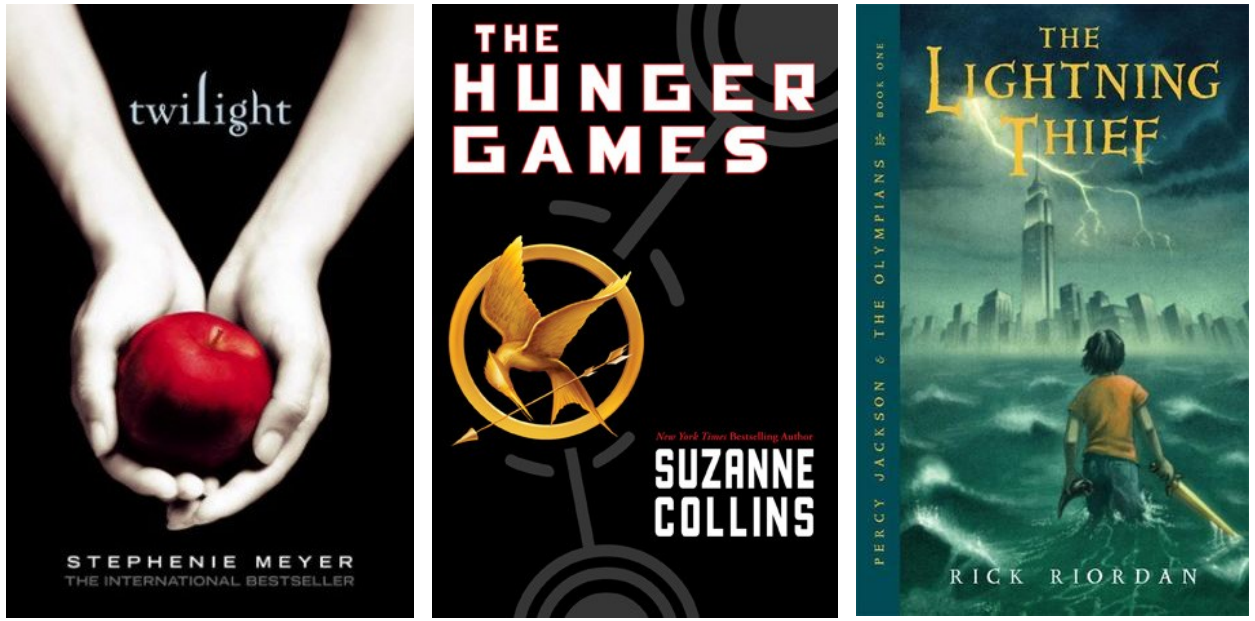
07-380 AI & ML II

Recommender Systems



★ ★ ★ ★ ☆ ★ ★ ★ ★ ★ ? ? ? ? ?

Instructors: Pat Virtue & Mohammad Salameh

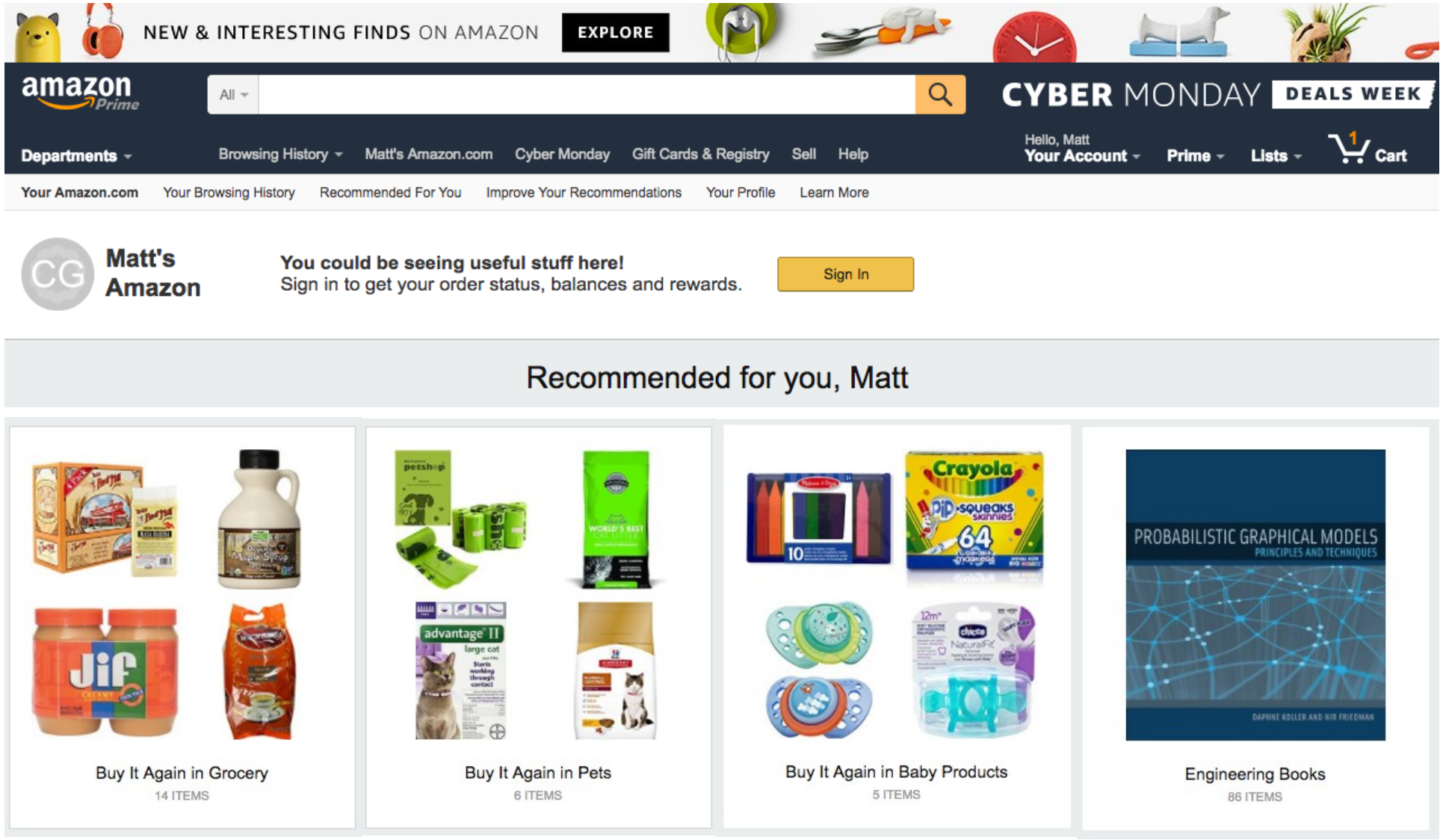
Recommender Systems

A Common Challenge:

- Assume you're a company selling items of some sort:
  movies, songs, products, etc.
- Company collects millions of ratings from users of their items
- To maximize profit / user happiness, you want to
  recommend items that users are likely to want

Slide credit: CMU MLD Matt Gormley

Recommender Systems



Slide credit: CMU MLD Matt Gormley

Poll 1

A user creates a new Spotify account and plays *Try Everything* by Shakira. What technique could we use to recommend the next song? (Note any assumptions you make with your choice.)

A. Item-based (content filtering)

B. User-based

C. Collaborative filtering



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

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)

	Minions	K-Pop Demon Hunters	Zootopia
Roz	1		5
BB-8	3	4	
C-3P0	3	5	2

Slide credit: CMU MLD Matt Gormley

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
- “Read any good books lately?”

