



10-315

Introduction to ML

Recommender Systems

Instructor: Pat Virtue

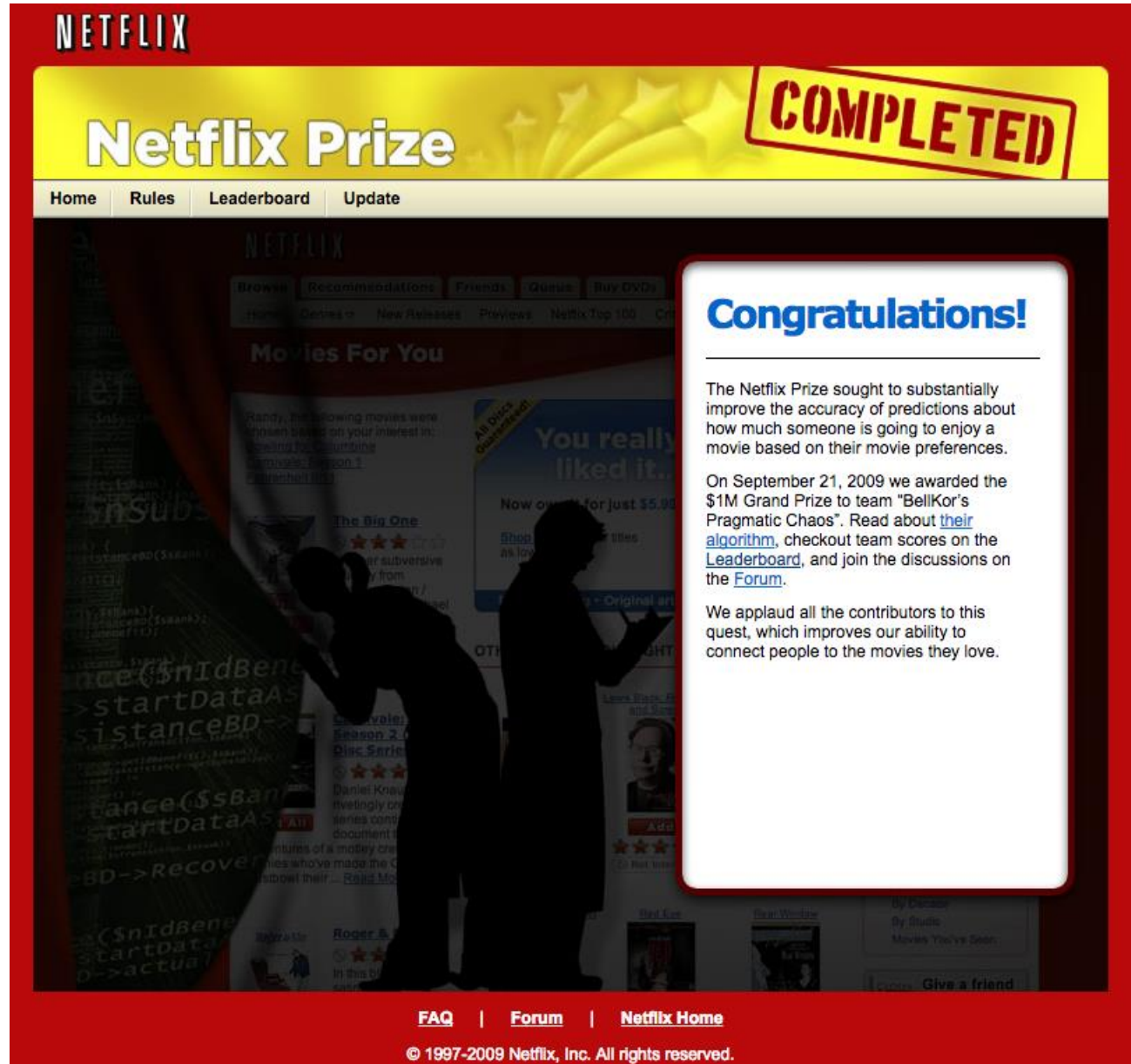
Recommender Systems

The screenshot displays the Amazon homepage with a personalized header for a user named Matt. The header includes the Amazon Prime logo, a search bar, and a 'CYBER MONDAY DEALS WEEK' banner. Below the header, there are links to 'Departments', 'Browsing History', 'Matt's Amazon.com', 'Cyber Monday', 'Gift Cards & Registry', 'Sell', and 'Help'. A 'Hello, Matt' greeting is followed by links to 'Your Account', 'Prime', 'Lists', and a shopping cart icon. A 'Sign In' button is also present.

The main section is titled 'Recommended for you, Matt' and features four categories of recommended items:

- Buy It Again in Grocery** (14 ITEMS): Includes products like Jif peanut butter, maple syrup, and various snack boxes.
- Buy It Again in Pets** (6 ITEMS): Includes pet products like Advantage II flea treatment, pet food, and pet toys.
- Buy It Again in Baby Products** (5 ITEMS): Includes baby products like Crayola crayons, baby wipes, and baby toys.
- Engineering Books** (86 ITEMS): Includes the book 'Probabilistic Graphical Models: Principles and Techniques' by Daphne Koller and Nir Friedman.

