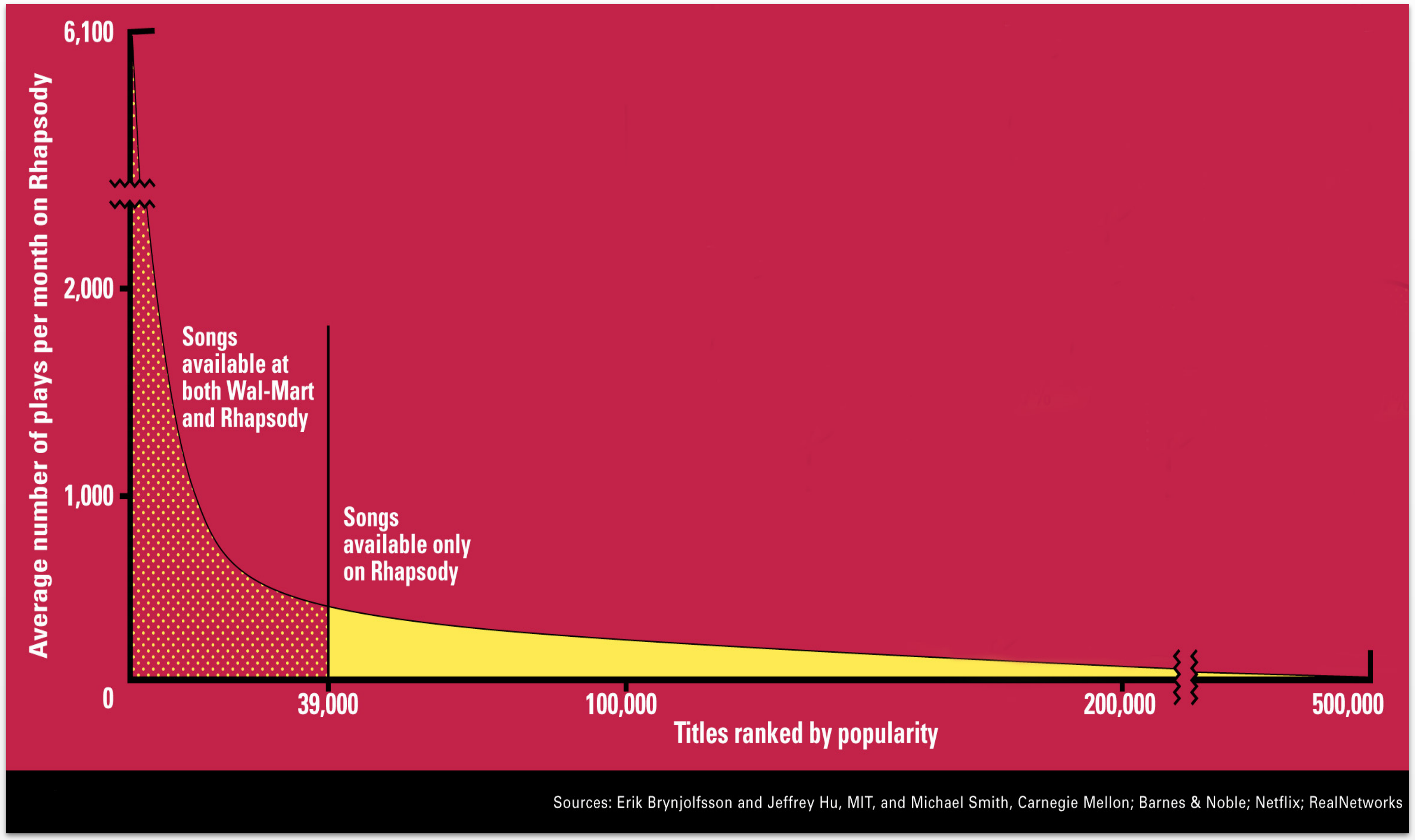
Recommender Systems

Shantanu Jain

Recommender Systems

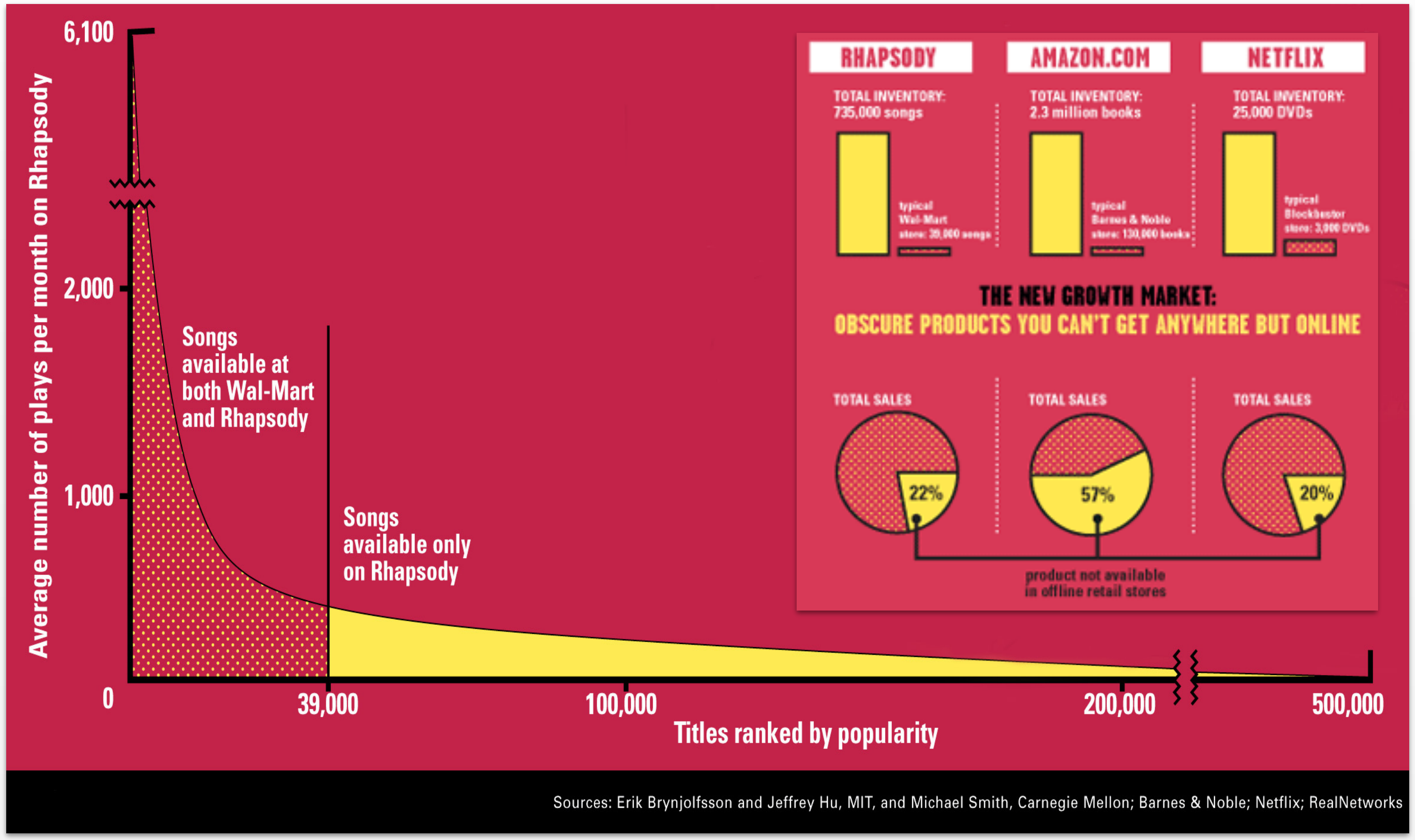
Reasoning about the Long Tail

The Long Tail



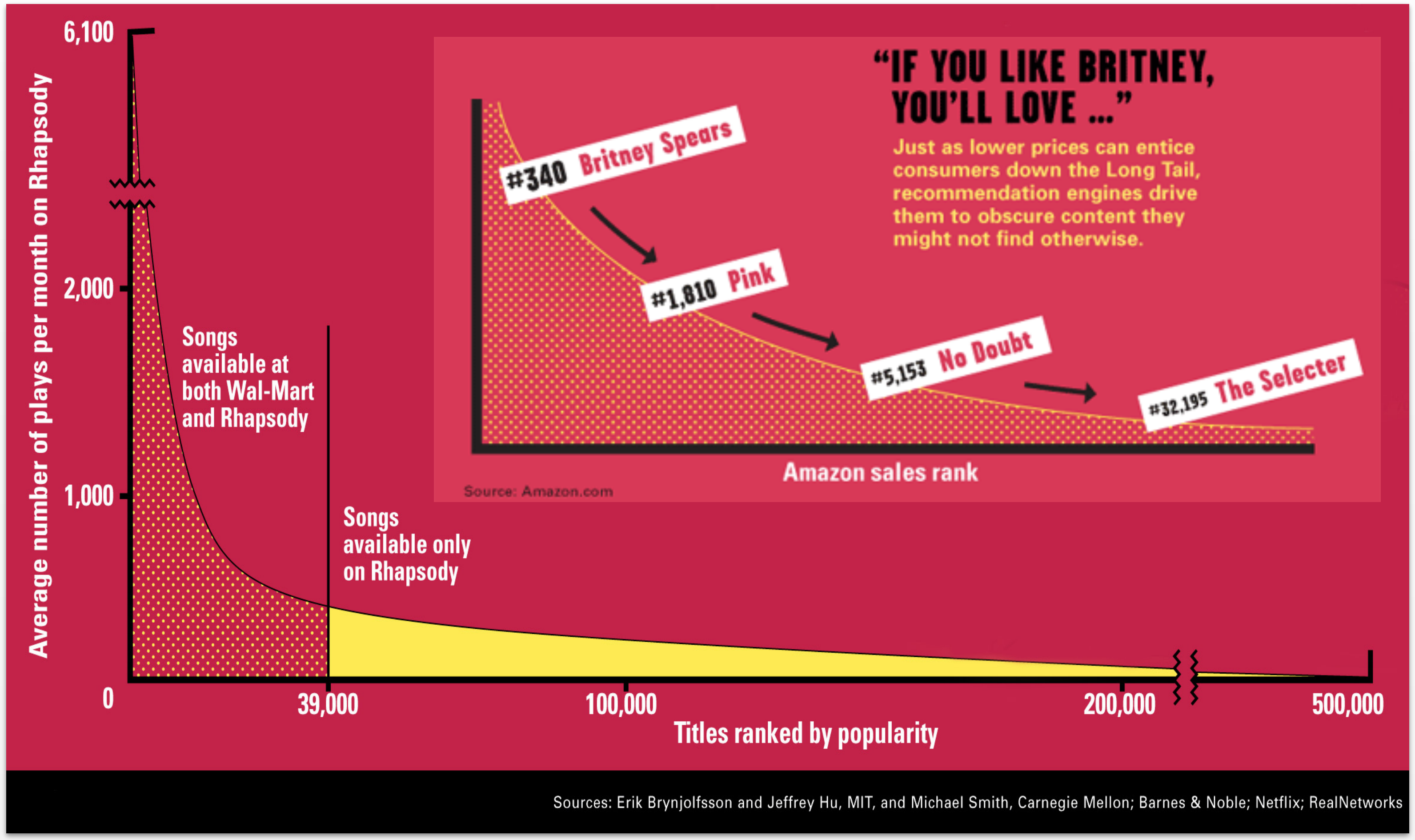
