

Information Storage and Retrieval

CSCE 670
Texas A&M University
Department of Computer Science & Engineering
Instructor: Prof. James Caverlee

Latent Factor Models
28/30 March 2017

Slides adapted from Jure Leskovec, Yehuda Koren, and others

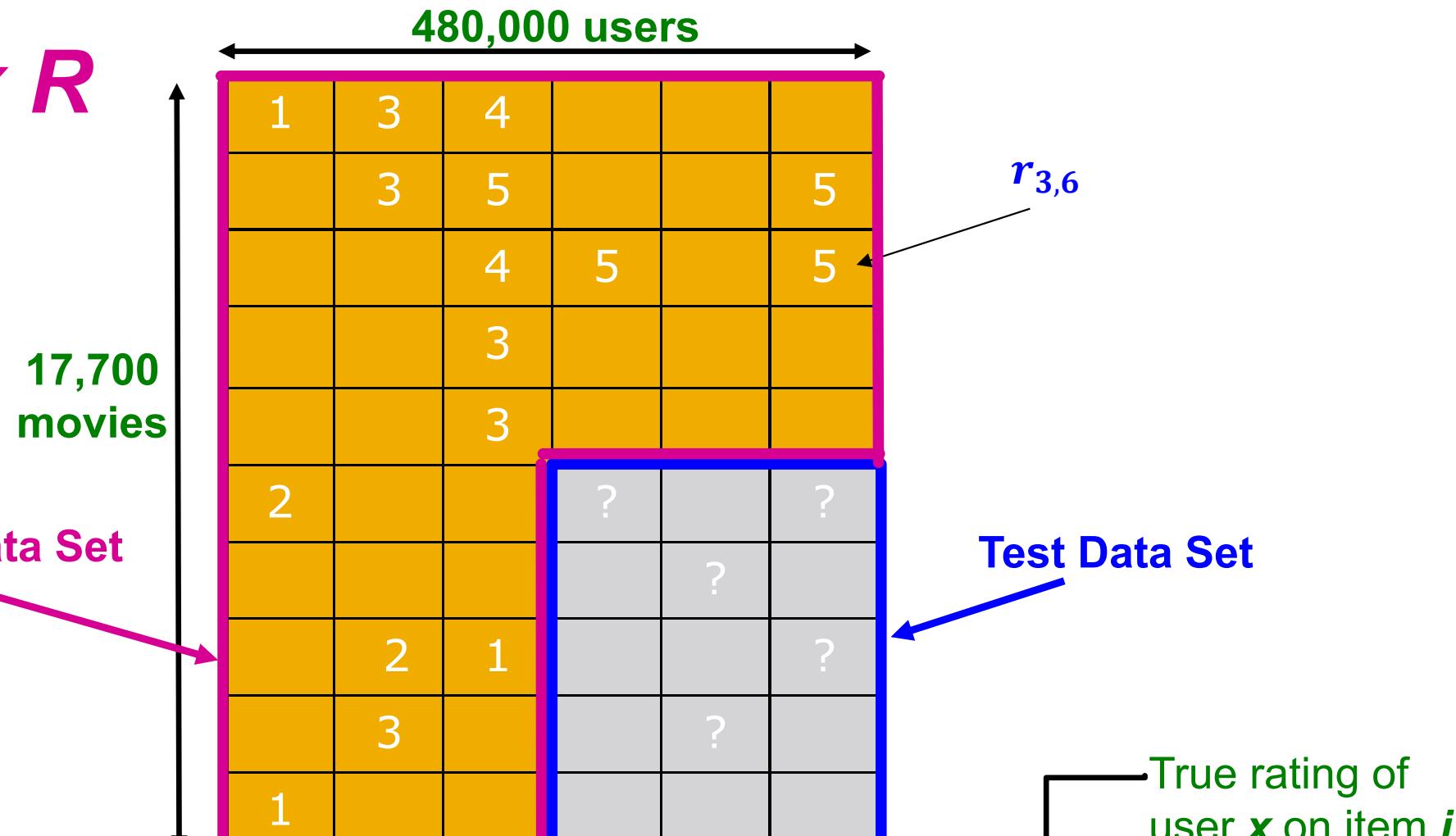
The Netflix Utility Matrix R

Matrix R

480,000 users					
17,700 movies	1	3	4		
		3	5		5
			4	5	5
				3	
				3	
	2			2	2
					5
		2	1		1
		3			3
	1				

Utility Matrix R: Evaluation

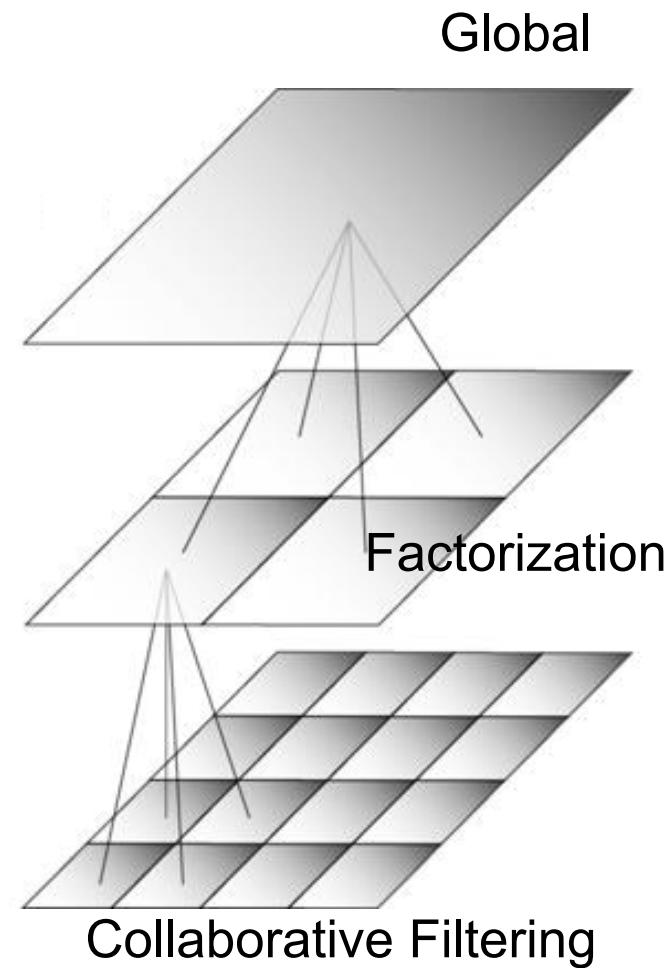
Matrix R



$$\text{RMSE} = \frac{1}{|R|} \sqrt{\sum_{(i,x) \in R} (\hat{r}_{xi} - r_{xi})^2}$$

BellKor Recommender System

- The winner of the Netflix Challenge
- Multi-scale modeling of the data
 - **Global:**
 - Overall deviations of users/movies
 - **Factorization:**
 - Addressing “regional” effects
 - **Collaborative Filtering:**
 - Extract local patterns



Modeling Local and Global Effects

- **Global:**

- Mean movie rating: **3.7 stars**
- *The Sixth Sense* is **0.5** stars above average
- Joe rates **0.2** stars below average
- => Baseline estimate: Joe will rate *The Sixth Sense* **4 stars**

- **Local Neighborhood (Collaborative Filtering)**

- Joe didn't like related movie *Signs*
- =>Final estimate: Joe will rate *The Sixth Sense* **3.8 stars**

Recap: Collaborative Filtering

- Earliest and most popular **collaborative filtering method**
- Derive unknown ratings from those of “similar” movies (item-item variant)
- Define **similarity measure s_{ij}** of items i and j
- Select k -nearest neighbors, compute the rating
 - **$N(i; x)$:** items most similar to i that were rated by x

$$\hat{r}_{xi} = \frac{\sum_{j \in N(i; x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i; x)} s_{ij}}$$

s_{ij} similarity of items i and j
 r_{xj} rating of user x on item j
 $N(i; x)$ set of items similar to item i that were rated by x

Modeling Local and Global Effects

- In practice we get better estimates if we model deviations:

$$\hat{r}_{xi} = b_{xi} + \frac{\sum_{j \in N(i; x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i; x)} s_{ij}}$$

baseline estimate for r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

μ = overall mean rating

b_x = rating deviation of user x

= (avg. rating of user x) – μ

b_i = (avg. rating of movie i) – μ

Problems/Issues:

- 1) Similarity measures are “arbitrary”
- 2) Pairwise similarities neglect interdependencies among users
- 3) Taking a weighted average can be restricting

Solution: Instead of s_{ij} use w_{ij} that we estimate directly from data

Idea: Interpolation Weights

- Use a **weighted sum** rather than **weighted avg.**:

$$\widehat{r}_{xi} = b_{xi} + \sum_{j \in N(i; x)} w_{ij} (r_{xj} - b_{xj})$$

- **A few notes:**

- $N(i; x)$... set of movies rated by user x that are similar to movie i
- w_{ij} is the interpolation weight (some real number)
 - We allow: $\sum_{j \in N(i, x)} w_{ij} \neq 1$
- w_{ij} models interaction between pairs of movies (it does not depend on user x)

- $\hat{r}_{xi} = b_{xi} + \sum_{j \in N(i,x)} w_{ij} (r_{xj} - b_{xj})$
- How to set w_{ij} ?

- Remember, error metric is: $\frac{1}{|R|} \sqrt{\sum_{(i,x) \in R} (\hat{r}_{xi} - r_{xi})^2}$
or equivalently SSE: $\sum_{(i,x) \in R} (\hat{r}_{xi} - r_{xi})^2$
- Find w_{ij} that minimize SSE on training data!
 - Models relationships between item i and its neighbors j
- w_{ij} can be learned/estimated based on x and all other users that rated i

Why is this a good idea?

Recommendations via Optimization

- **Goal: Make good recommendations**

- Quantify goodness using **RMSE**:
Lower RMSE \Rightarrow better recommendations
- Want to make good recommendations on items that user has not yet seen. **Can't really do this!**
- **Let's set build a system such that it works well on known (user, item) ratings**
And **hope** the system will also predict well the **unknown ratings**

1	3	4		
3	5		5	
	4	5	5	
	3			
	3			
2		2	2	2
			5	
	2	1		1
	3		3	
1				

Recommendations via Optimization

- Idea: Let's set values w such that they work well on known (user, item) ratings
- How to find such values w ?
- Idea: Define an objective function and solve the optimization problem
- Find w_{ij} that minimize SSE on training data!

