Announcements

Assignments

- HW8 (written + programming)
  - Due Thu 4/9, 11:59 pm

Introduction to Machine Learning

Recommender Systems

Instructor: Pat Virtue

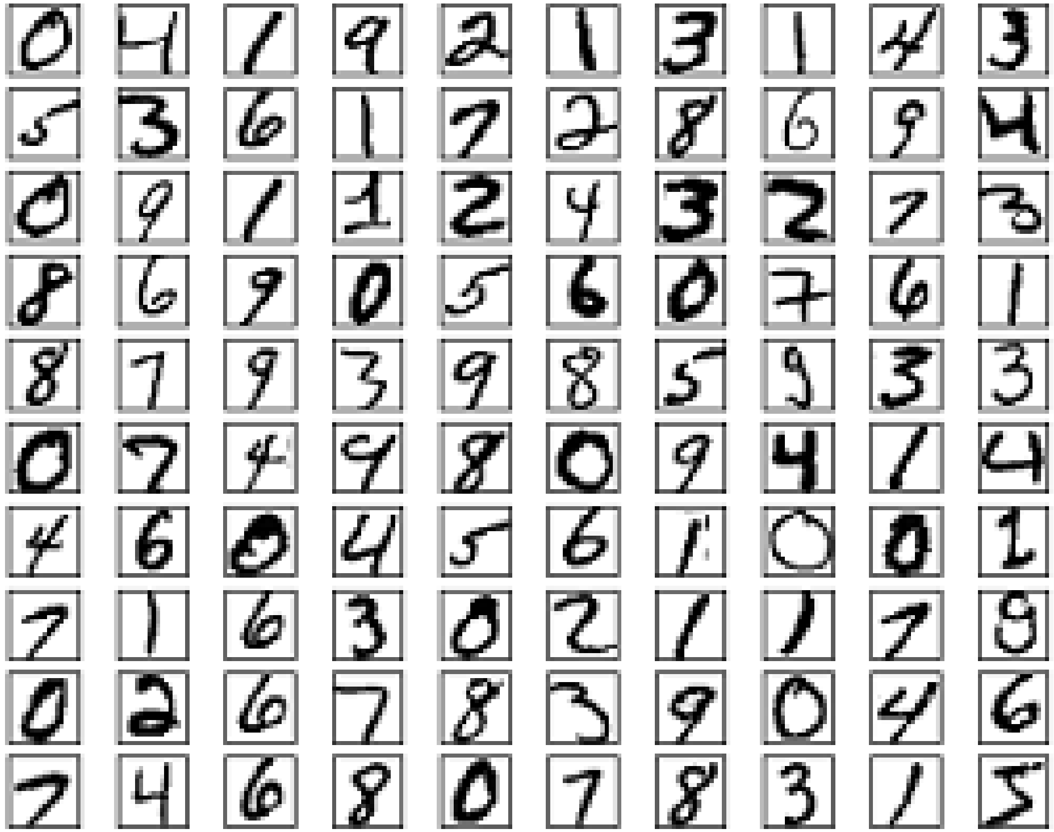
Background: Low Dimensional Embeddings

PCA: What did we do?

Background: Low Dimensional Embeddings

Why might low dimensional embeddings be useful?

- Example: MNIST digit classification with nearest neighbor



Background: Low Dimensional Embeddings

Why might low dimensional embeddings be useful?

- Example: MNIST digit classification with nearest neighbor

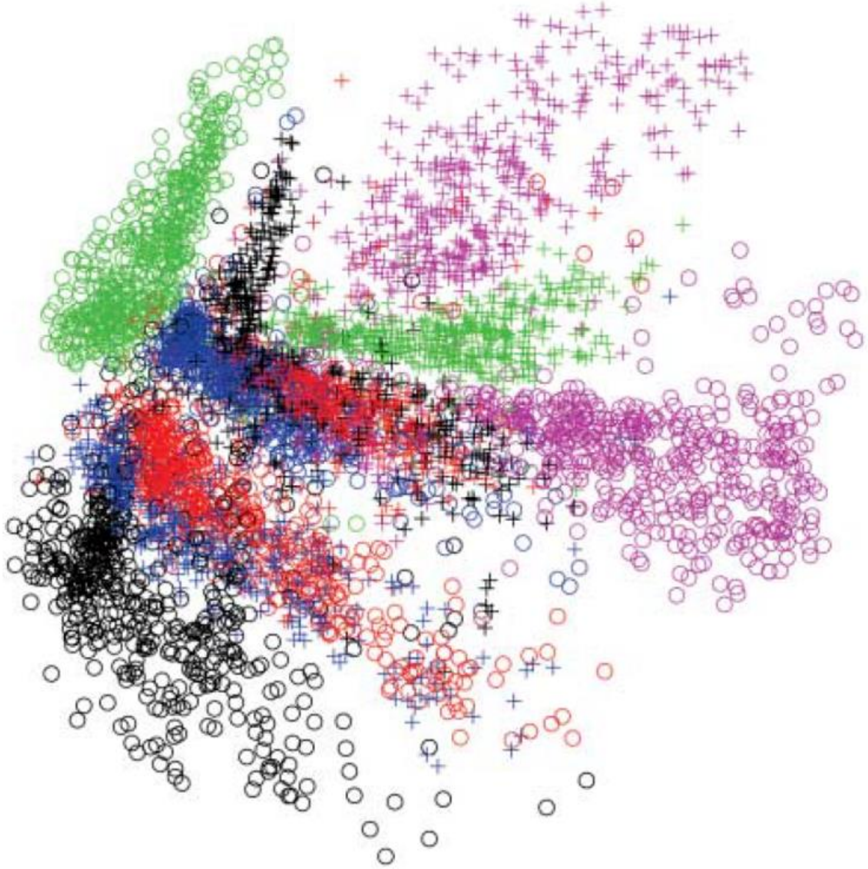


Image: Hinton & Salakhutdinov. *Science* 313.5786 (2006): 504-507.