Recommender Systems



ML System Design: Movie Recommendation

Task

$$\text{userid} \quad \text{itemid} \quad \text{rating}$$
$$h(i, j) \rightarrow \hat{r}$$

Experience

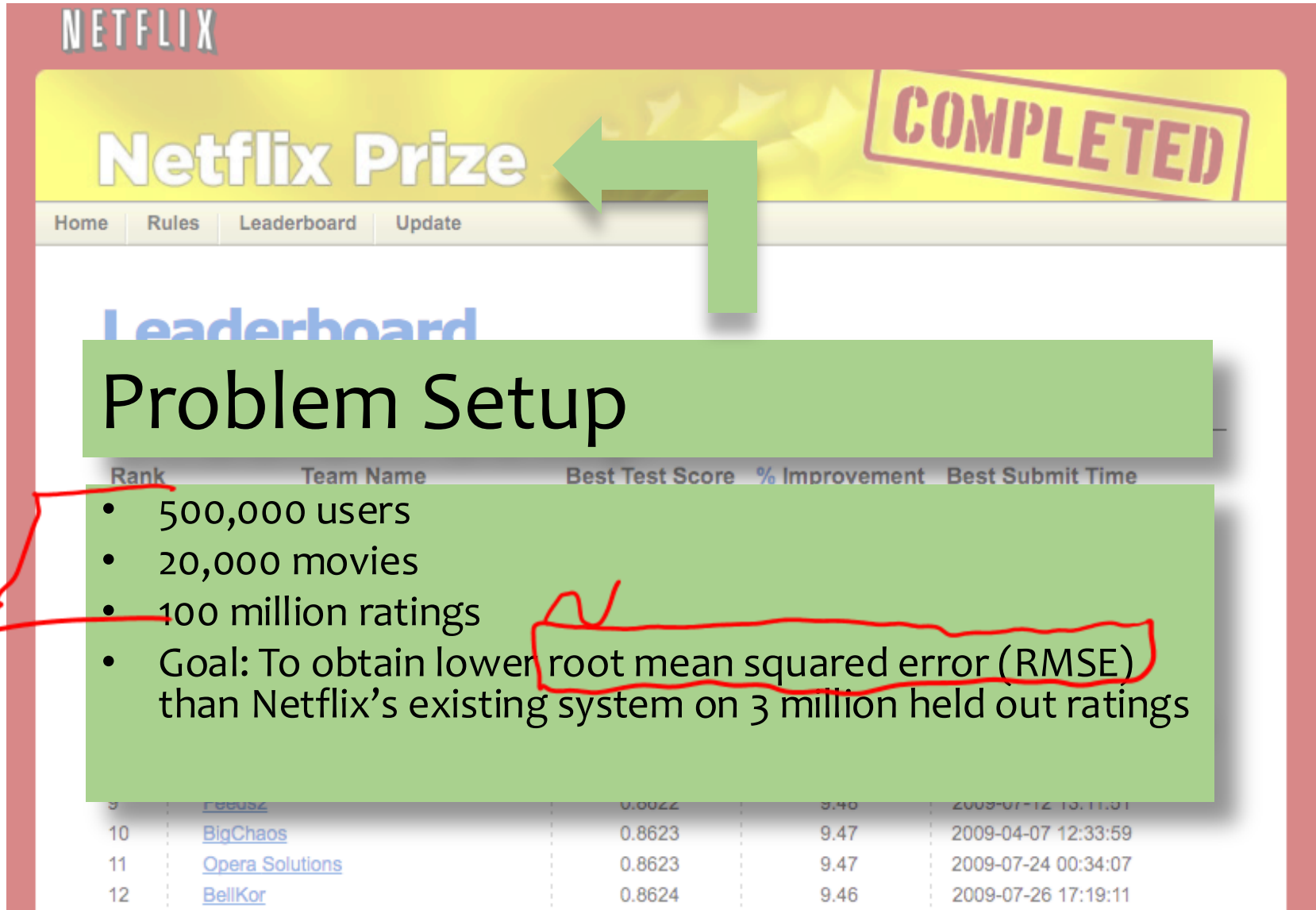
$$\{(i, j, r)\}_{i=1}^N$$



Performance measure

$$l(r, \hat{r}) = (r - \hat{r})^2$$

Recommender Systems



The image is a screenshot of the Netflix Prize website. At the top, the Netflix logo is visible. Below it, a yellow banner reads "Netflix Prize" with a green arrow pointing to it from the left. To the right of the banner is a red stamp that says "COMPLETED". Below the banner is a navigation bar with links: "Home", "Rules", "Leaderboard", and "Update". The "Leaderboard" link is highlighted. Below the navigation bar, the word "Leaderboard" is written in large blue letters. A green box with the text "Problem Setup" is overlaid on the page. Below this box is a table with the following columns: "Rank", "Team Name", "Best Test Score", "% Improvement", and "Best Submit Time". The table lists several teams, including "Feus2", "BigChaos", "Opera Solutions", and "BellKor". A red line is drawn around the "100 million ratings" bullet point in the list. Another red line is drawn around the "Goal: To obtain lower root mean squared error (RMSE) than Netflix's existing system on 3 million held out ratings" text.

NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update

Leaderboard

Problem Setup

- 500,000 users
- 20,000 movies
- 100 million ratings
- Goal: To obtain lower root mean squared error (RMSE) than Netflix's existing system on 3 million held out ratings

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
9	Feus2	0.8622	9.48	2009-07-12 15:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11

ML System Design: Movie Recommendation

Model



$$\hat{r}_{ij} \leftarrow h(i, j) = \vec{u}_i^T \vec{v}_j$$

$$J(U, V) = \sum_{ij, r \in S} \left(\|r_{ij} - \vec{u}_i^T \vec{v}_j\|_2^2 \right)$$

ML System Design: Movie Recommendation

Model

$$\|R - UV^T\|_F^2$$

Recommender Systems

Setup:

- Items:
movies, songs, products, etc.
(often many thousands)
- Users:
watchers, listeners, purchasers, etc.
(often many millions)
- Feedback:
5-star ratings, not-clicking 'next', purchases,
etc.

