



Recommender Systems + Matrix Factorization

Matt Gormley & Geoff Gordon

Lecture 23

Nov. 17, 2025

Reminders

- **Homework 8: Deep RL**
 - **Out: Sun, Nov. 16**
 - **Due: Mon, Nov. 24 at 11:59pm**

Learning Paradigms

Paradigm	Data
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \quad \mathbf{x} \sim p^*(\cdot) \text{ and } y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \dots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N \quad \mathbf{x} \sim p^*(\cdot)$
$\hookrightarrow$ Clustering	predict $\{z^{(i)}\}_{i=1}^N$ where $z^{(i)} \in \{1, \dots, K\}$
$\hookrightarrow$ Dimensionality Reduction	convert each $\mathbf{x}^{(i)} \in \mathbb{R}^M$ to $\mathbf{u}^{(i)} \in \mathbb{R}^K$ with $K \ll M$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \dots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \dots\}$
Reinforcement Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \dots\}$

ML Big Picture

Learning Paradigms:

What data is available and when? What form of prediction?

- supervised learning
- unsupervised learning
- semi-supervised learning
- reinforcement learning
- active learning
- imitation learning
- domain adaptation
- online learning
- density estimation
- recommender systems
- feature learning
- manifold learning
- dimensionality reduction
- ensemble learning
- distant supervision
- hyperparameter optimization

Theoretical Foundations:

What principles guide learning?

- ☐ probabilistic
- ☐ information theoretic
- ☐ evolutionary search
- ☐ ML as optimization

Problem Formulation:

What is the structure of our output prediction?

boolean	Binary Classification
categorical	Multiclass Classification
ordinal	Ordinal Classification
real	Regression
ordering	Ranking
multiple discrete	Structured Prediction
multiple continuous	(e.g. dynamical systems)
both discrete & cont.	(e.g. mixed graphical models)

Facets of Building ML Systems:

How to build systems that are robust, efficient, adaptive, effective?

1. Data prep
2. Model selection
3. Training (optimization / search)
4. Hyperparameter tuning on validation data
5. (Blind) Assessment on test data

Big Ideas in ML:

Which are the ideas driving development of the field?

- inductive bias
- generalization / overfitting
- bias-variance decomposition
- generative vs. discriminative
- deep nets, graphical models
- PAC learning
- distant rewards

Application Areas

Key challenges?

NLP, Speech, Computer Vision, Robotics, Medicine, Search

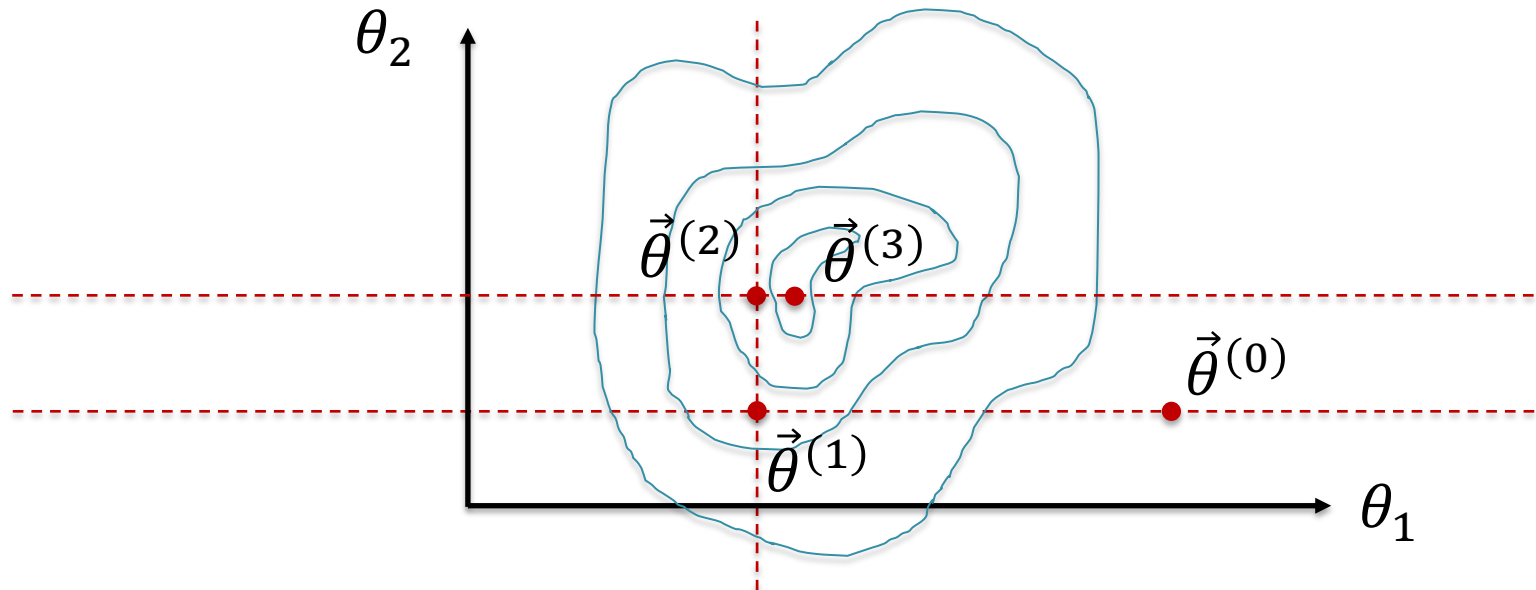
OPTIMIZATION BACKGROUND

Coordinate Descent

- Goal: minimize some objective

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}} J(\vec{\theta})$$

- Idea: iteratively pick one variable and minimize the objective w.r.t. just that one variable, *keeping all the others fixed*.



Block Coordinate Descent

- Goal: minimize some objective (with 2 blocks)

$$\vec{\alpha}^*, \vec{\beta}^* = \underset{\vec{\alpha}, \vec{\beta}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

- Idea: iteratively pick one *block* of variables ($\vec{\alpha}$ or $\vec{\beta}$) and minimize the objective w.r.t. that block, keeping the other(s) fixed.

while not converged:

$$\vec{\alpha} = \underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

$$\vec{\beta} = \underset{\vec{\beta}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

RECOMMENDER SYSTEMS

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

NEW & INTERESTING FINDS ON AMAZON EXPLORE

amazon Prime

All

CYBER MONDAY DEALS WEEK

Departments Browsing History Matt's Amazon.com Cyber Monday Gift Cards & Registry Sell Help

Hello, Matt Your Account Prime Lists Cart

Your Amazon.com Your Browsing History Recommended For You Improve Your Recommendations Your Profile Learn More

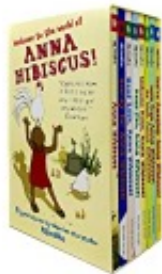


**Matt's
Amazon**

You could be seeing useful stuff here!
Sign in to get your order status, balances and rewards.

Sign In

Recommended for you, Matt



World of Anna Hibiscus 8 Books...

★★★★★ 77

\$34.98

70 pts

✓prime FREE Delivery

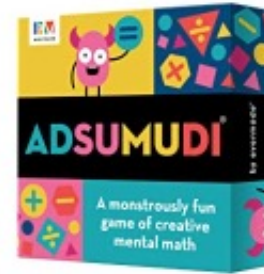


Biodegradable Dental Floss with a
Refillable Glass Holder |...

★★★★★ 1,806

\$9.99

✓prime FREE Delivery



Adsumudi Math Game - The
Monstrously Fun, Smart Game...