Background: Measure of Similarity

We've been using Euclidean distance

- $d(\boldsymbol{x}, \boldsymbol{z}) = ||\boldsymbol{x} - \boldsymbol{z}||_2$

Cosine similarity

- To vectors are similar if the angle between them is small
- $d(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^T \boldsymbol{z}$

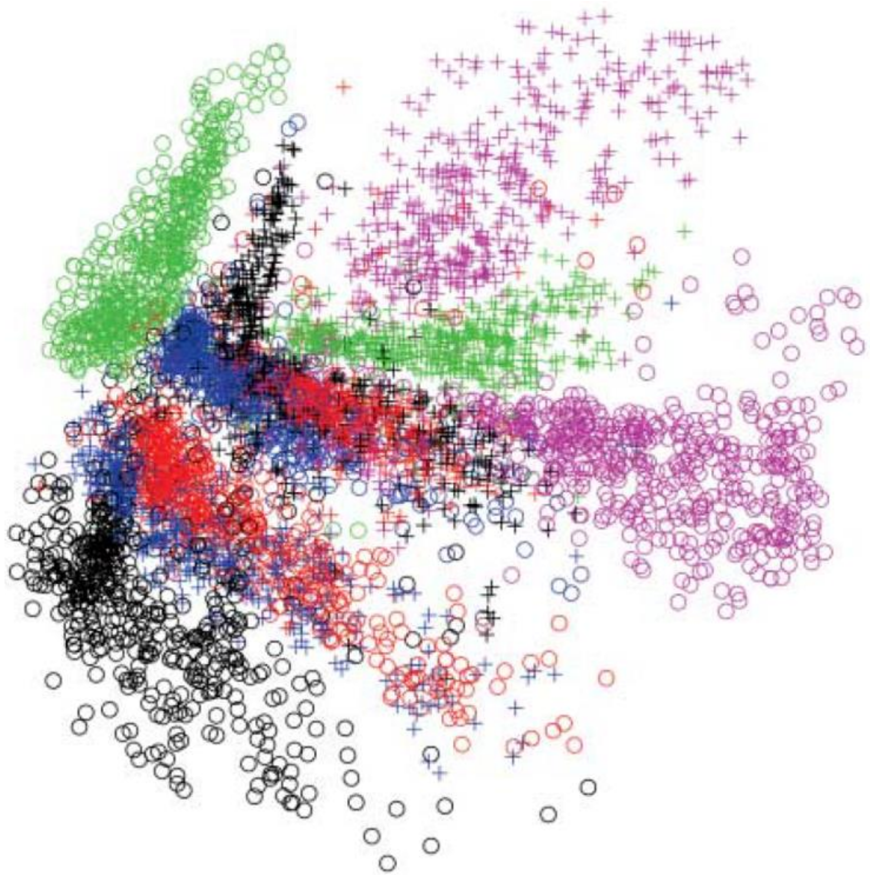


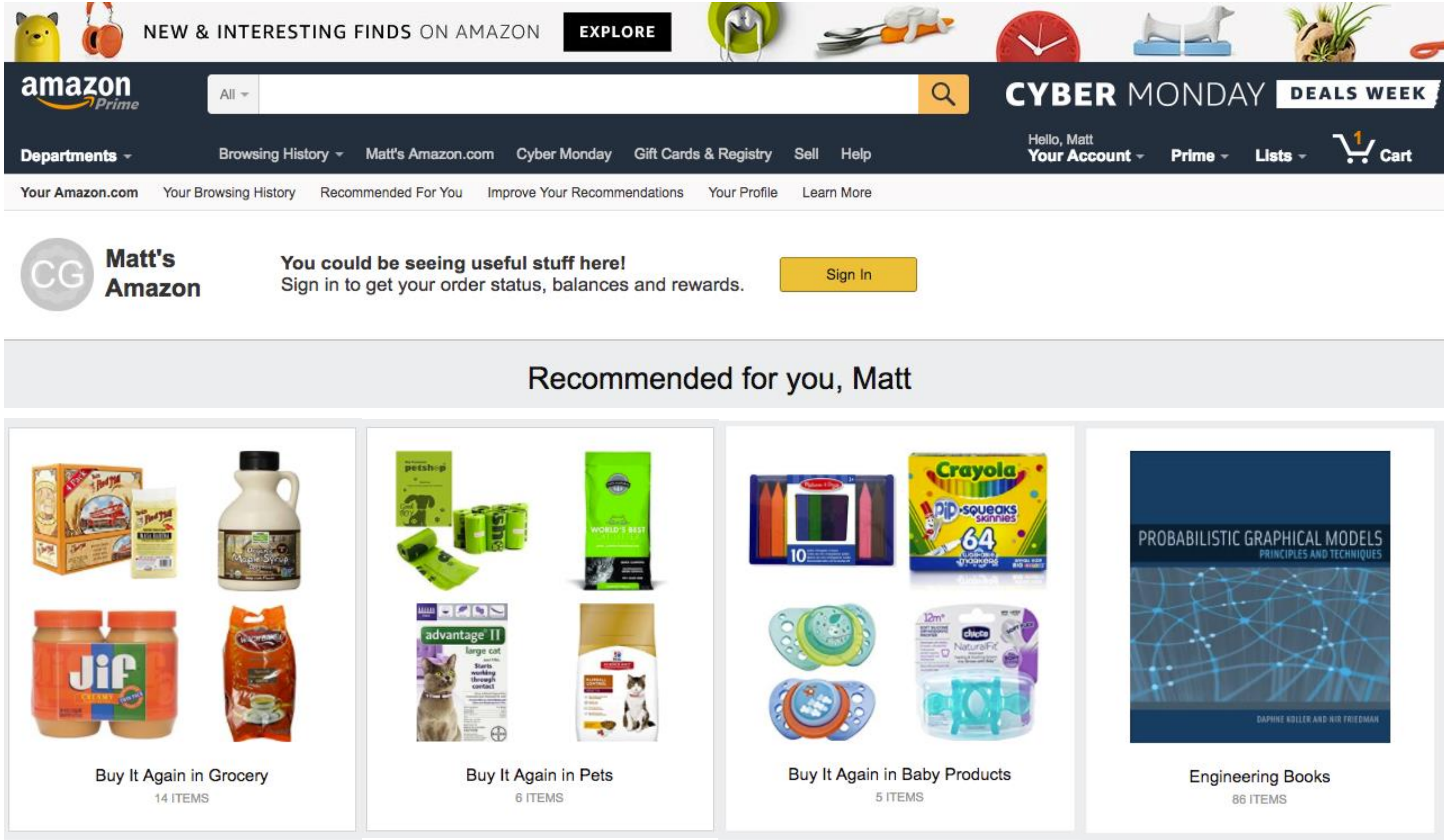
Image: Hinton & Salakhutdinov. *Science* 313.5786 (2006): 504-507.