Challenge:

- Users only rate a small number of items
(the user/item rating data is sparse)

R

1
2
3

	1 Doctor Strange	2 Star Trek: Beyond	3 Zootopia
Alita	1	?	5
BB-8	3	4	?
C-3P0	3	5	2

$$R_{2,1} = 3$$

Different Approaches

Item-based (*Content filtering*)

- Features about each item
- Given an item, other “close” items have similar values
- e.g. Pandora.com, music genome project

Different Approaches

Item-based (*Content filtering*)

- Features about each item
- Given an item, other “close” items have similar values
- e.g. Pandora.com, music genome project

User-based

- Features about each user
- Given a user, other “close” users have similar preferences
- *Market segmentation*

Learning user-item relationship

- Can be done without features on either user or item
- Collaborative filtering techniques 

Collaborative Filtering

Collaborative Filtering

Everyday Examples of Collaborative Filtering...

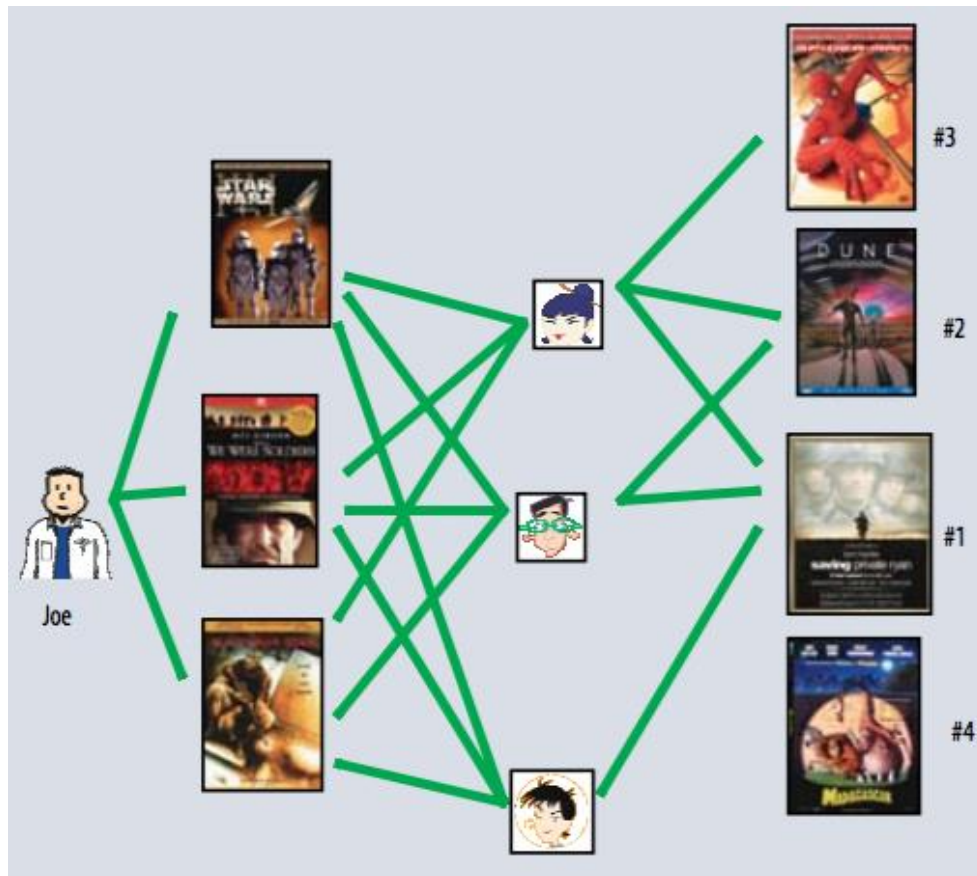
- Bestseller lists
- Top 40 music lists
- The “recent returns” shelf at the library
- Unmarked but well-used paths thru the woods
- The printer room at work
- “Read any good books lately?”
- ...

Common insight: personal tastes are correlated

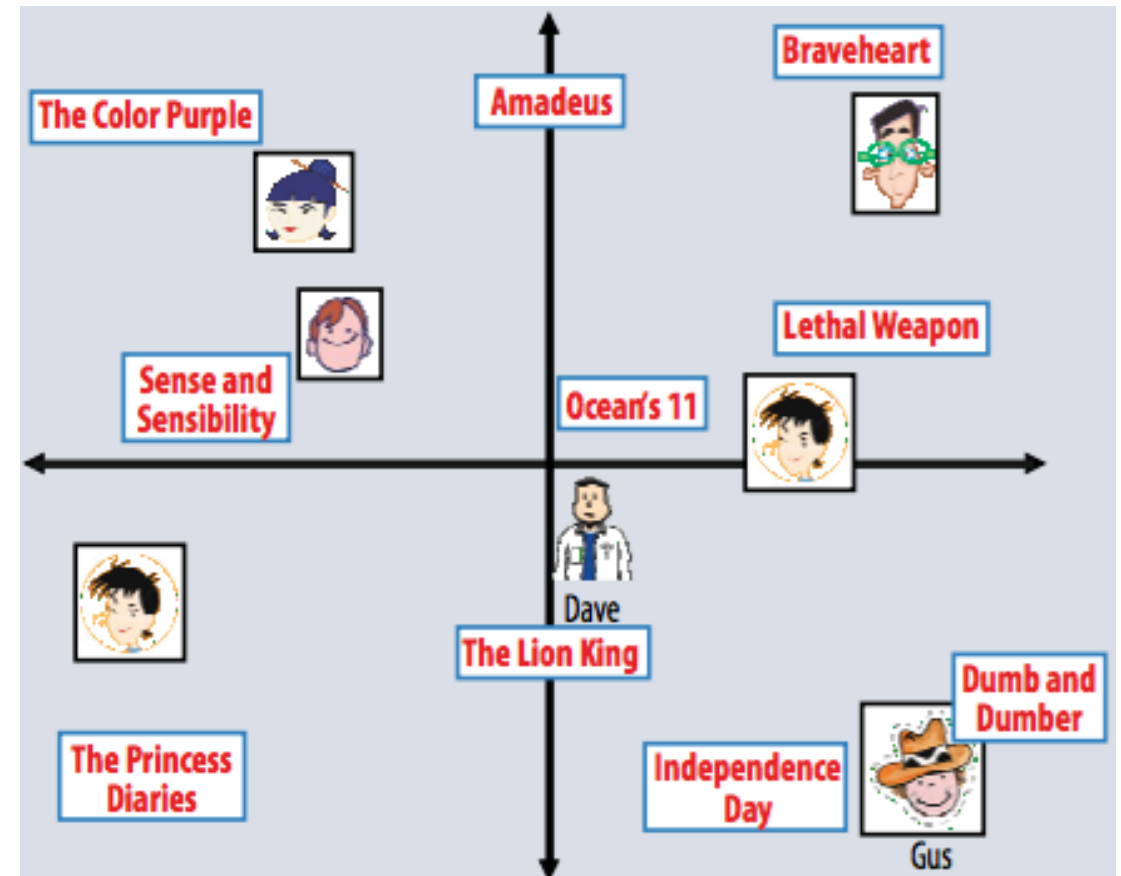
- If Alita and BB-8 both like X and Alita likes Y then BB-8 is more likely to like Y
- especially (perhaps) if BB-8 knows Alita