★★★★★ 551

\$17.99

✓prime FREE Delivery



Yamamotoyama - Jasmine Tea 16
bags

★★★★★ 531

\$6.30

✓prime FREE Delivery

Recommender Systems



The image is a screenshot of the Netflix Prize announcement page. At the top, the Netflix logo is on the left, and a yellow banner with the text "Netfix Prize" and a red "COMPLETED" stamp is on the right. Below the banner is a navigation bar with links: Home, Rules, Leaderboard, and Update. The main content area features a "Movies For You" section with a recommendation for "The Big One" and a "You really liked it..." section. A large white box on the right contains the "Congratulations!" message. In the background, silhouettes of three people are shown celebrating, with one person holding a trophy. The bottom of the page has a red bar with links for FAQ, Forum, and Netflix Home, and a copyright notice for 1997-2009 Netflix, Inc.

NETFLIX

Netfix Prize **COMPLETED**

Home Rules Leaderboard Update

NETFLIX

Browse Recommendations Friends Queue Buy DVDs

Home Genres New Releases Previews Netflix Top 100 Crit

Movies For You

Randy, the following movies were chosen based on your interest in:
[Dawson's Creek: Season 1](#)
[The Freshman](#)

The Big One

★★★★★
A subversive comedy from the mind of director Daniel Knauf, this series continues the adventures of a motley crew of misfits who've made the most of their small town in the shadow of their... [Read More](#)

You really liked it...

Now only for just \$5.98

Shop as low as

Original art

OT

Lenox Black, R and S

Add

Not in

By Decade
By Studio
Movies You've Seen

Give a friend

Congratulations!

The Netflix Prize sought to substantially improve the accuracy of predictions about how much someone is going to enjoy a movie based on their movie preferences.

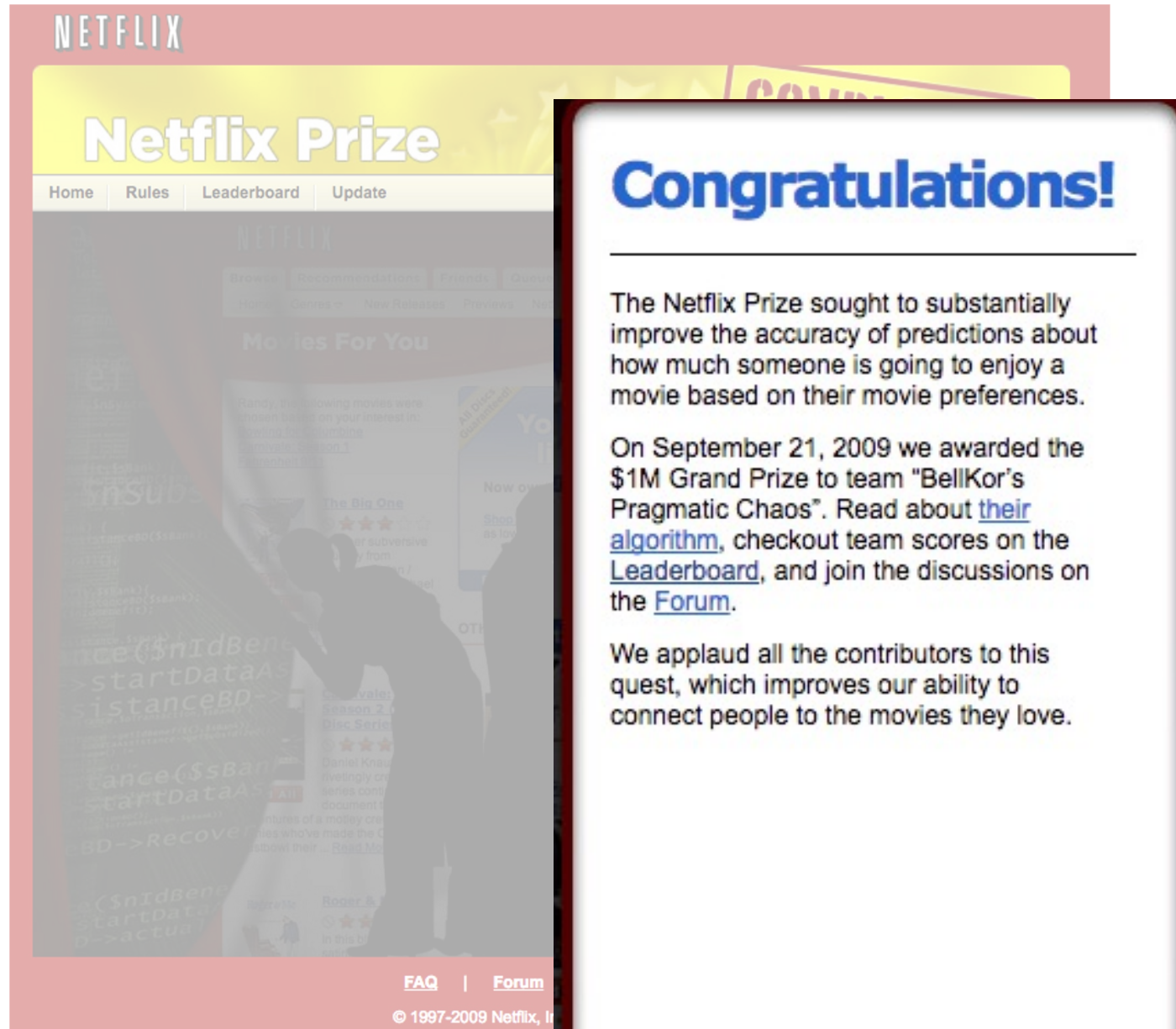
On September 21, 2009 we awarded the \$1M Grand Prize to team "BellKor's Pragmatic Chaos". Read about [their algorithm](#), checkout team scores on the [Leaderboard](#), and join the discussions on the [Forum](#).

We applaud all the contributors to this quest, which improves our ability to connect people to the movies they love.

