

MATRIX FACTORIZATION

&

RECOMMENDER SYSTEMS

Outline


- **Recommender Systems**
 - Content Filtering
 - Collaborative Filtering
 - CF: Neighborhood Methods
 - CF: Latent Factor Methods
- **Matrix Factorization**
 - User / item vectors
 - Prediction model
 - Training by SGD
- **Extra: Matrix Multiplication in ML**
 - Matrix Factorization
 - Linear Regression
 - PCA
 - (Autoencoders)
 - K-means

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



All ▾


Q

CYBER MONDAY DEALS WEEK

Hello, Matt
Your Account ▾ Prime ▾ Lists ▾  Cart

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

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



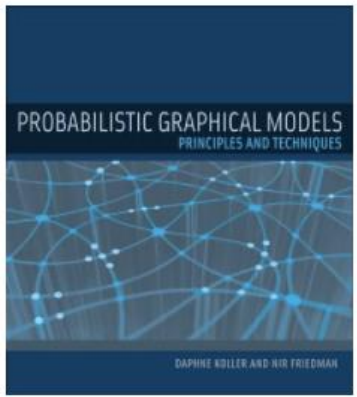
Buy It Again in Grocery
14 ITEMS



Buy It Again in Pets
6 ITEMS



Buy It Again in Baby Products
5 ITEMS



Engineering Books
86 ITEMS

NETFLIX

COMPLETED

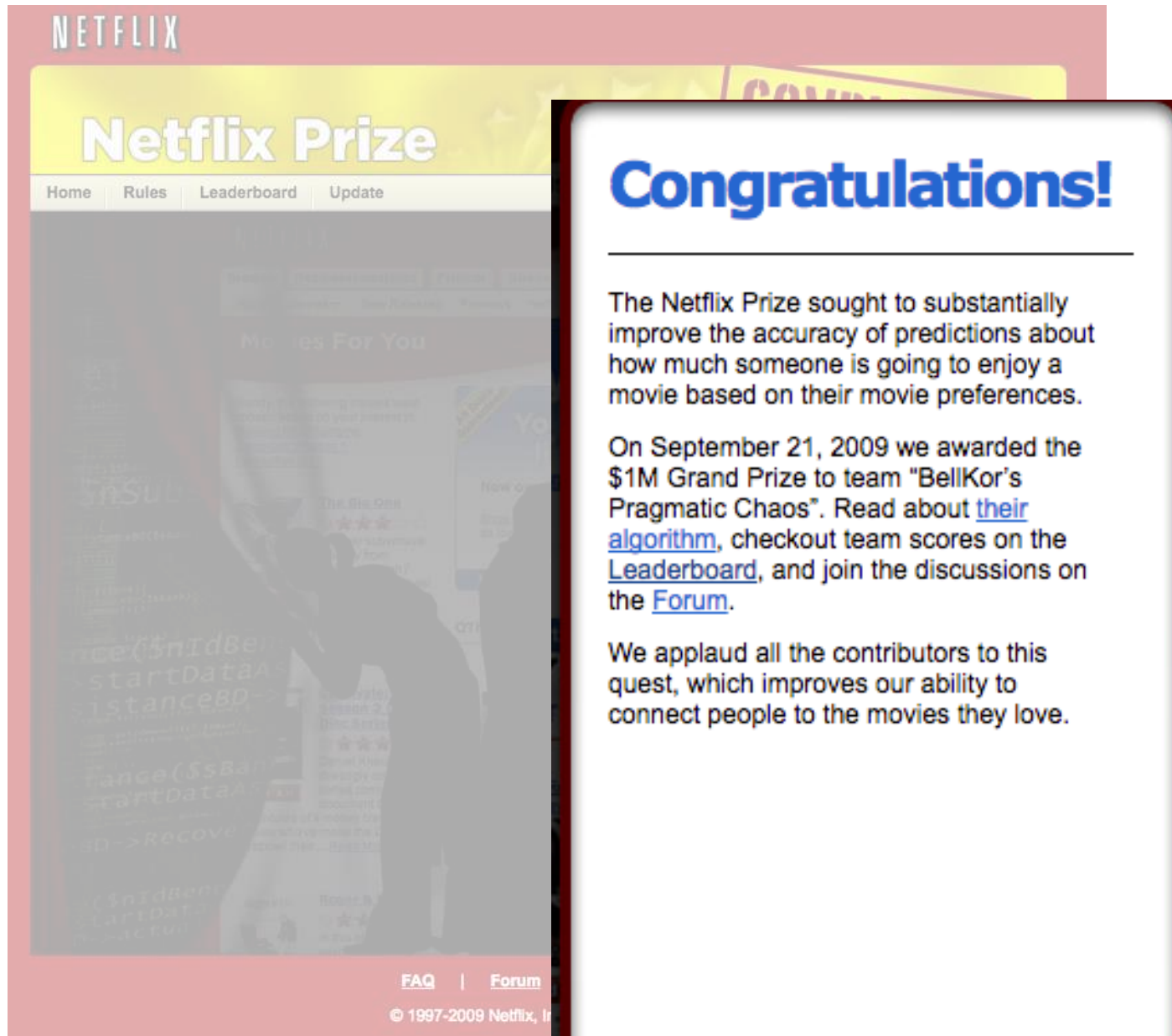
Congratulations!

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.

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

Recommender Systems



The image shows a screenshot of the Netflix Prize website. The top navigation bar includes links for Home, Rules, Leaderboard, and Update. The main heading is "Netflix Prize". Below this, there is a section titled "Movies For You" which displays a list of movie recommendations. A large, semi-transparent overlay box is positioned on the right side of the page, containing a congratulatory message. The message states that the \$1M Grand Prize was awarded to the team "BellKor's Pragmatic Chaos" on September 21, 2009. It encourages visitors to read about the team's algorithm, check out team scores on the Leaderboard, and join discussions on the Forum. The footer of the page includes links for FAQ and Forum, and a copyright notice for 1997-2009 Netflix, Inc.

NETFLIX

Netflix Prize

Home Rules Leaderboard Update

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.

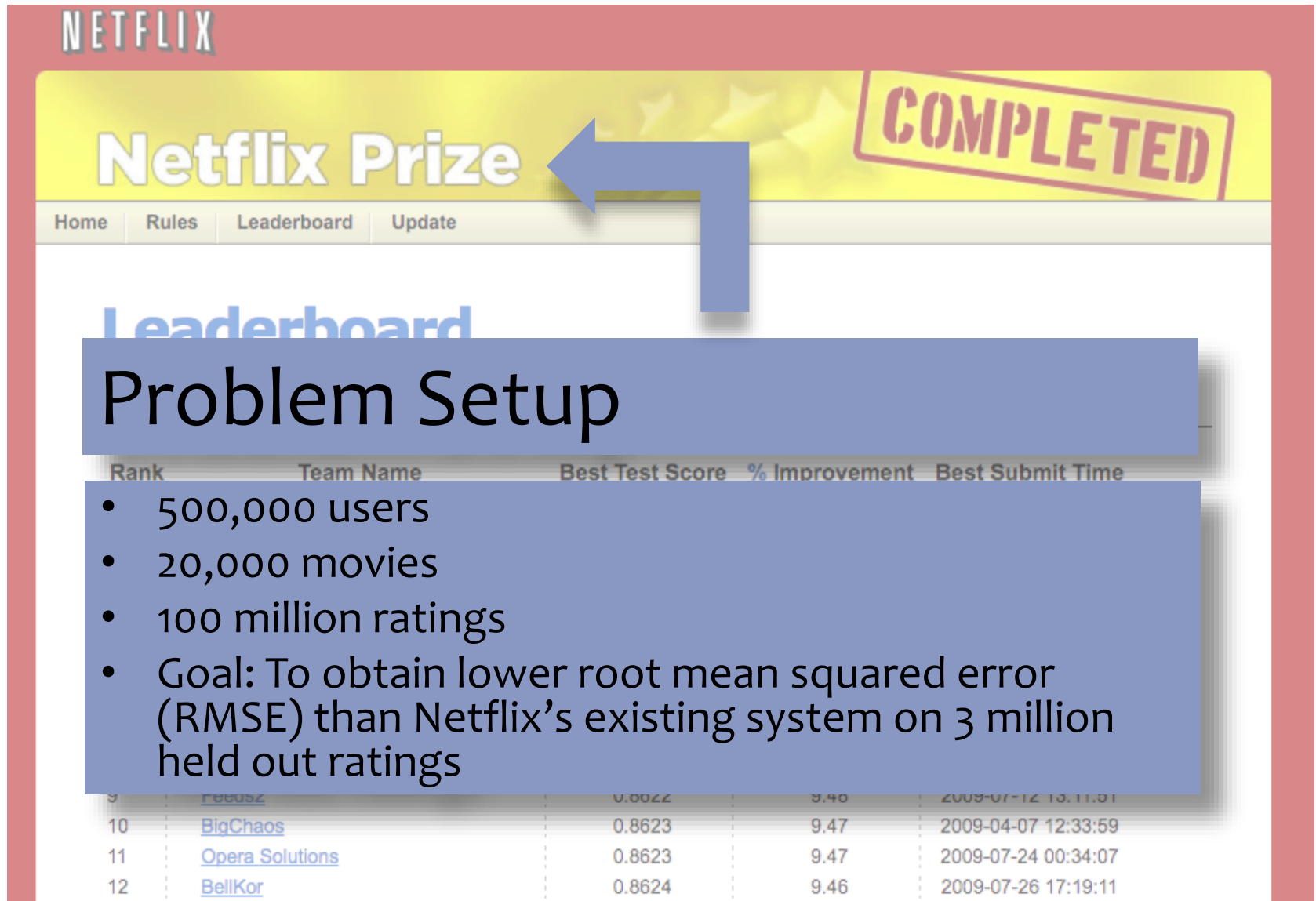
On September 21, 2009 we awarded the \$1M Grand Prize to team "BellKor's Pragmatic Chaos". Read about [their algorithm](#), checkout team scores on the [Leaderboard](#), and join the discussions on the [Forum](#).