Common insight: personal tastes are correlated

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

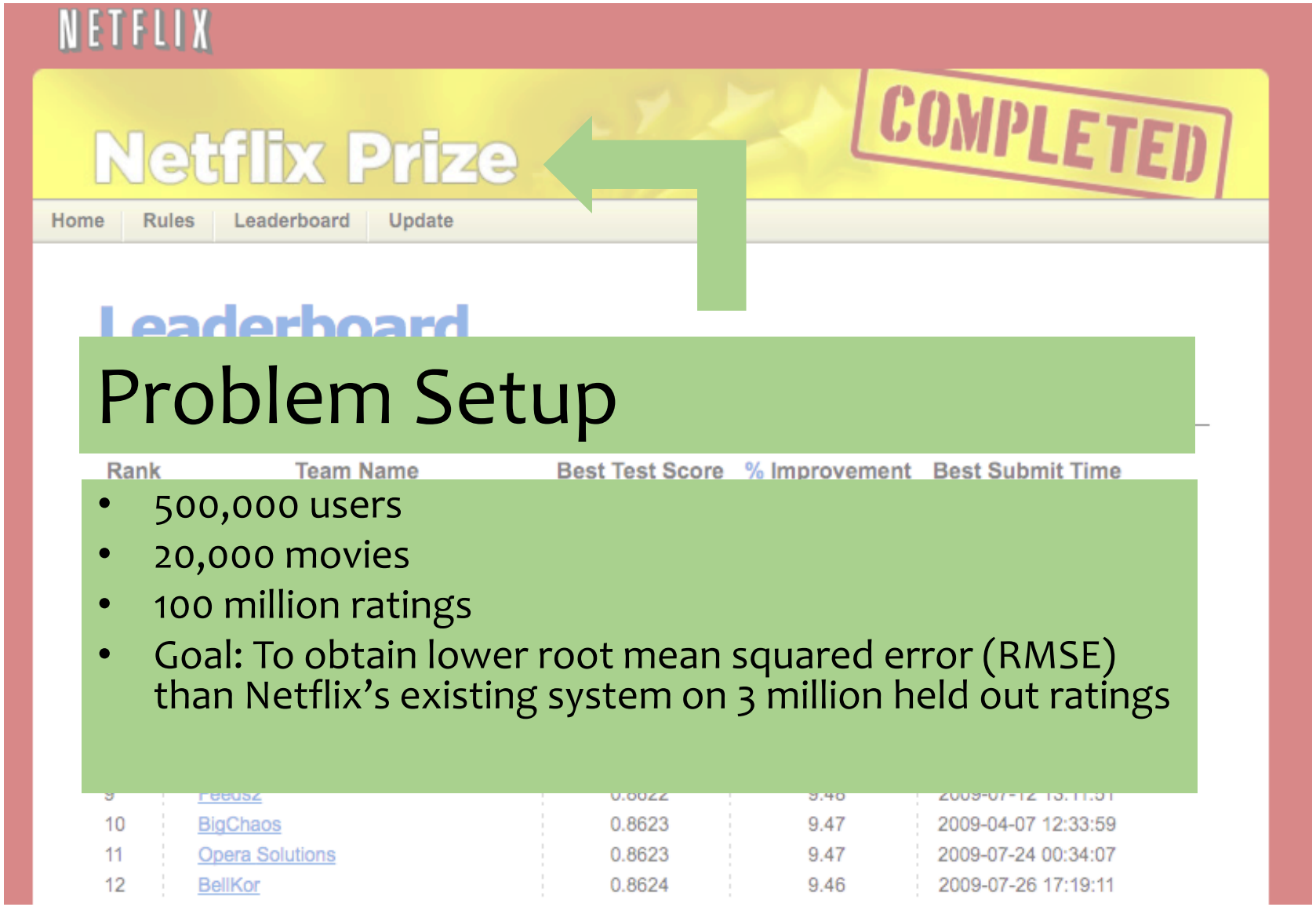
Slide from William Cohen

Recommender Systems



Slide credit: CMU MLD Matt Gormley

Recommender Systems



Slide credit: CMU MLD Matt Gormley

ML System Design: Movie Recommendation

Task

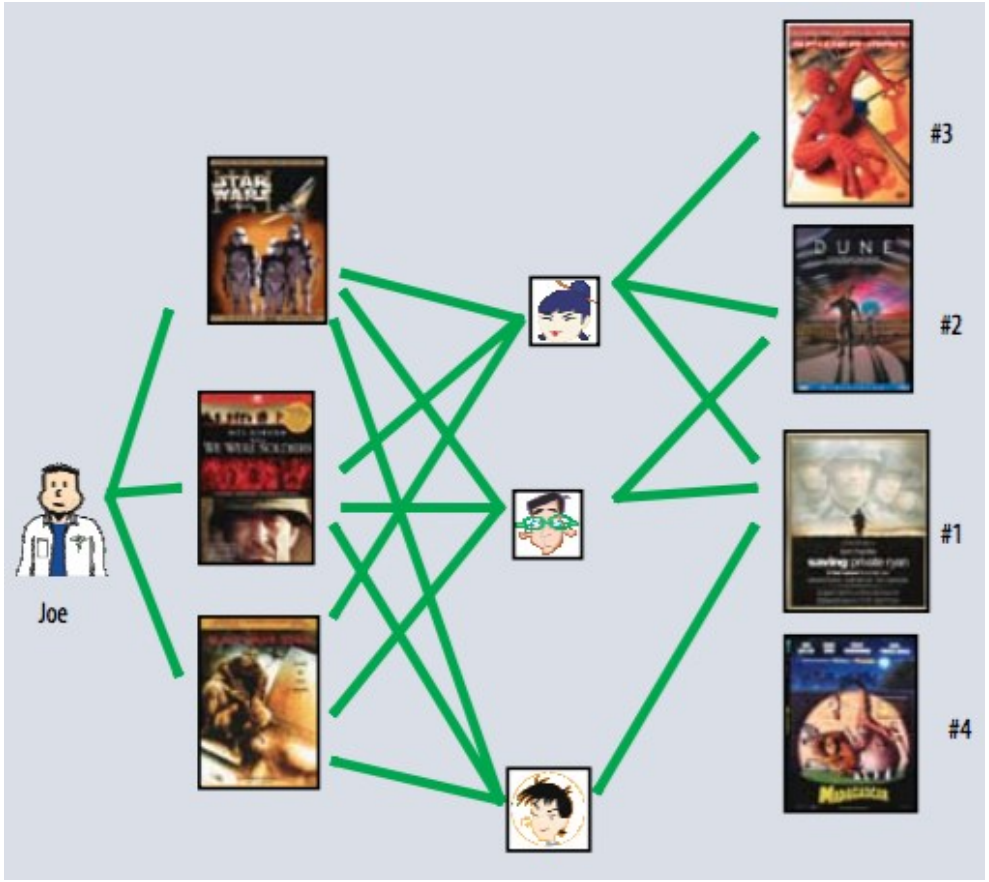
- 500,000 users
- 20,000 movies
- 100 million ratings
- Goal: Lower (RMSE) than Netflix