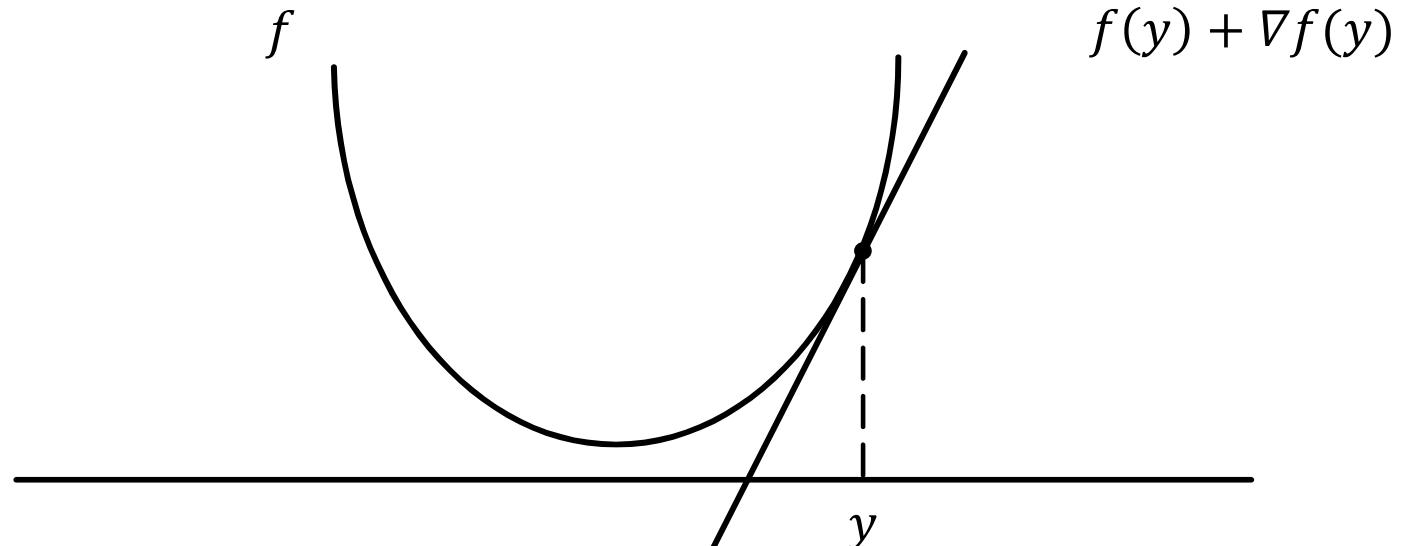
$$J(w) = \sum_{x,i} \left(\underbrace{\left[b_{xi} + \sum_{j \in N(i;x)} w_{ij}(r_{xj} - b_{xj}) \right]}_{\text{Predicted rating}} - r_{xi} \right)^2$$

True rating

- Think of w as a vector of numbers

Detour: Minimizing a function

- **A simple way to minimize a function $f(x)$:**
 - Compute the take a derivative ∇f
 - Start at some point y and evaluate $\nabla f(y)$
 - Make a step in the reverse direction of the gradient: $y = y - \nabla f(y)$
 - Repeat until converged



Interpolation Weights

- We have the optimization problem, now what?

$$J(w) = \sum_x \left(\left[b_{xi} + \sum_{j \in N(i;x)} w_{ij} (r_{xj} - b_{xj}) \right] - r_{xi} \right)^2$$

- Gradient decent:

- Iterate until convergence: $w \leftarrow w - \eta \nabla_w J$ η ... learning rate

- where $\nabla_w J$ is the gradient (derivative evaluated on data):

$$\nabla_w J = \left[\frac{\partial J(w)}{\partial w_{ij}} \right] = 2 \sum_{x,i} \left(\left[b_{xi} + \sum_{k \in N(i;x)} w_{ik} (r_{xk} - b_{xk}) \right] - r_{xi} \right) (r_{xj} - b_{xj})$$

for $j \in \{N(i; x), \forall i, \forall x\}$

else $\frac{\partial J(w)}{\partial w_{ij}} = \mathbf{0}$

- Note: We fix movie i , go over all r_{xi} , for every movie $j \in N(i; x)$, we compute $\frac{\partial J(w)}{\partial w_{ij}}$

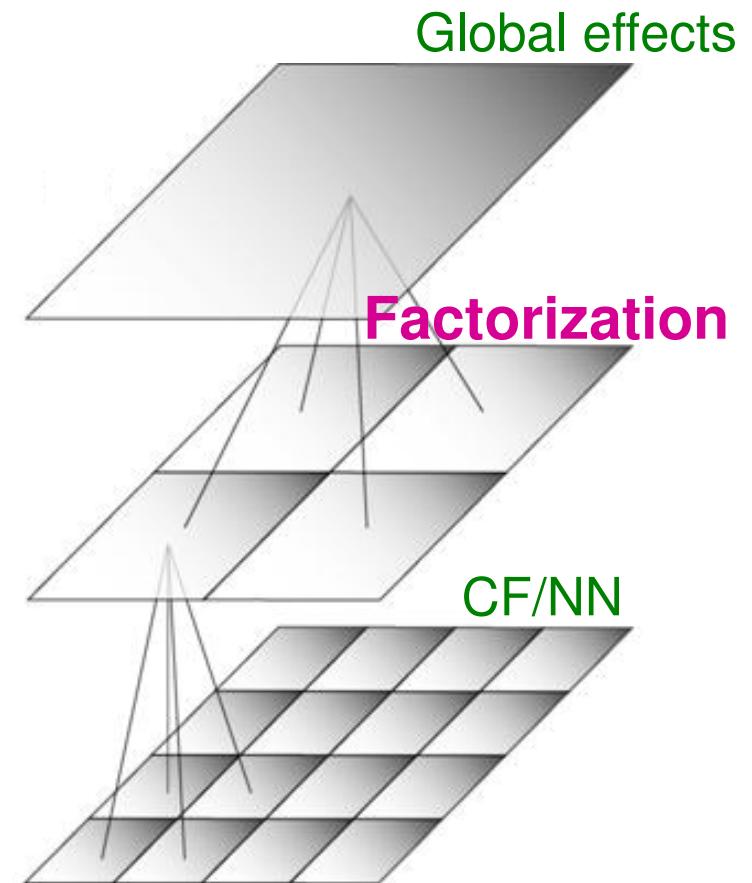
while $|w_{new} - w_{old}| > \varepsilon$:

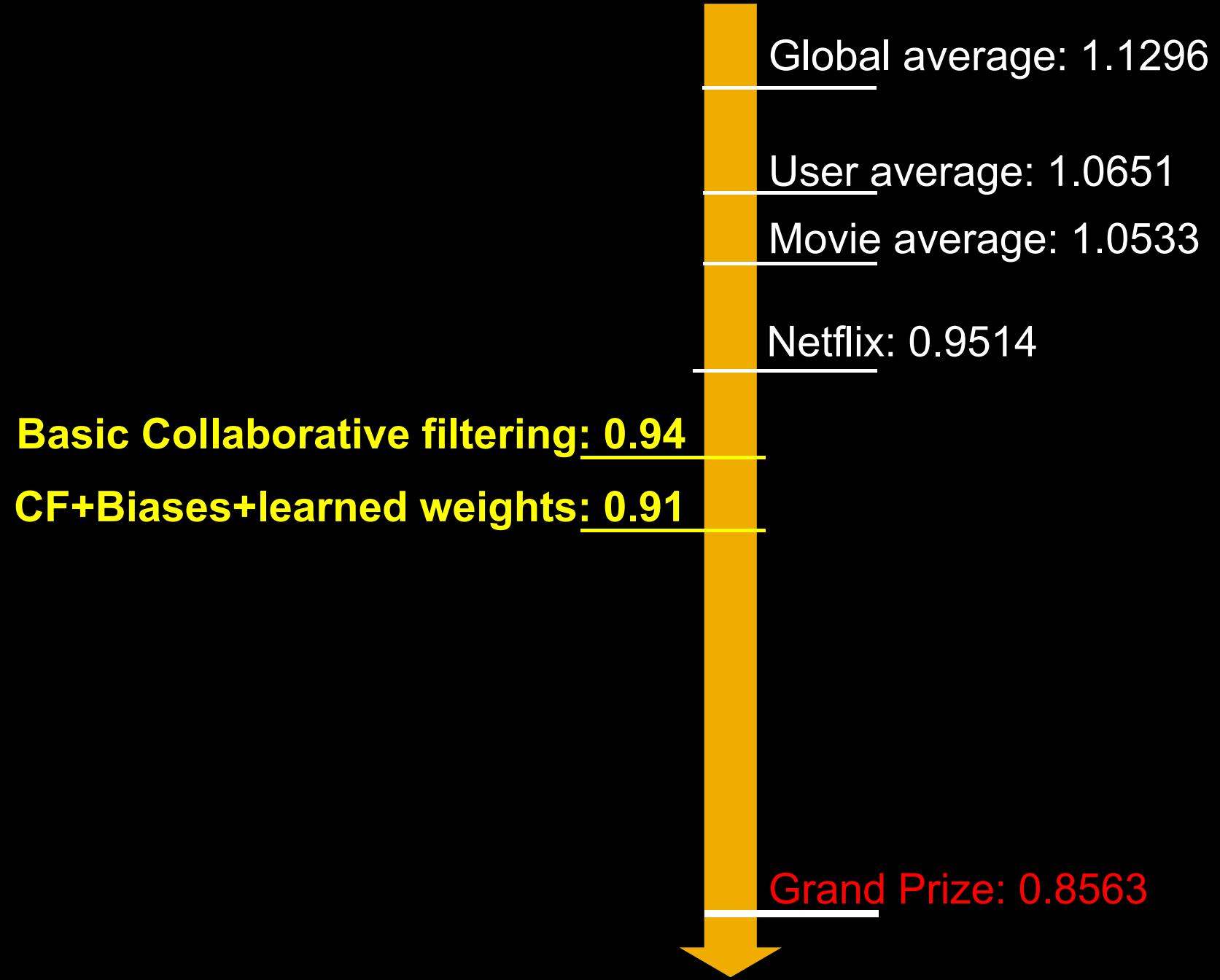
$w_{old} = w_{new}$

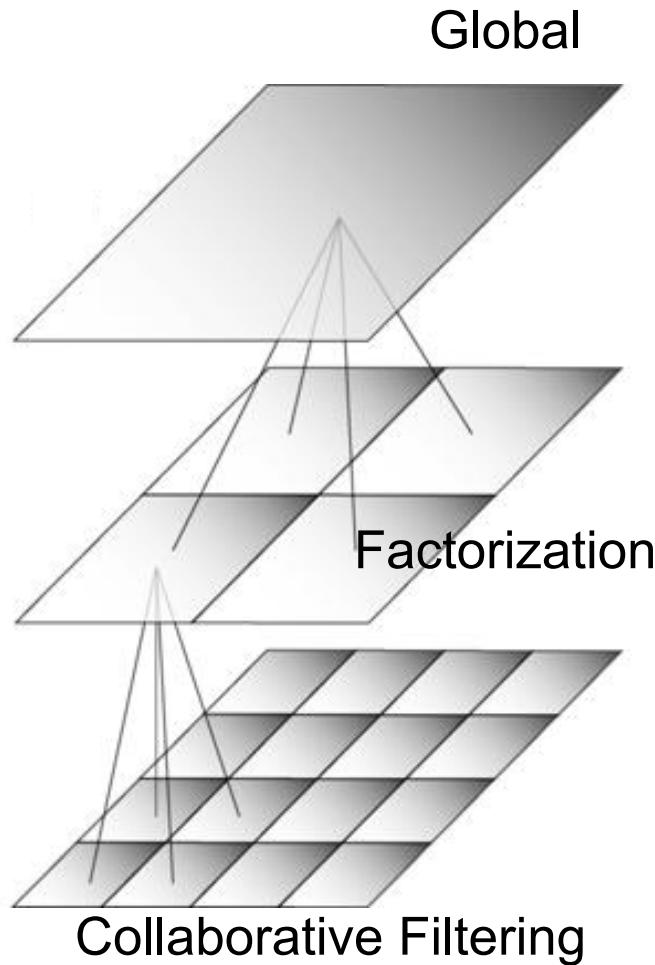
$w_{new} = w_{old} - \eta \cdot \nabla_w w_{old}$