(from: https://www.wired.com/2004/10/tail/)

The Long Tail



(from: https://www.wired.com/2004/10/tail/)

The Long Tail

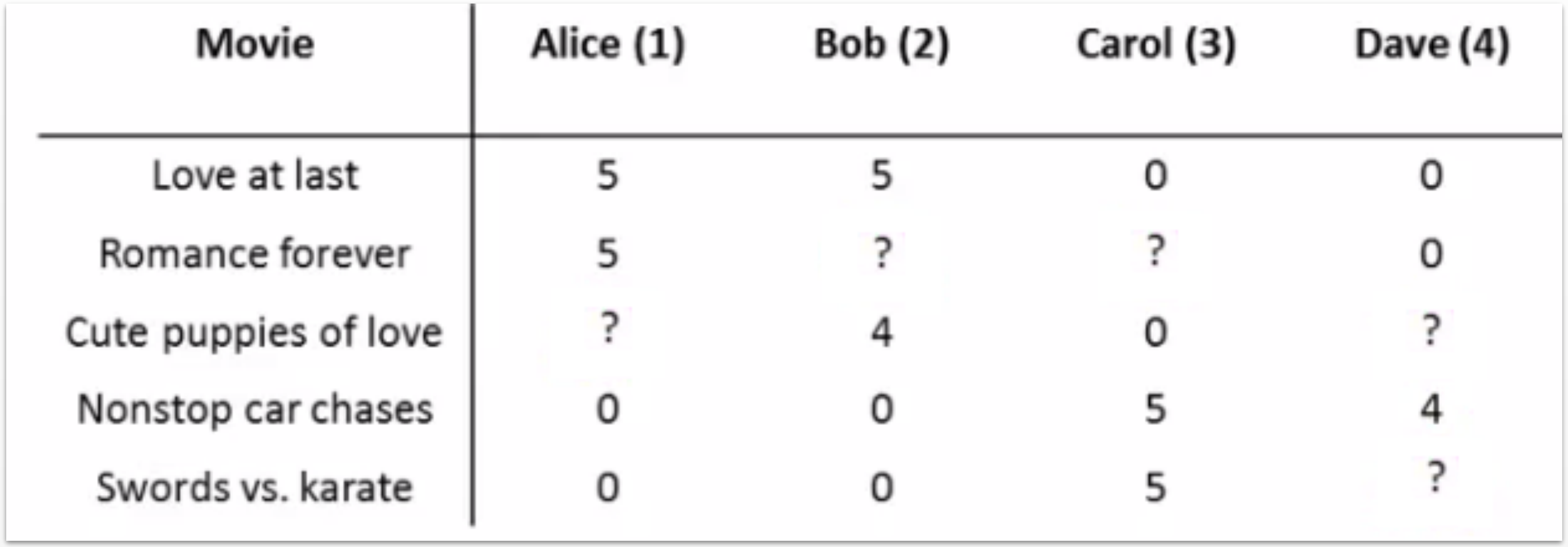


(from: https://www.wired.com/2004/10/tail/)