Two Types of Collaborative Filtering

1. Neighborhood Methods

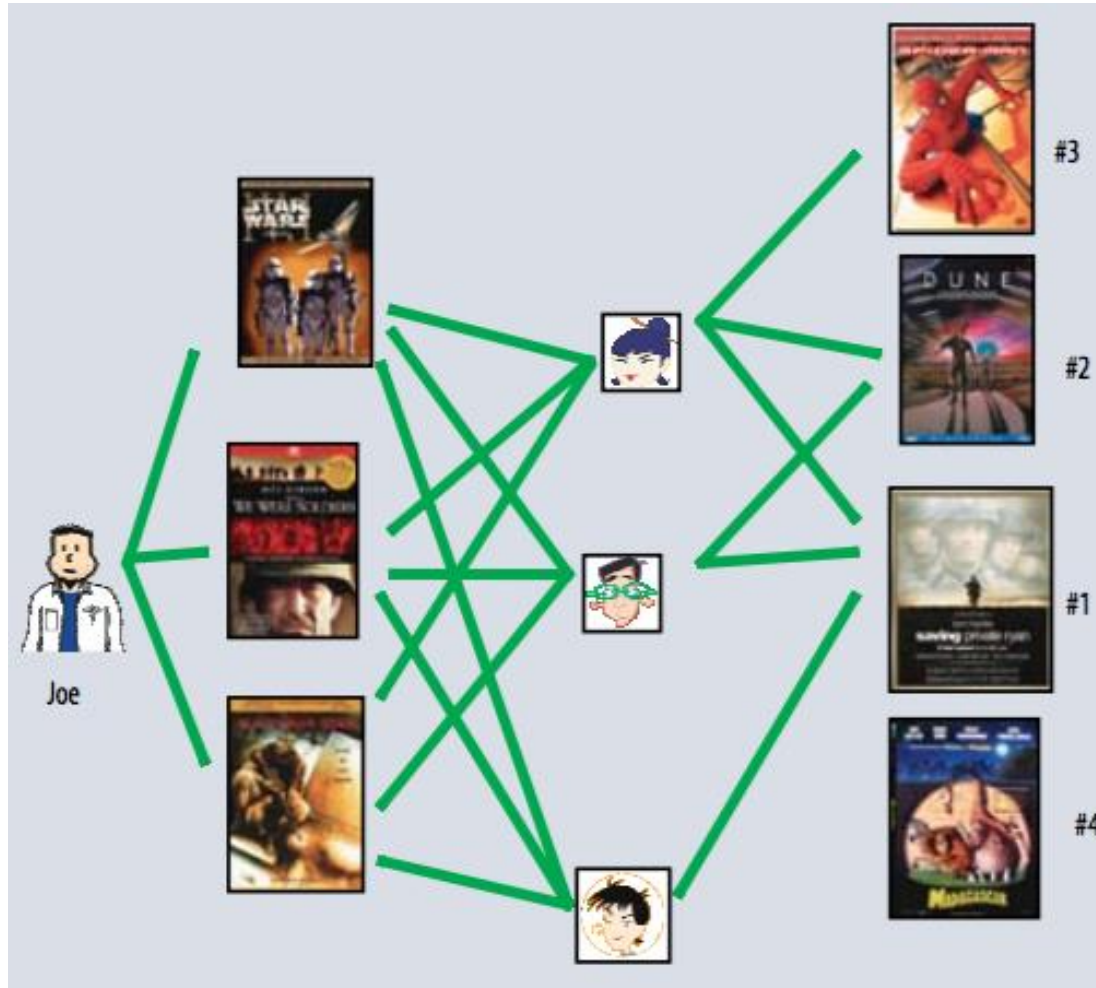


2. Latent Factor Methods



Two Types of Collaborative Filtering

1. Neighborhood Methods



In the figure, assume that a green line indicates the movie was **watched**

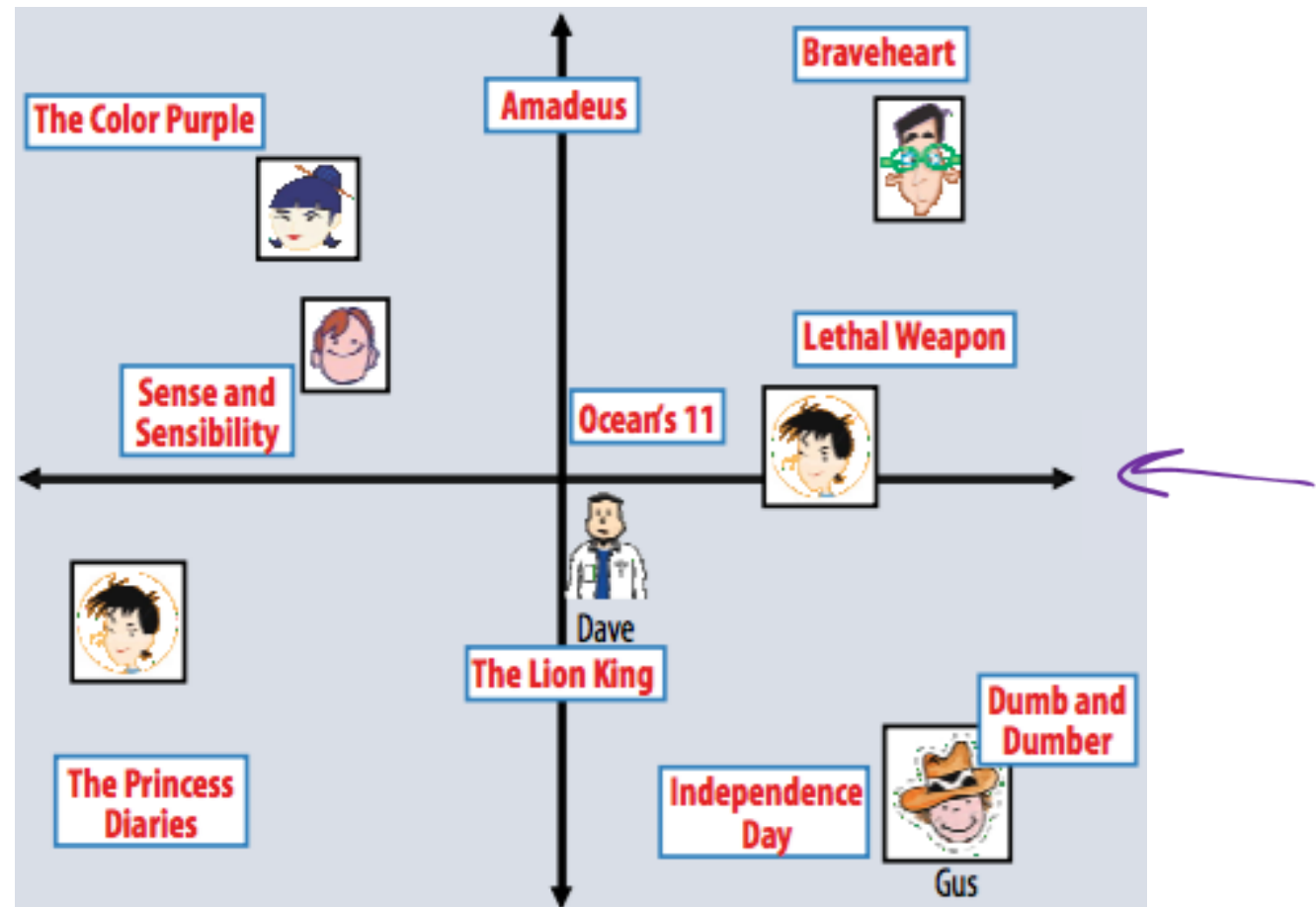
Algorithm:

1. **Find neighbors** based on similarity of movie preferences