Experience

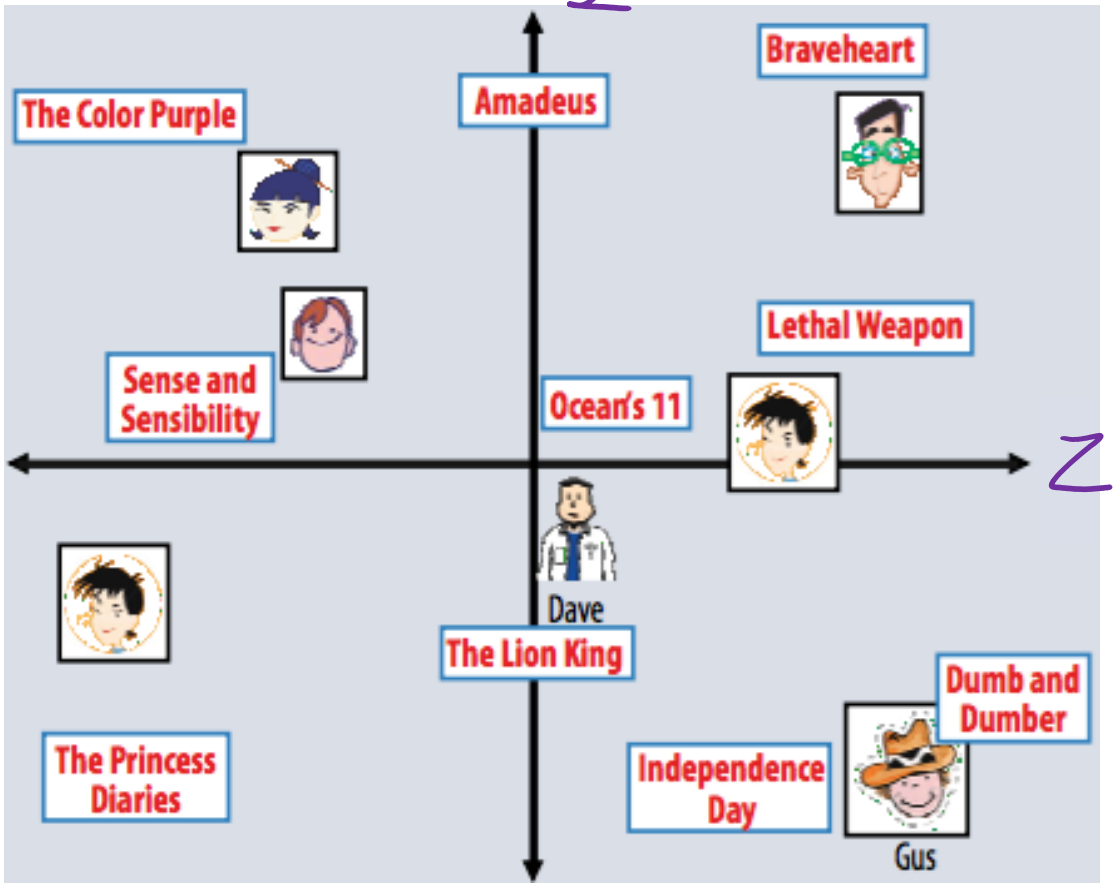
Performance measure

Two Types of Collaborative Filtering

1. Neighborhood Methods



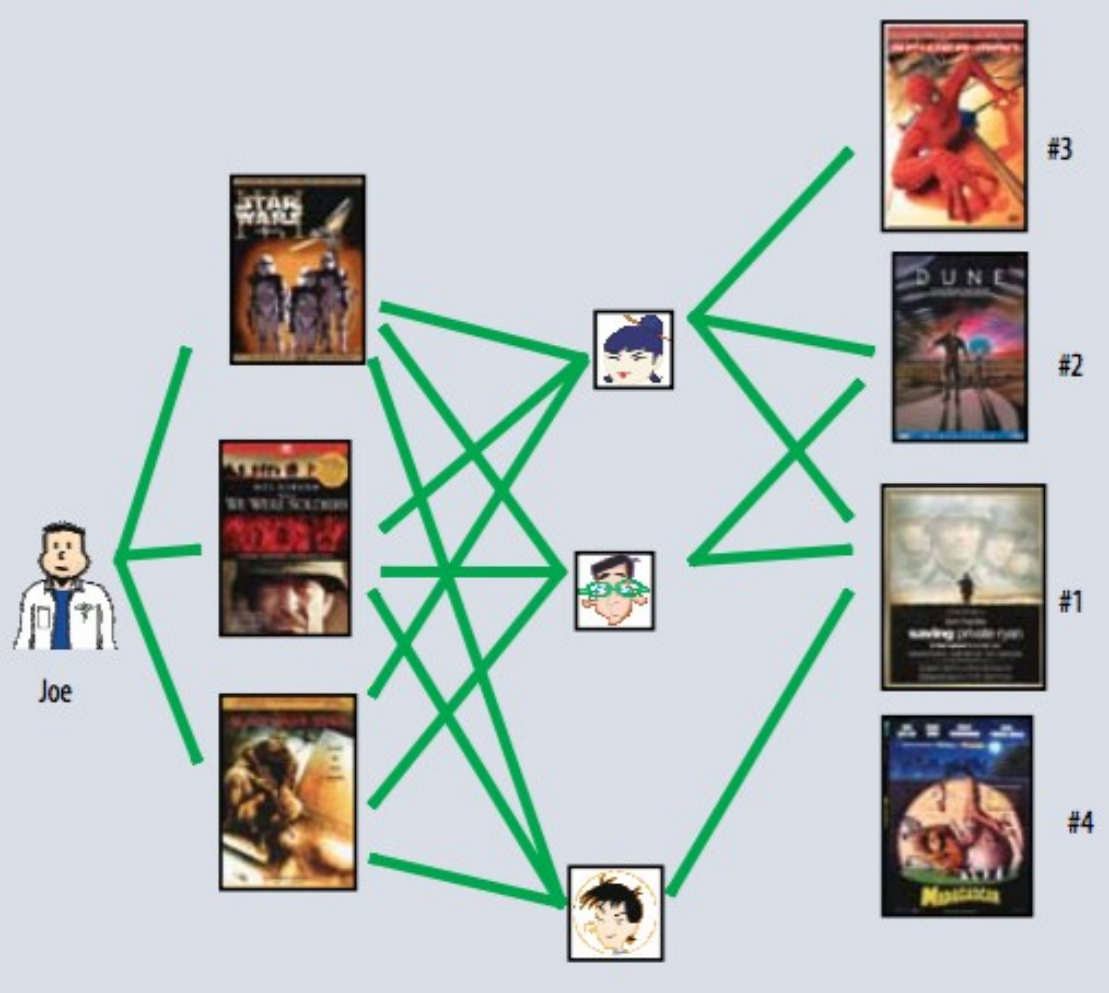
2. Latent Factor Methods



Figures from Koren et al. (2009)

Two Types of Collaborative Filtering

1. Neighborhood Methods



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

Algorithm:

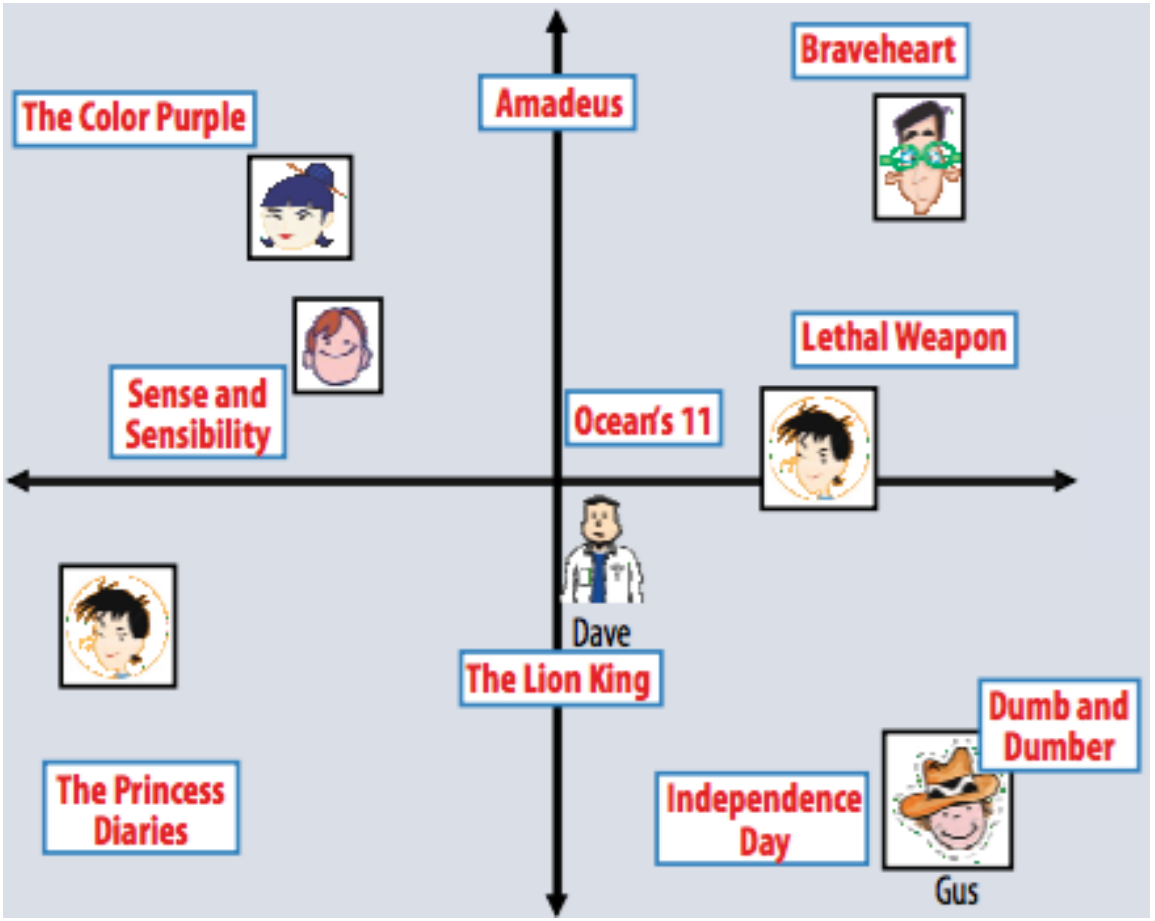
1. **Find neighbors** based on similarity of movie preferences
2. **Recommend** movies that those neighbors watched

Figures from Koren et al. (2009)