Interpolation Weights

- So far: $\widehat{r}_{xi} = b_{xi} + \sum_{j \in N(i;x)} w_{ij}(r_{xj} - b_{xj})$
 - Weights w_{ij} derived based on their role; **no use of an arbitrary similarity measure** ($w_{ij} \neq s_{ij}$)
 - Explicitly account for interrelationships among the neighboring movies
- **Next: Latent factor model**
 - Extract “regional” correlations

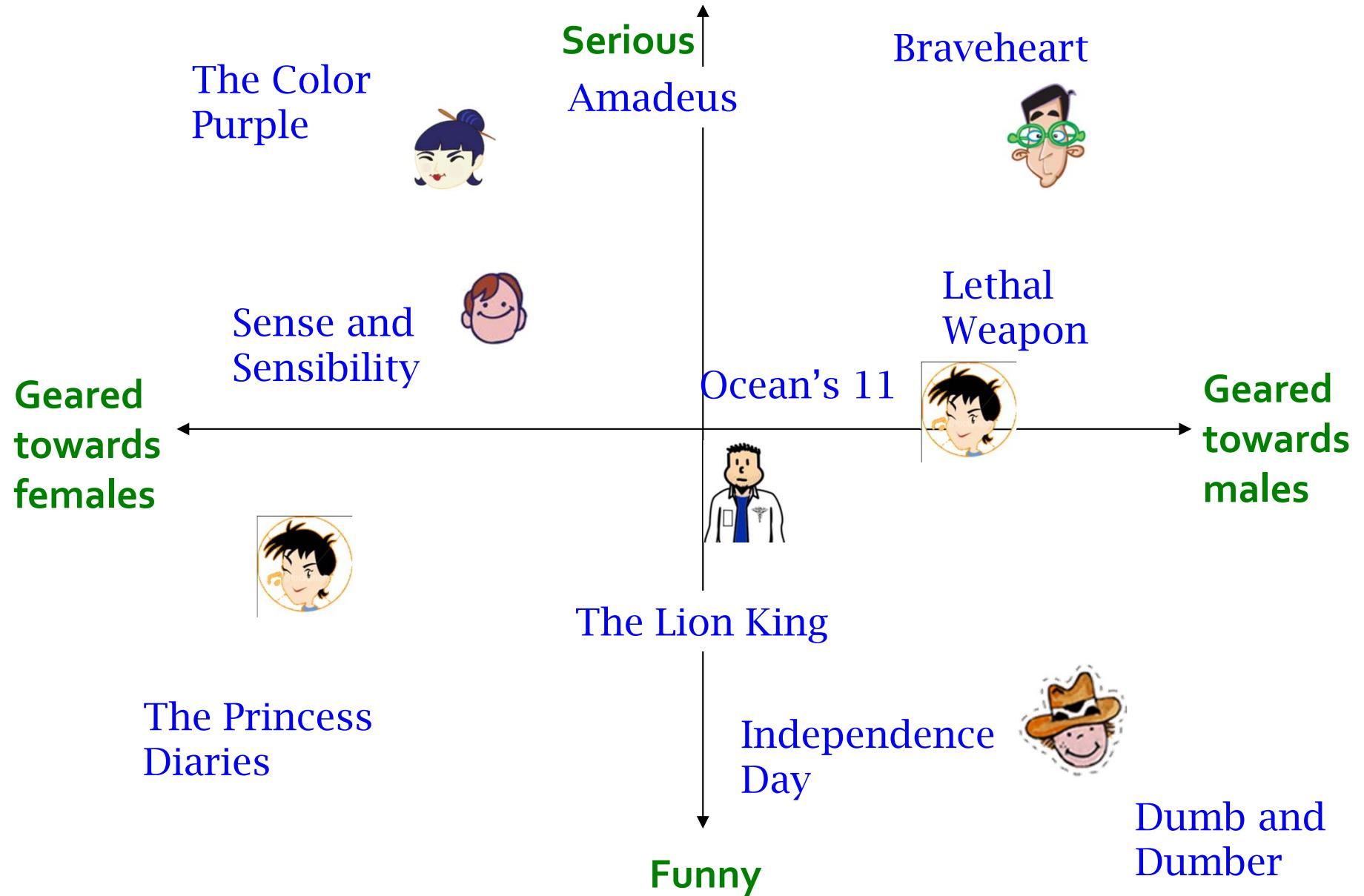




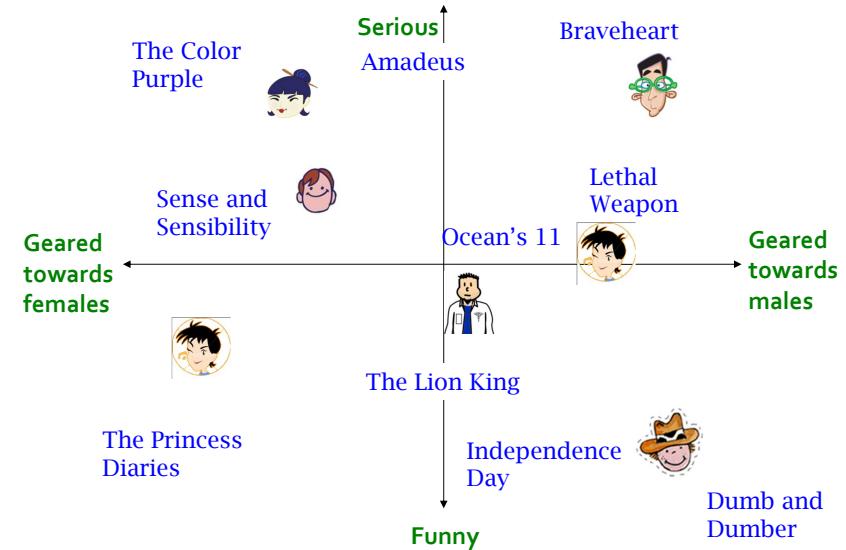


Latent Factor Model:
Extract “regional”
correlations

Latent Factor Models



- » How would you map from high dimensional features (ratings) to lower dimensional “latent factors”?
- » Or ... how would you map many individual users to smaller “latent” groups?
- » Or ... how would you map many individual items to smaller “latent” groups?



Latent factor models

$$\text{SVD: } A = U \Sigma V^T$$

- “SVD” on Netflix data: $R \approx Q \cdot P^T$

users

1		3		5		5		4	
		5	4		4		2	1	3
2	4		1	2		3	4	3	5
	2	4		5		4		2	
		4	3	4	2			2	5
1		3		3		2		4	

items

\approx

factors

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

items

R

Q

users

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6

factors

P^T

Latent factor models

users

1	3		5		5		4	
		5	4		4		2	1
2	4		1	2	3		4	3
	2	4		5		4		2
		4	3	4	2			2
1	3	3			2			4

~

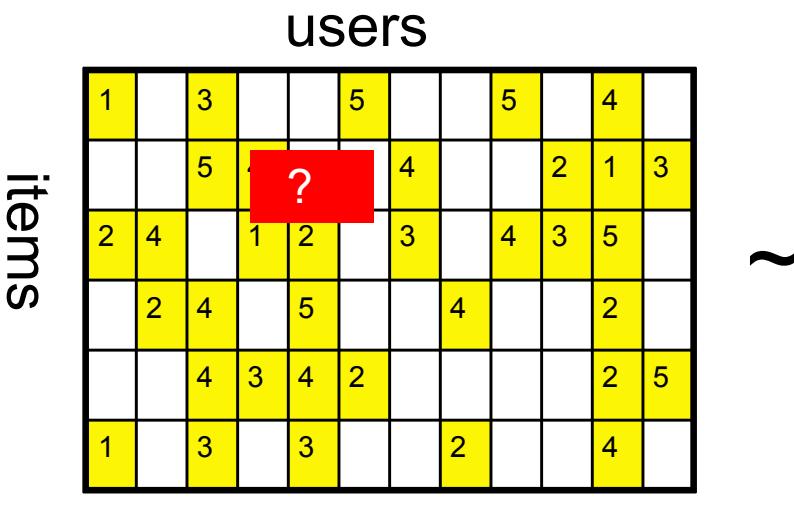
users

.1	-.4	.2	1.1	-.2	.3	.5	-.2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.5	.6	.5	-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
-.2	.3	.5	2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1
1.1	2.1	.3												
-.7	2.1	-2												
-1	.7	.3												

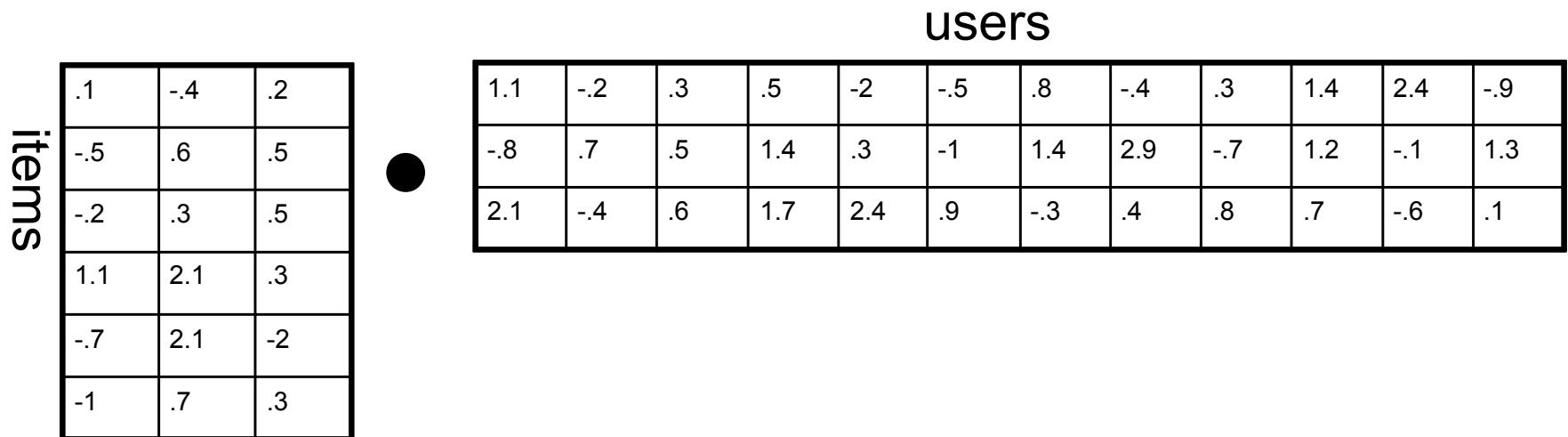
~

A rank-3 SVD approximation

Estimate unknown ratings as inner-products of factors:

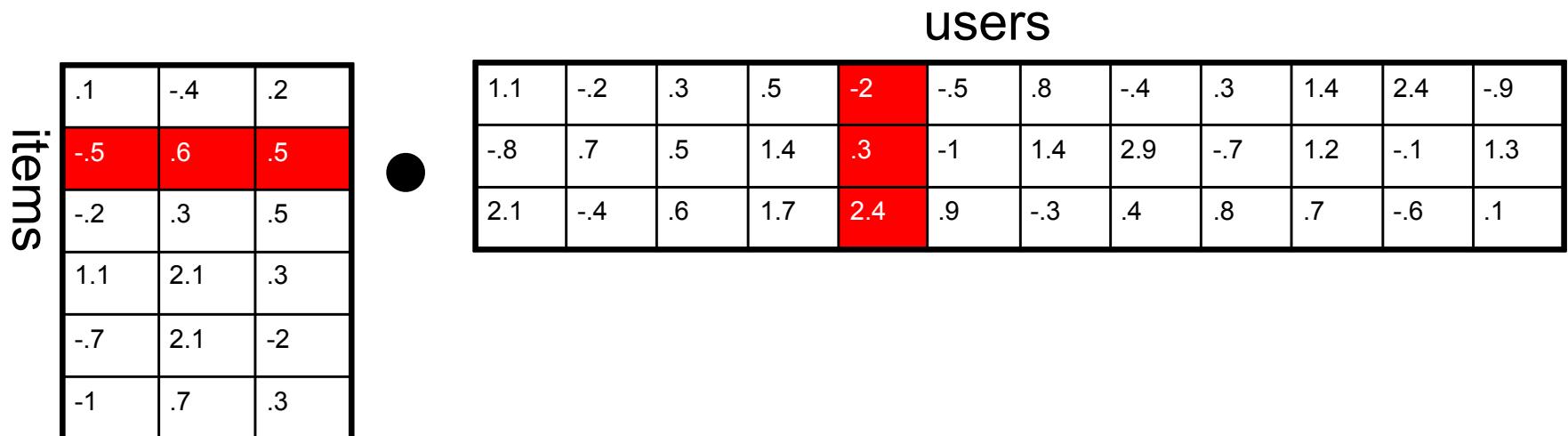
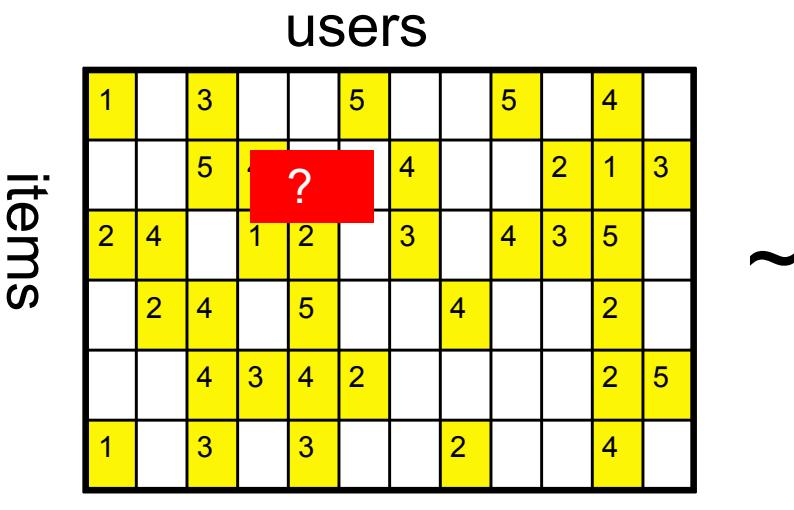


~



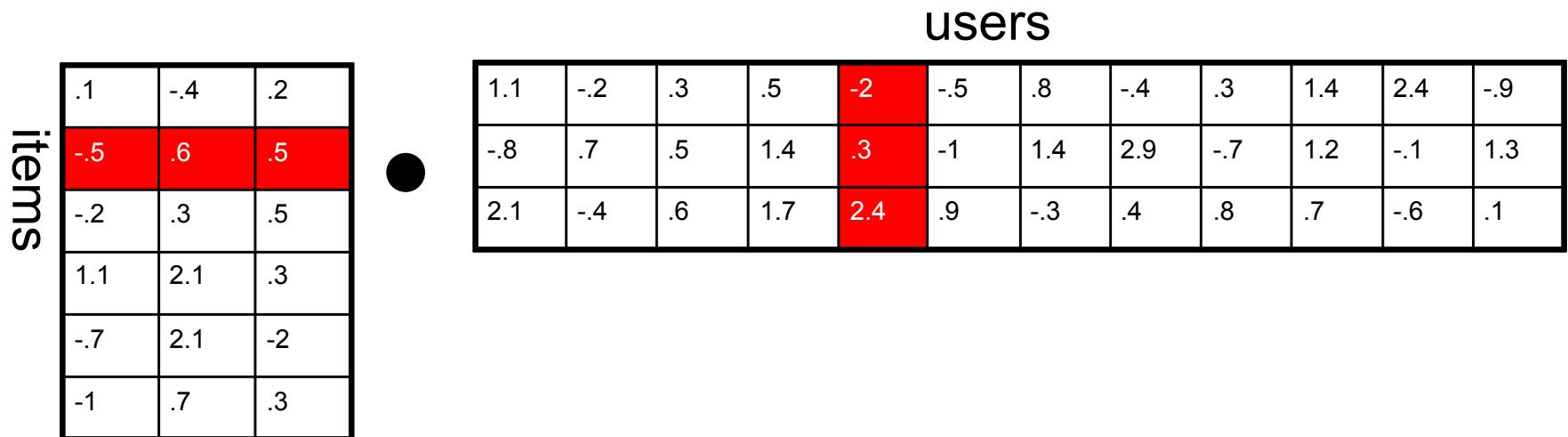
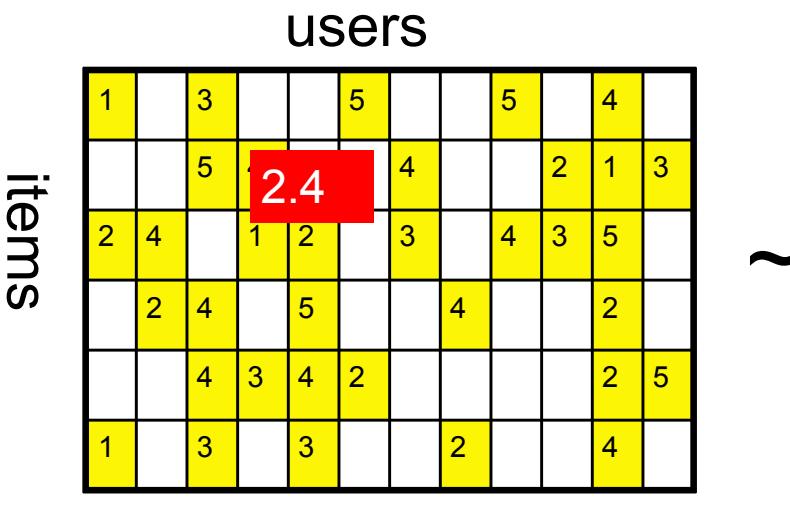
A rank-3 SVD approximation

Estimate unknown ratings as inner-products of factors:



A rank-3 SVD approximation

Estimate unknown ratings as inner-products of factors:



A rank-3 SVD approximation

SVD

SVD - Definition

$$A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T$$

■ A: Input data matrix

- $m \times n$ matrix (e.g., m documents, n terms)

■ U: Left singular vectors

- $m \times r$ matrix (m documents, r concepts)

■ Σ : Singular values

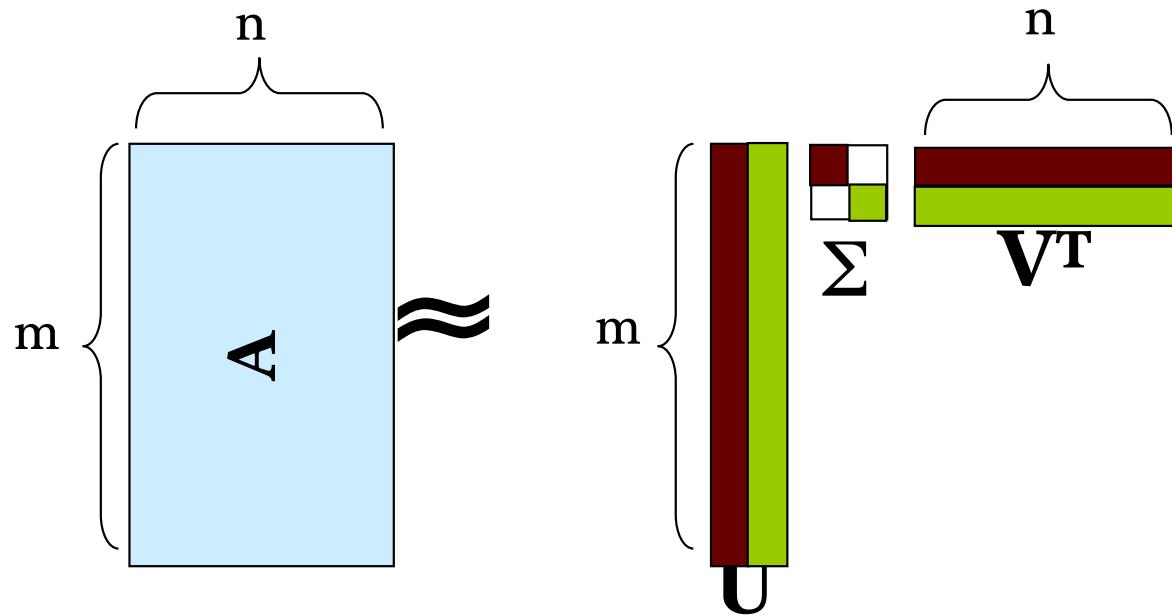
- $r \times r$ diagonal matrix (strength of each ‘concept’)
(r : rank of the matrix A)

■ V: Right singular vectors

- $n \times r$ matrix (n terms, r concepts)

SVD

$$\mathbf{A} \approx \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^\top$$