2. **Recommend** movies that those neighbors watched

Two Types of Collaborative Filtering

2. Latent Factor Methods

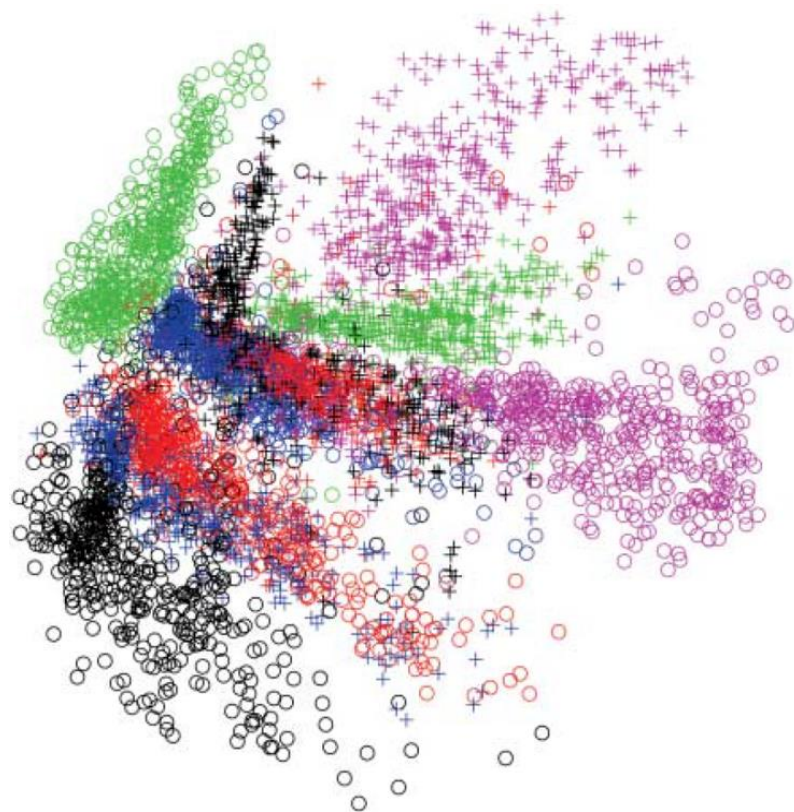
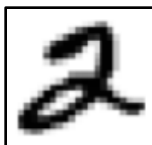
- Assume that both movies and users can be mapped to the same **feature space**
- **Recommend** a movie based on its **proximity** to the user in the feature (latent) space
- **Example algorithm:**
Latent factor method