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

Recommender Systems

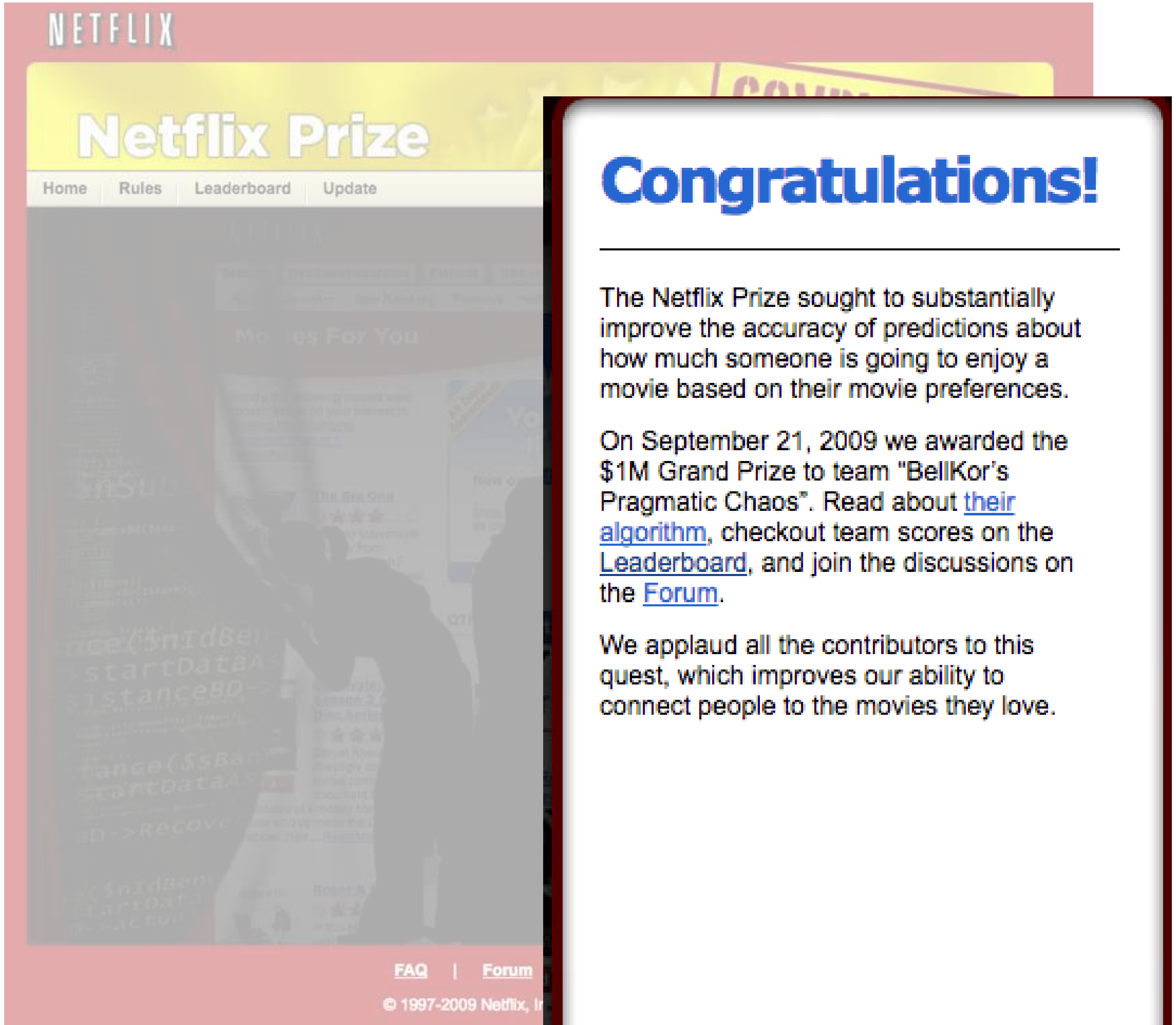


Slide credit: CMU MLD Matt Gormley

Recommender Systems

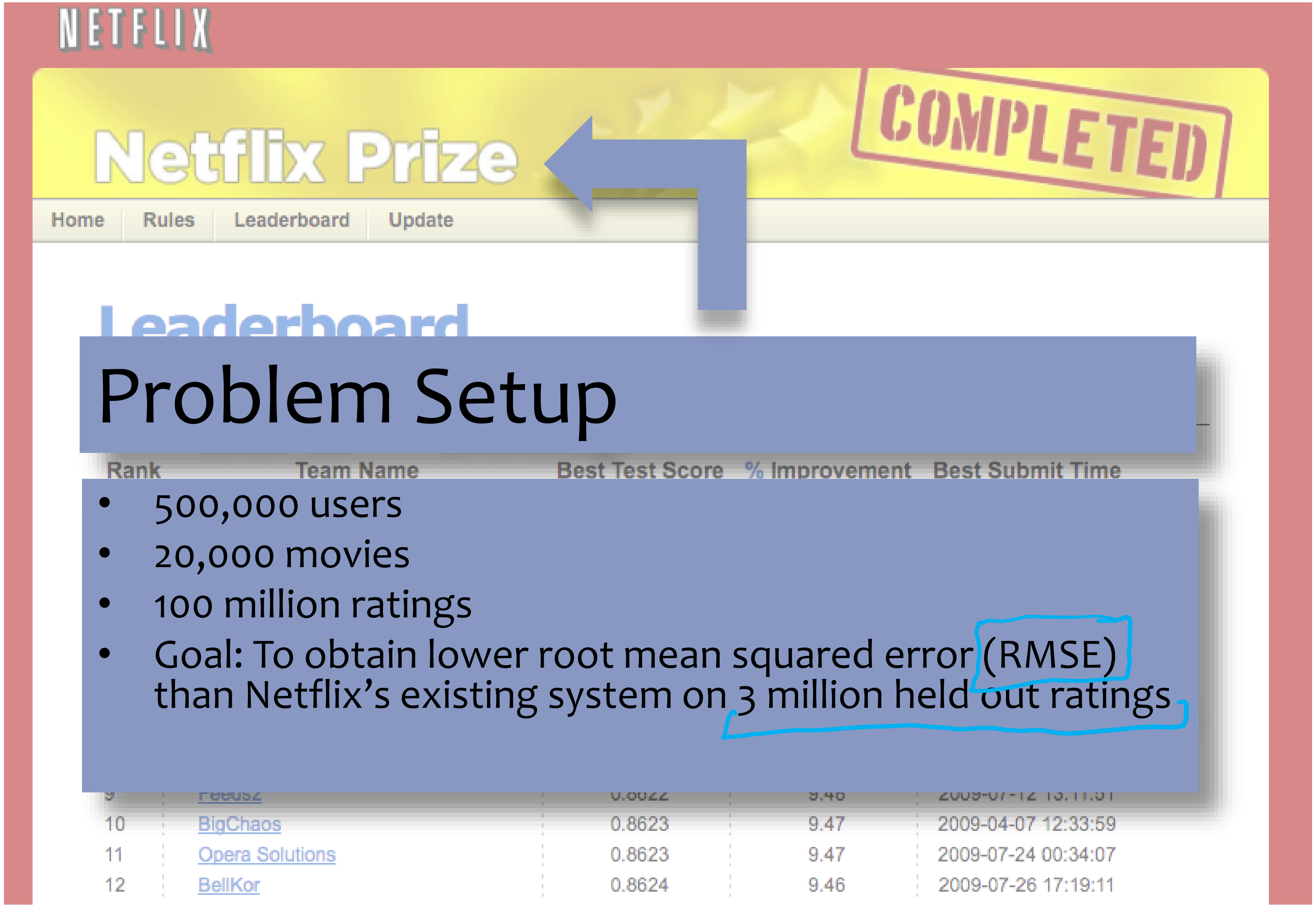
Slide credit: CMU MLD Matt Gormley

Recommender Systems



Slide credit: CMU MLD Matt Gormley

Recommender Systems



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

Slide credit: CMU MLD Matt Gormley

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.
- **Key Assumptions:**
  - Can represent ratings numerically as a user/item matrix
  - Users only rate a small number of items (the matrix is sparse)