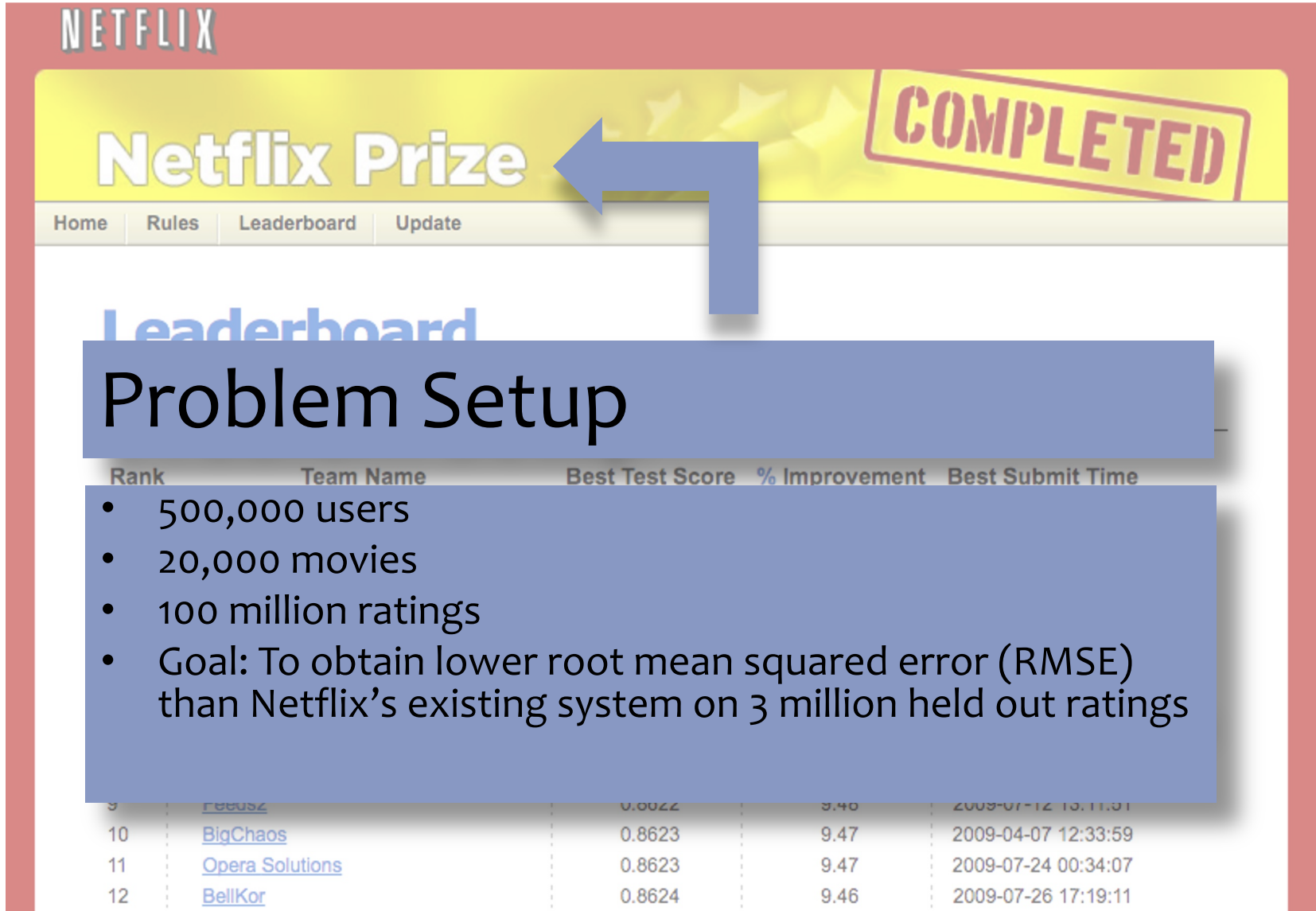
[FAQ](#) | [Forum](#) | [Netflix Home](#)

© 1997-2009 Netflix, Inc. All rights reserved.

Recommender Systems



Recommender Systems



The screenshot shows the Netflix Prize website interface. At the top, the Netflix logo is on the left, and a yellow banner contains the text 'Netflix Prize' and a 'COMPLETED' stamp. A blue arrow points from the 'COMPLETED' stamp to the 'Netflix Prize' text. 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 large blue box with the text 'Problem Setup' is overlaid on the page. Below this box, a table shows the leaderboard data.

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	Feedz	0.8622	9.48	2009-07-12 15:11:01
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

Recommender Systems

NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update

Leaderboard

Showing Test Score. [Click here to show quiz score](#)

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
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

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
Alice	1		5
Bob	3	4	
Charlie	3	5	2

Two Types of Recommender Systems

Content Filtering

- **Definition:** Recommendation approach that analyzes **item attributes** (features, metadata, keywords, categories) and matches them to a **user's known preferences**.
- **Example:** Pandora.com music recommendations (Music Genome Project)
- **Con:** Assumes access to **side information** about items (e.g. properties of a song)
- **Pro:** Got a **new item** to add? No problem, just be sure to include the side information

Collaborative Filtering

- **Definition:** A recommendation approach that analyzes **user-item interaction patterns** (ratings, clicks, purchases) to find similar users or items based on behavior.
- **Example:** Netflix movie recommendations
- **Pro:** Does not assume access to **side information** about items (e.g. does not need to know about movie genres)
- **Con:** Does not work on **new items** that have no ratings

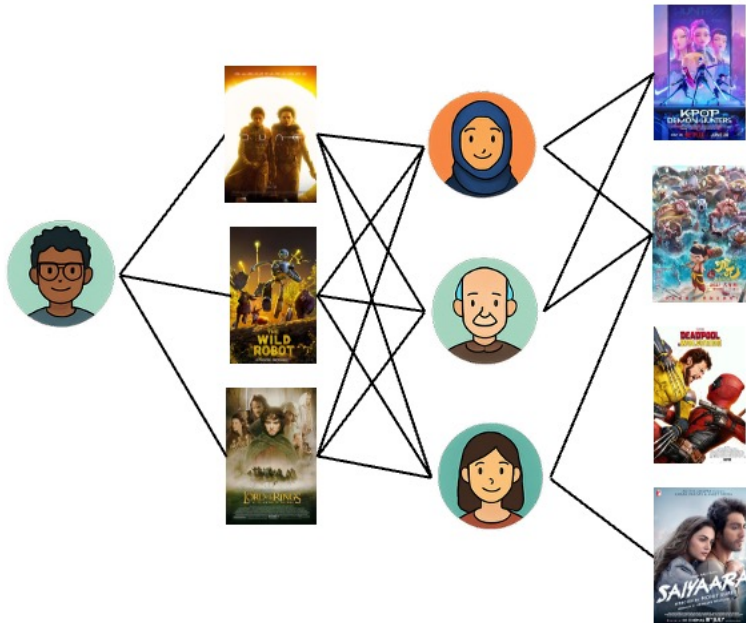
COLLABORATIVE FILTERING

Collaborative Filtering

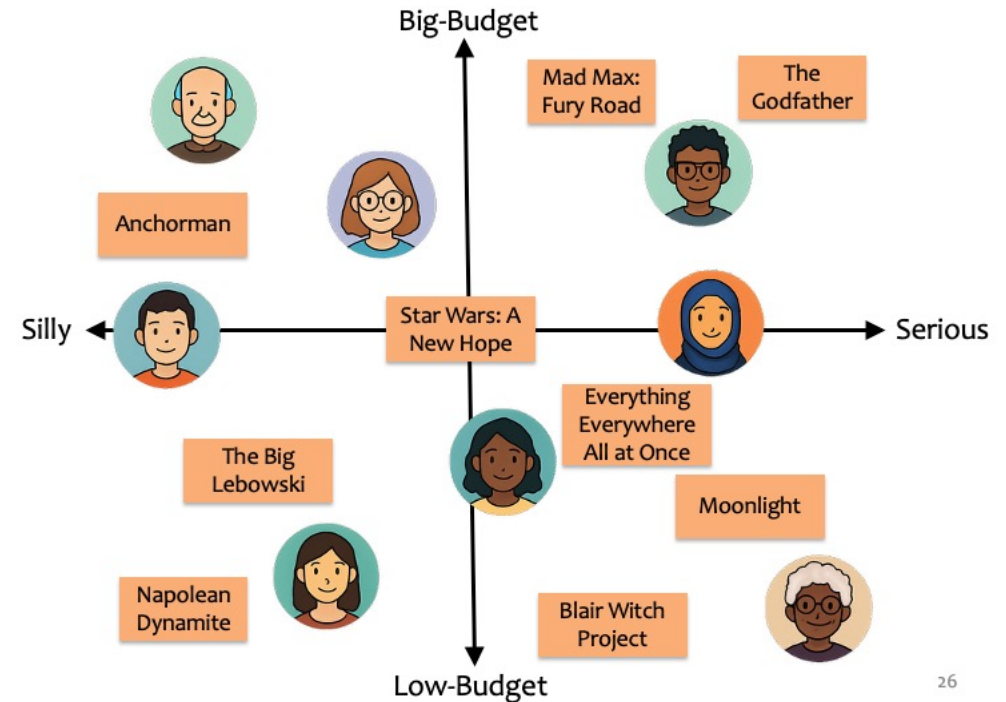
- **Everyday Examples of Collaborative Filtering...**
 - Bestseller lists
 - Top 40 music lists
 - The “recent returns” shelf at the library
 - Unmarked but well-used paths thru the woods
 - The printer room at work
 - “Read any good books lately?”
 - ...
- **Common insight:** personal tastes are correlated
 - If Alice and Bob both like X and Alice likes Y then Bob is more likely to like Y
 - especially (perhaps) if Bob knows Alice