Collaborative Filtering:
Latent Factor (Matrix Factorization) Method

Background: Learn a Feature Space

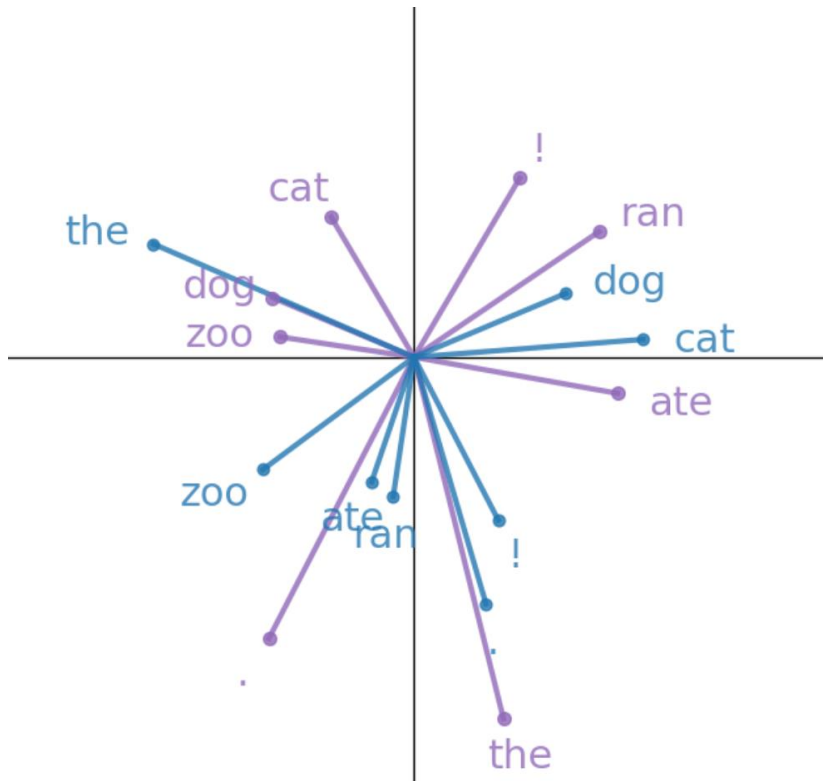
Autoencoders



Background: Learn a Feature Space

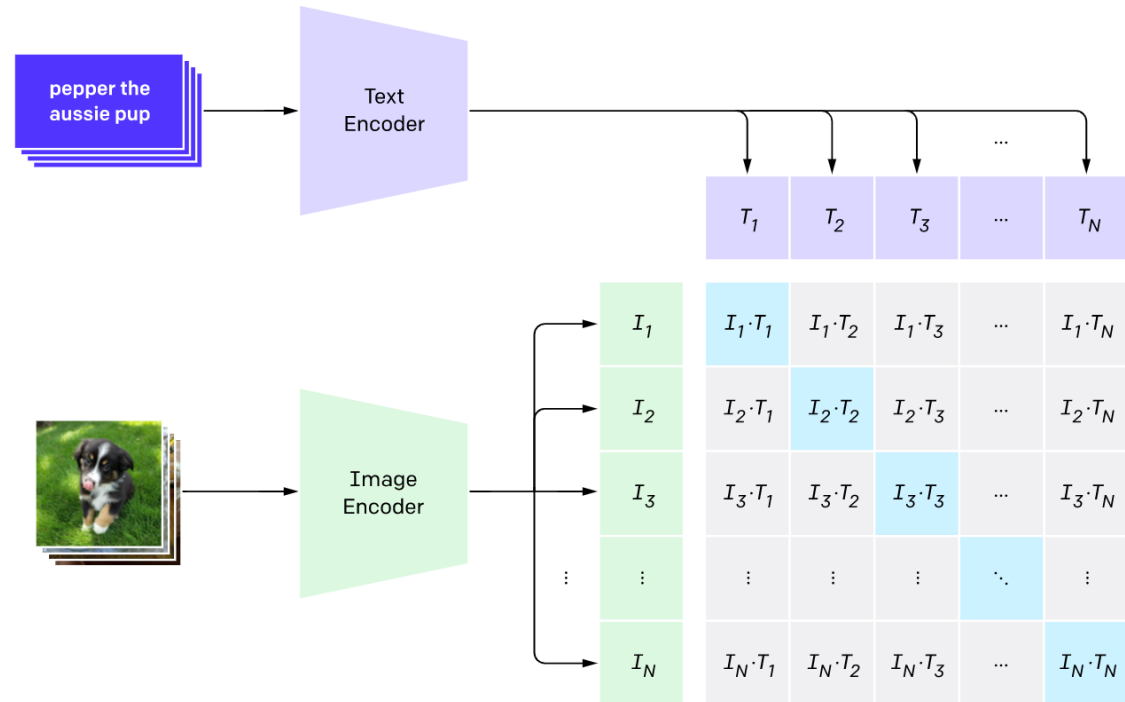
Word2vec:

Feature space for words



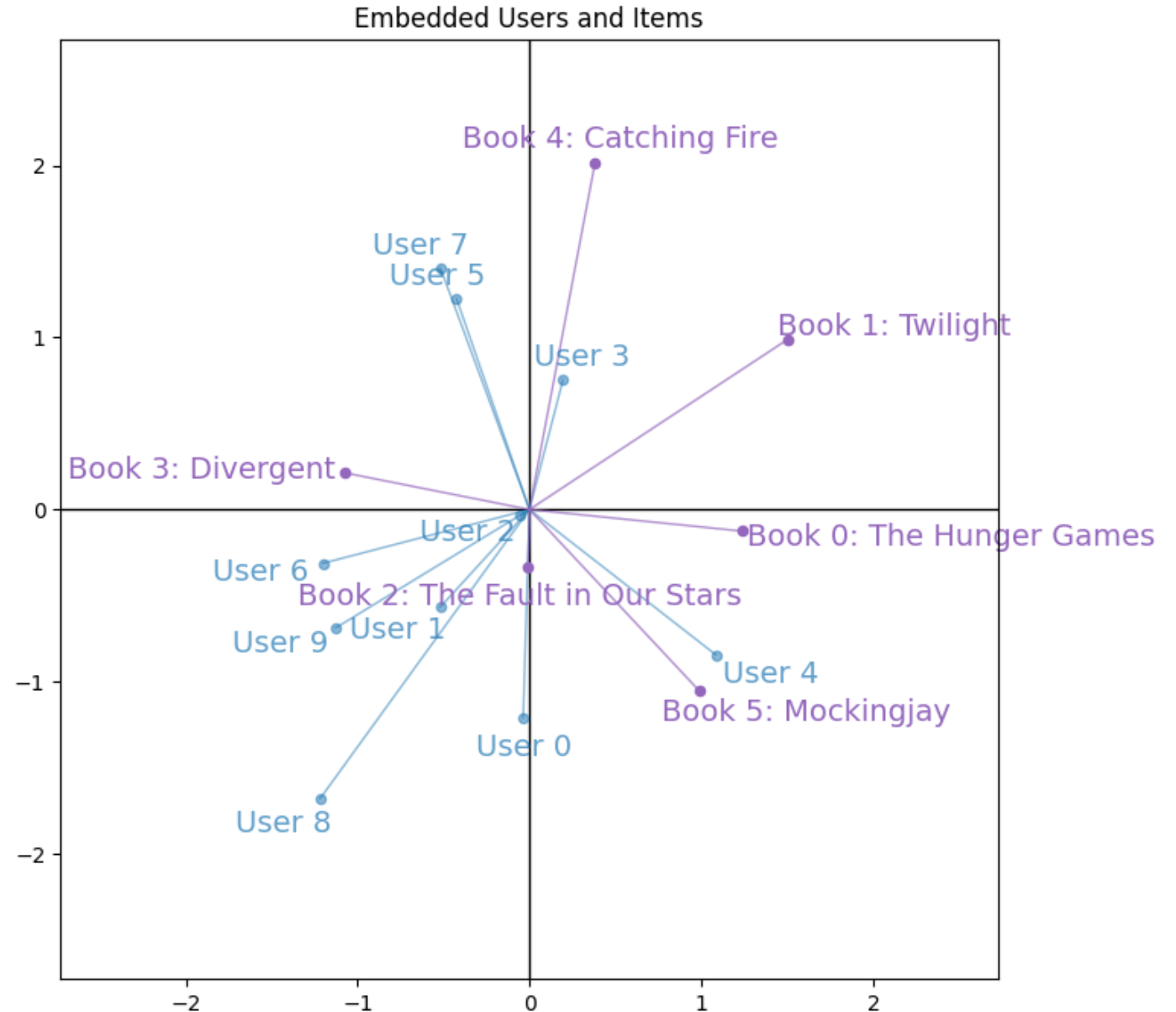
CLIP:

Common feature space for text and images



Recommender System: Latent Factor Model

Learning to map items and users to the same lower dimensional space



Recommender System: Latent Factor Model

Optimization: Objective function using only the labels we have

$$J(U, V) = \sum_{i, j, r \in \mathcal{D}} \left(r - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right)^2$$

Notation alert!

- Let $\mathbf{u}^{(i)}$ be the transpose of the i -th row of U
- Let $\mathbf{v}^{(j)}$ be the transpose of the j -th row of V

Latent Factor (Matrix Factorization) Method

SGD Optimization

Recommender System: Matrix Factorization

Optimization: Objective function using only the labels we have

$$J(U, V) = \sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right)^2$$

SGD objective function: Just one tuple in $\mathcal{D}$: (r_{ij}, i, j)

$$J(\mathbf{u}^{(i)}, \mathbf{v}^{(j)}) = \frac{1}{2} \left(r_{ij} - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right)^2$$

SGD gradient steps

$$\frac{\partial J}{\partial \mathbf{u}^{(j)}} = - \left(r_{ij} - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right) \mathbf{v}^{(j)}$$

$$\frac{\partial J}{\partial \mathbf{v}^{(j)}} = - \left(r_{ij} - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right) \mathbf{u}^{(i)}$$

Notation alert!

- Let $\mathbf{u}^{(i)}$ be the transpose of the i -th row of U
- Let $\mathbf{v}^{(j)}$ be the transpose of the j -th row of V

Latent Factor Method

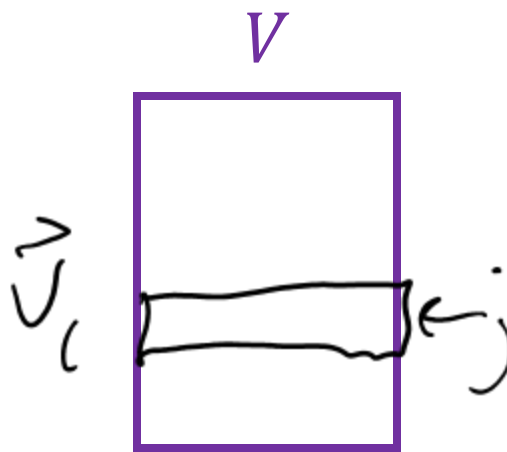
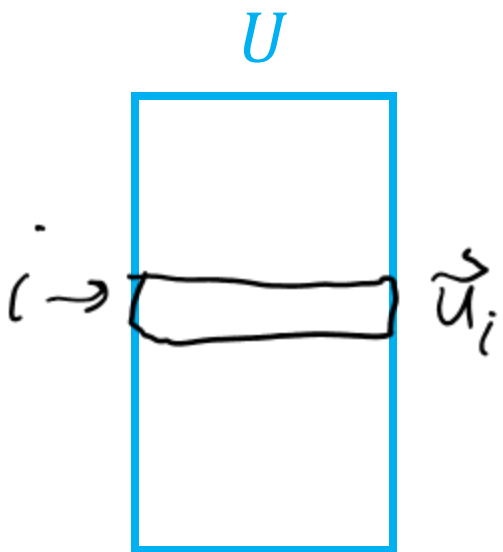
Inference

Inference for Recommender Systems

First solve:

$$\min_{U,V} J(U,V) \quad J(U,V) = \sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right)^2$$

What then?