Applications of Recommender Systems

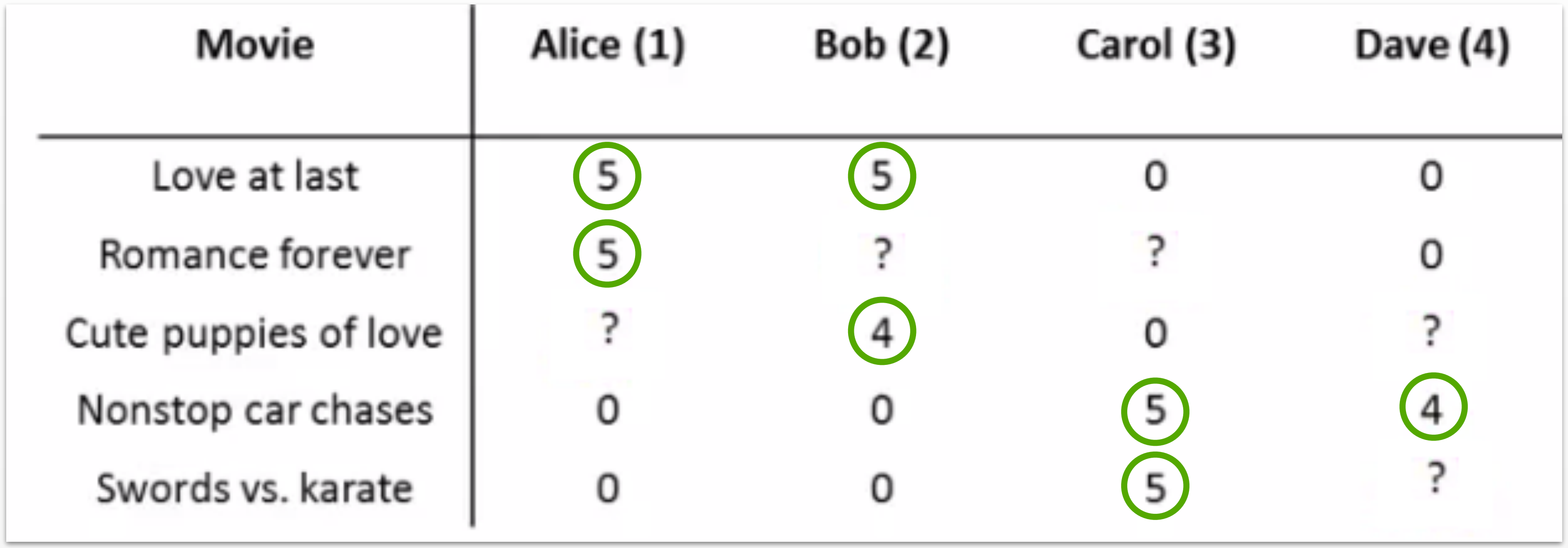
- Movie recommendation (Netflix)
- Related product recommendation (Amazon)
- Web page ranking (Google)
- Social recommendation (Facebook)
- Priority inbox & spam filtering (Google)
- Online dating (OK Cupid)
- Computational Advertising (Everyone)

Problem Setting



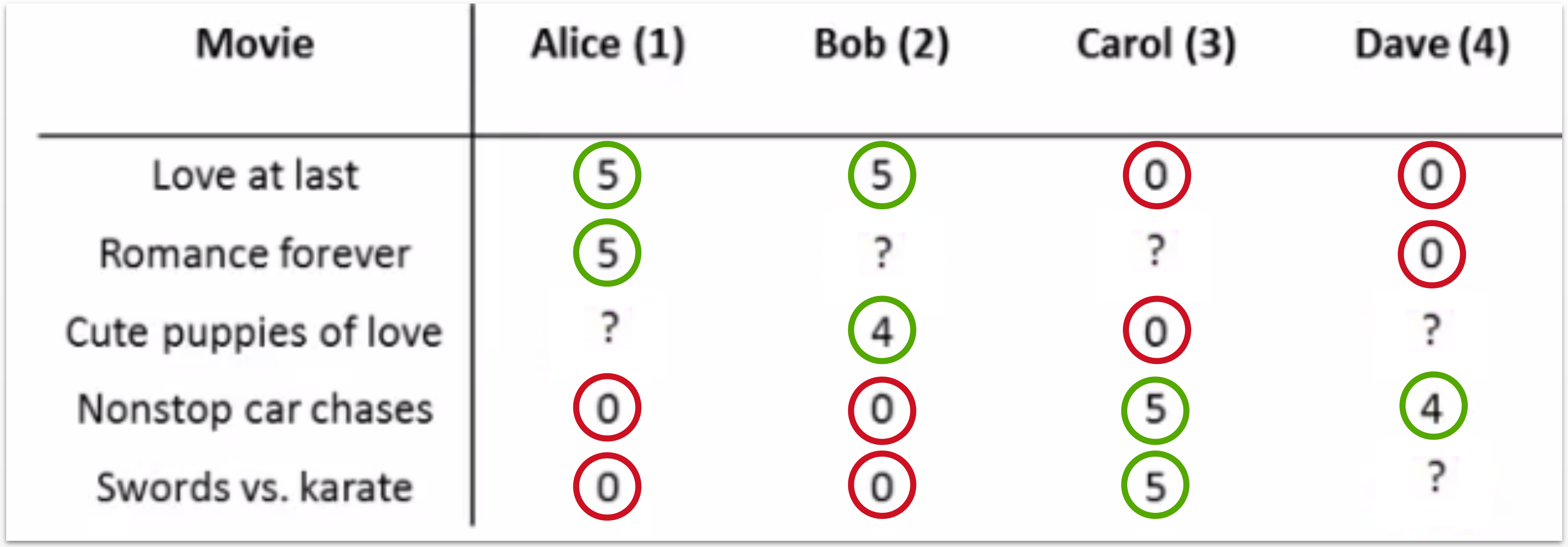
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