Two Types of Collaborative Filtering

1. Neighborhood Methods

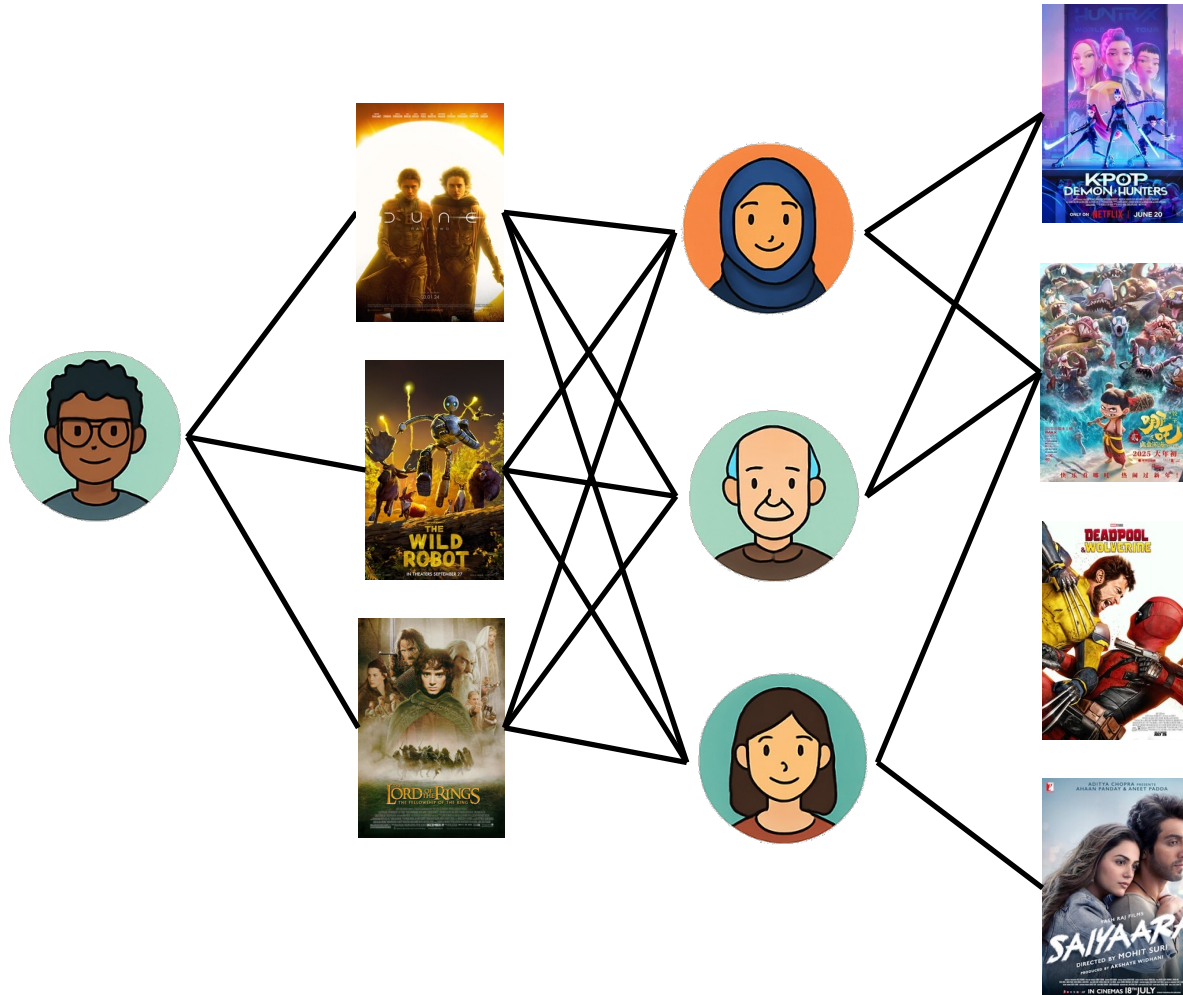


2. Latent Factor Methods



Two Types of Collaborative Filtering

1. Neighborhood Methods



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

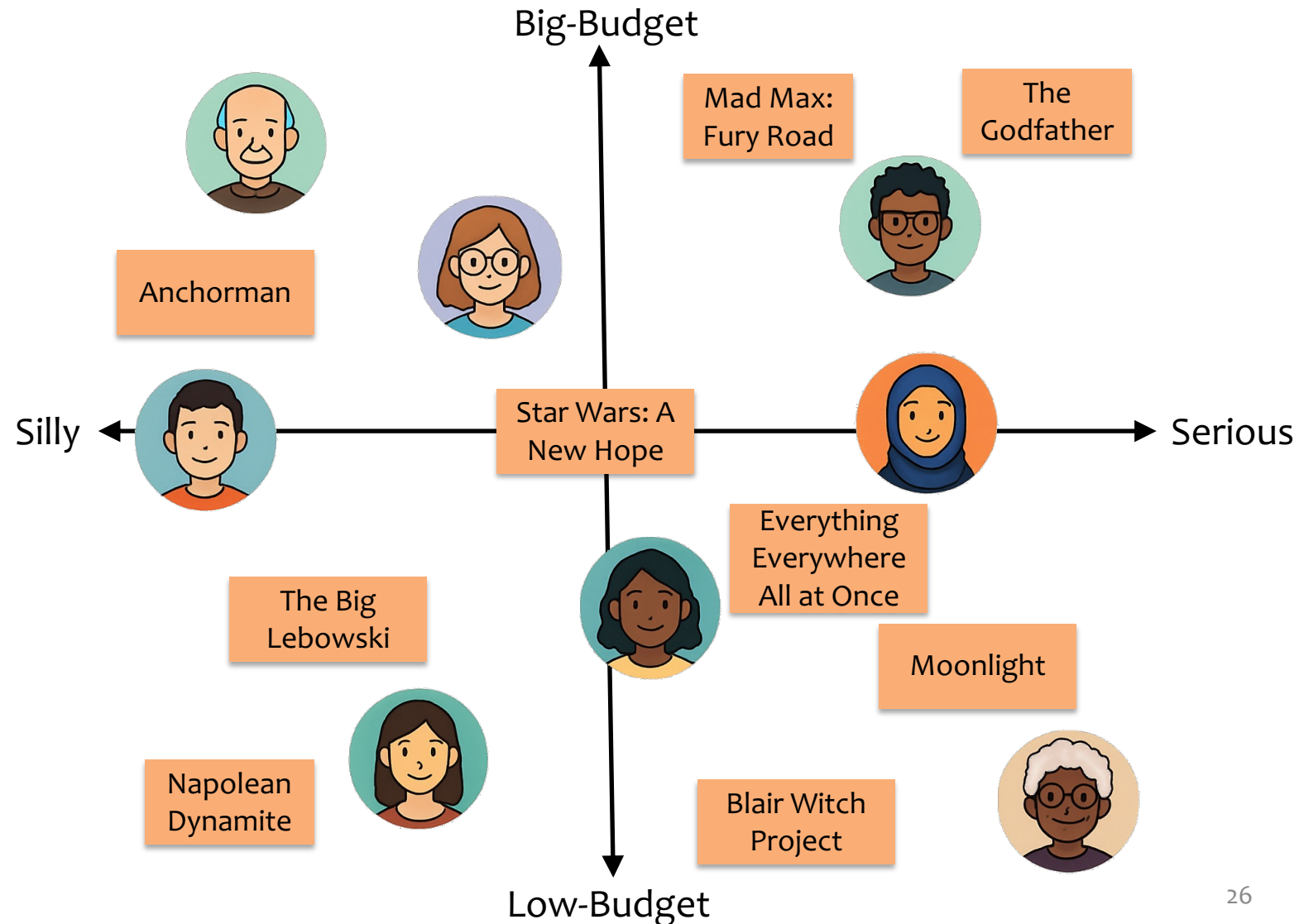
Algorithm:

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 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



Recommending Movies

Question:

Applied to the Netflix Prize problem, which of the following methods *always* requires side information about the users and movies?