SVD - Properties

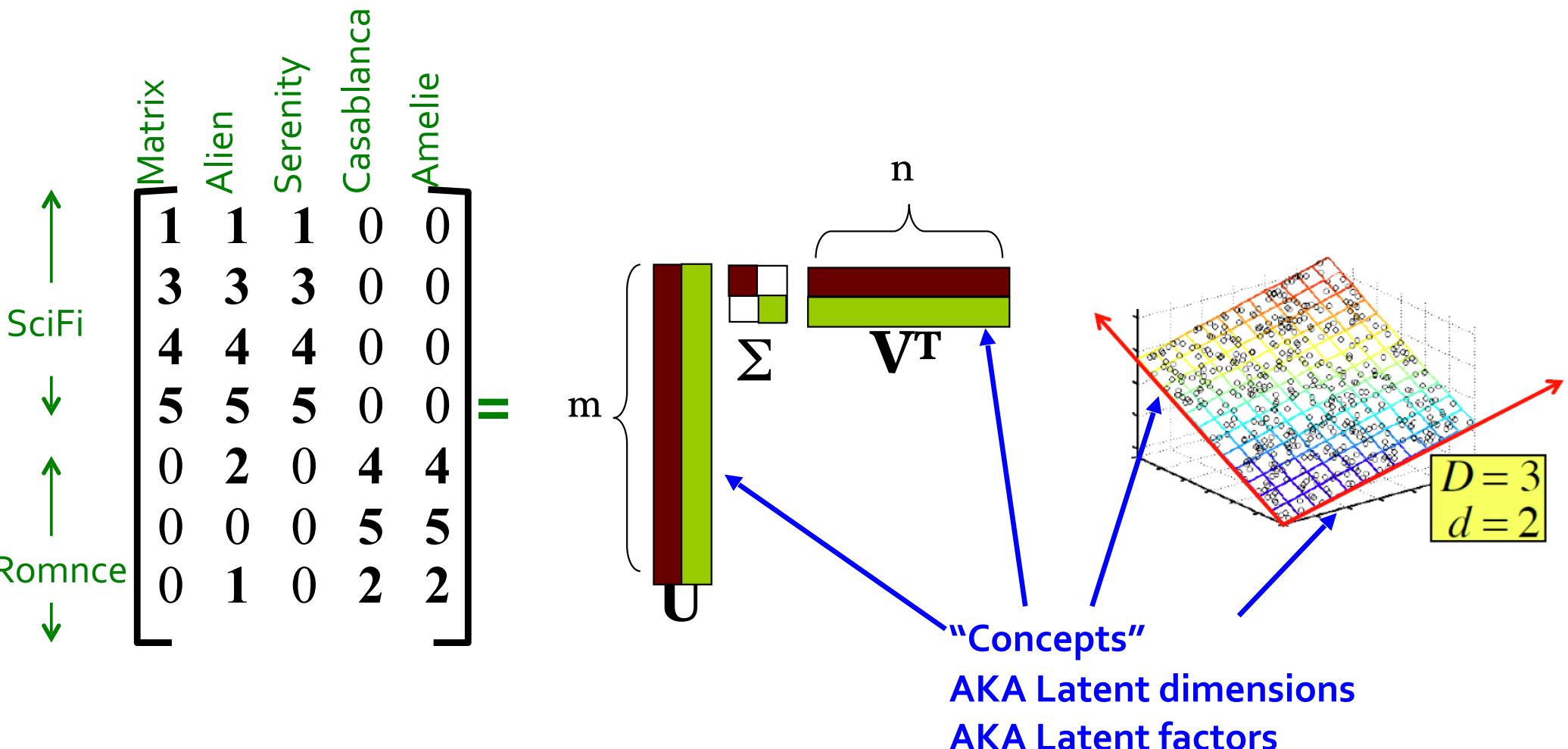
It is **always** possible to decompose a real matrix \mathbf{A} into $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T$, where

- $\mathbf{U}, \Sigma, \mathbf{V}$: unique
- \mathbf{U}, \mathbf{V} : column orthonormal
 - $\mathbf{U}^T \mathbf{U} = \mathbf{I}; \mathbf{V}^T \mathbf{V} = \mathbf{I}$ (\mathbf{I} : identity matrix)
 - (Columns are orthogonal unit vectors)
- Σ : diagonal
 - Entries (**singular values**) are positive, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \dots \geq 0$)

Nice proof of uniqueness: <http://www.mpi-inf.mpg.de/~bast/ir-seminar-wso4/lecture2.pdf>

SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies



SVD – Example: Users-to-Movies

- $A = U \Sigma V^T$ - example: Users to Movies

Matrix

$$\begin{matrix}
 & \begin{matrix} \text{SciFi} \\ \downarrow \end{matrix} & \begin{matrix} \text{Romance} \\ \downarrow \end{matrix} & \begin{matrix} \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{matrix} \\
 \begin{matrix} \uparrow \\ \text{SciFi} \end{matrix} & \left[\begin{matrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{matrix} \right] & = & \left[\begin{matrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{matrix} \right] & \times & \left[\begin{matrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{matrix} \right] & \times \\
 & \begin{matrix} \uparrow \\ \text{Romance} \end{matrix} & \downarrow & & & & & \begin{matrix} \uparrow \\ \text{Romance} \end{matrix} & \\
 & & & & & & & & \left[\begin{matrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{matrix} \right]
 \end{matrix}$$

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example: Users to Movies

Matrix

	Alien	Serenity	Casablanca	Amelie	
SciFi	1	1	1	0	0
↓	3	3	3	0	0
↑	4	4	4	0	0
Romnce	5	5	5	0	0
↓	0	2	0	4	4
↑	0	0	0	5	5
	0	1	0	2	2

$=$ X

0.13	0.02	-0.01
0.41	0.07	-0.03
0.55	0.09	-0.04
0.68	0.11	-0.05
0.15	-0.59	0.65
0.07	-0.73	-0.67
0.07	-0.29	0.32

SciFi-concept Romance-concept

X

12.4	0	0
0	9.5	0
0	0	1.3

X

$\begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:

U is “user-to-concept” similarity matrix

$$\begin{array}{c}
 \text{Matrix} \\
 \left[\begin{array}{ccccc}
 & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \\
 \text{SciFi} & 1 & 1 & 1 & 0 & 0 \\
 \downarrow & 3 & 3 & 3 & 0 & 0 \\
 & 4 & 4 & 4 & 0 & 0 \\
 \text{Romance} & 5 & 5 & 5 & 0 & 0 \\
 \downarrow & 0 & 2 & 0 & 4 & 4 \\
 & 0 & 0 & 0 & 5 & 5 \\
 \downarrow & 0 & 1 & 0 & 2 & 2
 \end{array} \right] = \left[\begin{array}{ccc}
 & \text{SciFi-concept} & \text{Romance-concept} \\
 \text{SciFi} & 0.13 & 0.02 & -0.01 \\
 \downarrow & 0.41 & 0.07 & -0.03 \\
 & 0.55 & 0.09 & -0.04 \\
 & 0.68 & 0.11 & -0.05 \\
 & 0.15 & -0.59 & 0.65 \\
 & 0.07 & -0.73 & -0.67 \\
 & 0.07 & -0.29 & 0.32
 \end{array} \right] \times \left[\begin{array}{ccc}
 12.4 & 0 & 0 \\
 0 & 9.5 & 0 \\
 0 & 0 & 1.3
 \end{array} \right] \times
 \left[\begin{array}{ccccc}
 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
 0.40 & -0.80 & 0.40 & 0.09 & 0.09
 \end{array} \right]
 \end{array}$$

SVD – Example: Users-to-Movies

- ## ■ $A = U \Sigma V^T$ - example:

The diagram illustrates the decomposition of a movie rating matrix into two components. The original matrix (Matrix) is decomposed into a SciFi-concept matrix and a Romance-concept matrix.

Matrix:

	Matrix	Alien	Serenity	Casablanca	Amelie
SciFi	1	1	1	0	0
↓	3	3	3	0	0
↑	4	4	4	0	0
Romance	5	5	5	0	0
↓	0	2	0	4	4
↑	0	0	0	5	5
	0	1	0	2	2

SciFi-concept:

0.13	0.02	-0.01
0.41	0.07	-0.03
0.55	0.09	-0.04
0.68	0.11	-0.05
0.15	-0.59	0.65
0.07	-0.73	-0.67
0.07	-0.29	0.32

Romance-concept:

0.56	0.59	0.56	0.09	0.09
0.12	-0.02	0.12	-0.69	-0.69
0.40	-0.80	0.40	0.09	0.09

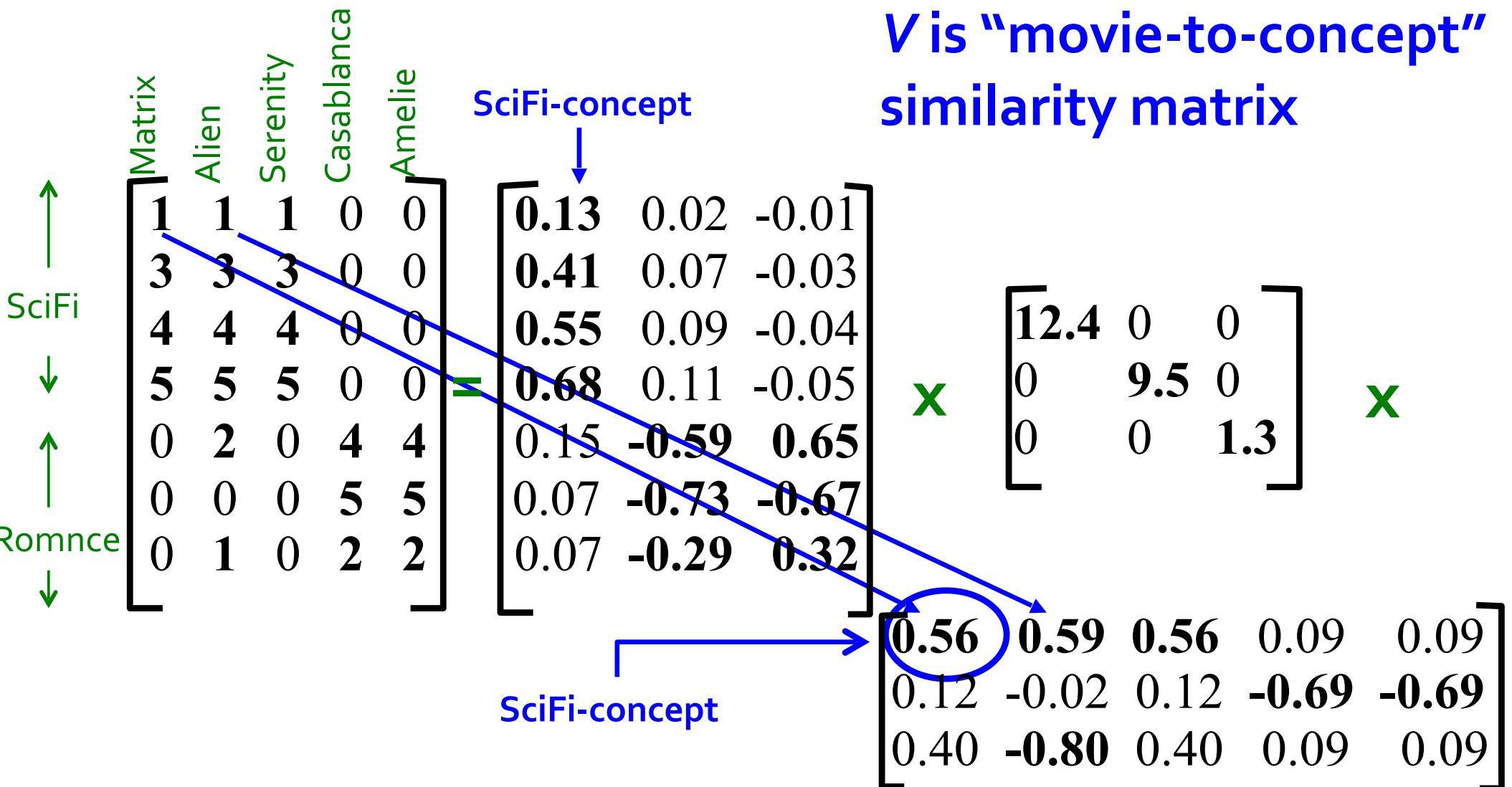
"strength" of the SciFi-concept:

A blue arrow points from the value **12.4** in the SciFi-concept matrix to the following matrix:

12.4	0	0
0	9.5	0
0	0	1.3

SVD – Example: Users-to-Movies

■ $A = U \Sigma V^T$ - example:



SVD - Interpretation #1

‘movies’, ‘users’ and ‘concepts’:

- U : user-to-concept similarity matrix
- V : movie-to-concept similarity matrix
- Σ : its diagonal elements:
‘strength’ of each concept

Dimensionality Reduction with SVD

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & \cancel{1.3} \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

The diagram illustrates the SVD decomposition of a matrix. The original matrix on the left is approximately equal to the product of three matrices: a matrix Q (containing the first three columns of the original matrix), a diagonal matrix D (containing the singular values 12.4, 9.5, and 1.3), and a matrix V (containing the first three columns of the inverse of the transpose of Q). Red lines and a red 'X' mark the zero singular value and its corresponding column in the second matrix, indicating that the third dimension is being reduced by setting the smallest singular value to zero.