Problem Setting



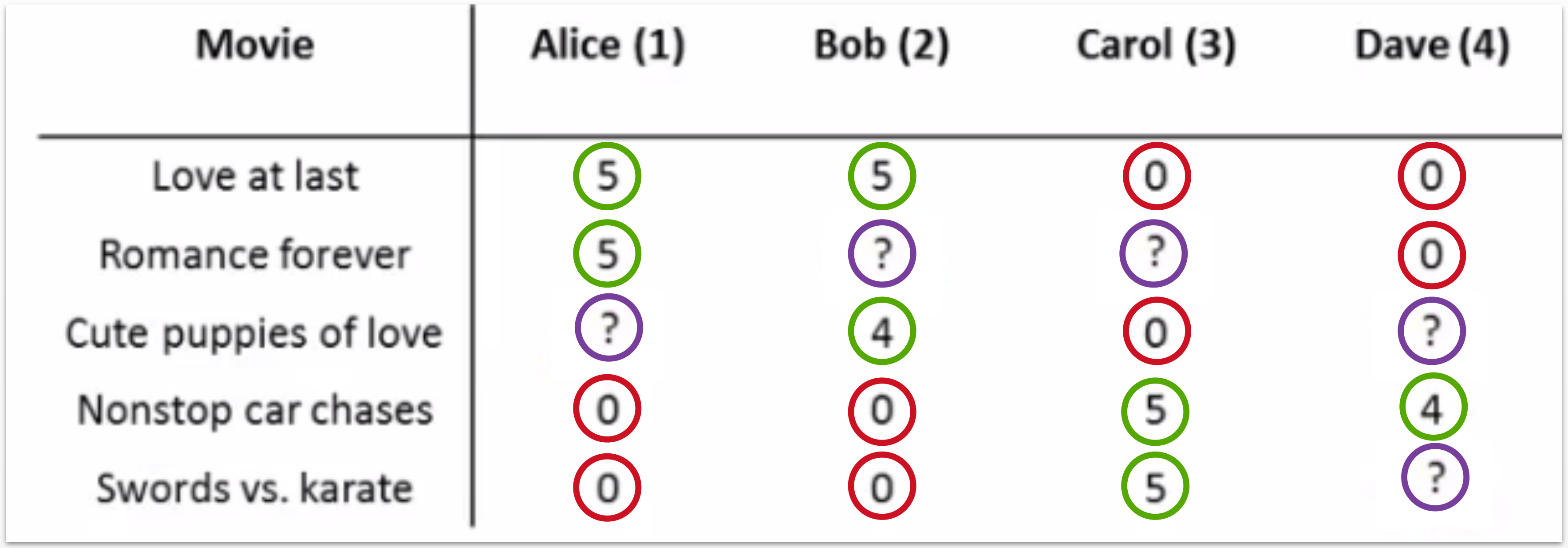
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