Select all that apply

- A. principal component analysis
- B. collaborative filtering
- C. latent factor methods
- D. ensemble methods
- E. content filtering
- F. neighborhood methods
- G. recommender systems

Answer:

MATRIX FACTORIZATION

Matrix Factorization

- Many different ways of factorizing a matrix
- We'll consider three:
 1. Unconstrained Matrix Factorization
 2. Singular Value Decomposition
 3. Non-negative Matrix Factorization
- MF is just another example of a **common recipe**:
 1. define a model
 2. define an objective function
 3. optimize with SGD

Linear Algebra Background

Rank of a Matrix: For an $m \times n$ matrix $\mathbf{R}$, the *rank* is the number of linearly independent columns (equivalently, rows).

$\text{rank}(\mathbf{R}) = k \iff \mathbf{R} \text{ has } k \text{ linearly independent columns.}$

Example.

$$\mathbf{R} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

The first two columns are independent; the third is their sum. Thus $\text{rank}(\mathbf{R}) = 2$.

Basis Vectors: A *basis* is a smallest set of independent vectors that can generate all vectors in a space. If $\mathbf{R}$ has rank k , then there exist k basis vectors that span all its columns.

Example. A basis for the $\mathbf{R}$ above is given by its first two columns:

$$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{u}_2 = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}.$$

The third column satisfies

$$\mathbf{r}_3 = 1 \cdot \mathbf{u}_1 + 1 \cdot \mathbf{u}_2.$$

Low-rank Matrix Factorization

Case 1: Exact Factorization

Case 2: Approximate Factorization

Approximation Error:

Low-rank Matrix Factorization

Low-Rank Factorization: We seek matrices $\mathbf{U}$ and $\mathbf{V}$ with k columns such that

$$\mathbf{R} = \mathbf{UV}^\top, \quad \text{rank}(\mathbf{UV}^\top) \leq k.$$

Example (Exact Rank-2 Factorization). Let

$$\mathbf{U} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{V}^\top = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

Then

$$\mathbf{UV}^\top = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \mathbf{R}.$$

Approximation Viewpoint: If $\mathbf{R}$ has rank $\ell > k$, then it cannot be represented exactly with rank k , but we can choose $\mathbf{U}, \mathbf{V}$ to make the residual $\mathbf{E} = \mathbf{R} - \mathbf{UV}^\top$ as small as possible.

Example (Forced Rank-1 Approximation). Let

$$\mathbf{U} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{V}^\top = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}.$$

Then

$$\mathbf{UV}^\top = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The residual is

$$\mathbf{E} = \mathbf{R} - \mathbf{UV}^\top = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad \|\mathbf{E}\|_F = \sqrt{2}.$$

Example: MF for Netflix Problem

		NERO	JULIUS CAESAR	CLEOPATRA	SLEEPLESS IN SEATTLE	PRETTY WOMAN	CASABLANCA
HISTORY	1	1	1	1	0	0	0
	2	1	1	1	0	0	0
	3	1	1	1	0	0	0
BOTH	4	1	1	1	1	1	1
ROMANCE	5	-1	-1	-1	1	1	1
	6	-1	-1	1	1	1	1
	7	-1	-1	-1	1	1	1

<

(a) Example of rank-2 matrix factorization

		NERO	JULIUS CAESAR	CLEOPATRA	SLEEPLESS IN SEATTLE	PRETTY WOMAN	CASABLANCA
HISTORY	1	0	0	0	0	0	0
	2	0	0	0	0	0	0
	3	0	0	0	0	0	0
BOTH	4	0	0	-1	0	0	0
	5	0	0	-1	0	0	0
ROMANCE	6	0	0	1	0	0	0
	7	0	0	-1	0	0	0