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \begin{bmatrix} 0.13 & 0.02 \\ 0.41 & 0.07 \\ 0.55 & 0.09 \\ 0.68 & 0.11 \\ 0.15 & -0.59 \\ 0.07 & -0.73 \\ 0.07 & -0.29 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \end{bmatrix}$$

SVD - Interpretation #2

More details

- Q: How exactly is dim. reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.92 & 0.95 & 0.92 & 0.01 & 0.01 \\ 2.91 & 3.01 & 2.91 & -0.01 & -0.01 \\ 3.90 & 4.04 & 3.90 & 0.01 & 0.01 \\ 4.82 & 5.00 & 4.82 & 0.03 & 0.03 \\ 0.70 & 0.53 & 0.70 & 4.11 & 4.11 \\ -0.69 & 1.34 & -0.69 & 4.78 & 4.78 \\ 0.32 & 0.23 & 0.32 & 2.01 & 2.01 \end{bmatrix}$$

Frobenius norm:

$$\|M\|_F = \sqrt{\sum_{ij} M_{ij}^2}$$

$$\|A-B\|_F = \sqrt{\sum_{ij} (A_{ij}-B_{ij})^2}$$

is “small”

SVD – Best Low Rank Approx.

$$A = U \Sigma V^T$$

B is best approximation of A

$$B = U \Sigma V^T$$

Example of SVD & Conclusion

Case study: How to query?

- Q: Find users that like ‘Matrix’
- A: Map query into a ‘concept space’ – how?

$$\begin{matrix} \text{↑} & \text{Matrix} \\ \text{SciFi} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \\ \text{↓} & \end{matrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

↑ SciFi
↓ Romance

Matrix
Alien Serenity Casablanca Amelie

0.13 0.02 -0.01
0.41 0.07 -0.03
0.55 0.09 -0.04
0.68 0.11 -0.05
0.15 -0.59 0.65
0.07 -0.73 -0.67
0.07 -0.29 0.32

12.4 0 0
0 9.5 0
0 0 1.3

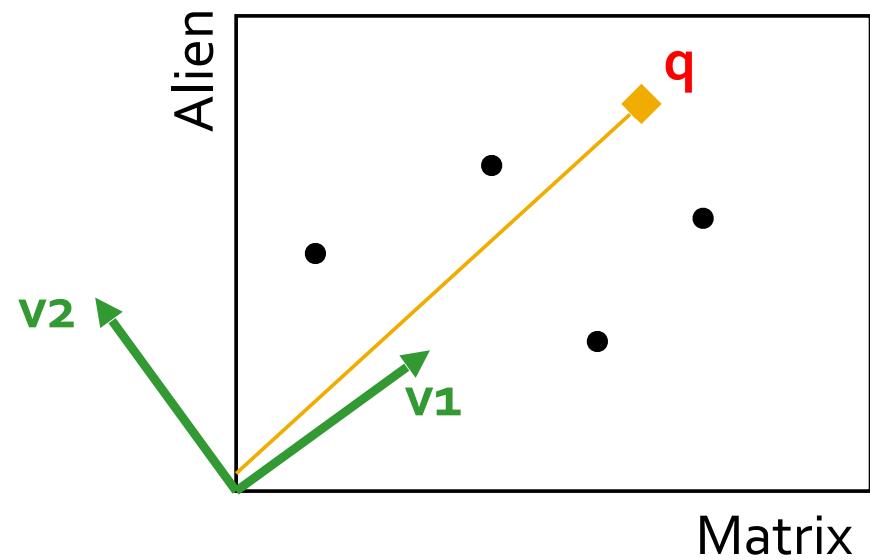
0.56 0.59 0.56 0.09 0.09
0.12 -0.02 0.12 -0.69 -0.69
0.40 -0.80 0.40 0.09 0.09

Case study: How to query?

- Q: Find users that like ‘Matrix’
- A: Map query into a ‘concept space’ – how?

$$q = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Project into concept space:
Inner product with each
'concept' vector v_i

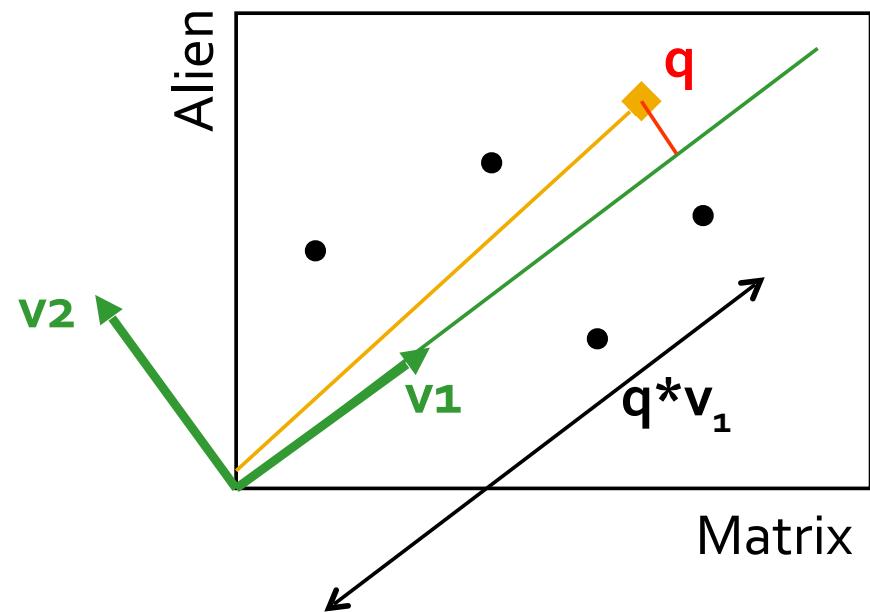


Case study: How to query?

- Q: Find users that like ‘Matrix’
- A: Map query into a ‘concept space’ – how?

$$q = \begin{bmatrix} \text{Matrix} \\ \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Project into concept space:
Inner product with each
'concept' vector v_i



Case study: How to query?

Compactly, we have:

$$\mathbf{q}_{\text{concept}} = \mathbf{q} \mathbf{V}$$

E.g.:

$$\mathbf{q} = \begin{bmatrix} \text{Matrix} \\ 5 \end{bmatrix} \times \begin{bmatrix} \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix} \quad \mathbf{V} = \begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix}$$

movie-to-concept
similarities (\mathbf{V})

SciFi-concept
↓
 $= \begin{bmatrix} 2.8 & 0.6 \end{bmatrix}$

Case study: How to query?

- How would the user d that rated ('Alien', 'Serenity') be handled?

$$d_{\text{concept}} = d \vee$$

E.g.:

$$\mathbf{q} = \begin{bmatrix} \text{Matrix} \\ 0 & 4 & 5 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} \text{Alien} \\ \text{Serenity} \\ \text{Casablanca} \\ \text{Amelie} \end{bmatrix}$$

\times

$$\begin{bmatrix} 0.56 & 0.12 \\ 0.59 & -0.02 \\ 0.56 & 0.12 \\ 0.09 & -0.69 \\ 0.09 & -0.69 \end{bmatrix}$$

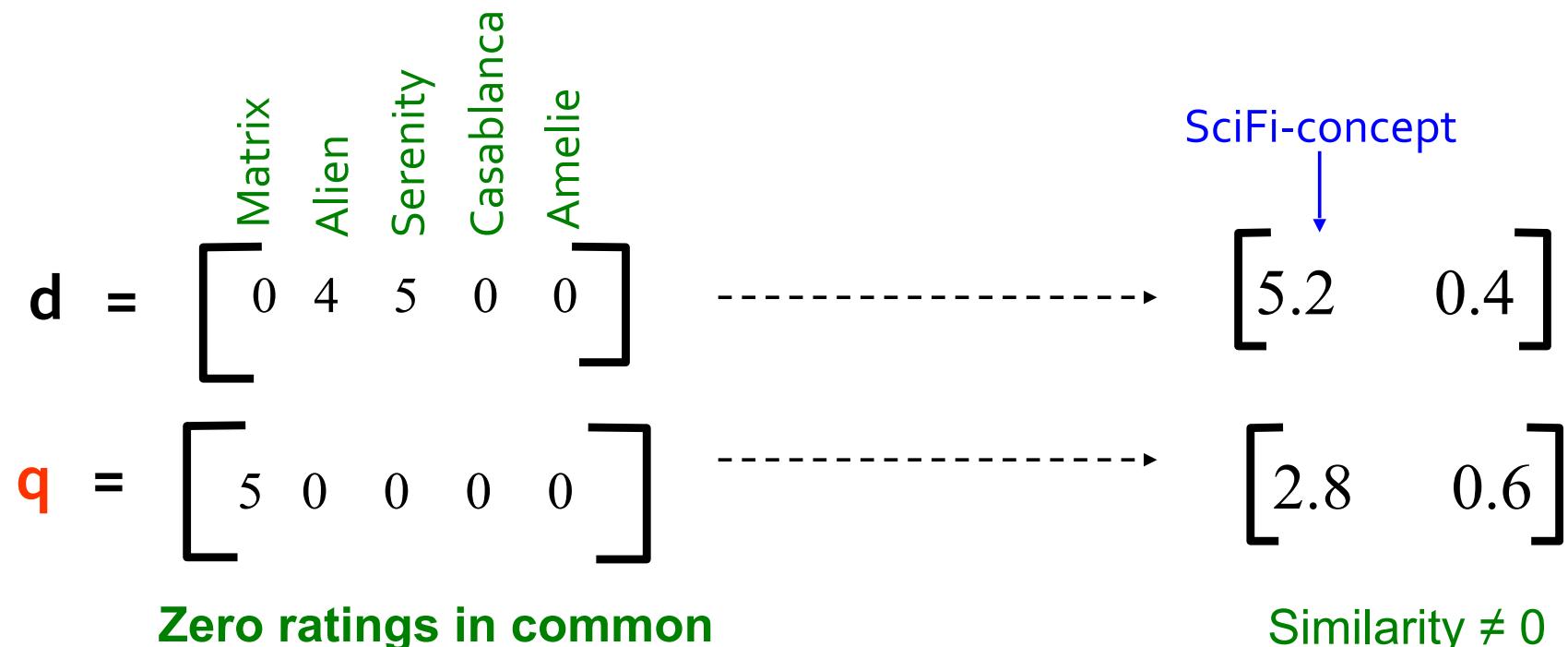
movie-to-concept
similarities (V)

SciFi-concept

$= \begin{bmatrix} 5.2 \\ 0.4 \end{bmatrix}$

Case study: How to query?