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

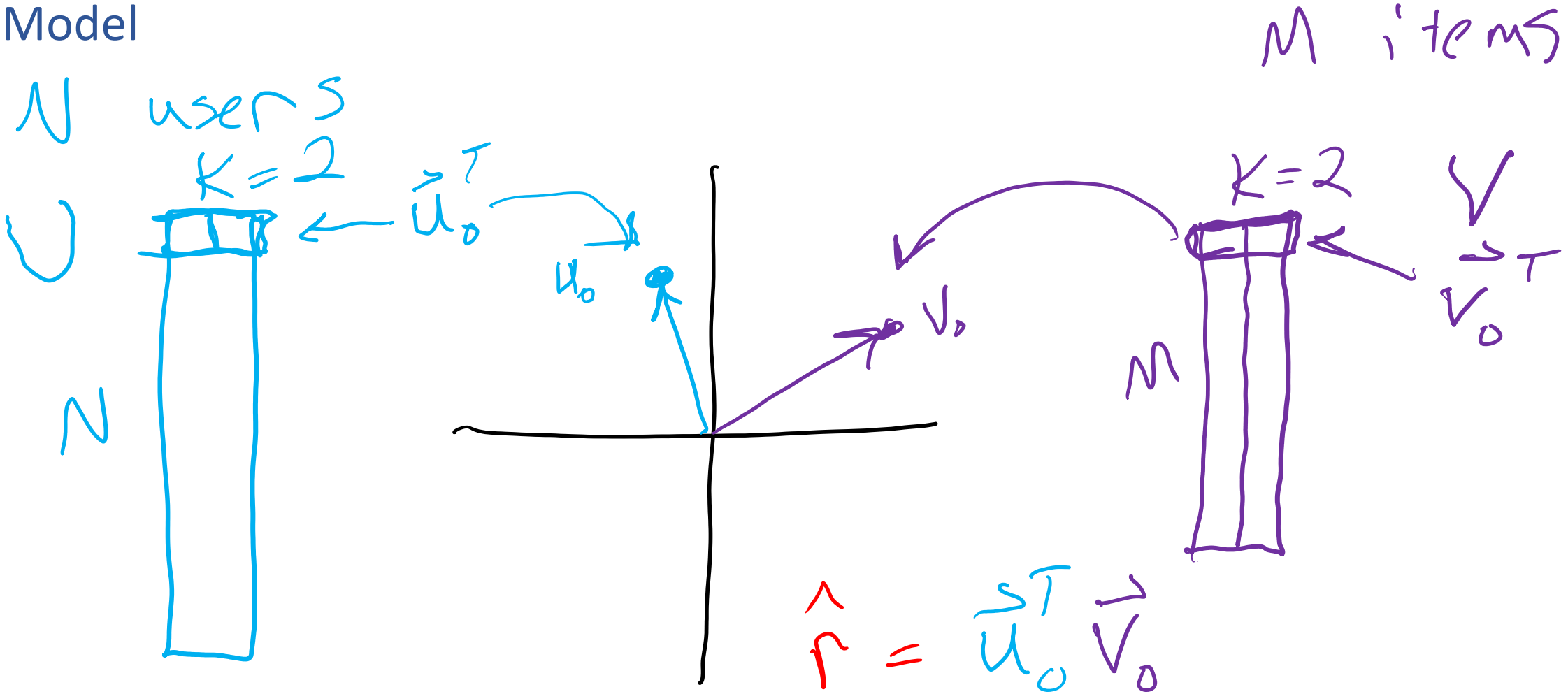


Figures from Koren et al. (2009)

Collaborative Filtering as Embedded Features

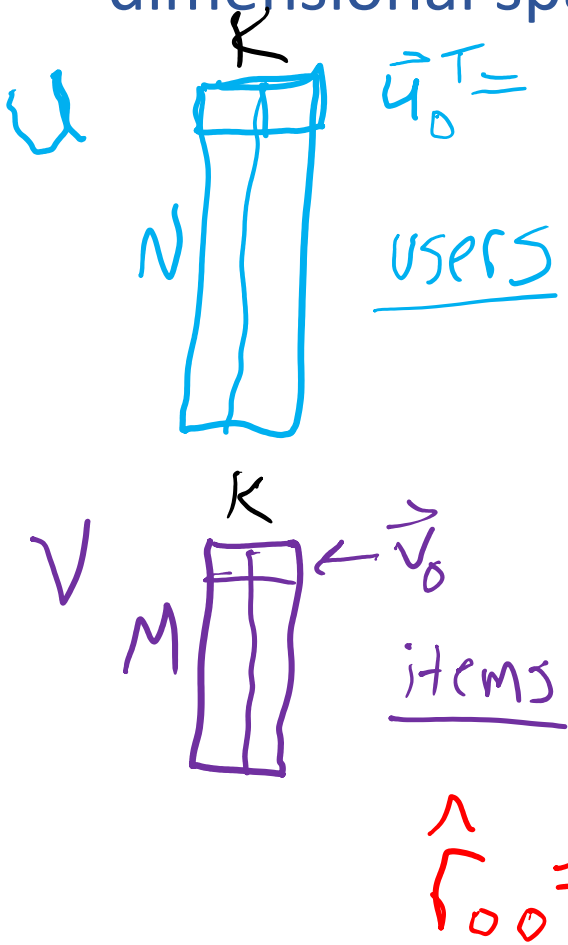
ML System Design: Movie Recommendation

Model



Recommender System: Latent Factor Model

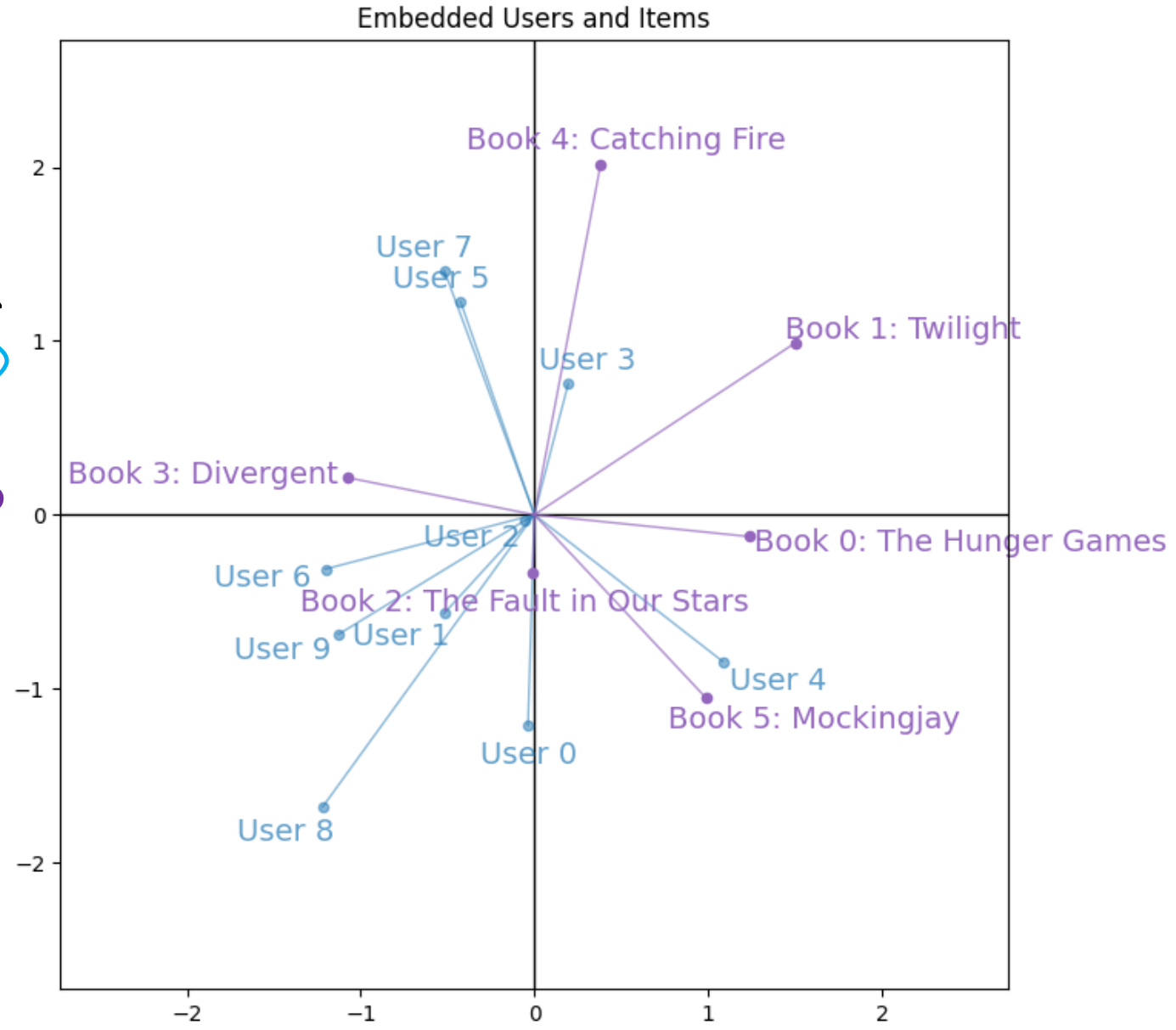
Learning to map items and users to the same lower dimensional space