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

FAQ | Forum

© 1997-2009 Netflix, Inc.

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	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

Recommender Systems



Netflix Prize



[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

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

Two Types of Recommender Systems

Content Filtering

- *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

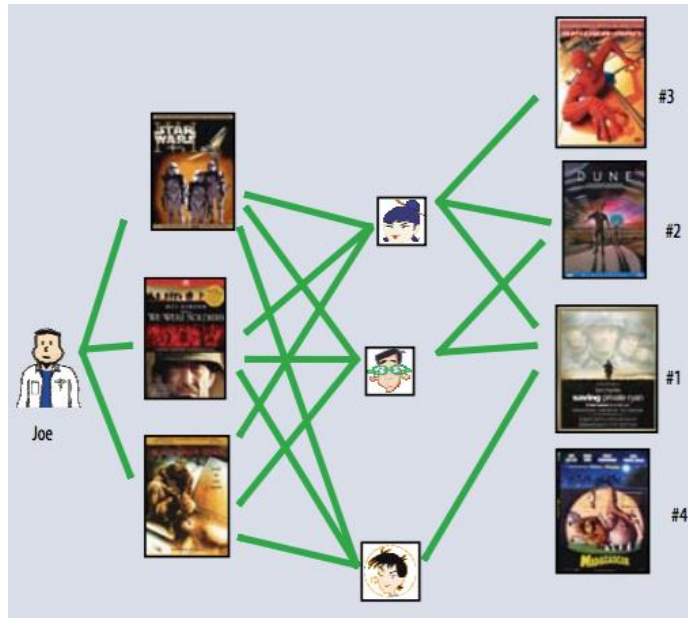
- *Example:* **Netflix** movie recommendations
- **Pro:** Does not assume access to **side information** about items (e.g. does not need to know about movie genres)
- **Con:** Does not work on **new items** that have no ratings

Collaborative Filtering

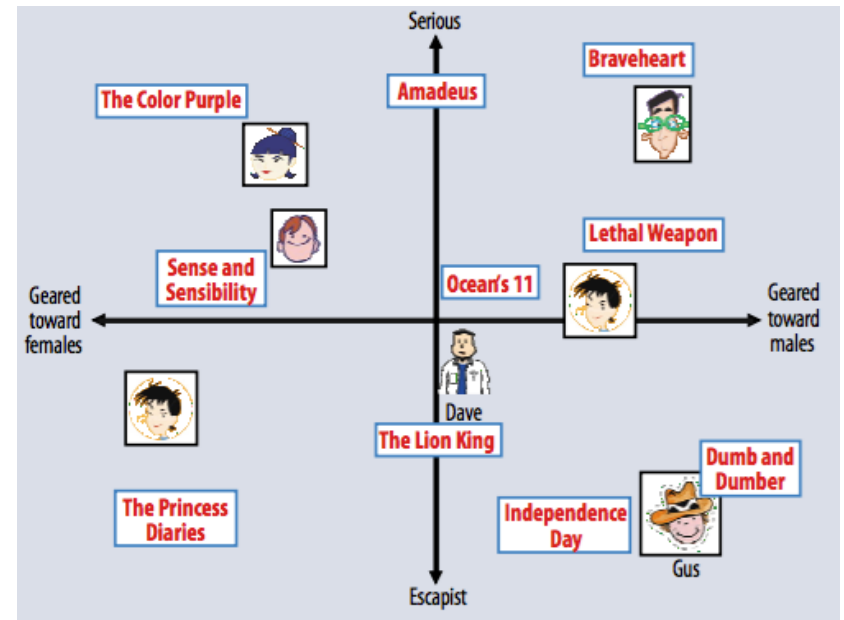
- **Everyday Examples of Collaborative Filtering...**
 - Bestseller lists
 - Top 40 music lists
 - The “recent returns” shelf at the library
 - Unmarked but well-used paths thru the woods
 - The printer room at work
 - “Read any good books lately?”
 - ...
- **Common insight:** personal tastes are correlated
 - 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