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

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

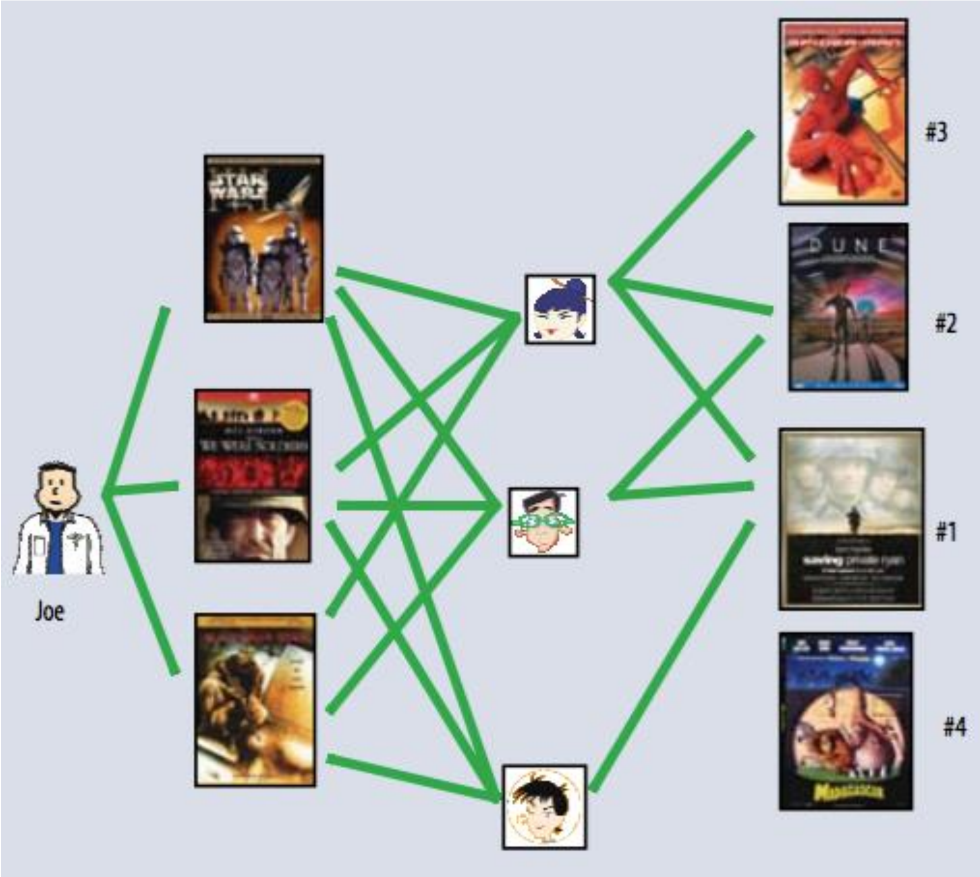
Collaborative Filtering

- **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