$K = 2$

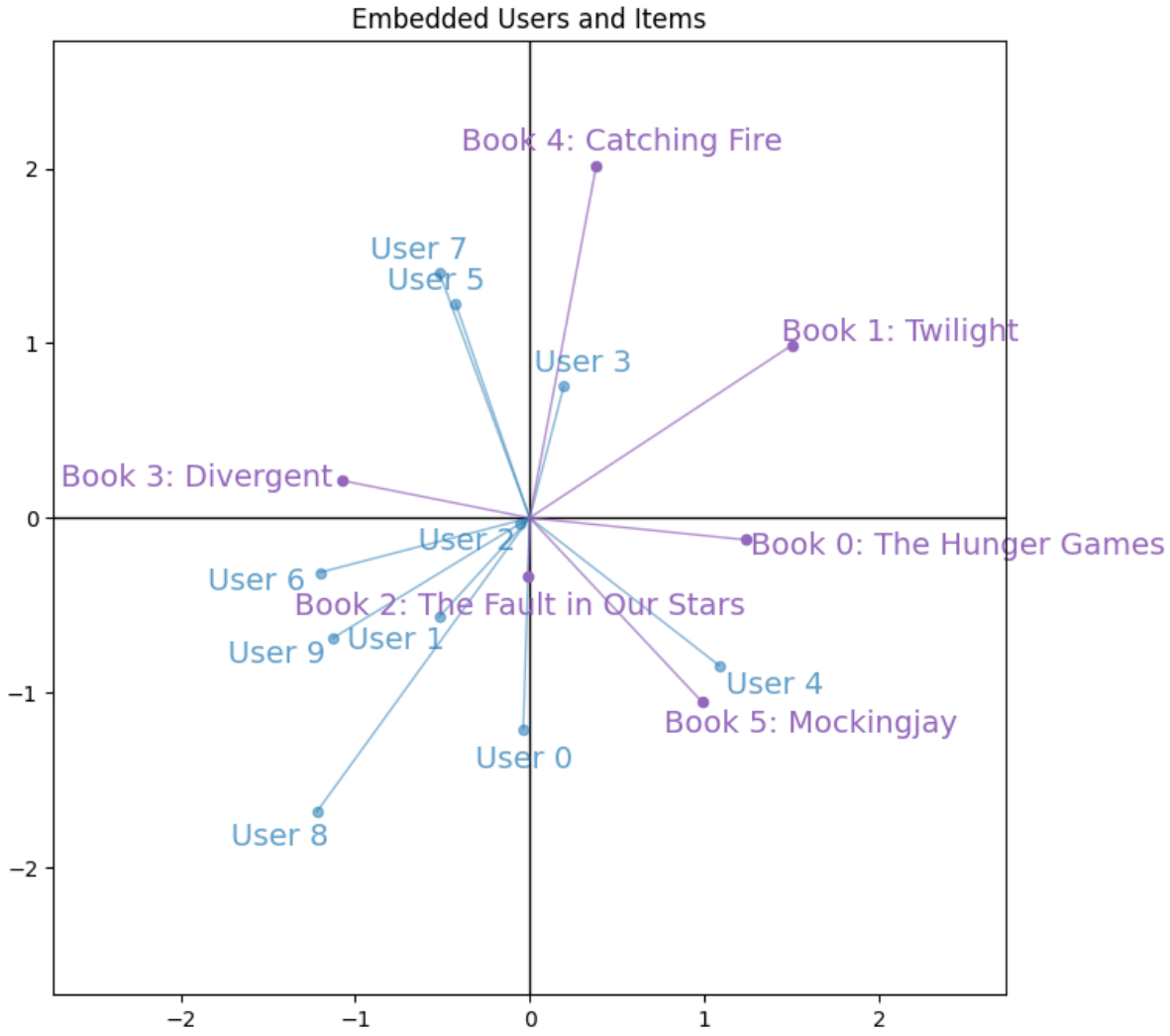
$N = 10$

$M = 6$



Poll 2

Where else have we seen learned feature spaces?



Background: Learn a Feature Space

Autoencoders

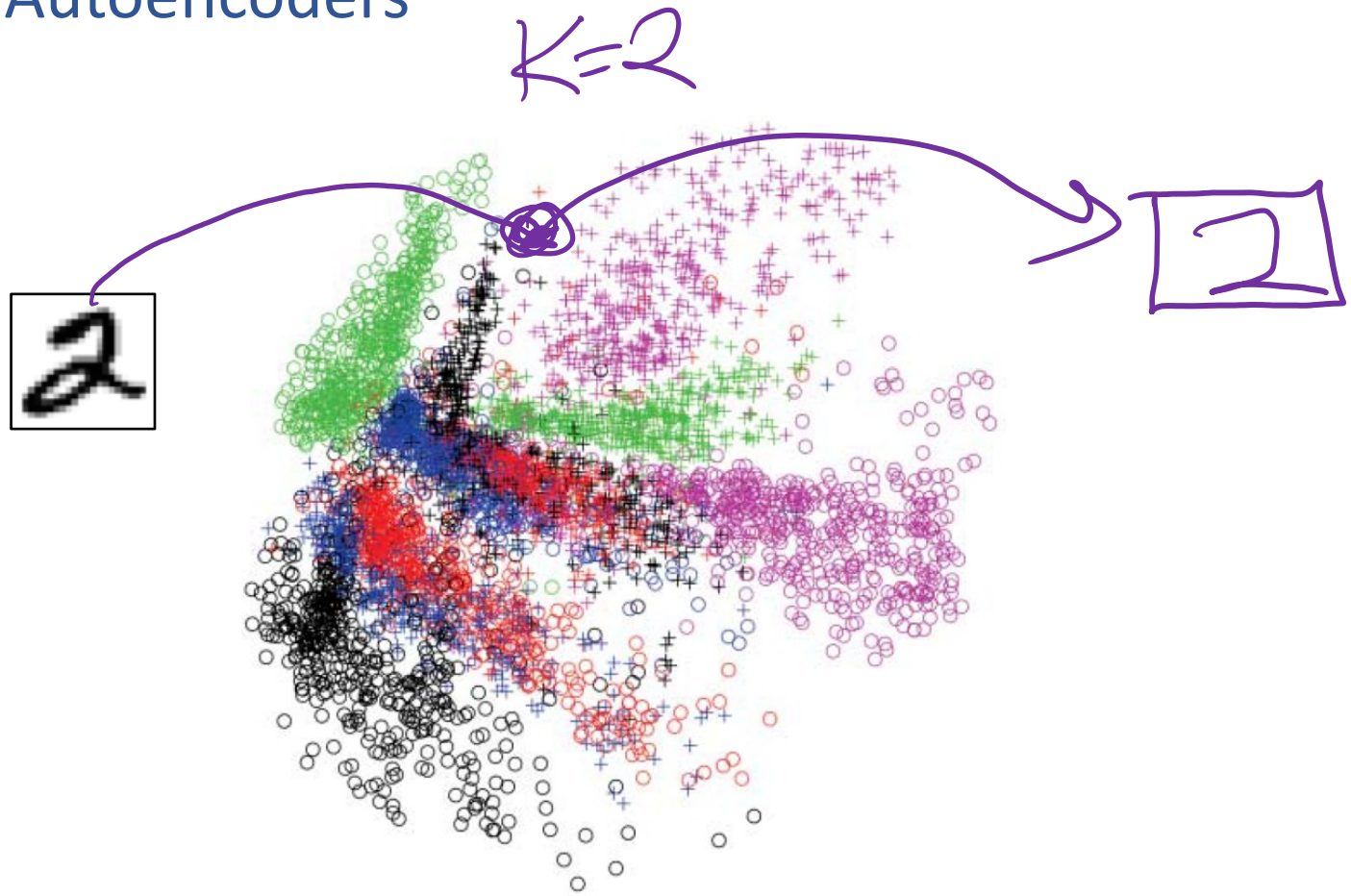
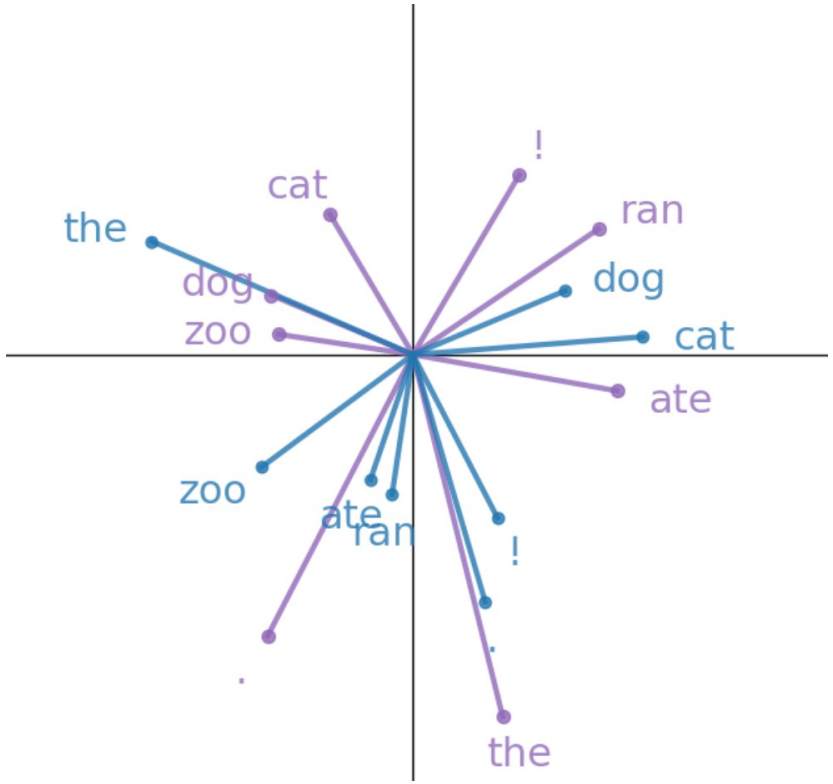