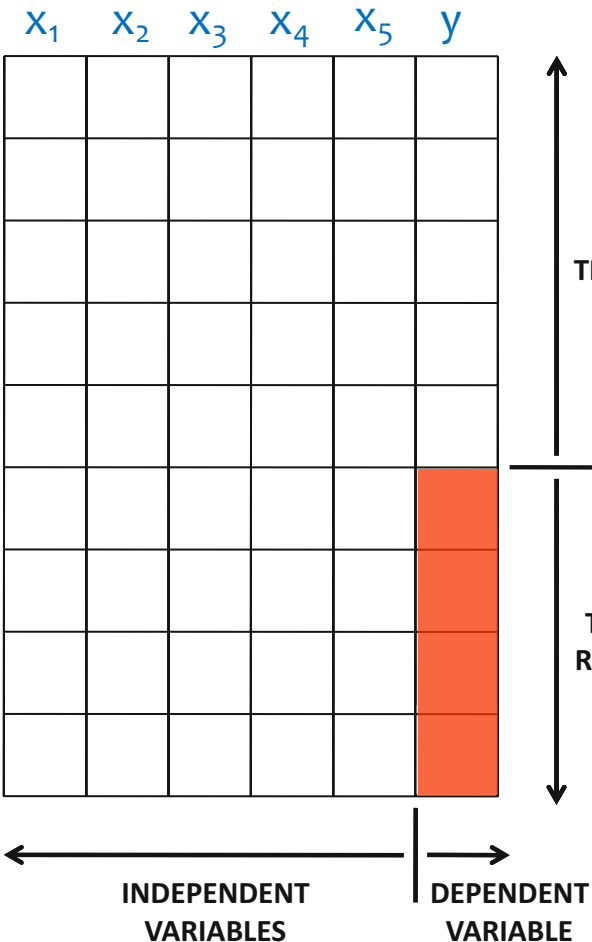
 R

(b) Residual matrix

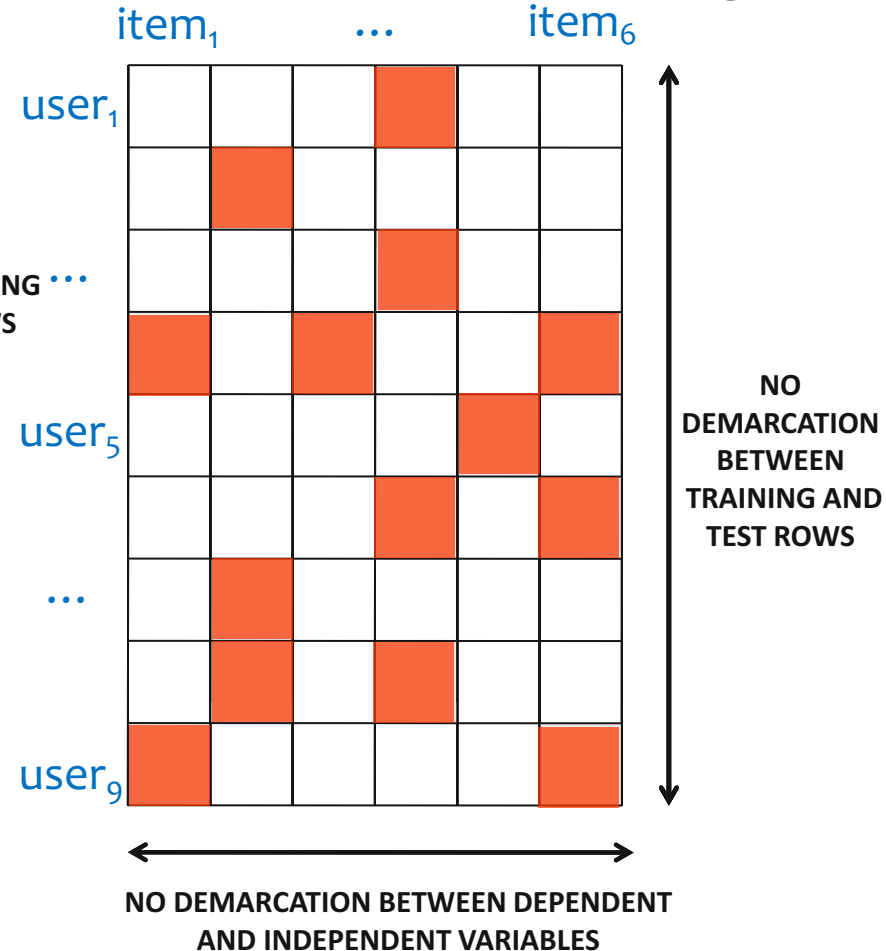
Regression vs. Collaborative Filtering

Goal of each problem is to predict the values of the missing squares 

Regression



Collaborative Filtering



UNCONSTRAINED MATRIX FACTORIZATION

Unconstrained Matrix Factorization

Opt. Problem #1 (fully observed R)

Model Predictions:

Opt. Problem #2 (partially observed R)

Gradient Descent:

Unconstrained Matrix Factorization

SGD for UMF:

while not converged:

1. Sample (i, j) from $\mathcal{Z}$ uniformly at random
2. Compute $e_{ij} = r_{ij} - \mathbf{u}_i^T \mathbf{v}_j$
3. Update:

$$\mathbf{u}_i \leftarrow \mathbf{u}_i - \eta \nabla_{\mathbf{u}_i} J_{ij}(\mathbf{U}, \mathbf{V})$$

$$\mathbf{v}_j \leftarrow \mathbf{v}_j - \eta \nabla_{\mathbf{v}_j} J_{ij}(\mathbf{U}, \mathbf{V})$$

where:

with Regularization

$$J_{ij}(\mathbf{U}, \mathbf{V}) = \frac{1}{2}(r_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2 + \lambda(\|\mathbf{u}_i\|_2^2 + \|\mathbf{v}_j\|_2^2)$$

$$\nabla_{\mathbf{u}_i} J_{ij}(\mathbf{U}, \mathbf{V}) = -e_{ij} \mathbf{v}_j + \lambda \mathbf{u}_i$$

$$\nabla_{\mathbf{v}_j} J_{ij}(\mathbf{U}, \mathbf{V}) = -e_{ij} \mathbf{u}_i + \lambda \mathbf{v}_j$$

User/Item Bias terms

$$\hat{r}_{ij} = o_i + p_j + \mathbf{u}_i^T \mathbf{v}_j$$

matrix trick:

$$\mathbf{U} = \begin{bmatrix} - & \mathbf{u}_1 & - & o_1 & 1 \\ - & \mathbf{u}_2 & - & o_2 & 1 \\ & \vdots & & & \\ - & \mathbf{u}_m & - & o_m & 1 \end{bmatrix}$$
$$\mathbf{V} = \begin{bmatrix} - & \mathbf{p}_1 & - & 1 & p_1 \\ - & \mathbf{p}_2 & - & 1 & p_2 \\ & \vdots & & & \\ - & \mathbf{p}_n & - & 1 & p_n \end{bmatrix}$$

Unconstrained Matrix Factorization

Alternating Least Squares (ALS) for UMF:

Block Coordinate Descent:

while not converged:

1. $\mathbf{U} = \arg \min_{\mathbf{U}} J(\mathbf{U}, \mathbf{V})$
2. $\mathbf{V} = \arg \min_{\mathbf{V}} J(\mathbf{U}, \mathbf{V})$

Applied to UMF:

$$J(\mathbf{U}, \mathbf{V}) = \frac{1}{2} \sum_{(i,j) \in \mathcal{Z}} (r_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2$$

→ If $\mathbf{U}$ is fixed: Least Squares to solve for $\mathbf{V}$

→ If $\mathbf{V}$ is fixed: Least Squares to solve for $\mathbf{U}$

The Least Squares Problem:

(i.e. solving Linear Regression in closed form)

$$J(\Theta) = \frac{1}{2} \sum_{i=1}^N \left(y_i - \hat{\Theta}^\top \mathbf{x}_i \right)^2$$

take derivatives, set to zero, and solve in closed form.

Solving $J(\mathbf{U}, \mathbf{V})$ in closed form directly isn't easy and $J(\mathbf{U}, \mathbf{V})$ is nonconvex.