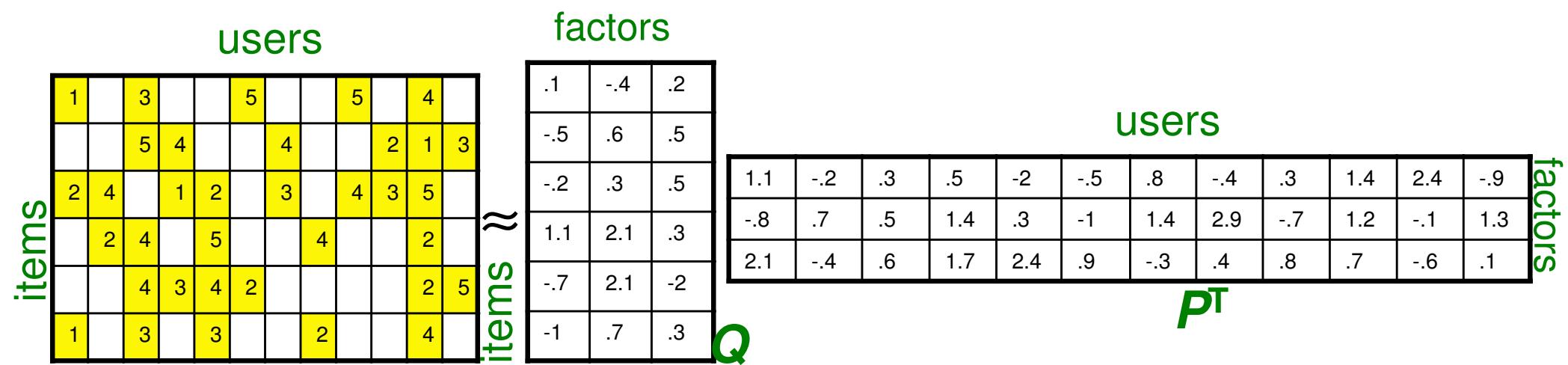
- **Observation:** User d that rated ('Alien', 'Serenity') will be **similar** to user q that rated ('Matrix'), although d and q have **zero ratings in common**!



SVD

Our goal is to find P and Q such that:

$$\min_{P,Q} \sum_{(i,x) \in R} (r_{xi} - q_i \cdot p_x)^2$$



- Want to minimize SSE for unseen test data
- Idea: Minimize SSE on training data
 - Want large k (# of factors) to capture all the signals
 - But, SSE on test data begins to rise for $k > 2$
- This is a classical example of **overfitting**:
 - With too much freedom (too many free parameters) the model starts fitting noise
 - That is it fits too well the training data and thus **not generalizing** well to unseen test data

1	3	4		
3	5		5	
4	5		5	
3				
3				
2			?	?
			?	?
2	1			?
3			?	?
1				

1	3	4		
3	5		5	
4	5		5	
3				
3				
2		?	?	?
	2	1		?
	3		?	
1				

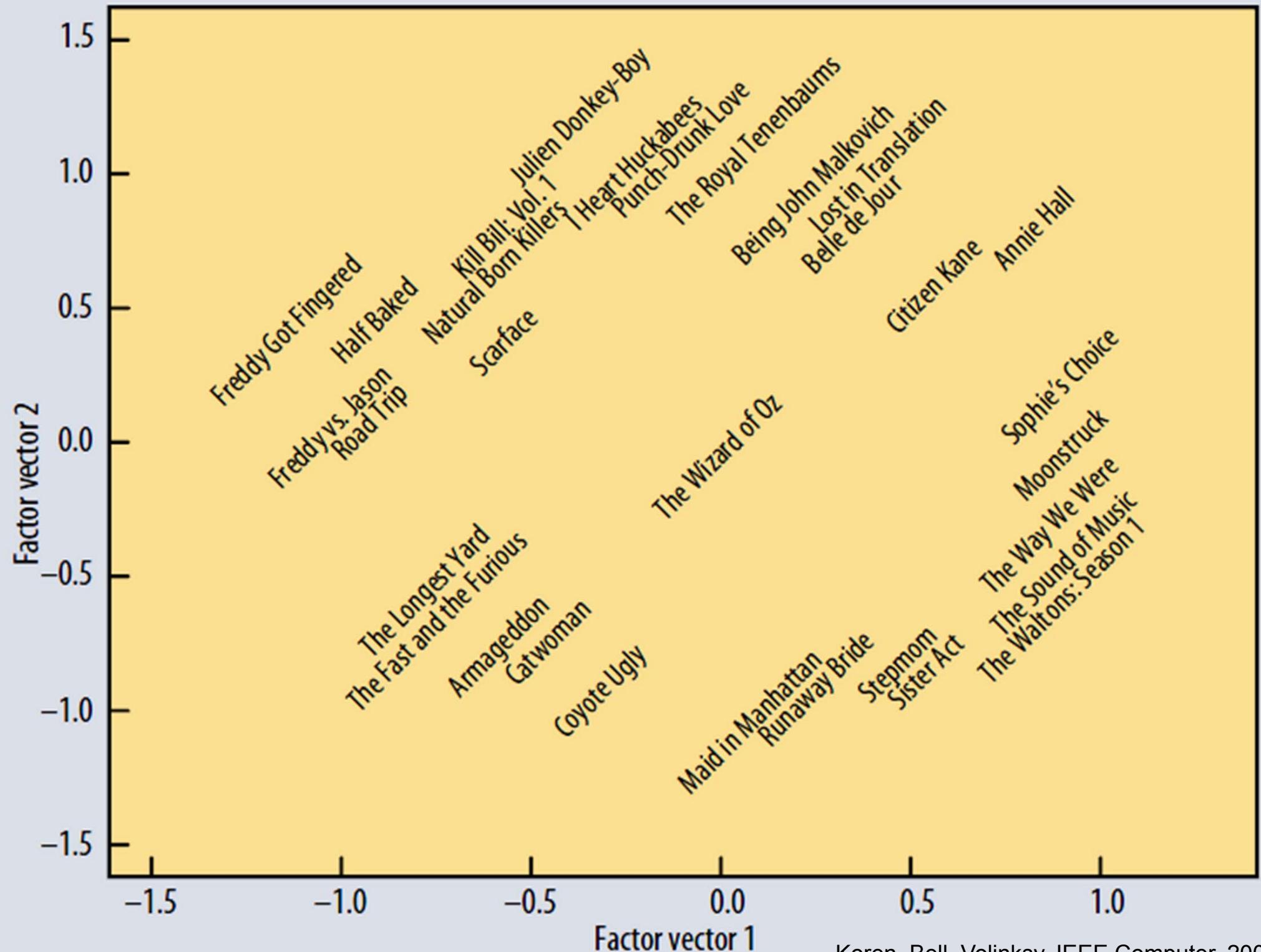
- To solve overfitting we introduce regularization:

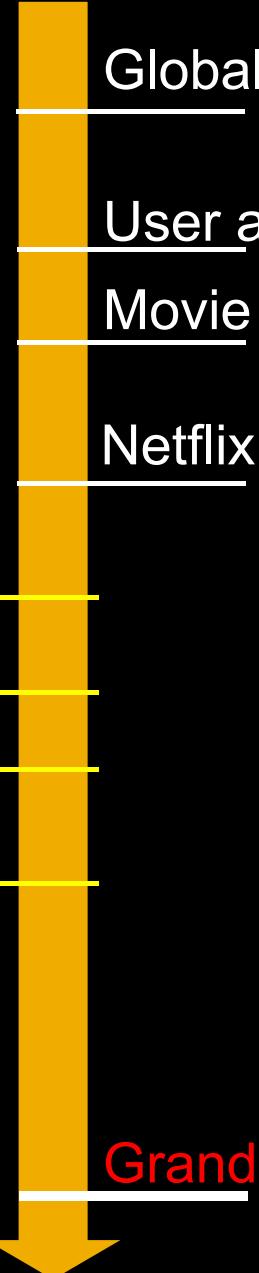
- Allow rich model where there are sufficient data
- Shrink aggressively where data are scarce

$$\min_{P,Q} \underbrace{\sum_{\text{training}} (r_{xi} - q_i p_x)^2}_{\text{"error"}} + \left[\underbrace{\lambda_1 \sum_x \|p_x\|^2}_{\text{"length"}^x} + \underbrace{\lambda_2 \sum_i \|q_i\|^2}_{\text{"length"}^i} \right]$$

$\lambda_1, \lambda_2 \dots$ user set regularization parameters

Note: We do not care about the “raw” value of the objective function, but we care in P,Q that achieve the minimum of the objective





Global average: 1.1296

User average: 1.0651

Movie average: 1.0533

Netflix: 0.9514

Basic Collaborative filtering: 0.94

Collaborative filtering++: 0.91

Latent factors: 0.90

Latent factors+Biases: 0.89

Grand Prize: 0.8563

Modeling Biases and Interactions

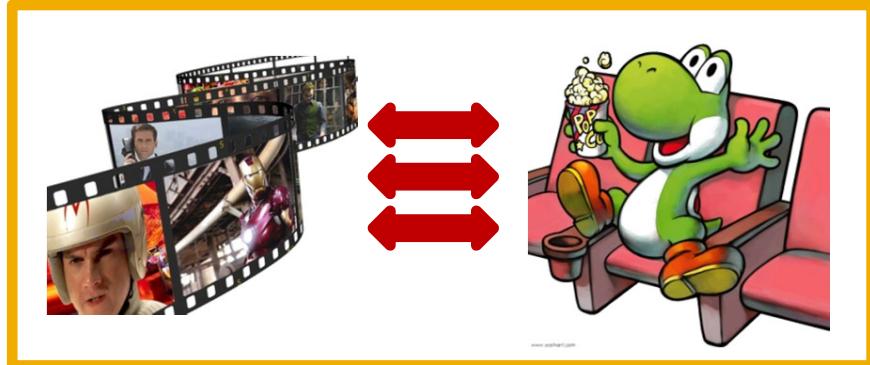
user bias



movie bias



user-movie interaction



Baseline predictor

- Separates users and movies
- Benefits from insights into user's behavior
- Among the main practical contributions of the competition

User-Movie interaction

- Characterizes the matching between users and movies
- Attracts most research in the field
- Benefits from algorithmic and mathematical innovations

- μ = overall mean rating
- b_x = bias of user x
- b_i = bias of movie i

Baseline Predictor

- We have expectations on the rating by user x of movie i , even without estimating x 's attitude towards movies like i