Problem Setting



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

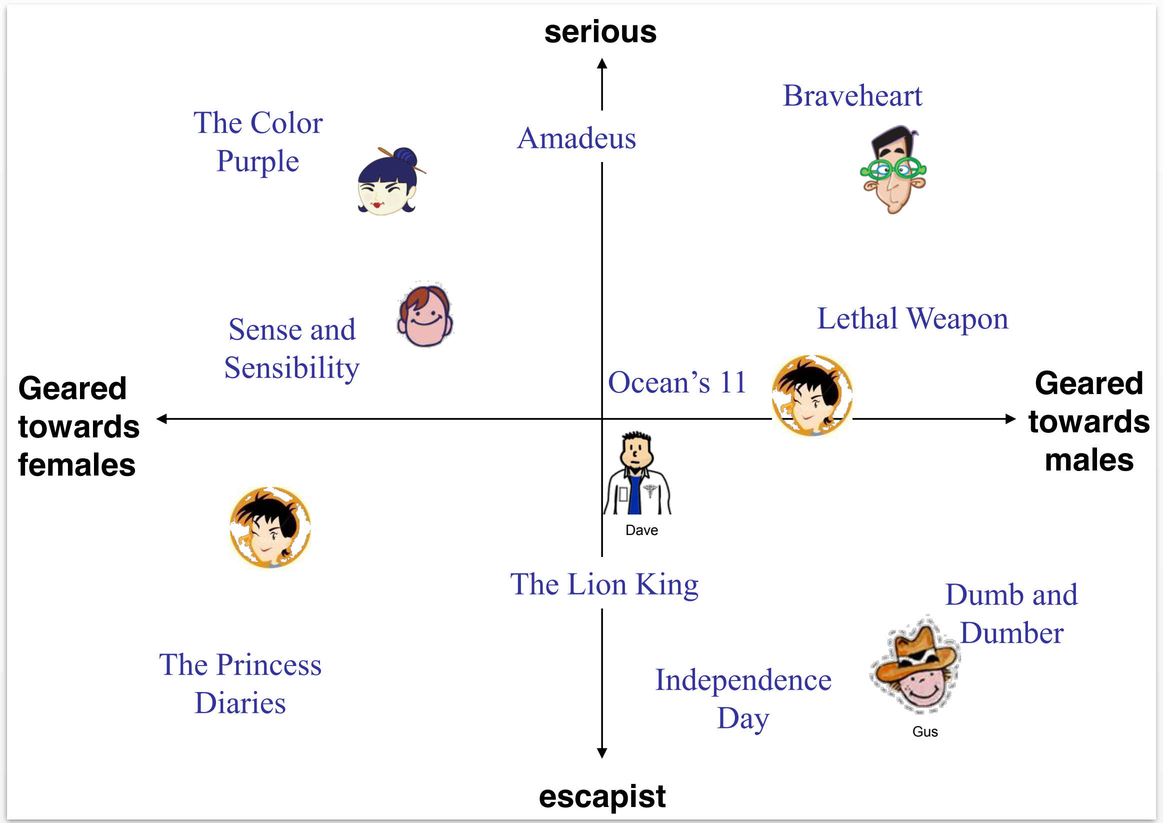
Problem Setting



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

- *Task*: Predict user preferences for unseen items

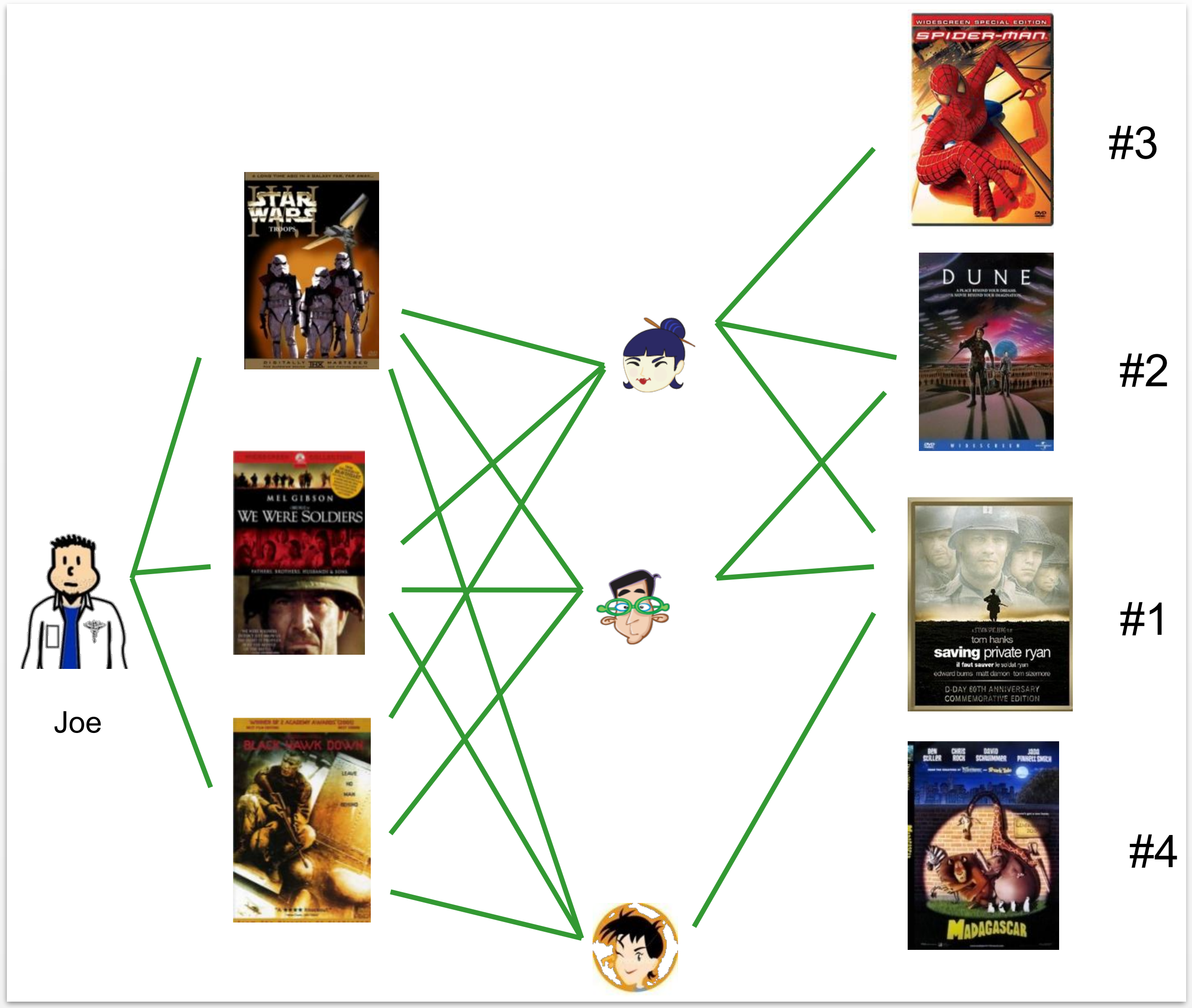
Content-based Filtering



Two Approaches:

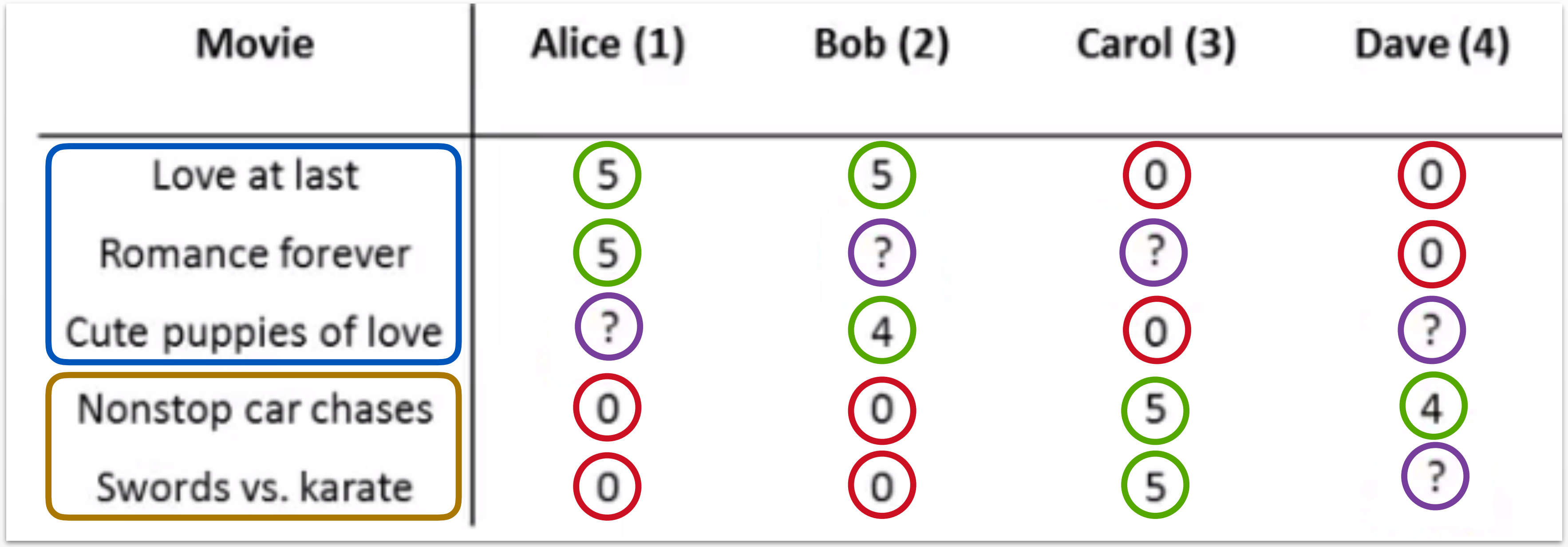
1. Predict rating using ***item*** features on a ***per-user*** basis
2. Predict rating using ***user*** features on a ***per-item*** basis