Image: Hinton & Salakhutdinov. *Science* 313.5786 (2006): 504-507.

Background: Learn a Feature Space

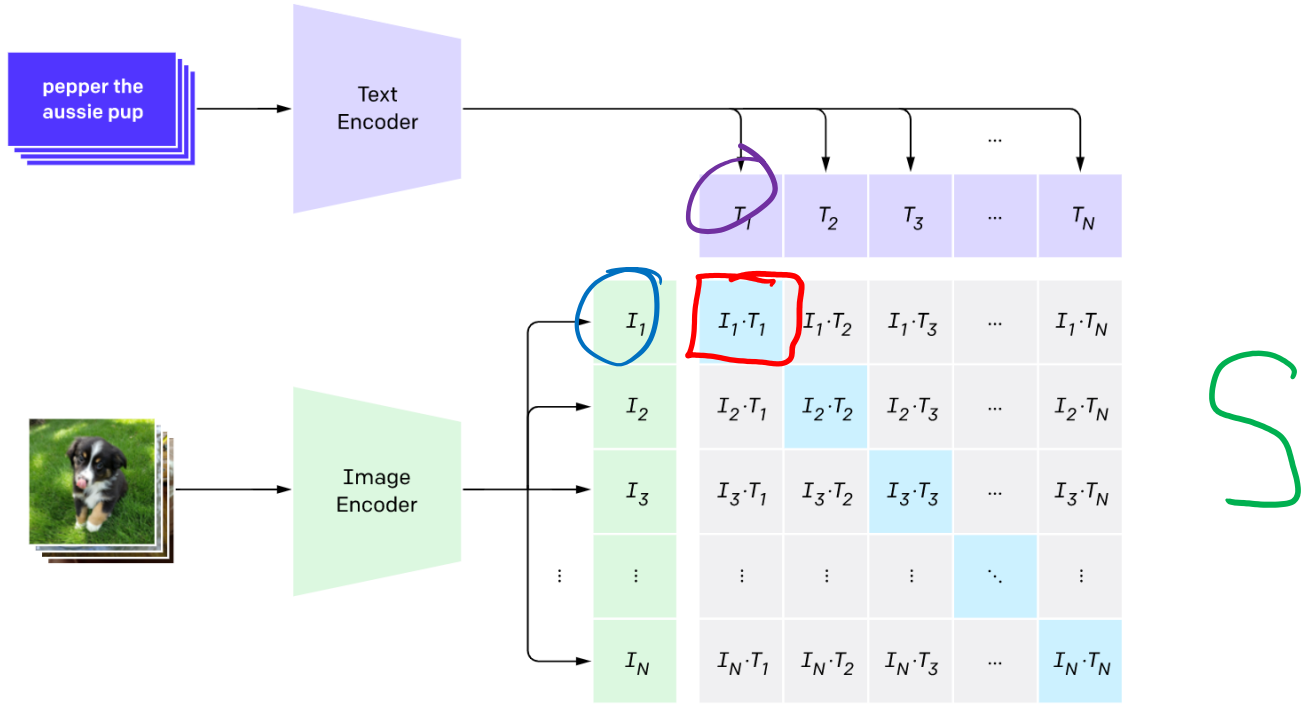
Word2vec:

Feature space for words



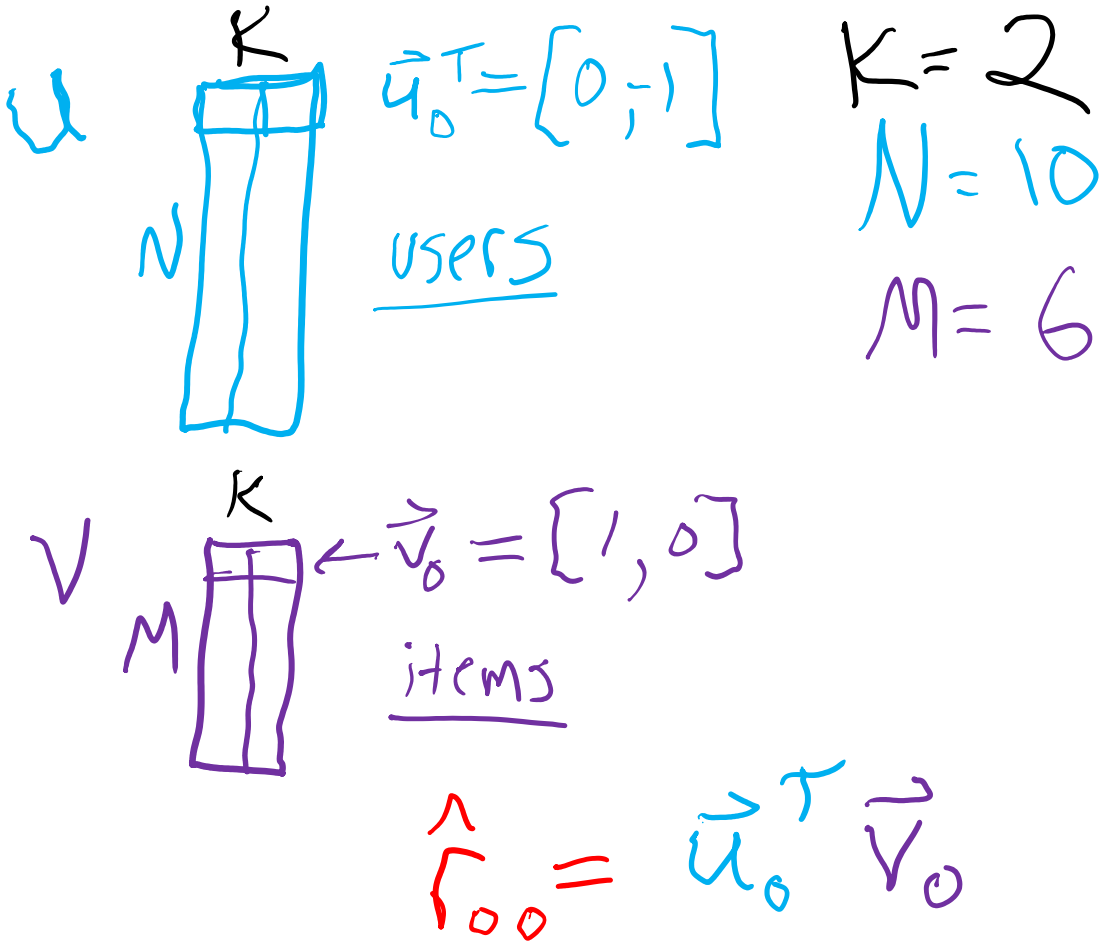
CLIP:

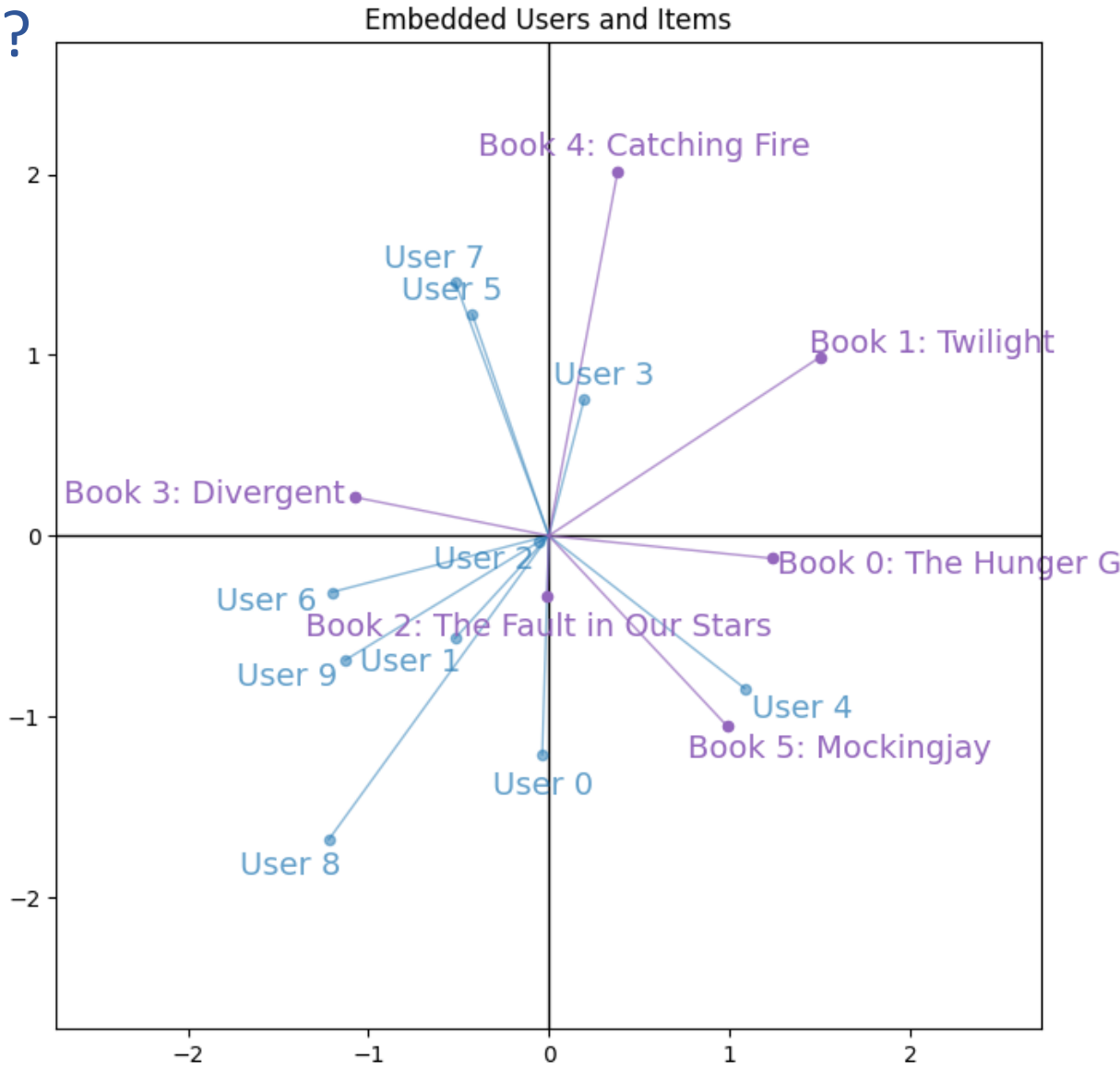
Common feature space for text and images



Poll 3

How many parameters are in this model?





Recommender System: Latent Factor Model

Optimization: Objective function using only the labels we have

$$J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}}\left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

Notation alert!

- Let $\mathbf{u}_i$ be the transpose of the i-the row of U
- Let $\mathbf{v}_j$ be the transpose of the j-the row of V

Collaborative Filtering

SGD Optimization

Poll 4

Is this objective function quadratic?

$$J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

Poll 5

Is this objective function convex?

$$J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

Recommender System: Latent Factor Model

Optimization: Objective function using only the labels we have

$$J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

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

$$J^{(i,j,r)}(\mathbf{u}_i, \mathbf{v}_j) = \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

SGD gradient steps

$$\frac{\partial J^{(i,j,r)}}{\partial \mathbf{u}_i} = -2\left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)\mathbf{v}_j$$

$$\frac{\partial J^{(i,j,r)}}{\partial \mathbf{v}_j} = -2\left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)\mathbf{u}_i$$

Notation alert!

- Let $\mathbf{u}_i$ be the transpose of the i-the row of U
- Let $\mathbf{v}_j$ be the transpose of the j-the row of V

Collaborative Filtering

Inference

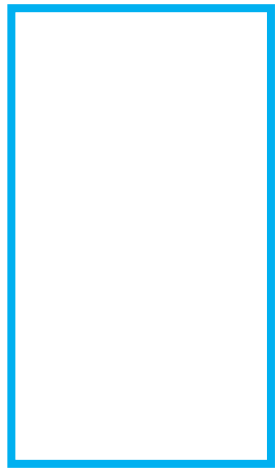
Inference for Recommender Systems

First solve:

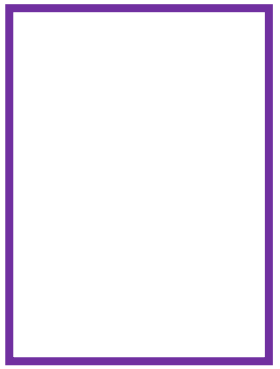
$$\min_{U,V} J(U,V) \qquad J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

What then?