Matrix Factorization

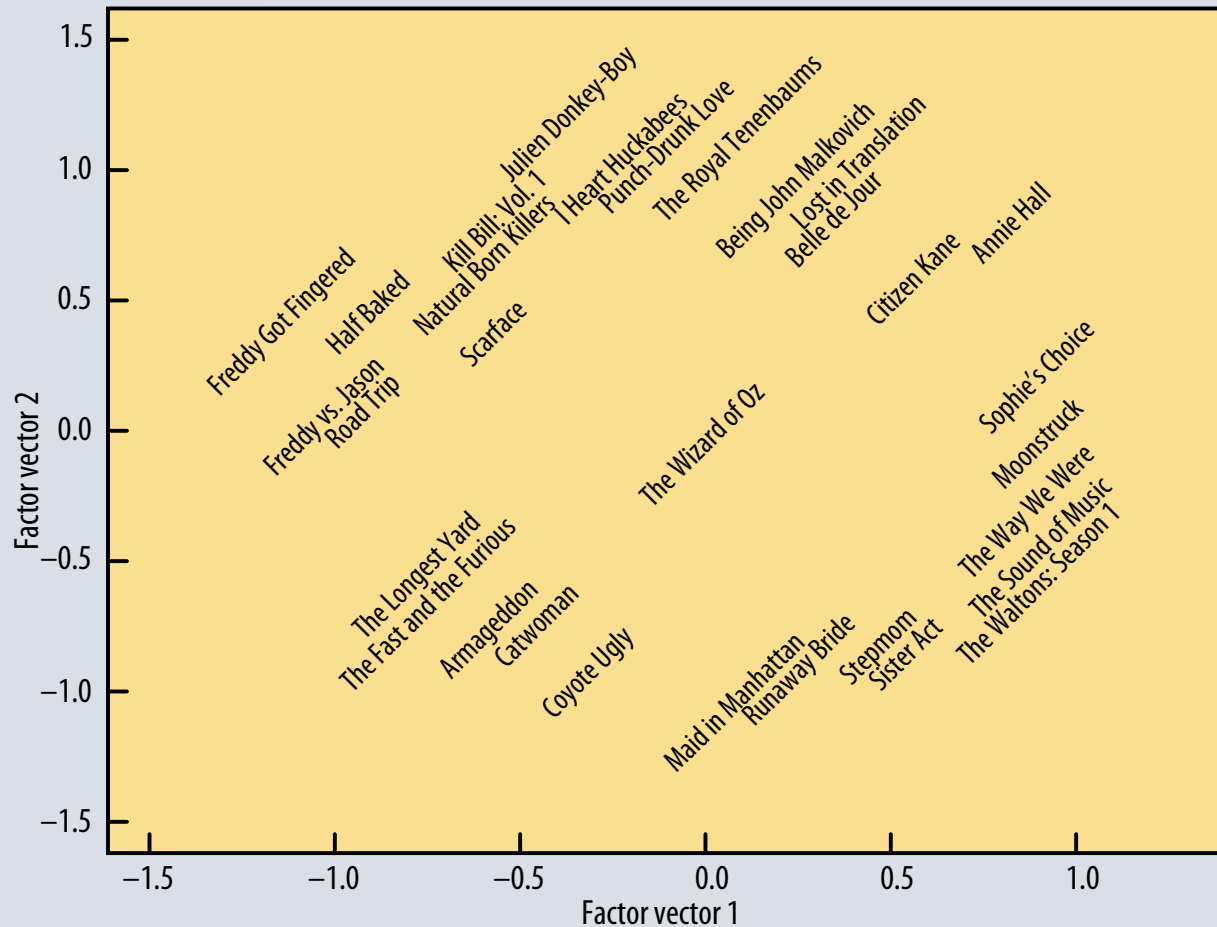


Figure 3. The first two vectors from a matrix decomposition of the Netflix Prize data. Selected movies are placed at the appropriate spot based on their factor vectors in two dimensions. The plot reveals distinct genres, including clusters of movies with strong female leads, fraternity humor, and quirky independent films.

Figure from Koren et al. (2009)

Question: Write an algorithm for visualizing two latent factors.

Answer:

Matrix Factorization

Comparison of Optimization Algorithms

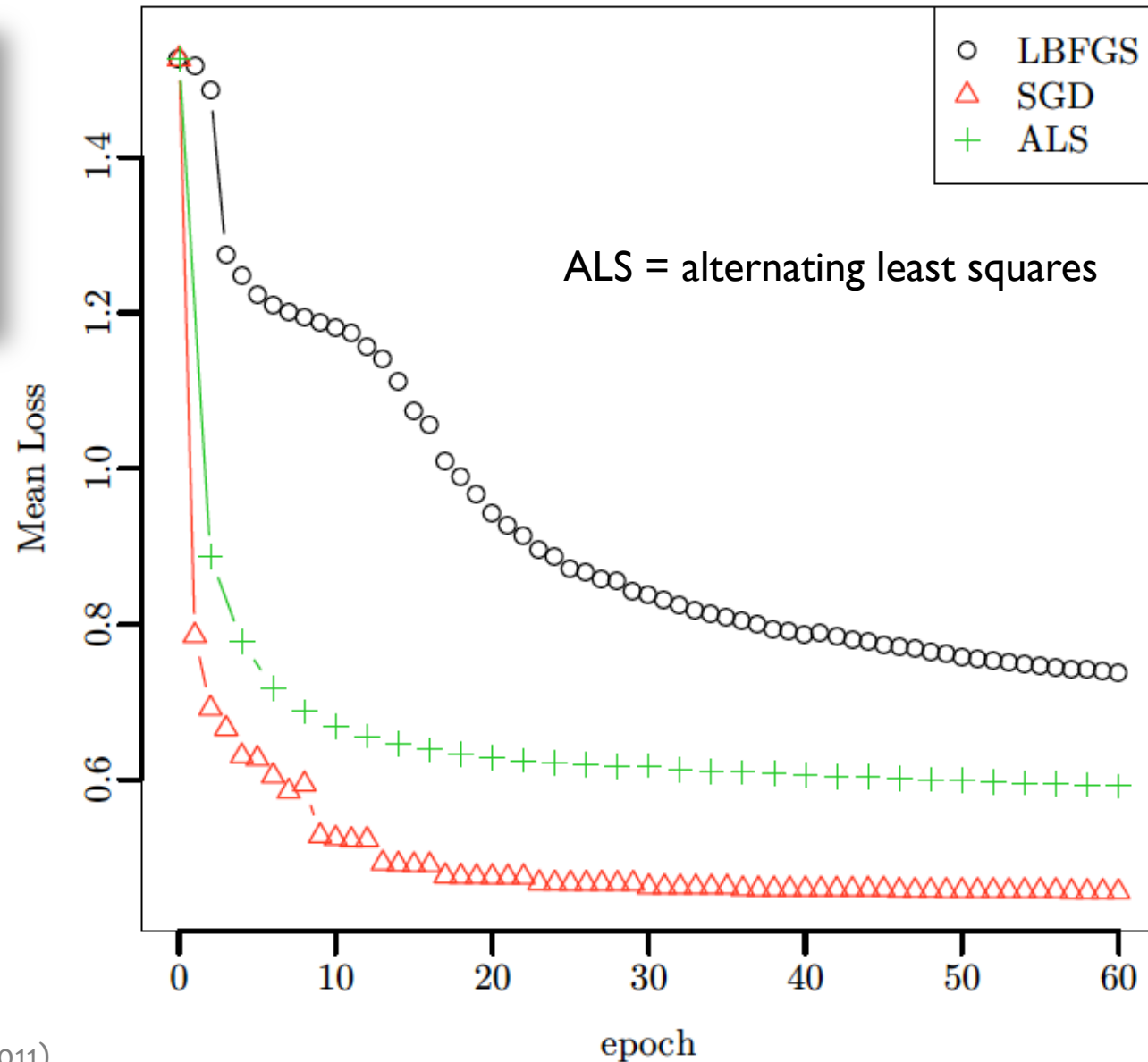


Figure from Gemulla et al. (2011)

SVD FOR COLLABORATIVE FILTERING

Singular Value Decomposition

Definition: Any $m \times n$ matrix $\mathbf{R}$ can be factored as

$$\mathbf{R} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\top,$$

where

$$\mathbf{U} \in \mathbb{R}^{m \times m}, \quad \mathbf{V} \in \mathbb{R}^{n \times n}, \quad \mathbf{\Sigma} \in \mathbb{R}^{m \times n}.$$

Properties:

- Columns of $\mathbf{U}$ are orthonormal $\Rightarrow \mathbf{U}^\top \mathbf{U} = \mathbf{I}$.
- Columns of $\mathbf{V}$ are orthonormal $\Rightarrow \mathbf{V}^\top \mathbf{V} = \mathbf{I}$.
- $\mathbf{\Sigma}$ is diagonal (possibly rectangular) with entries

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq 0,$$

called the *singular values*.

Interpretation: SVD writes $\mathbf{R}$ as a sum of rank-1 pieces:

$$\mathbf{R} = \sum_{k=1}^{\text{rank}(\mathbf{R})} \sigma_k \mathbf{u}_k \mathbf{v}_k^\top.$$

The first few components (largest σ_k) capture the most important structure; smaller ones refine details.

Best Rank- k Approximation: Keeping only the first k singular values gives

$$\mathbf{R}_k = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^\top,$$

the closest rank- k matrix to $\mathbf{R}$ under the Frobenius norm.

Singular Value Decomposition for Collaborative Filtering

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$$

where $\mathbf{\Lambda}$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

Suppose we have the SVD of our ratings matrix

$$R = Q\Sigma P^T,$$

but then we truncate each of Q , Σ , and P s.t. Q and P have only k columns and Σ is $k \times k$:

$$R \approx Q_k \Sigma_k P_k^T$$

For collaborative filtering, let:

$$U \triangleq Q_k \Sigma_k$$

$$V \triangleq P_k$$

$$\Rightarrow U, V = \operatorname{argmin}_{U, V} \frac{1}{2} \|R - UV^T\|_2^2$$

s.t. columns of U are mutually orthogonal

s.t. columns of V are mutually orthogonal

Theorem: If R fully observed and no regularization, the optimal UV^T from SVD equals the optimal UV^T from Unconstrained MF

NON-NEGATIVE MATRIX FACTORIZATION

Implicit Feedback Datasets

- What information does a five-star rating contain?