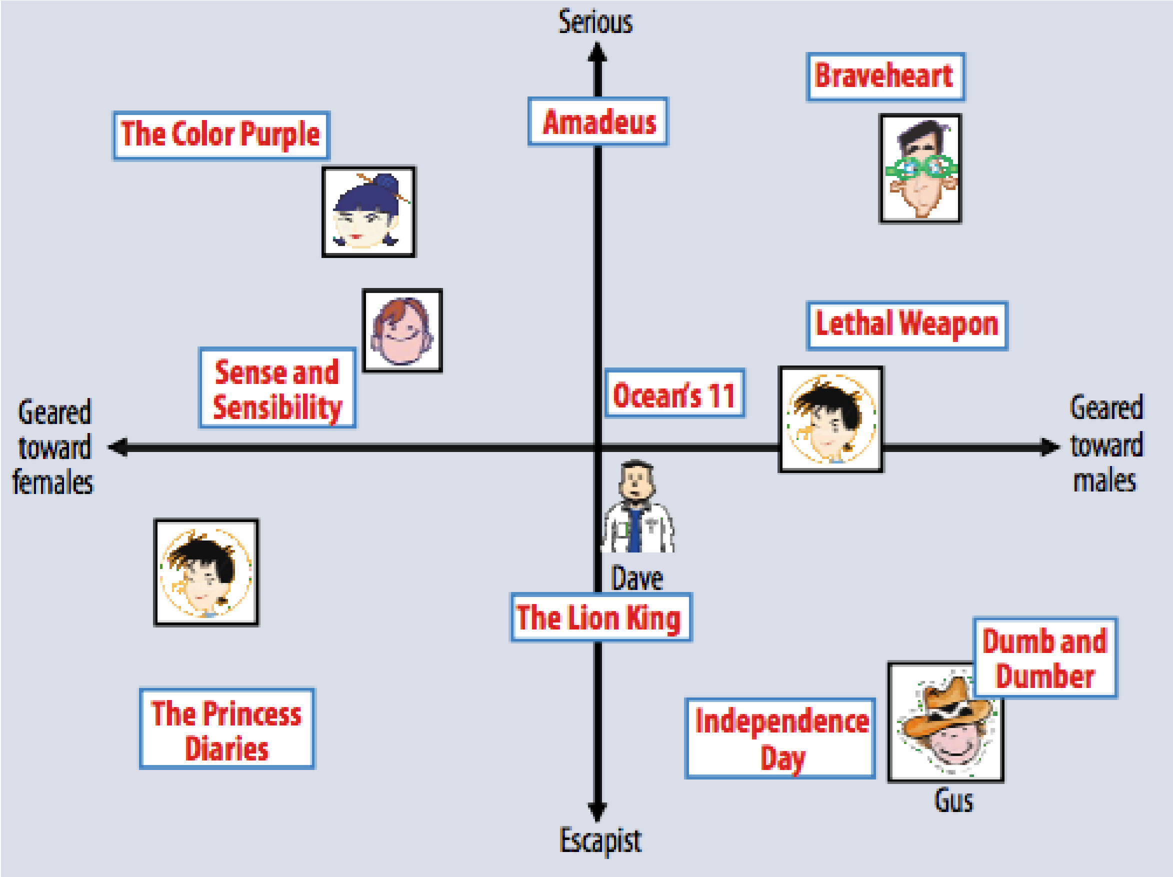
Slide from William Cohen

Two Types of Collaborative Filtering

1. Neighborhood Methods



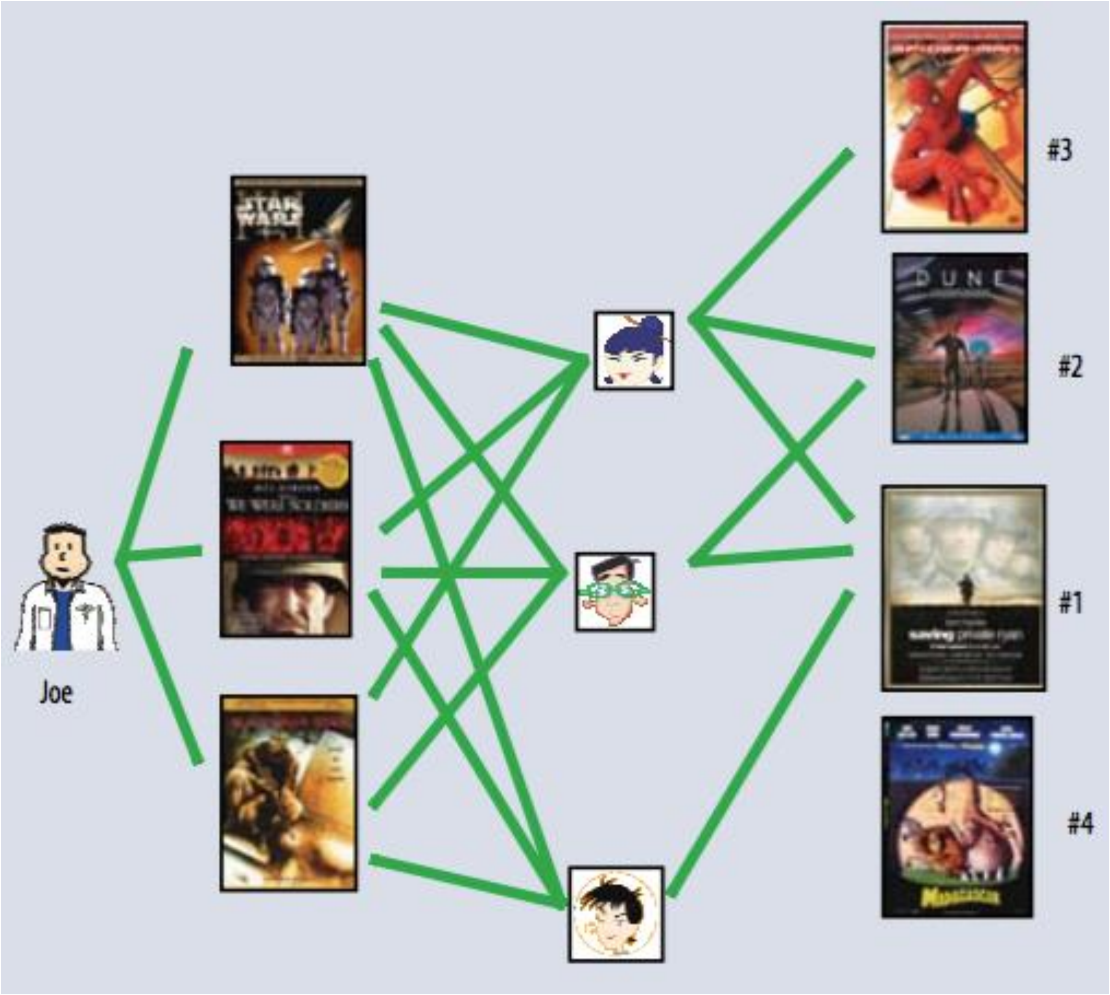
2. Latent Factor Methods



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:

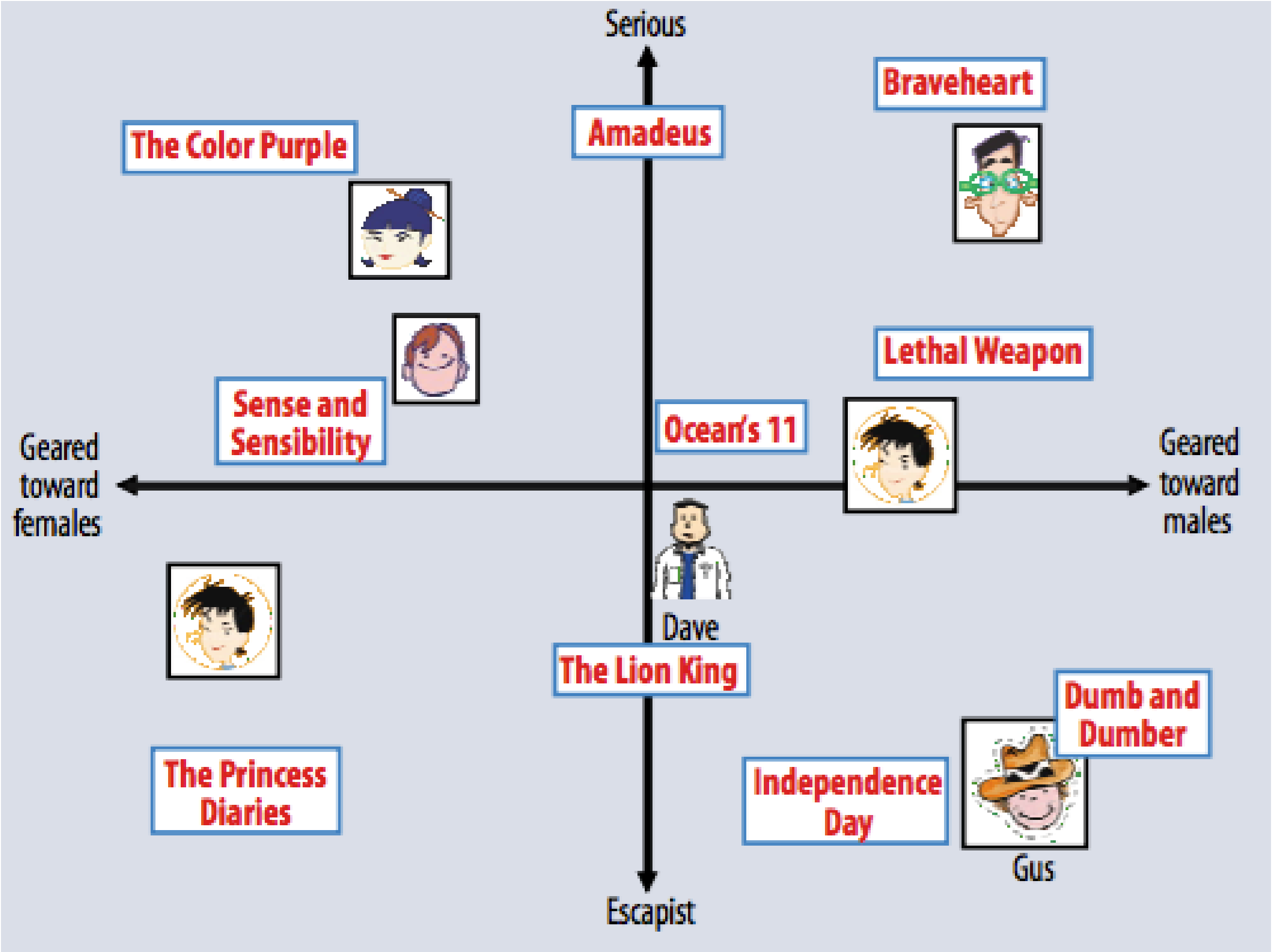
1. **Find neighbors** based on similarity of movie preferences
2. **Recommend** movies that those neighbors watched

Figures from Koren et al. (2009)

Two Types of Collaborative Filtering

2. Latent Factor Methods

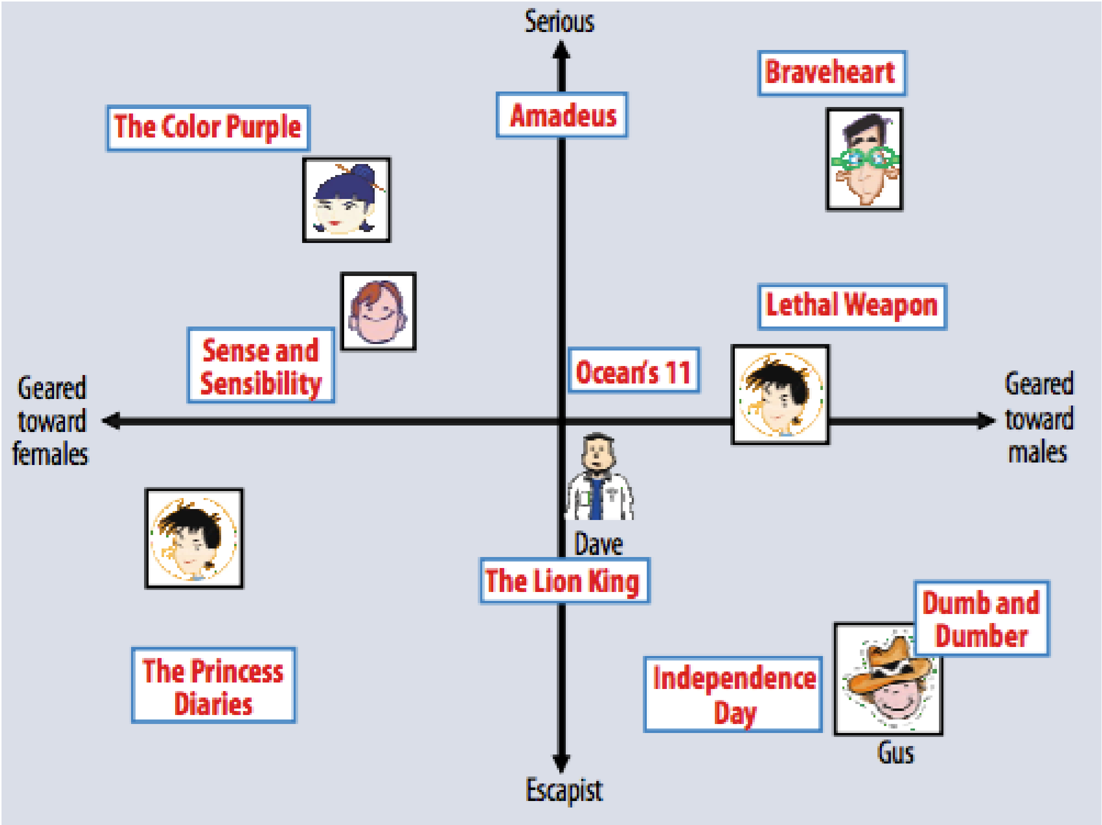
- Assume that both movies and users live in some **low-dimensional space** describing their properties
- **Recommend** a movie based on its **proximity** to the user in the latent space
- **Example Algorithm:** Matrix Factorization