U



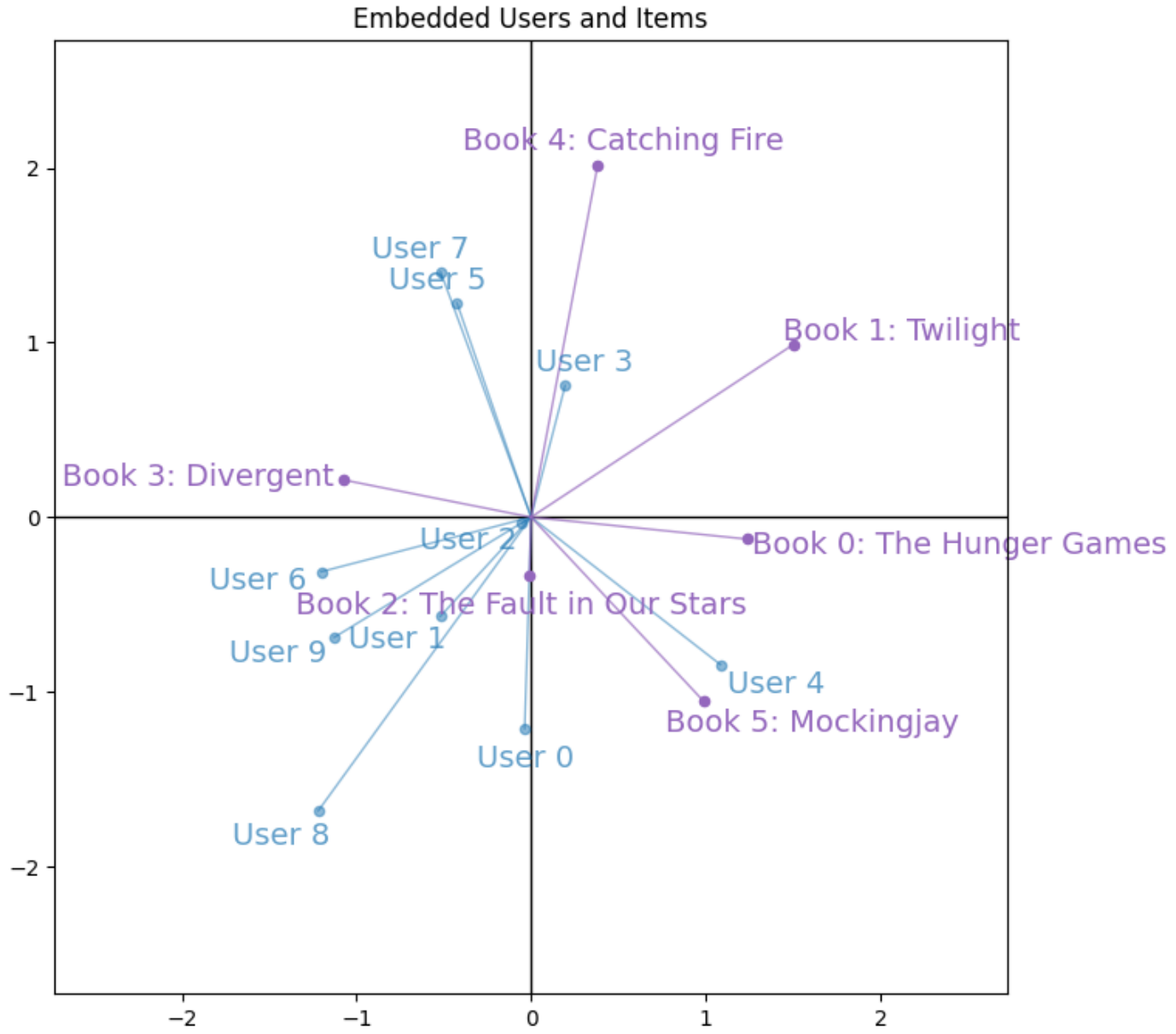
V



	Minions	K-Pop Demon Hunters	Zootopia
Roz	1		5
BB-8	3	4	
C-3P0	3	5	2

Inference for Recommender Systems

Recommendations



Collaborative Filtering

Practical Considerations

Latent Factor Regularization

Add regularization to avoid overfitting

$$J(U,V) = \frac{1}{|\mathcal{D}|}\sum_{i,j,r \in \mathcal{D}} \left(r - \mathbf{u}_i^\top \mathbf{v}_j\right)^2$$

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

Centering users and items

Some users always rate items super high

Some items always have high ratings

Idea: pre-process the data to focus on these smaller ranges of ratings

- For each user, subtract the mean of that users ratings
- For each item, subtract the mean of that items ratings

Note: Don't forget to store these means to add them back when we want absolution ratings

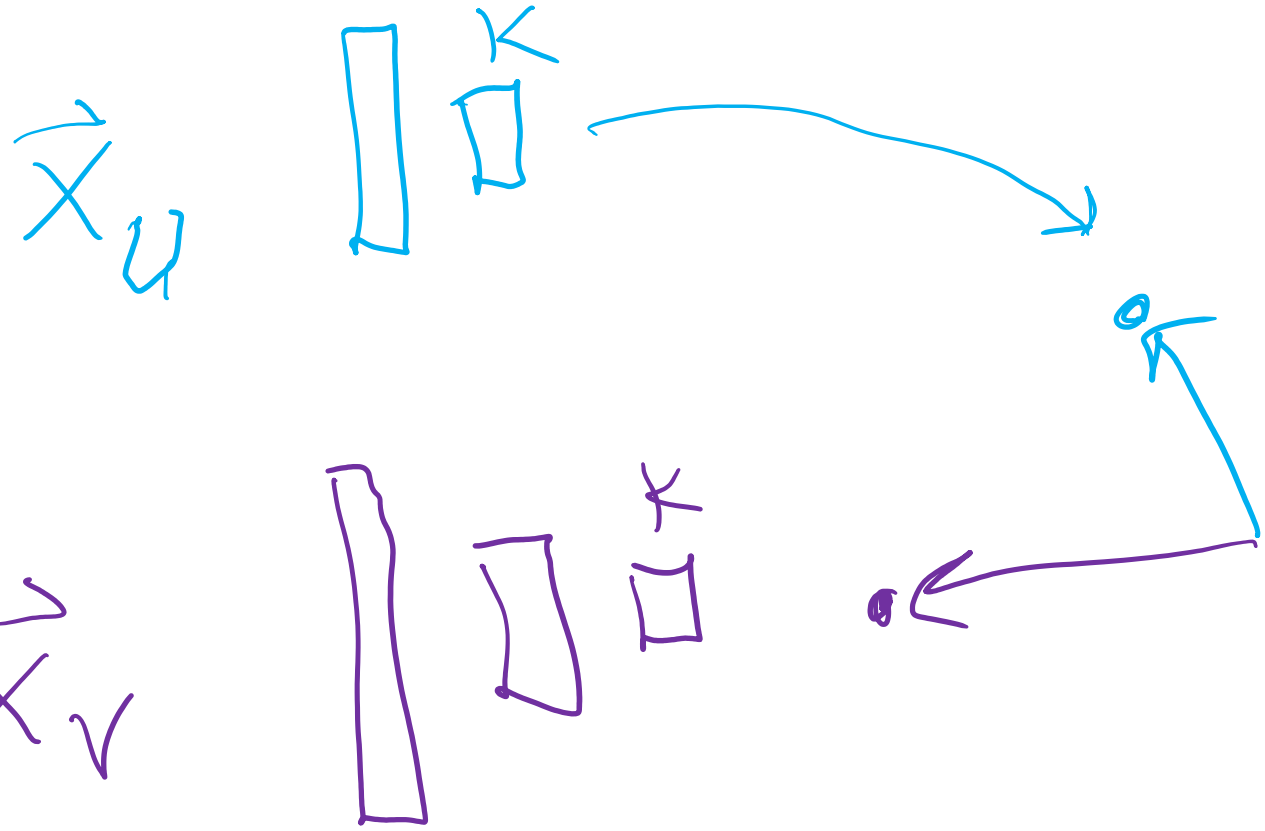
Bias in Recommender Systems

What high-level problems can occur from recommender systems?

Additional Techniques for Rec Systems

Neural Collaborative Filtering

User feature encoder network $f_\theta(\vec{x})$



$$\min_{\theta,\phi} \frac{1}{N} \sum_{i,j,r} \left(r_{ij} - f_\theta(\vec{x}_i)^T g_\phi(\vec{x}_j) \right)$$

Item feature encoder network $g_\phi(\vec{x})$

Low Rank Optimization

We've effectively been reconstructing a low-rank matrix of all predicted user ratings:

$$\hat{R} = \mathrm{U}\mathrm{V}^{\top}$$

Low Rank Optimization

We've effectively been reconstructing a low-rank matrix of all predicted user ratings:

$$\hat{R} = UV^T$$

R

	Minions	K-Pop Demon Hunters	Zootopia
Roz	1		5
BB-8	3	4	
C-3P0	3	5	2

$\hat{R}$

	Minions	K-Pop Demon Hunters	Zootopia
Roz			
BB-8			
C-3P0			

Low Rank Optimization

Idea: Directly estimate the (low-rank) matrix of all predicted ratings

$$\hat{R} = \mathrm{U}\mathrm{V}^{\top} \rightarrow \quad M$$

R

	Minions	K-Pop Demon Hunters	Zootopia
Roz	1		5
BB-8	3	4	
C-3P0	3	5	2

M

	Minions	K-Pop Demon Hunters	Zootopia
Roz			
BB-8			
C-3P0			

Low Rank Optimization

Minimize the nuclear norm of the predicted ratings matrix, M, while trying to keep the predicted rating $m_{i,j}$ as close as possible to the corresponding r rating for all user-item ratings, i, j, r, in our training data

The nuclear norm of a matrix, $\|M\|_*$, is the sum of singular values of M:

$$\|M\|_* = \sum \sigma_i$$

$$\underset{\substack{M \\ s.t. \|M\|_* < \tau}}{\operatorname{argmin}} \frac{1}{|\mathcal{D}|} \sum_{i,j,r \in \mathcal{D}} \left(r - m_{i,j}\right)^2$$

Low Rank Optimization

The nuclear norm of a matrix, $\|M\|_*$, is the sum of singular values of M: $\|M\|_* = \|\boldsymbol{\sigma}\|_1 = \sum \sigma_i$, where $\boldsymbol{\sigma}$ is the diagonal of S in the SVD decomposition of $M = USV^\top$

$$\underset{\substack{M \\ s.t. \|M\|_* < \tau}}{\operatorname{argmin}} \frac{1}{|\mathcal{D}|} \sum_{i,j,r \in \mathcal{D}} \left(r - m_{i,j}\right)^2$$

Algorithm:

Randomly initialize M

Repeat the following until convergence:

1. Replace i, j entries in M with r for all i, j, r in the training data
2. Soft threshold by τ the singular values of M to get S'
3. Update M by reconstructing with S': $M \leftarrow US'V^\top$

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
- Neural Nets on user and item input features