Collaborative Filtering



Idea: Predict rating based on similarity to other users

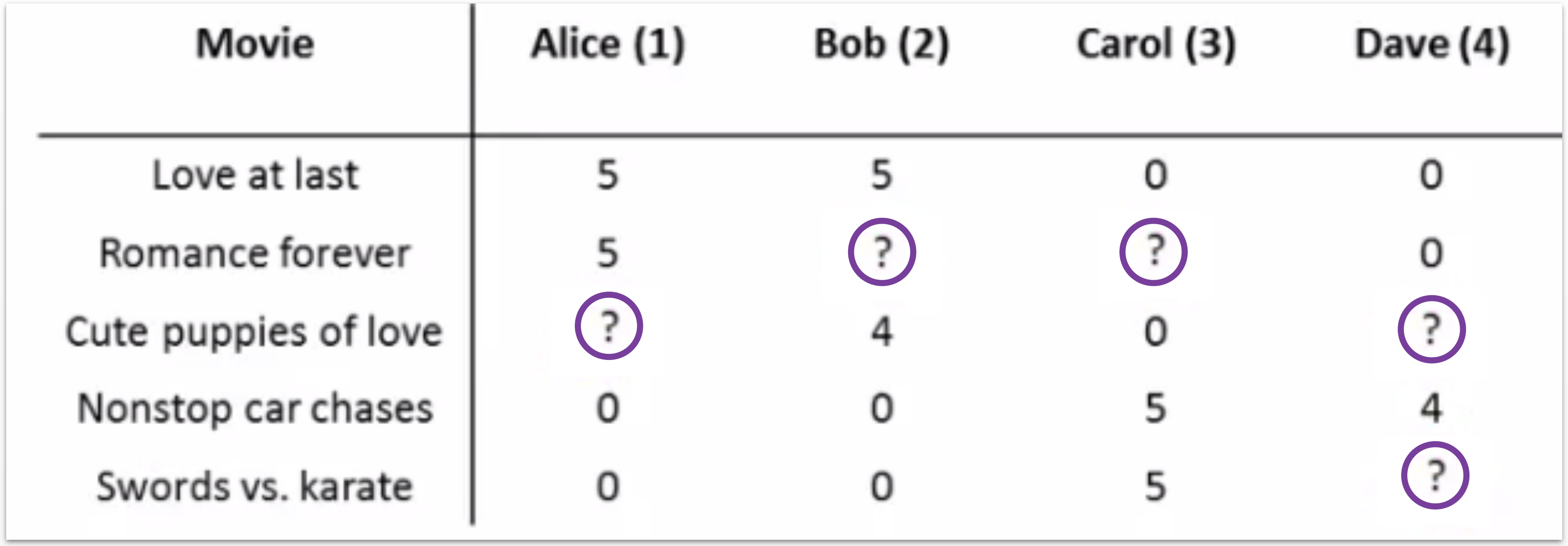
Problem Setting



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

- *Task*: Predict user preferences for unseen items
- *Content-based filtering*: Model user/item features
- *Collaborative filtering*: Implicit similarity of users or items

Running Yardstick: RMSE



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

$$\mathrm{rmse}(S) = \sqrt{|S|^{-1} \sum_{(i,u)\in S} (\hat{r}_{ui} - r_{ui})^2}$$