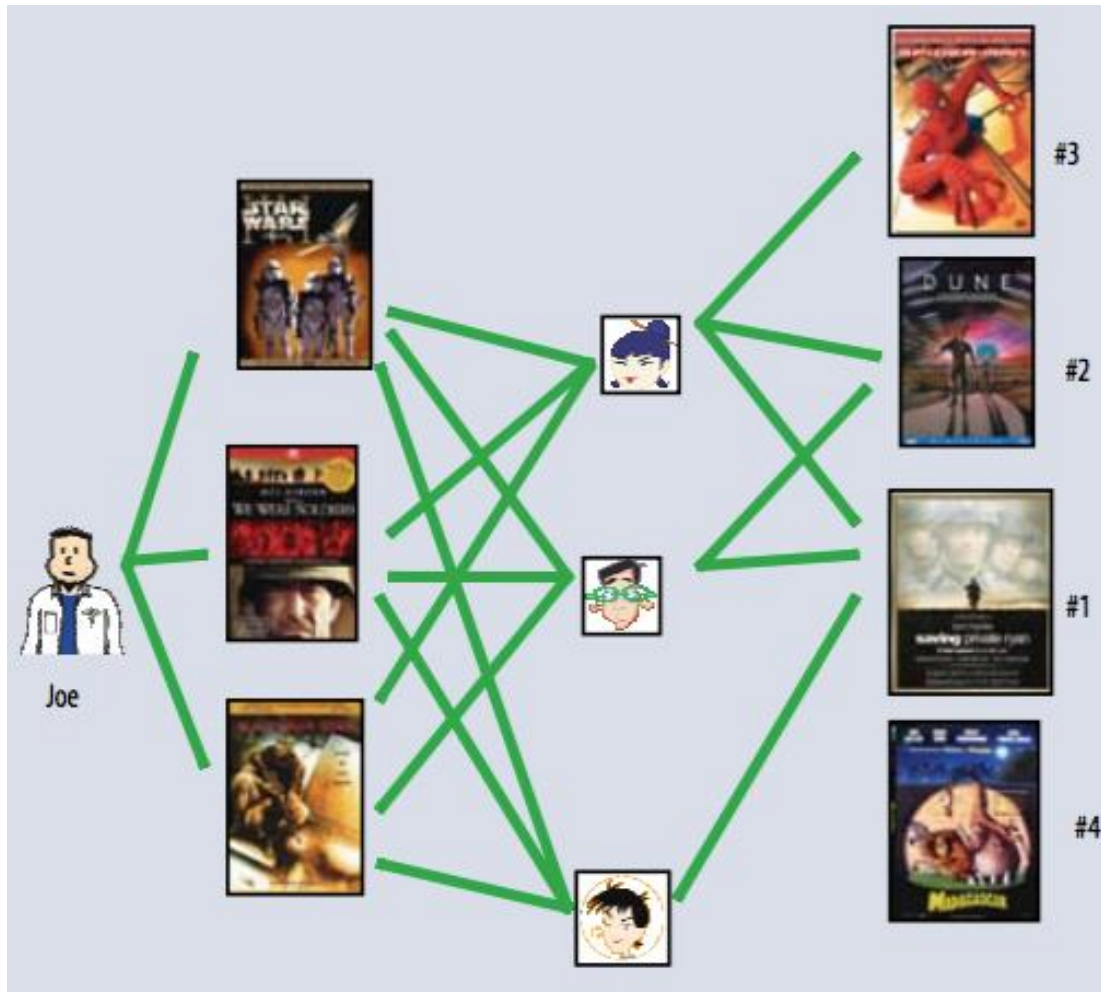


2. Latent Factor Methods



Two Types of Collaborative Filtering

1. Neighborhood Methods



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

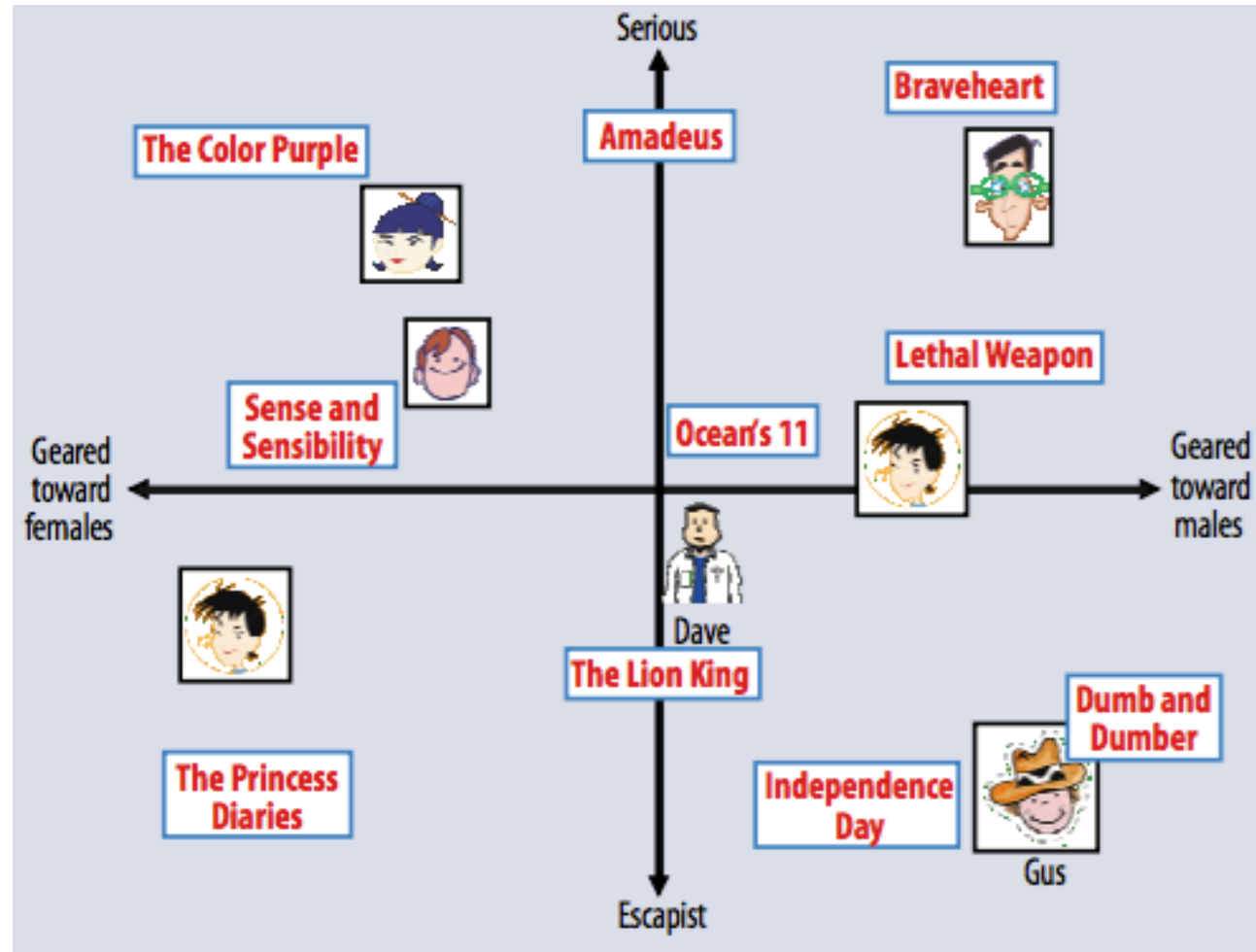
Algorithm:

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



MATRIX FACTORIZATION

Matrix Factorization (with matrices)

- User vectors:

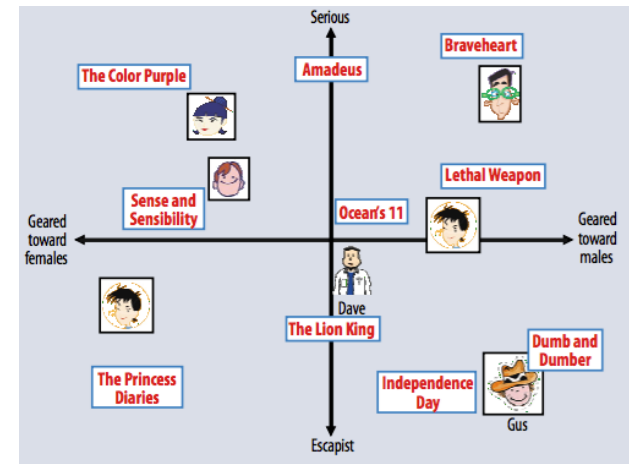
$$(W_{u*})^T \in \mathbb{R}^r$$

- Item vectors:

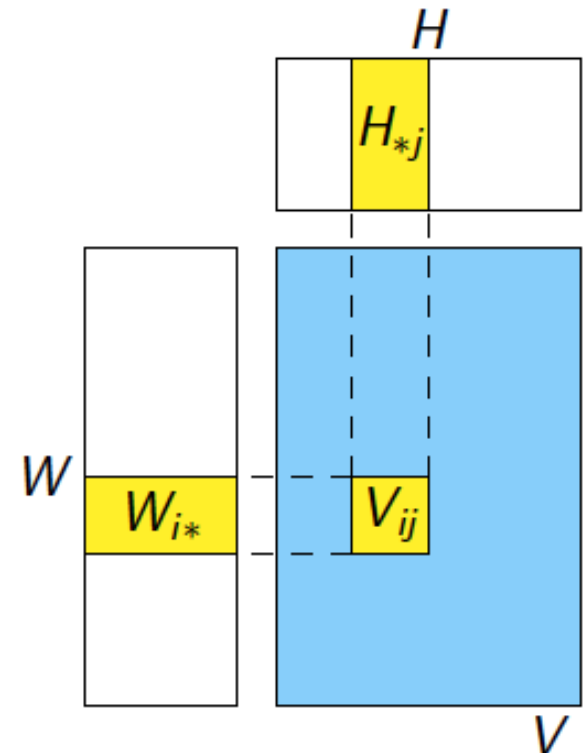
$$H_{*i} \in \mathbb{R}^r$$

- Rating prediction:

$$\begin{aligned} V_{ui} &= W_{u*} H_{*i} \\ &= [WH]_{ui} \end{aligned}$$



Figures from Koren et al. (2009)



Figures from Gemulla et al. (2011)₁₇

Matrix Factorization (with vectors)

- User vectors:

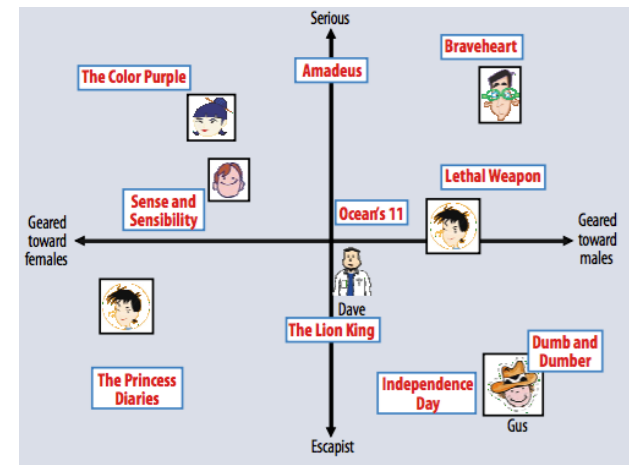
$$\mathbf{w}_u \in \mathbb{R}^r$$

- Item vectors:

$$\mathbf{h}_i \in \mathbb{R}^r$$

- Rating prediction:

$$v_{ui} = \mathbf{w}_u^T \mathbf{h}_i$$



Figures from Koren et al. (2009)

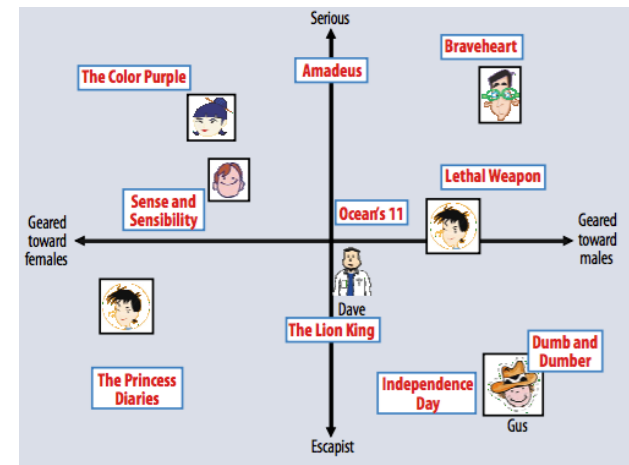
Matrix Factorization (with vectors)

- Set of non-zero entries:

$$\mathcal{Z} = \{(u, i) : v_{ui} \neq 0\}$$

- Objective:

$$\operatorname{argmin}_{\mathbf{w}, \mathbf{h}} \sum_{(u, i) \in \mathcal{Z}} (v_{ui} - \mathbf{w}_u^T \mathbf{h}_i)^2$$