- Implicit Feedback Datasets:
 - In many settings, users don't have a way of expressing *dislike* for an item (e.g. can't provide negative ratings)
 - The only mechanism for feedback is to “like” something
- Examples:
 - Facebook has a “Like” button, but no “Dislike” button
 - Google’s “+1” button
 - Pinterest pins
 - Purchasing an item on Amazon indicates a preference for it, but there are many reasons you might *not* purchase an item (besides dislike)
 - Search engines collect click data but don't have a clear mechanism for observing dislike of a webpage

Non-negative Matrix Factorization

Constrained Optimization Problem:

$$U, V = \operatorname{argmin}_{U, V} \frac{1}{2} \|R - UV^T\|_2^2$$

$$\text{s.t. } U_{ij} \geq 0$$

$$\text{s.t. } V_{ij} \geq 0$$

Multiplicative Updates: simple iterative algorithm for solving just involves multiplying a few entries together

Fighting Fire with Fire: Using Antidote Data to Improve Polarization and Fairness of Recommender Systems

Bashir Rastegarpanah
Boston University
bashir@bu.edu

Krishna P. Gummadi
MPI-SWS
gummadi@mpi-sws.org

Mark Crovella
Boston University
crovella@bu.edu

where $S_j = \sum_{i \in \Omega_j} u_i u_i^T + \tilde{U} \tilde{U}^T + \lambda I_\ell$.

By using (9) instead of the general formula in (5) we can significantly reduce the number of computations required for finding the gradient of the utility function with respect to the antidote data. Furthermore, the term $g_j^T U^T S_j^{-1}$ appears in all the partial derivatives that correspond to elements in column j of $\hat{X}$ and can be precomputed in each iteration of the algorithm and reused for computing partial derivatives with respect to different antidote users.

5 SOCIAL OBJECTIVE FUNCTIONS

The previous section developed a general framework for improving various properties of recommender systems; in this section we show how to apply that framework specifically to issues of polarization and fairness.

As described in Section 2, polarization is the degree to which opinions, views, and sentiments diverge within a population. Recommender systems can capture this effect through the ratings that they present for items. To formalize this notion, we define polarization in terms of the variability of predicted ratings when compared across users. In fact, we note that both very high variability, and very low variability of ratings may be undesirable. In the case of high variability, users have strongly divergent opinions, leading to conflict. Recent analyses of the YouTube recommendation system have suggested that it can enhance this effect [29, 30]. On the other hand, the convergence of user preferences, i.e., very low variability of ratings given to each item across users, corresponds to increased homogeneity, an undesirable phenomenon that may occur as users interact with a recommender system [11]. As a result, in what follows we consider using antidote data in both ways: to either increase or decrease polarization.

As also described in Section 2, unfairness is a topic of growing interest in machine learning. Following the discussion in that section, we consider a recommender system fair if it provides equal quality of service (i.e., prediction accuracy) to all users or all groups of users [36].

Next we formally define the metrics that specify the objective functions associated with each of the above objectives. Since the gradient of each objective function is used in the optimization algorithm, for reproducibility we provide the details about derivation of the gradients in appendix A.2.

5.1 Polarization

To capture polarization, we seek to measure the extent to which the user ratings *disagree*. Thus, to measure user polarization we consider the estimated ratings $\hat{X}$, and we define the polarization metric as the normalized sum of pairwise euclidean distances between estimated user ratings, i.e., between rows of $\hat{X}$. In particular:

$$R_{pol}(\hat{X}) = \frac{1}{n^2 d} \sum_{k=1}^n \sum_{l>k}^n \|\hat{x}^k - \hat{x}^l\|^2 \quad (10)$$

The normalization term $\frac{1}{n^2 d}$ in (10) makes the polarization metric identical to the following definition:⁴

$$R_{pol}(\hat{X}) = \frac{1}{d} \sum_{j=1}^d \sigma_j^2 \quad (11)$$

where σ_j^2 is the variance of estimated user ratings for item j . Thus this polarization metric can be interpreted either as the average of the variances of estimated ratings in each item, or equivalently as the average user disagreement over all items.

5.2 Fairness

Individual fairness. For each user i , we define ℓ_i , the loss of user i , as the mean squared estimation error over known ratings of user i :

$$\ell_i = \frac{\|P_{\Omega_i}(\hat{X}^i - \mathbf{x}^i)\|_2^2}{|\Omega_i|} \quad (12)$$

Then we define the individual unfairness as the variance of the user losses:⁵

$$R_{indv}(\mathbf{X}, \hat{X}) = \frac{1}{n^2} \sum_{k=1}^n \sum_{l>k}^n (\ell_k - \ell_l)^2 \quad (13)$$

To improve individual fairness, we seek to minimize R_{indv} .

Group fairness. Let I be the set of all users/items and $G = \{G_1 \dots G_g\}$ be a partition of users/items into g groups, i.e., $I = \bigcup_{i \in \{1, \dots, g\}} G_i$. We define the loss of group i as the mean squared estimation error over all known ratings in group i :

$$L_i = \frac{\|P_{\Omega_{G_i}}(\hat{X} - \mathbf{X})\|_2^2}{|\Omega_{G_i}|} \quad (14)$$

For a given partition G , we define the group unfairness as the variance of all group losses:

$$R_{grp}(\mathbf{X}, \hat{X}, G) = \frac{1}{g^2} \sum_{k=1}^g \sum_{l>k}^g (L_k - L_l)^2 \quad (15)$$

Again, to improve group fairness, we seek to minimize R_{grp} .

5.3 Accuracy vs. Social Welfare

Adding antidote data to the system to improve a social utility will also have an effect on the overall prediction accuracy. Previous works have considered social objectives as regularizers or constraints added to the recommender model (eg. [8, 25, 37]), implying a trade-off between the prediction accuracy and a social objective.

However, in the case of the metrics we define here, the relationship is not as simple. Considering polarization, we find that in general, increasing or decreasing polarization will tend to decrease system accuracy. In either case we find that system accuracy only declines slightly in our experiments; we report on the specific values in Section 6. Considering either individual or group unfairness, the situation is more subtle. Note that our unfairness metrics will be exactly zero for a system with zero error (perfect accuracy). As a

⁴We can derive it by rewriting (10) as $R_{pol}(\hat{X}) = \frac{1}{d} \sum_{j=1}^d \frac{1}{n^2} \sum_{k=1}^n \sum_{l>k}^n (\hat{x}_{kj} - \hat{x}_{lj})^2$.

⁵Note that for a set of equally likely values $x_1, \dots, x_n$ the variance can be expressed without referring to the mean as: $\frac{1}{n^2} \sum_{i=1}^n \sum_{j>i}^n (x_i - x_j)^2$.

Summary

- Recommender systems solve many **real-world** (*large-scale) **problems**
- Collaborative filtering by Matrix Factorization (MF) is an **efficient** and **effective** approach
- MF is just another example of a **common recipe**:
 1. define a model
 2. define an objective function
 3. optimize with your favorite black box optimizer
(e.g. SGD, Gradient Descent, Block Coordinate Descent aka. Alternating Least Squares)

Learning Objectives

Recommender Systems

You should be able to...

1. Compare and contrast the properties of various families of recommender system algorithms: content filtering, collaborative filtering, neighborhood methods, latent factor methods
2. Formulate a squared error objective function for the matrix factorization problem
3. Implement unconstrained matrix factorization with a variety of different optimization techniques: gradient descent, stochastic gradient descent, alternating least squares
4. Offer intuitions for why the parameters learned by matrix factorization can be understood as user factors and item factors

Recommender Systems

NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update

Leaderboard

Showing Test Score. [Click here to show quiz score](#)

Top performing systems were ensembles

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
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