Figures from Koren et al. (2009)

Recommender System: Matrix Factorization

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



Figures from Koren et al. (2009)

Recommender System: Matrix Factorization

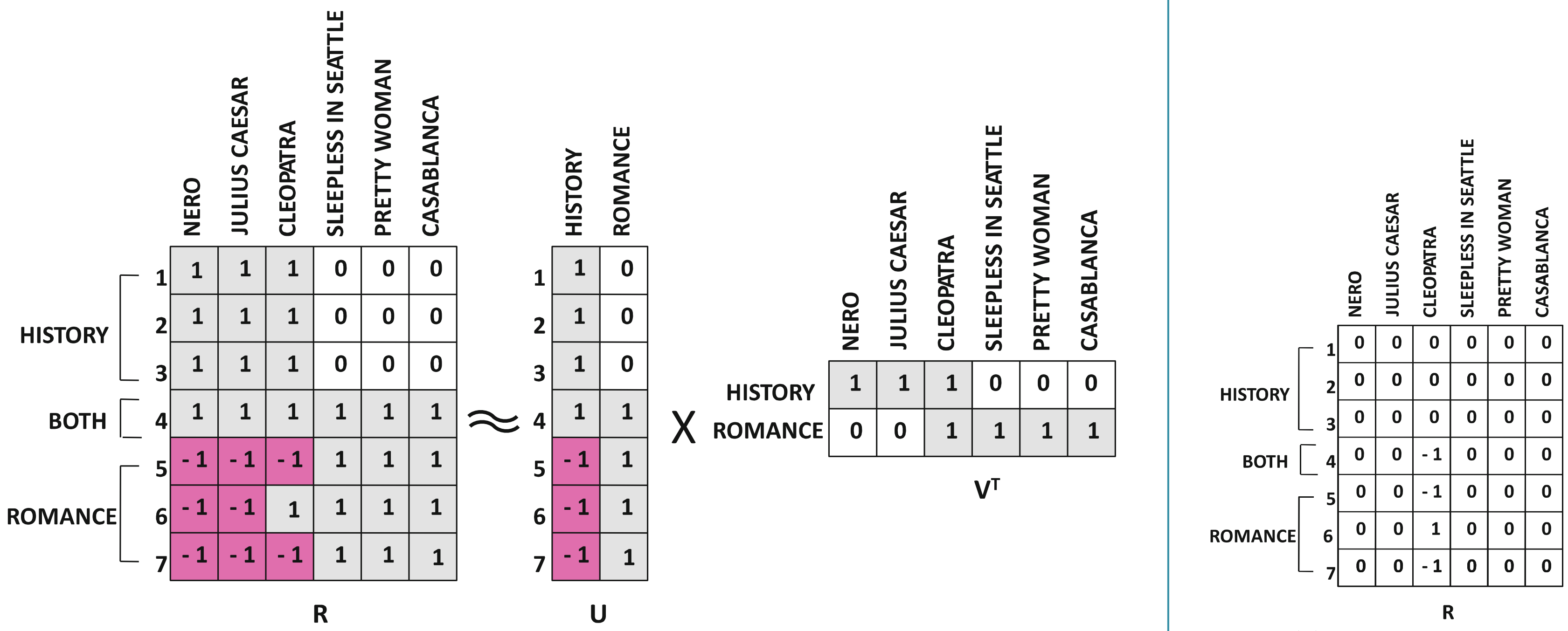
Optimization

Recommender System: Matrix Factorization

Sparse labels ☹

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

Example: MF for Netflix Problem

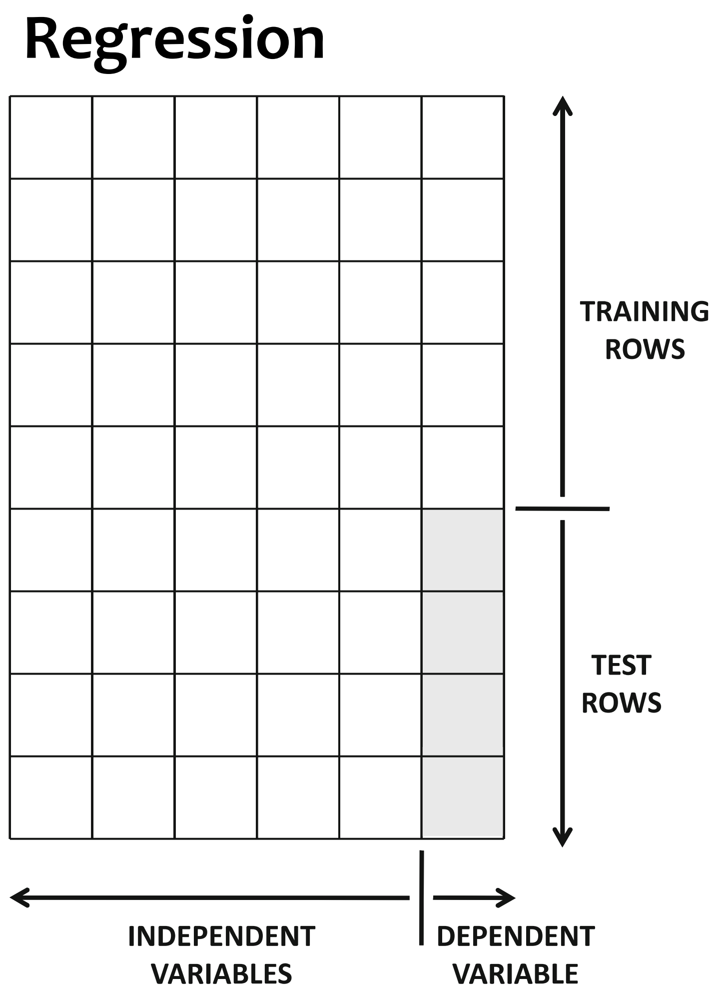


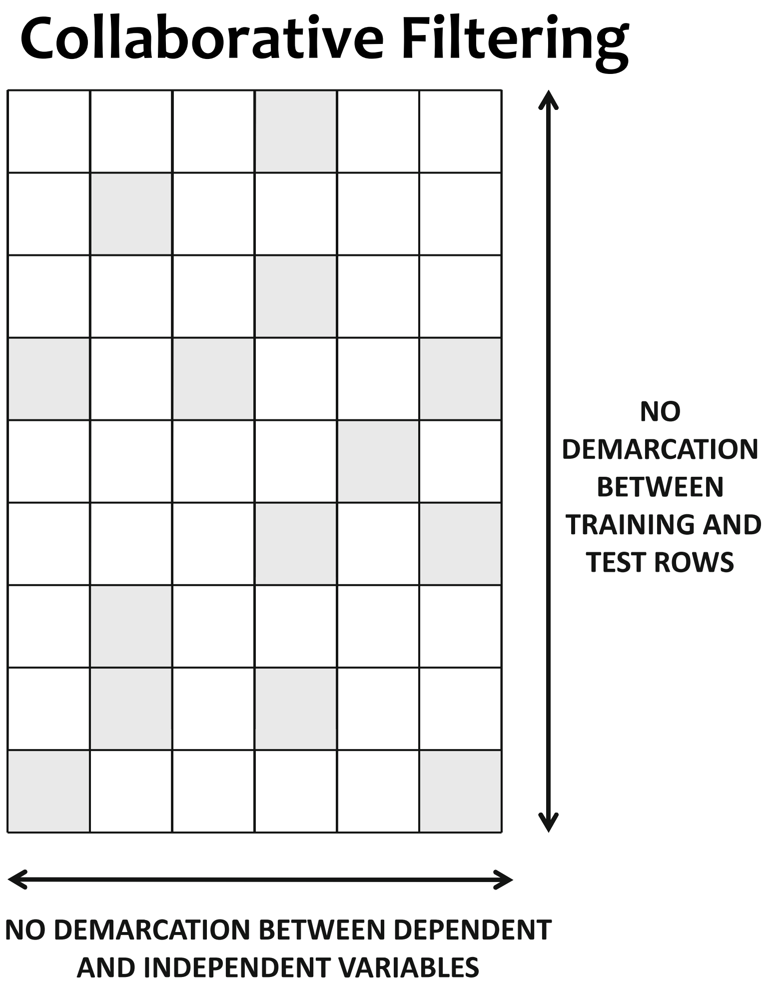
(a) Example of rank-2 matrix factorization

(b) Residual matrix

Figures from Aggarwal (2016)

Regression vs. Collaborative Filtering





Figures from Aggarwal (2016)

Matrix Factorization: SVD

We can use SVD, but as you'll see it has issues

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

Alita			
BB-8			
C-3Po			

Matrix Factorization: SGD

Objective function using only the labels we have

Piazza Poll 1

Is the following optimization a quadratic optimization?

$$\min_{\boldsymbol{U},\boldsymbol{V}} \sum_{i,j \in \mathcal{S}} \left(R_{ij} - {\boldsymbol{u}^{(i)}}^T \boldsymbol{v}^{(j)} \right)^2$$

A.

B.

C.

Matrix Factorization: SGD

Method of *alternating minimization*