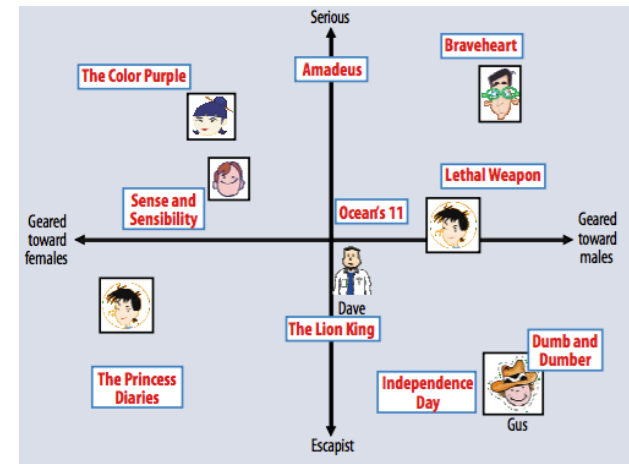


Figures from Koren et al. (2009)

Matrix Factorization (with vectors)

- Regularized Objective:

$$\begin{aligned} \operatorname{argmin}_{\mathbf{w}, \mathbf{h}} \sum_{(u,i) \in \mathcal{Z}} (v_{ui} - \mathbf{w}_u^T \mathbf{h}_i)^2 \\ + \lambda \left(\sum_i \|\mathbf{w}_i\|^2 + \sum_u \|\mathbf{h}_u\|^2 \right) \end{aligned}$$



Figures from Koren et al. (2009)

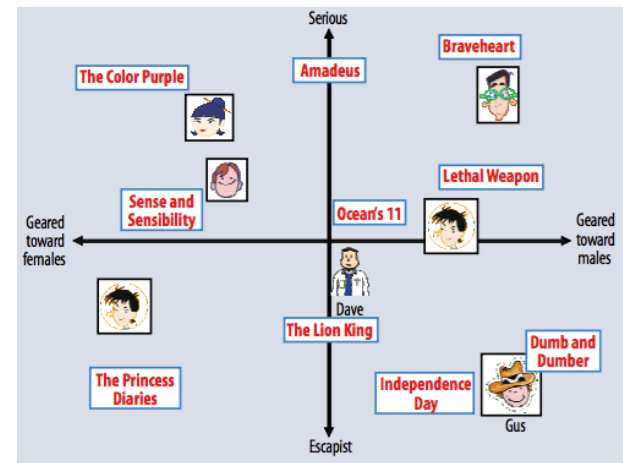
Matrix Factorization (with vectors)

- Regularized Objective:

$$\begin{aligned} \operatorname{argmin}_{\mathbf{w}, \mathbf{h}} \sum_{(u,i) \in \mathcal{Z}} (v_{ui} - \mathbf{w}_u^T \mathbf{h}_i)^2 \\ + \lambda \left(\sum_i \|\mathbf{w}_i\|^2 + \sum_u \|\mathbf{h}_u\|^2 \right) \end{aligned}$$

- Stochastic Gradient Descent (SGD) update for random (u,i):

$$\begin{aligned} e_{ui} &\leftarrow v_{ui} - \mathbf{w}_u^T \mathbf{h}_i \\ \mathbf{w}_u &\leftarrow \mathbf{w}_u + \gamma(e_{ui} \mathbf{h}_i - \lambda \mathbf{w}_u) \\ \mathbf{h}_i &\leftarrow \mathbf{h}_i + \gamma(e_{ui} \mathbf{w}_u - \lambda \mathbf{h}_i) \end{aligned}$$



Figures from Koren et al. (2009)

Matrix Factorization (with matrices)

- User vectors:

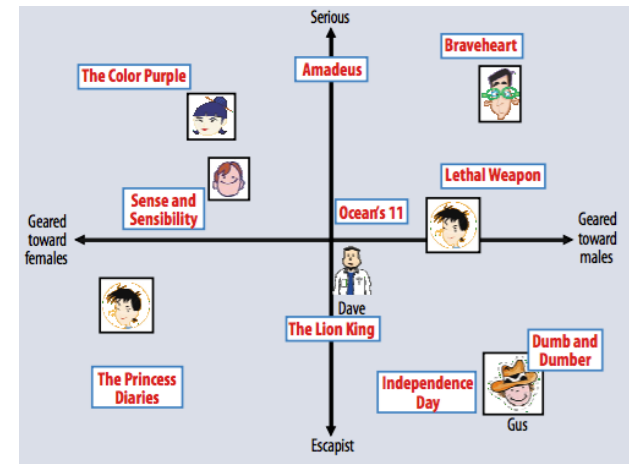
$$(W_{u*})^T \in \mathbb{R}^r$$

- Item vectors:

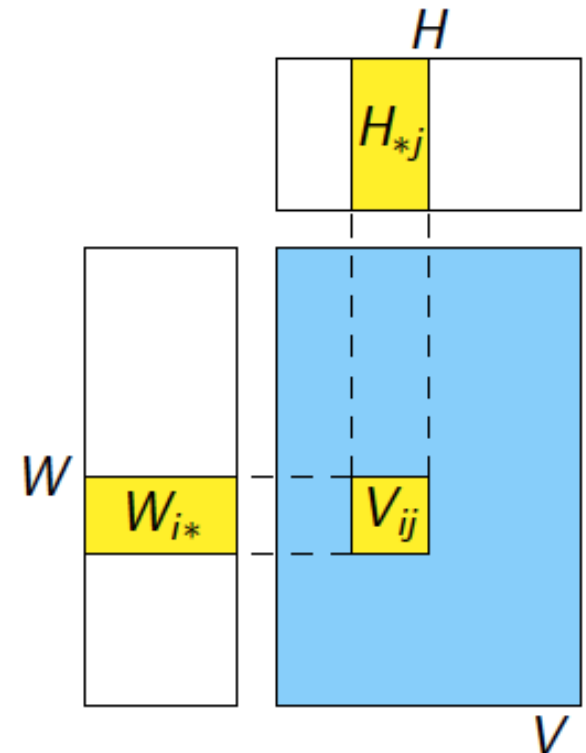
$$H_{*i} \in \mathbb{R}^r$$

- Rating prediction:

$$\begin{aligned} V_{ui} &= W_{u*} H_{*i} \\ &= [WH]_{ui} \end{aligned}$$



Figures from Koren et al. (2009)



Figures from Gemulla et al. (2011)₂₂

Matrix Factorization (with matrices)

- SGD (Stochastic Gradient Descent)

require that the loss can be written as

$$L = \sum_{(i,j) \in Z} l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j})$$

Algorithm 1 SGD for Matrix Factorization

Require: A training set Z , initial values \mathbf{W}_0 and \mathbf{H}_0

while not converged **do** {step}

 Select a training point $(i, j) \in Z$ uniformly at random.

$\mathbf{W}'_{i*} \leftarrow \mathbf{W}_{i*} - \epsilon_n N \frac{\partial}{\partial \mathbf{W}_{i*}} l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j})$

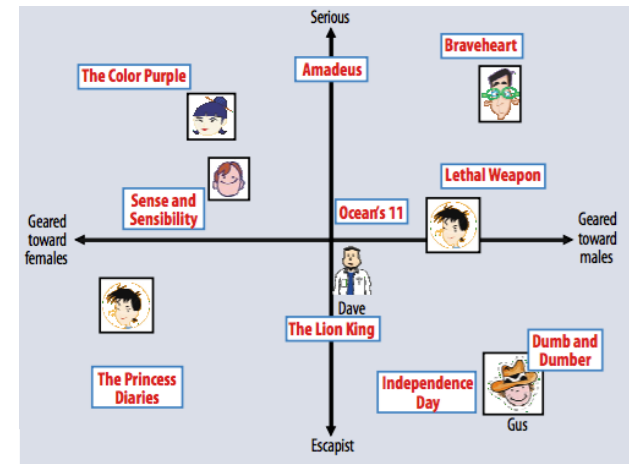
$\mathbf{H}_{*j} \leftarrow \mathbf{H}_{*j} - \epsilon_n N \frac{\partial}{\partial \mathbf{H}_{*j}} l(\mathbf{V}_{ij}, \mathbf{W}_{i*}, \mathbf{H}_{*j})$

$\mathbf{W}_{i*} \leftarrow \mathbf{W}'_{i*}$

end while

ϵ_n step size

Figure from Gemulla et al. (2011)



Figures from Koren et al. (2009)