S contains user-item pairs for which ratings are observed

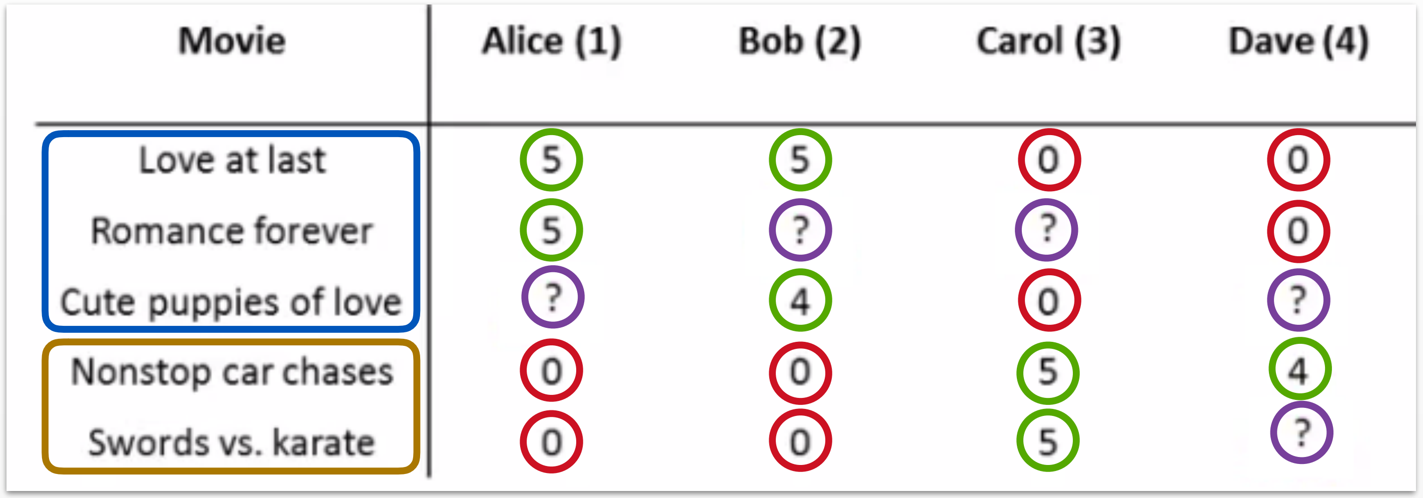
Recommender Systems

Shantanu Jain

Content-based Filtering

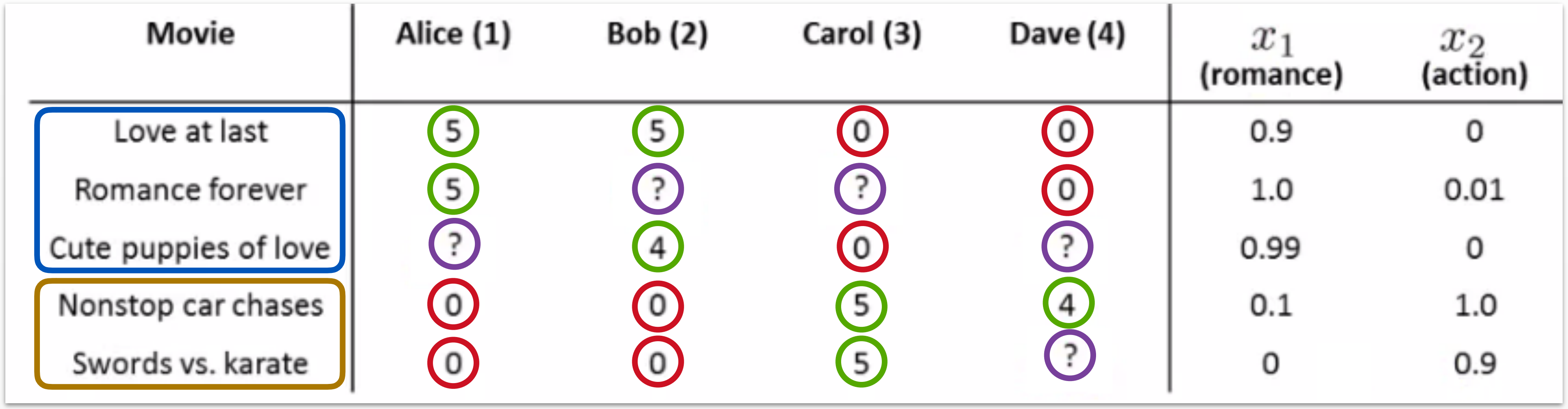
Feature-based recommendation

Item-based Features



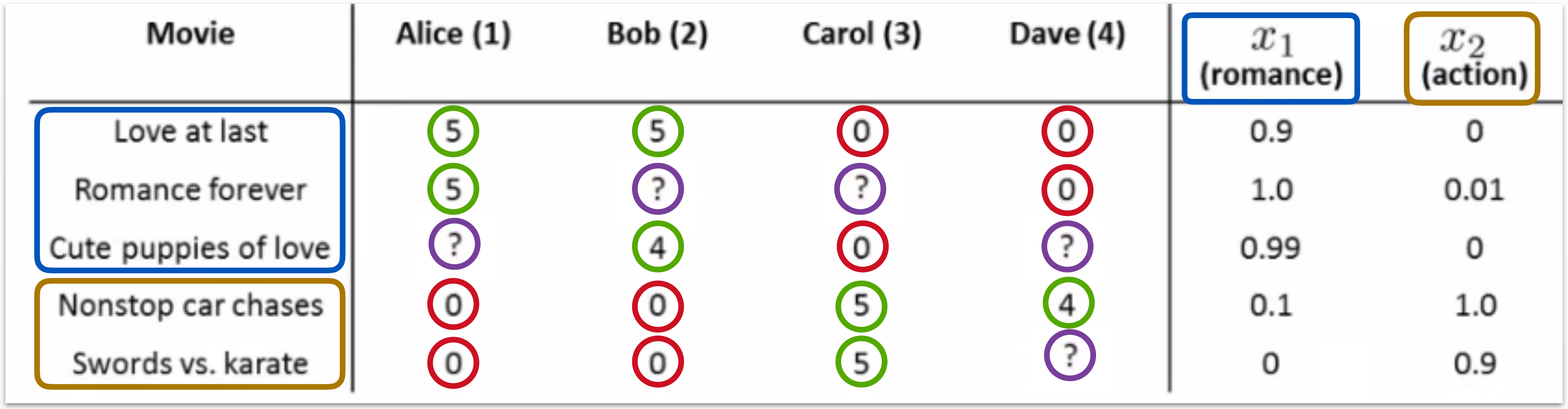
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

Item-based Features



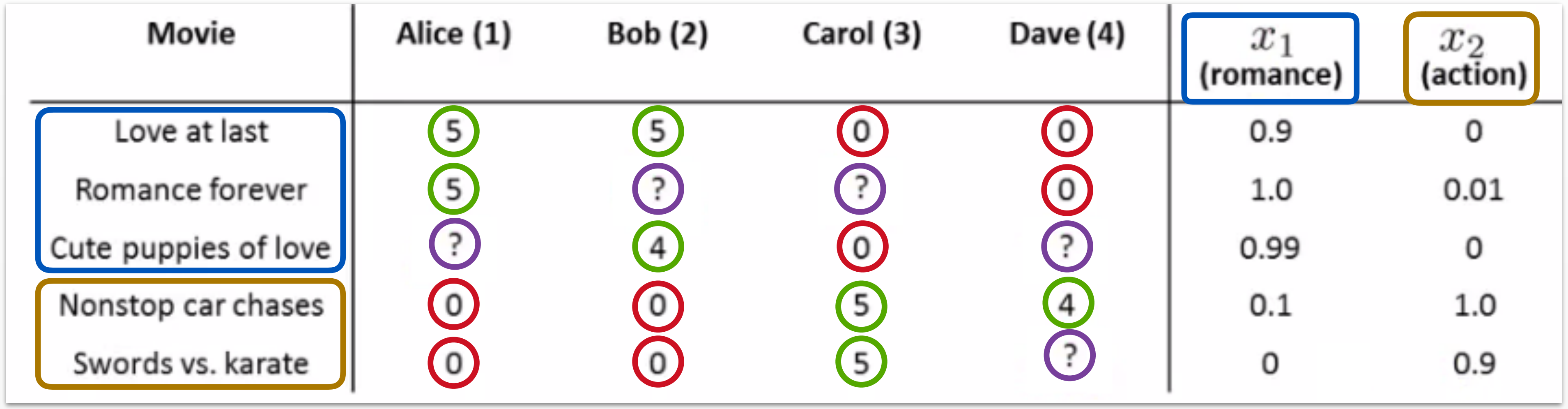
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Item-based Features



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Per-user Regression



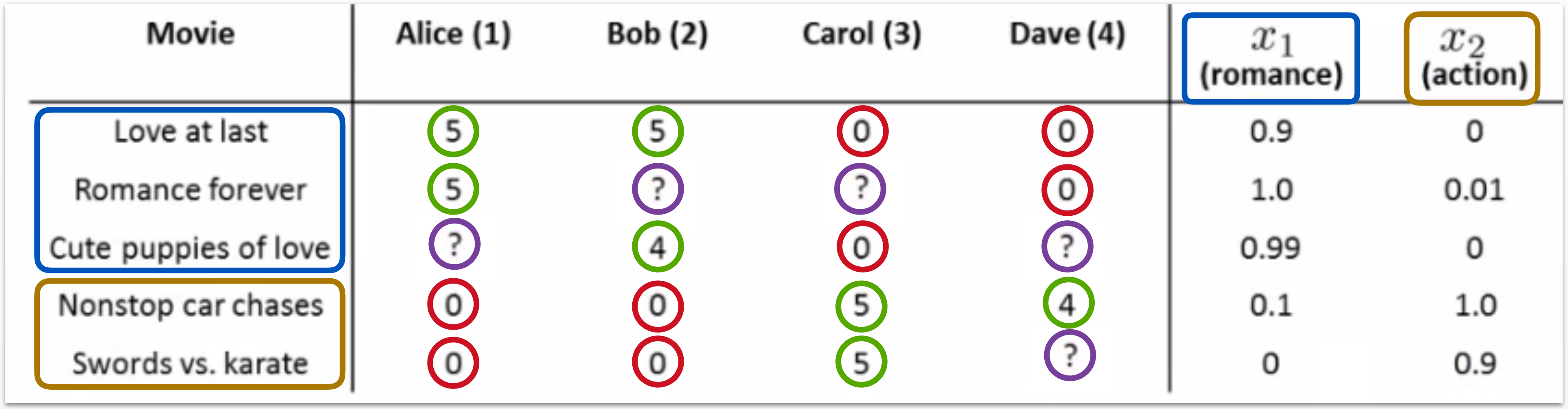
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Learn a set of regression coefficients for each user

$$\boldsymbol{w}_u = \underset{\boldsymbol{w}}{\operatorname{argmin}} |\boldsymbol{r}_u - \boldsymbol{X}\boldsymbol{w}|^2 \qquad w_u = (X^T X)^{-1} X^T r_u$$