$$\min_{\boldsymbol{U},\boldsymbol{V}} J(\boldsymbol{U},\boldsymbol{V}) \qquad J(\boldsymbol{U},\boldsymbol{V}) = \sum_{i,j\in\mathcal{S}} \left(R_{ij} - \boldsymbol{u}^{(i)^T} \boldsymbol{v}^{(j)}\right)^2$$

Matrix Factorization: SGD

Method of *alternating minimization*

$$\min_{\boldsymbol{U},\boldsymbol{V}} J(\boldsymbol{U},\boldsymbol{V}) \qquad J(\boldsymbol{U},\boldsymbol{V}) = \sum_{i,j\in\mathcal{S}} \left(R_{ij} - {\boldsymbol{u}^{(i)}}^T \boldsymbol{v}^{(j)}\right)^2$$

Matrix Factorization: SGD

Add regularization to avoid overfitting

$$\min_{\boldsymbol{U},\boldsymbol{V}} J(\boldsymbol{U},\boldsymbol{V}) \qquad J(\boldsymbol{U},\boldsymbol{V}) = \sum_{i,j\in\mathcal{S}} \left(R_{ij} - \boldsymbol{u}^{(i)^T}\boldsymbol{v}^{(j)}\right)^2$$

Summary

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

Collaborative filtering by Matrix Factorization (MF) is an **efficient** and **effective** approach

(SVD for MF is a bit broken)

MF is just another example of a **common recipe**:

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

Optimization

- Need alternating minimization
- Add regularization to avoid overfitting