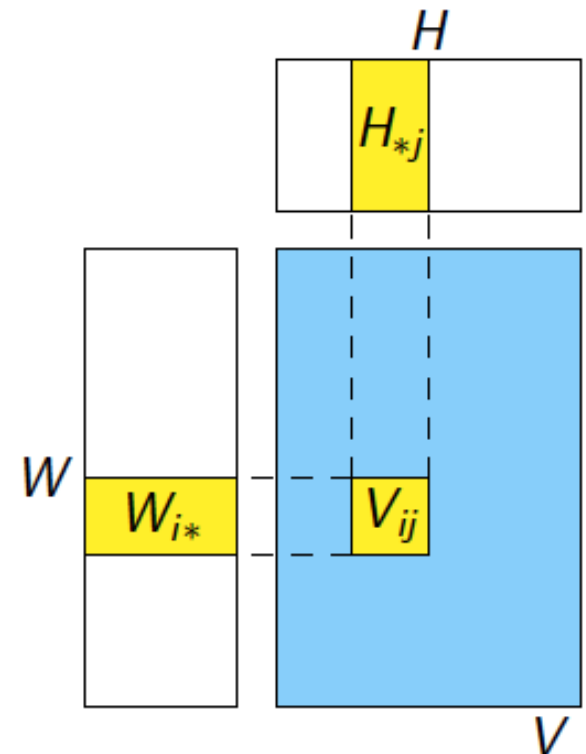


Figure from Gemulla et al. (2011)₂₃

Matrix Factorization

Example Factors

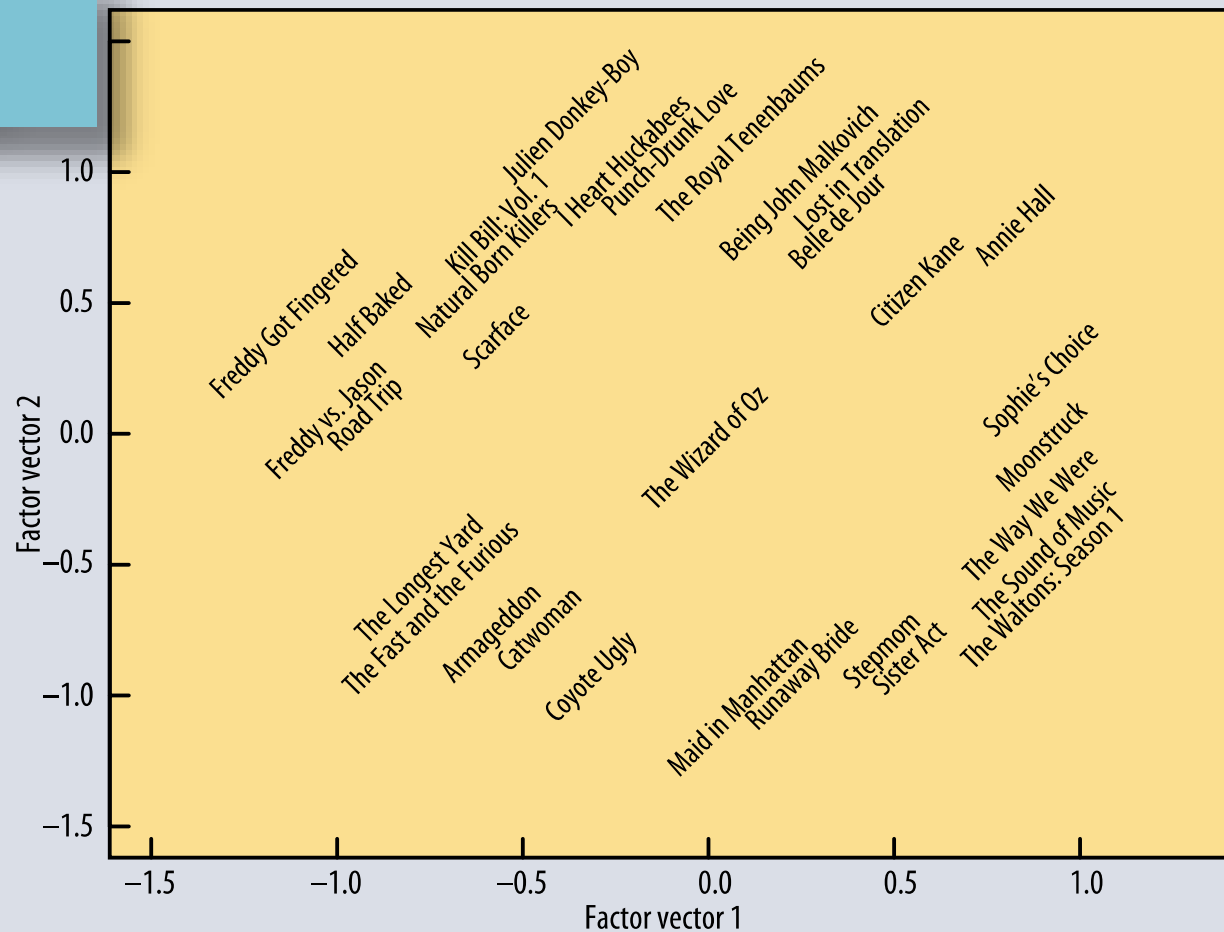
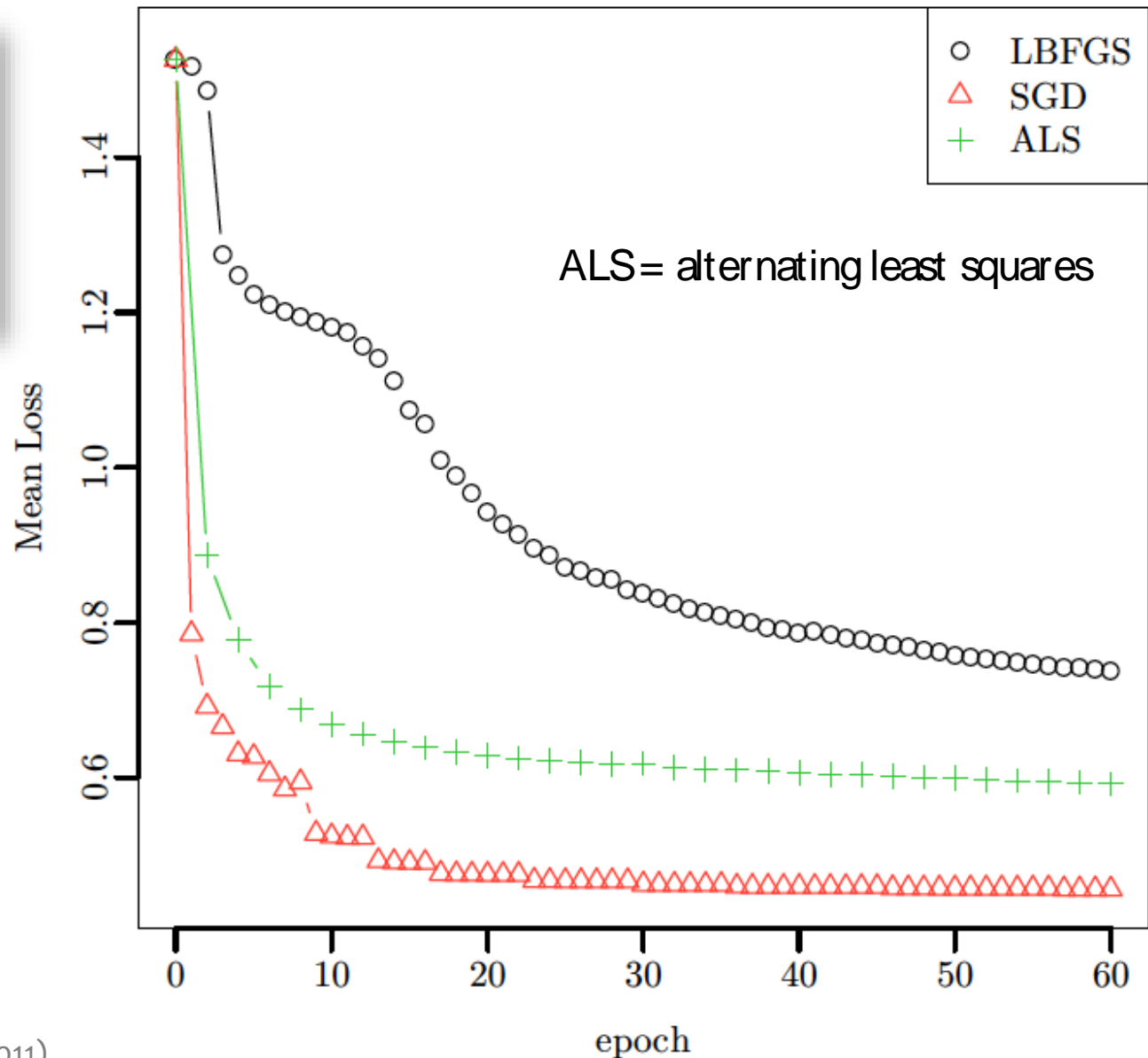


