1.5	- 1.5	
8			- 1		0	1

Similarity
0.18
0.61
-0.32
-0.35
-0.2
0.7
1
0

$$r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$$

- 2 Nearest users: $r_{73} = \frac{0.61 \cdot 0 + 0.18 \cdot 1}{0.61 + 0.18} = 0.25$
- What will be the predicted rating? **(Ans: 2.5+0.25)**

Predicting ratings: Example

Item-Based Collaborative Filtering

User/Movie	1	2	3	4	5	6
1	1	1	1		- 1	- 2
2		- 1	0	2		- 1
3	- 2	2	0	1	- 1	
4	- 1			0		1
5	0		1	-1	0	- 2
6	0	-1		1	0	
7	1.5	- 1.5	?	1.5	- 1.5	
8			- 1		0	1
Similarity	0.2	0.19	1	0.19	-0.28	-0.87

$$r_{xi} = \frac{\sum_{j \in M} s_{ij} \cdot r_{xj}}{\sum_{j \in M} s_{ij}}$$

- 2 Nearest movies: $r_{73} = \frac{0.2 \cdot 1.5 + 0.19(-1.5)}{0.2+0.19} = 0.03$
- What will be the predicted rating? **(Ans: 2.5+0.03)**

Latent-factor Recommendation system

Motivation: from Content-based (Quick Recap below)

Decision variable: $\{\theta^{(i)}\}, i = 1, 2, 3, \ldots$

$$\min \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{i} \sum_{k} \left(\theta_k^{(i)}\right)^2$$

$\theta^{(i)}$: user *i*'s para
$x^{(j)}$: movie *j*'s para
$w^{(i,j)}$: value = 1 if user *i* has rated movie *j*
$r^{(i,j)}$: rating value (movie *j* rated by user *i*)

Latent-factor Recommendation system

User/Movie	1	2	3	4	5	6	7
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					1

Features							
x_1	0.8	1	0.9	0.1	0.9	0.8	0.01
x_2	0.1	0.01	0.02	1	0.3	0.5	0.7

Decision variable: $\theta^{(i)}$: user i's para

User 1: $\min\ [4-\left(\theta_1^{(1)},\theta_2^{(1)}\right)\cdot(0.8,0.1)]^2+[5-\left(\theta_1^{(1)},\theta_2^{(1)}\right)\cdot(0.1,1)]^2+[1-\left(\theta_1^{(1)},\theta_2^{(1)}\right)\cdot(0.9,0.3)]^2$

User 2: $\min\ [5-\left(\theta_1^{(2)},\theta_2^{(2)}\right)\cdot(0.8,0.1)]^2+[5-\left(\theta_1^{(2)},\theta_2^{(2)}\right)\cdot(1,0.01)]^2+[4-\left(\theta_1^{(2)},\theta_2^{(2)}\right)\cdot(0.9,0.02)]^2$

…

Suppose it is hard to come up with features of the movies $x^{(j)}$,

What should we do?

Latent-factor Recommendation system

Suppose we do not have features of the movies $x^{(j)}$

We can still have an algorithm that learns $x^{(j)}$ by itself

Idea

$$\min \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{i} \sum_{k} \left(\theta_k^{(i)}\right)^2 + \lambda \sum_{j} \sum_{k} \left(x_k^{(j)}\right)^2$$

Given x, minimize θ
Given θ, minimize x

Latent-factor Recommendation system

More specifically

Fix $\{x^{(j)}\}$, $\quad min_{\{\theta^{(i)}\}} \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{i} \sum_{k} \left(\theta_k^{(i)}\right)^2$

Fix $\{\theta^{(i)}\}$, $\quad min_{\{x^{(j)}\}} \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{j} \sum_{k} \left(x_k^{(j)}\right)^2$