- Rating scale of user x
- Values of other ratings user gave recently (day-specific mood, anchoring, multi-user accounts)

- (Recent) popularity of movie i
- Selection bias; related to number of ratings user gave on the same day (“frequency”)

Putting it all together

$$r_{xi} = \mu + b_x + b_i + q_i \cdot p_x$$

Overall mean rating Bias for user x Bias for movie i User-Movie interaction

■ Example:

- Mean rating: $\mu = 3.7$
- You are a critical reviewer: your ratings are 1 star lower than the mean: $b_x = -1$
- Star Wars gets a mean rating of 0.5 higher than average movie: $b_i = +0.5$
- Predicted rating for you on Star Wars:
 $= 3.7 - 1 + 0.5 = 3.2$

Fitting the new model

- **Solve:**

$$\min_{Q,P} \sum_{(x,i) \in R} (r_{xi} - (\mu + b_x + b_i + q_i p_x))^2$$

goodness of fit

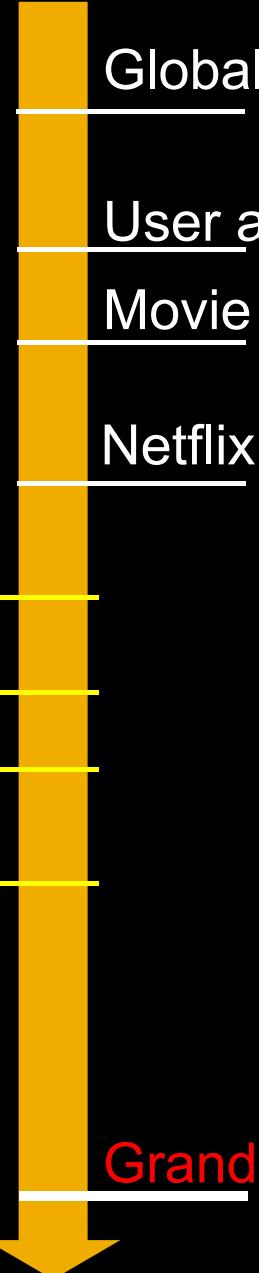
$$+ \left(\lambda_1 \sum_i \|q_i\|^2 + \lambda_2 \sum_x \|p_x\|^2 + \lambda_3 \sum_x \|b_x\|^2 + \lambda_4 \sum_i \|b_i\|^2 \right)$$

regularization

λ is selected via grid-search on a validation set

- **Stochastic gradient decent to find parameters**

- **Note:** Both biases b_x, b_i as well as interactions q_i, p_x are treated as parameters (we estimate them)



Global average: 1.1296

User average: 1.0651

Movie average: 1.0533

Netflix: 0.9514

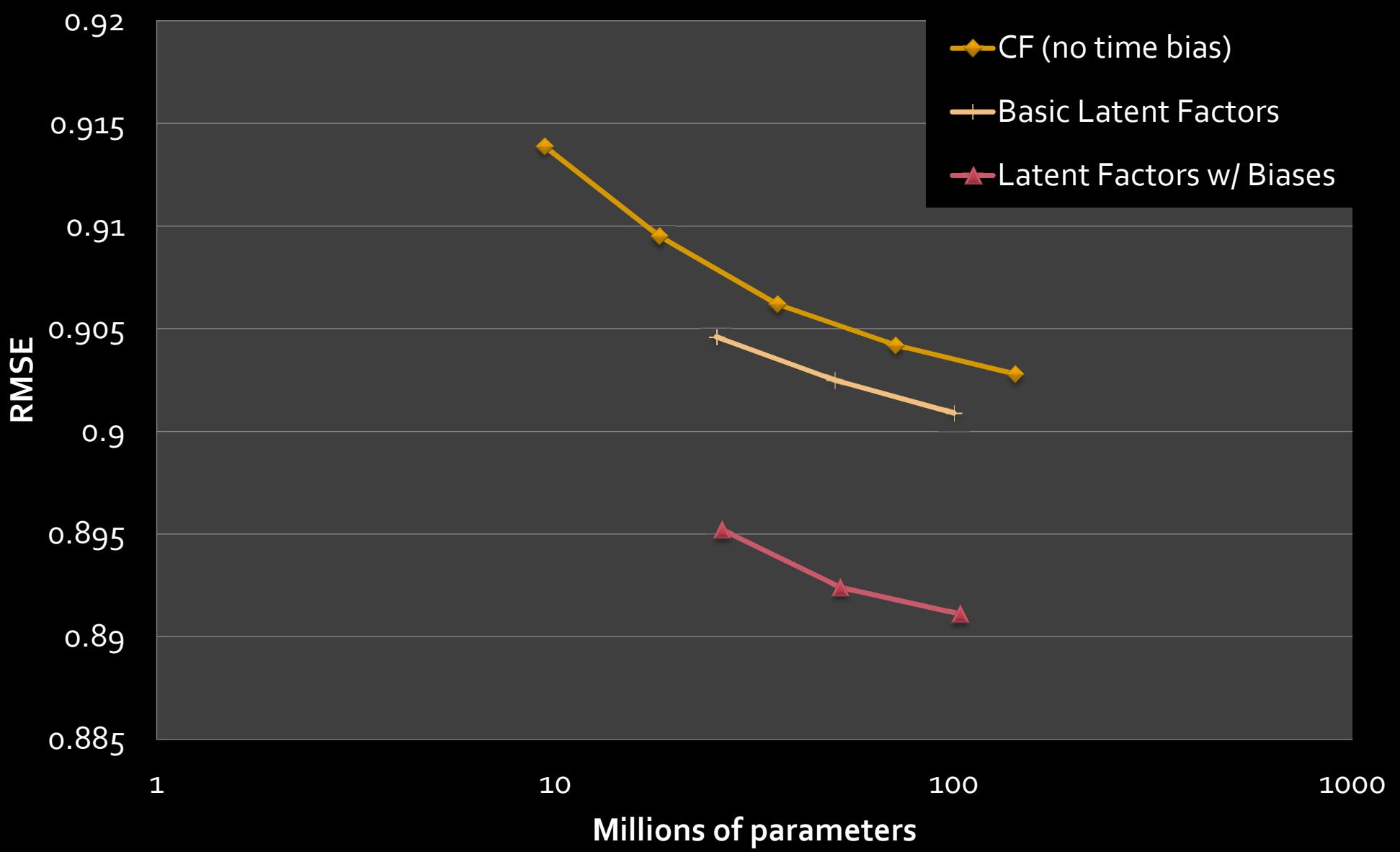
Basic Collaborative filtering: 0.94

Collaborative filtering++: 0.91

Latent factors: 0.90

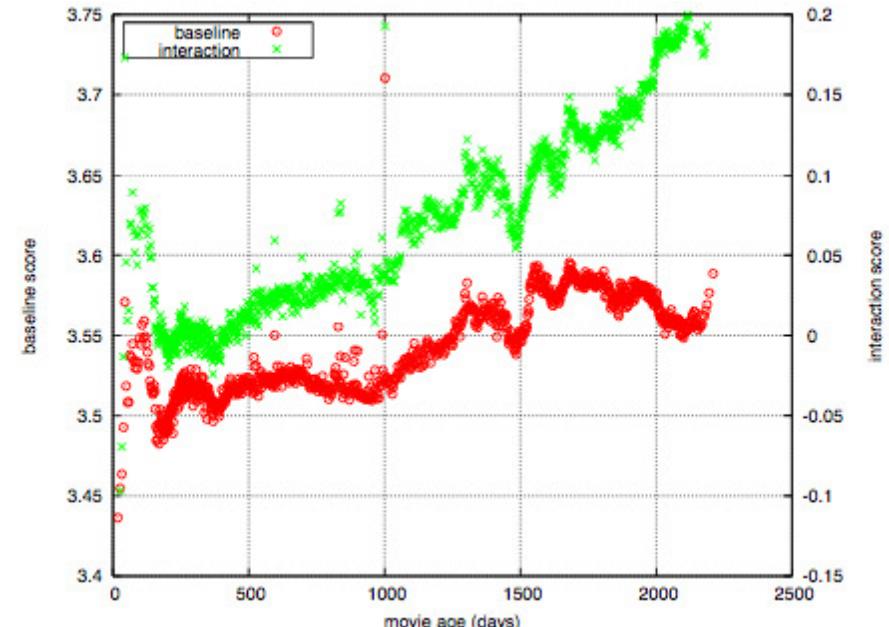
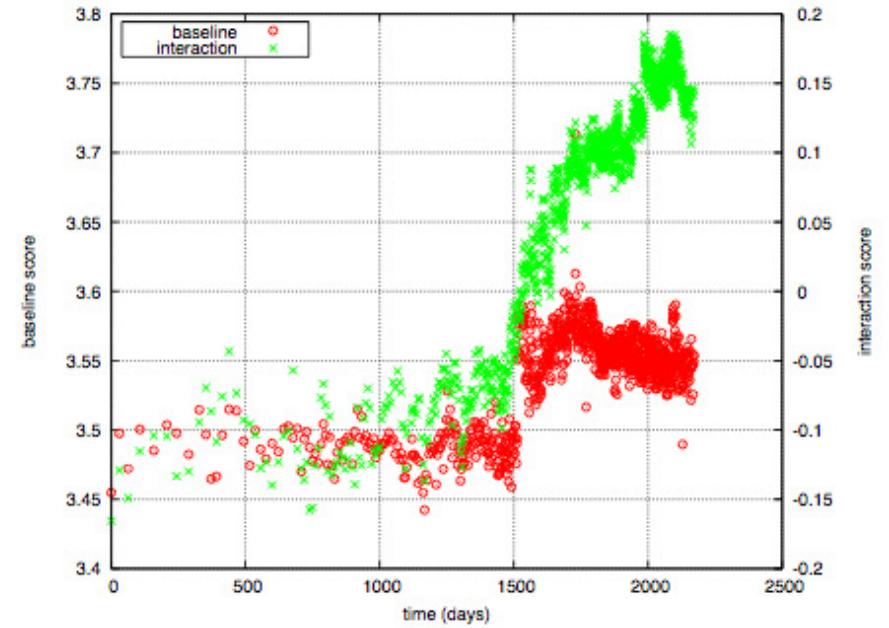
Latent factors+Biases: 0.89

Grand Prize: 0.8563



Temporal Biases of Users

- **Sudden rise in the average movie rating** (early 2004)
 - Improvements in Netflix
 - GUI improvements
 - Meaning of rating changed
- **Movie age**
 - Users prefer new movies without any reasons
 - Older movies are just inherently better than newer ones



Temporal Bias and Users

- **Original model:**

$$r_{xi} = \mu + b_x + b_i + q_i \cdot p_x$$

- **Add time dependence to biases:**

$$r_{xi} = \mu + b_x(t) + b_i(t) + q_i \cdot p_x$$

- Make parameters b_x and b_i to depend on time
- (1) Parameterize time-dependence by linear trends
- (2) Each bin corresponds to 10 consecutive weeks

$$b_i(t) = b_i + b_{i,\text{Bin}(t)}$$

- **Add temporal dependence to factors**

- $p_x(t)$... user preference vector on day t