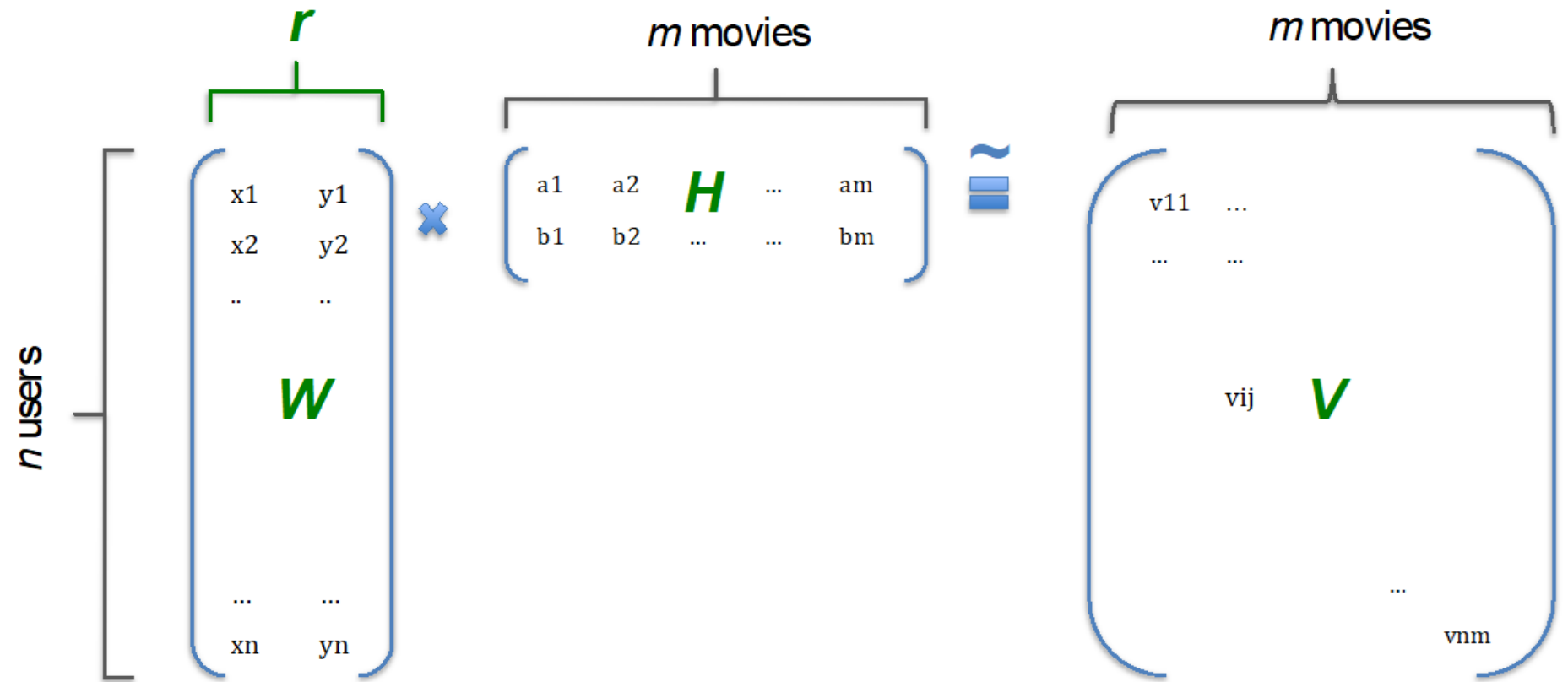
Figure 3. The first two vectors from a matrix decomposition of the Netflix Prize data. Selected movies are placed at the appropriate spot based on their factor vectors in two dimensions. The plot reveals distinct genres, including clusters of movies with strong female leads, fraternity humor, and quirky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