Latent-factor Recommendation system: Example

User/Movie	1	2
A	5	?
B	2	4

Features		
x_1		
x_2		

Goal: predict $r^{(1,2)}$

Latent-factor Recommendation system: Example

User/Movie	1	2
A	5	?
B	2	4

Features		
x_1	0.02	0.05
x_2	-0.04	0.03

Goal: predict $r^{(1,2)}$

Initialization: assign some random values $\{x^{(j)}\}$

Latent-factor Recommendation system: Example

User/Movie	1	2
A	5	?
B	2	4

Features		
x_1	0.02	0.05
x_2	-0.04	0.03

Goal: predict $r^{(1,2)}$

Given $\{x^{(j)}\}$ $$min_{\{\theta^{(i)}\}} \sum_i \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_i \sum_k \left(\theta_k^{(i)}\right)^2$$

Latent-factor Recommendation system: Example

User/Movie	1	2
A	5	?
B	2	4

Features		
x_1	0.02	0.05
x_2	-0.04	0.03

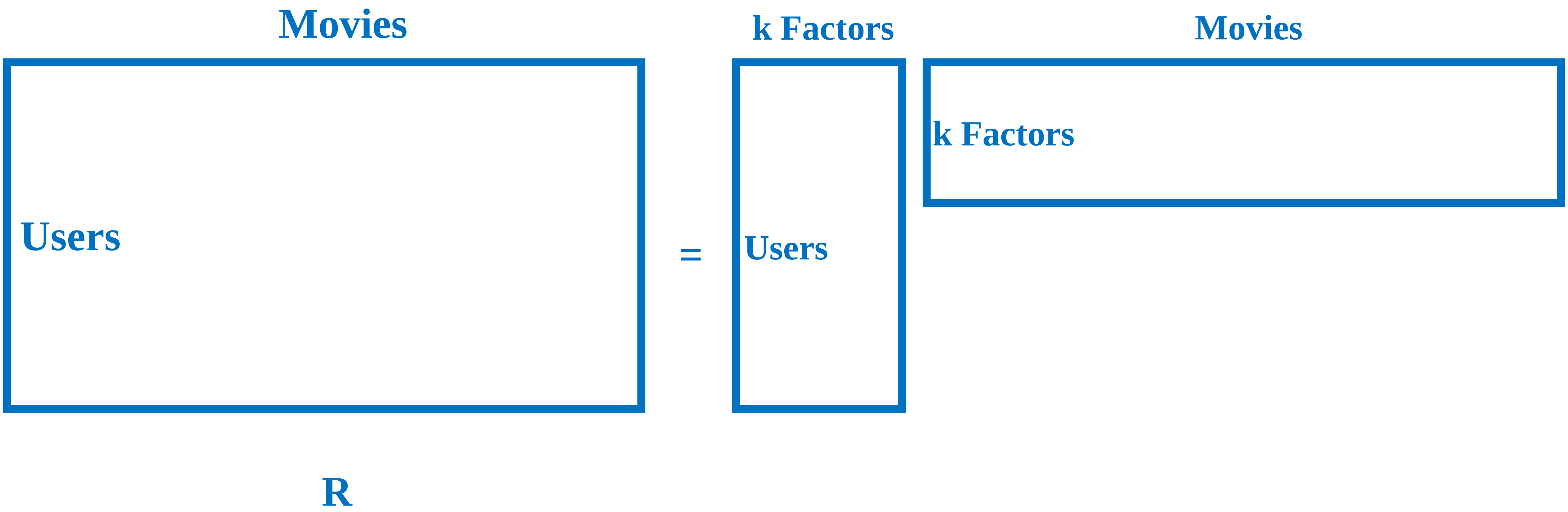
Goal: predict $r^{(1,2)}$

Given $\{\theta^{(i)}\}$

$$min_{\{x^{(j)}\}} \sum_{i} \sum_{j|w^{(i,j)}=1} (\theta^{(i)\prime} x^{(j)} - r^{(i,j)})^2 + \lambda \sum_{j} \sum_{k} \left(x_k^{(j)}\right)^2$$

Latent-factor Recommendation system

High level:



Each user and movie: represented by k-dim vector

Other Issues

Cold-start problem (for new user)

- Ask users to list their preferences (e.g. “tell me your 3 favorite movies”)
- Try to get user’s other info. (e.g. region, gender, browse history, etc.)
- Recommend popular & diverse items

1st-rater problem (for new item)

- Pay a group of people to rate them
- Content-based algo.