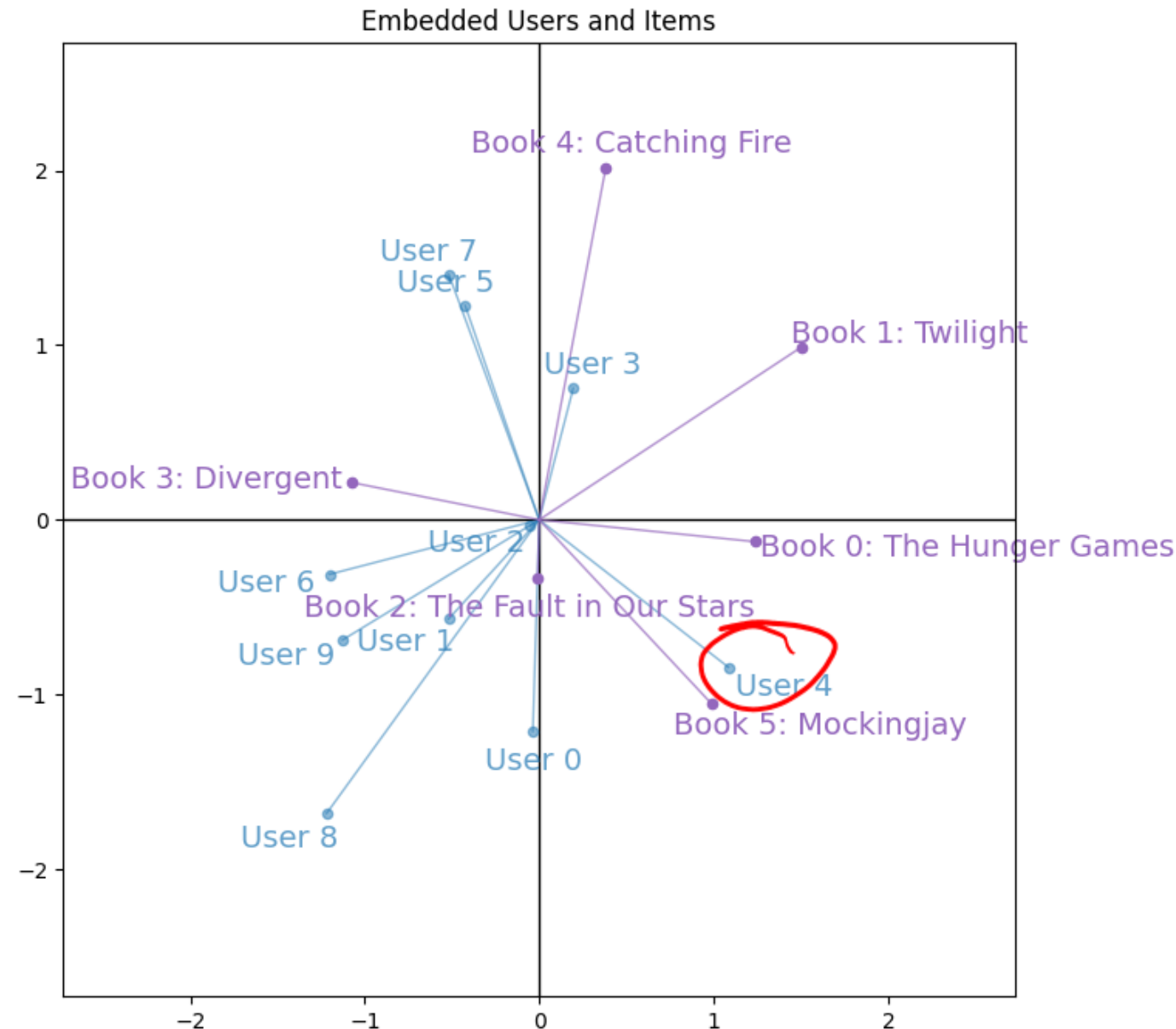
↓

	Doctor Strange	Star Trek: Beyond	Zootopia
Alita	1		5
BB-8	3	4	
C-3Po	3	5	2



Inference for Recommender Systems

Recommendations



Latent Factor Method

Practical Considerations

Latent Factor Regularization

Add regularization to avoid overfitting

$$\min_{U,V} J(U,V)$$

$$J(U,V) = \sum_{i,j \in \mathcal{S}} \left(r_{ij} - \mathbf{u}^{(i)T} \mathbf{v}^{(j)} \right)^2 + \lambda \|\mathbf{u}_i\|_2^2 + \lambda \|\mathbf{v}_j\|_2^2$$

Bias in Recommender Systems

What high-level problems can occur from recommender systems?

Summary

Recommender systems solve many **real-world** (*large-scale) **problems**

Collaborative filtering using a latent factor model can be done without any features about the users or items other than their past ratings

Solving for latent factors is just another example of a **common recipe**:

1. define a model
2. define an objective function
3. optimize

Optimization

- Use SGD
- Add regularization to avoid overfitting