Matrix Factorization

Comparison of Optimization Algorithms

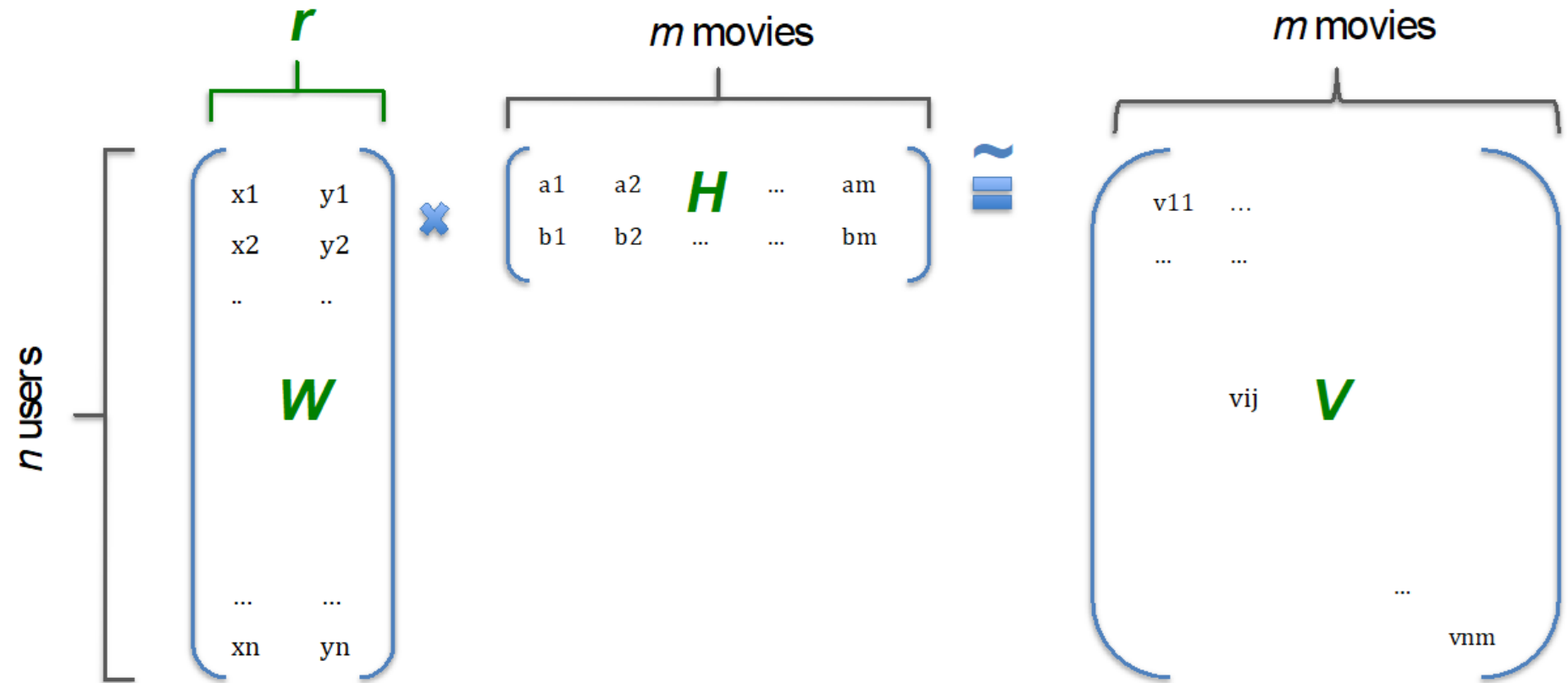


MATRIX MULTIPLICATION IN MACHINE LEARNING



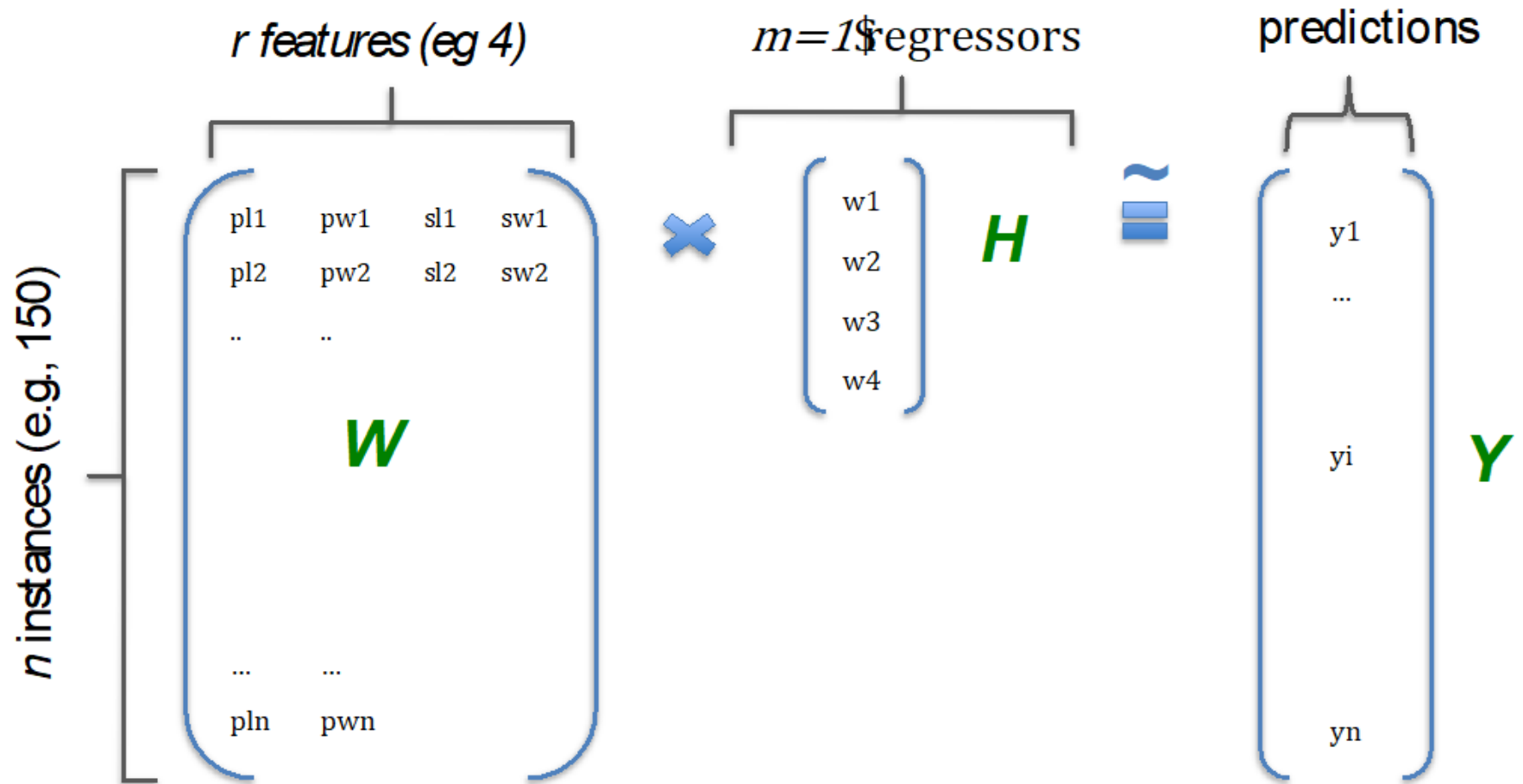
$V[i,j]$ = user i 's rating of movie j

Recovering Latent Factors

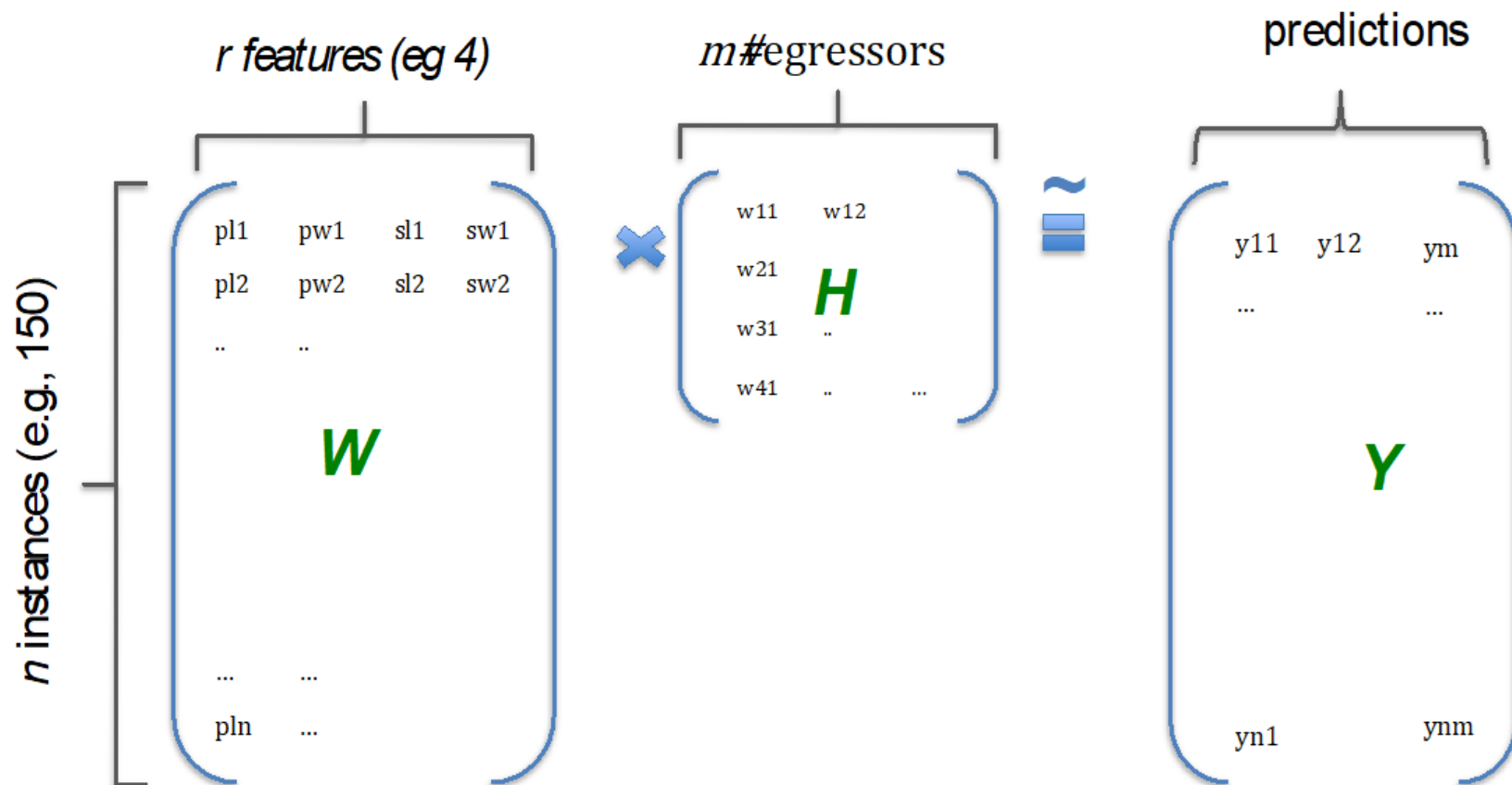


$V[i,j]$ = user i 's rating of movie j

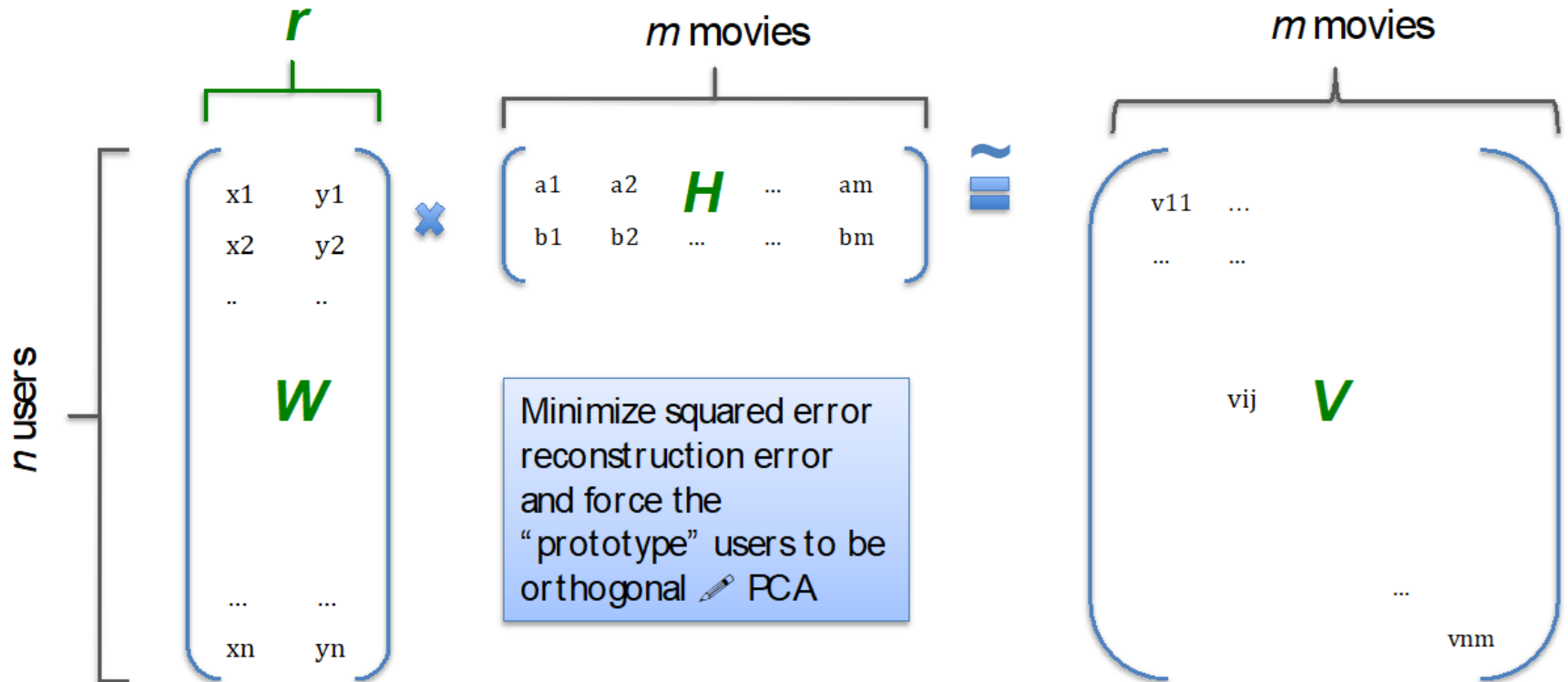
Is like Regression



Many Output at Once



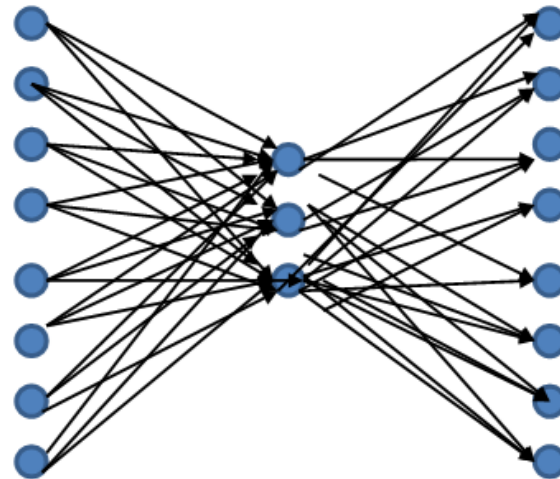
Similar like PCA



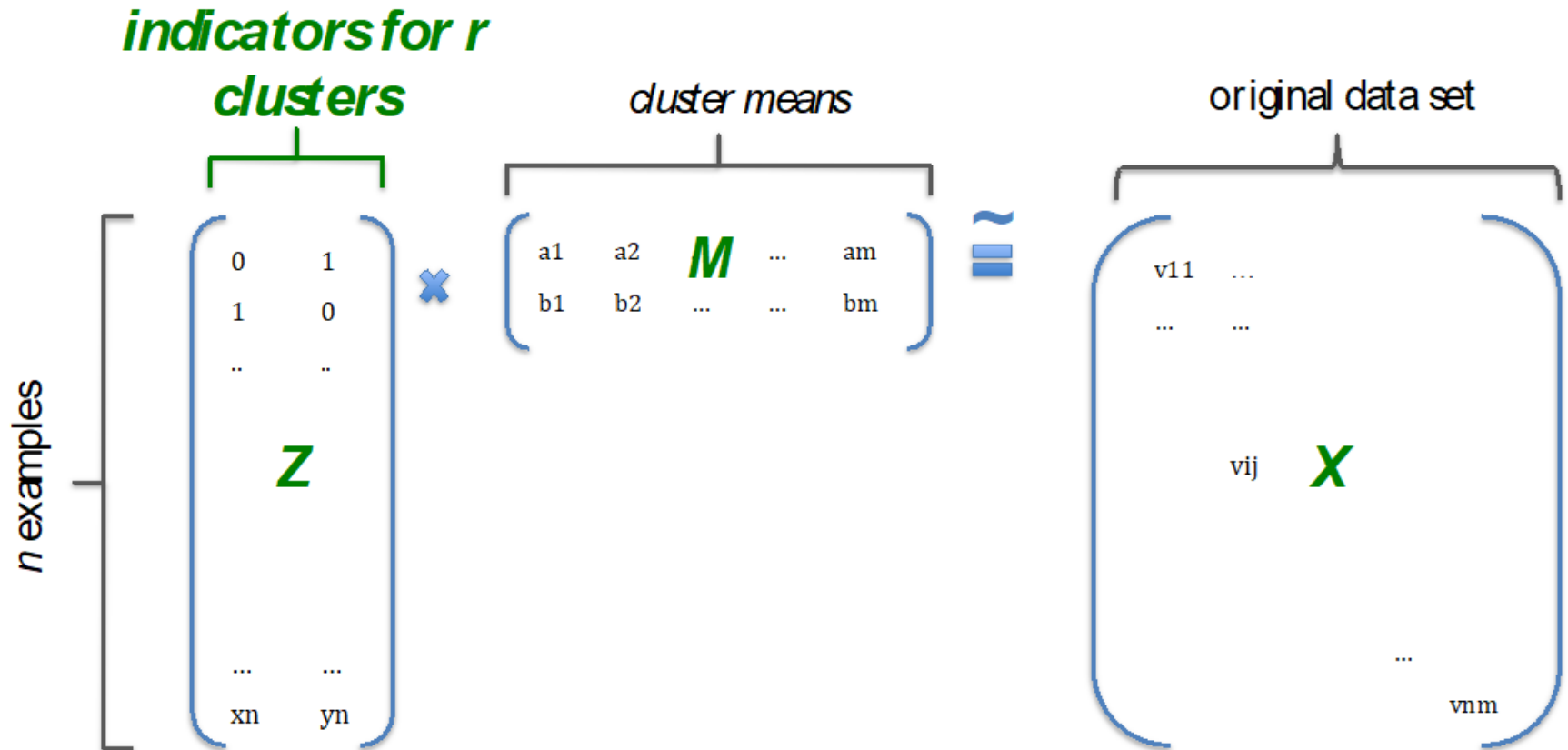
Auto-encoder and Non-Linear PCA

- Assume we would like to learn the following (trivial?) output function:
- Using the following network:
- With *linear* hidden units, how do the weights match up to W and H ?

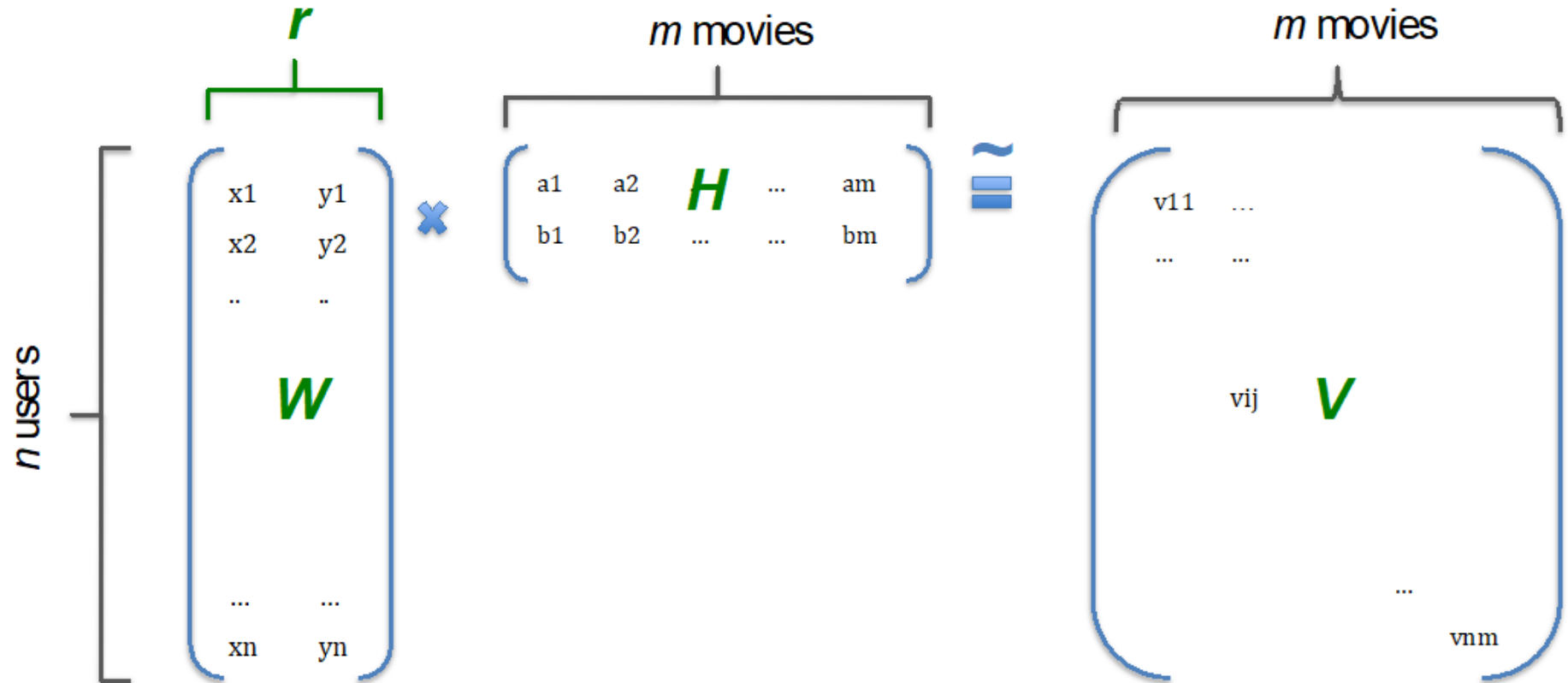
Input	Output
00000001	00000001
00000010	00000010
00000101	00000100
00001000	00001000
00010000	00010000
00100000	00100000
01000000	01000000
10000000	10000000



Like K-Means Clustering



Recovering Latent Factors in Matrix



$V[i,j]$ = user i 's rating of movie j

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 SGD