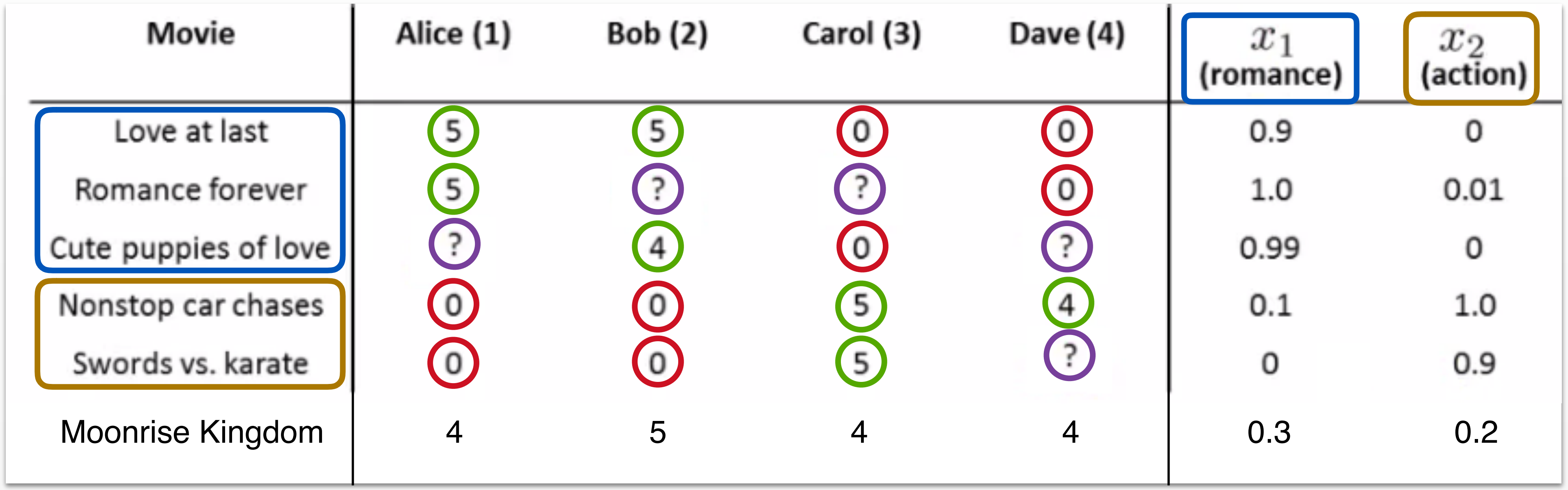
Each row of X contains an encoding for an item.

Bias



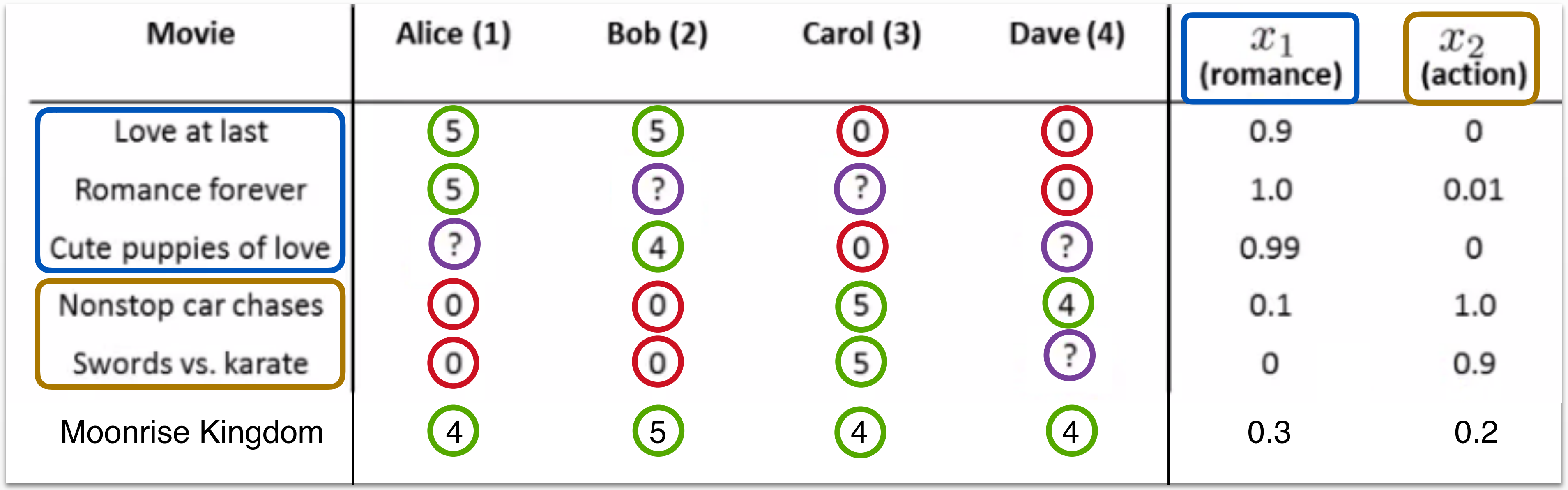
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Bias



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9
Moonrise Kingdom	4	5	4	4	0.3	0.2

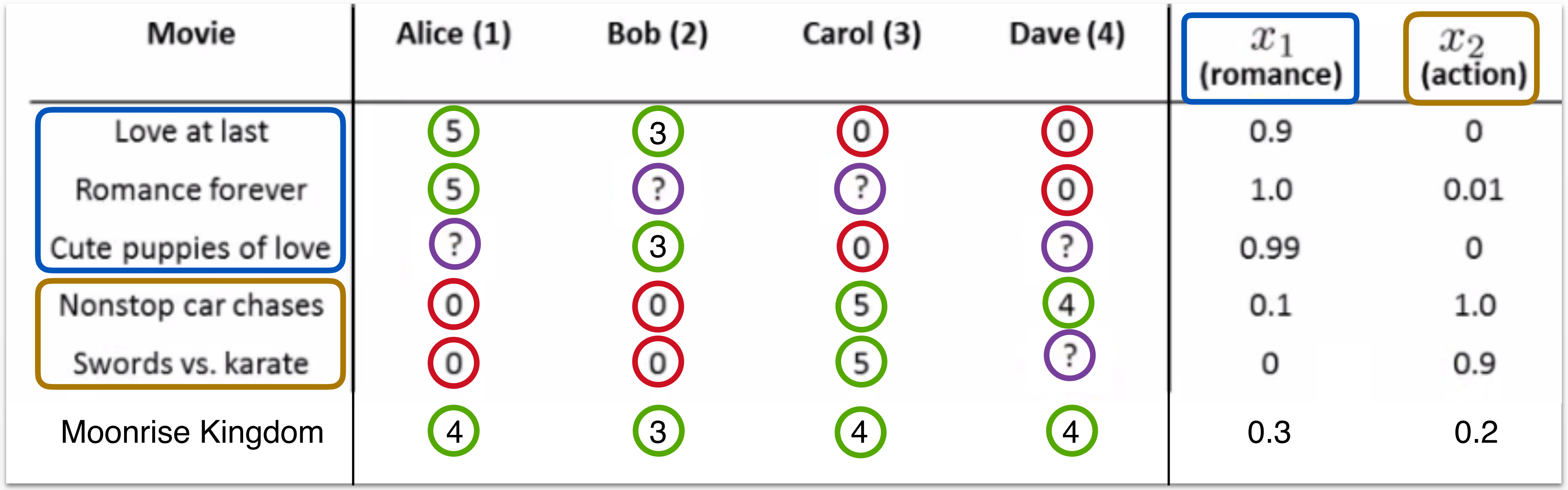
Bias



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9
Moonrise Kingdom	4	5	4	4	0.3	0.2

Problem: Some movies are universally loved / hated

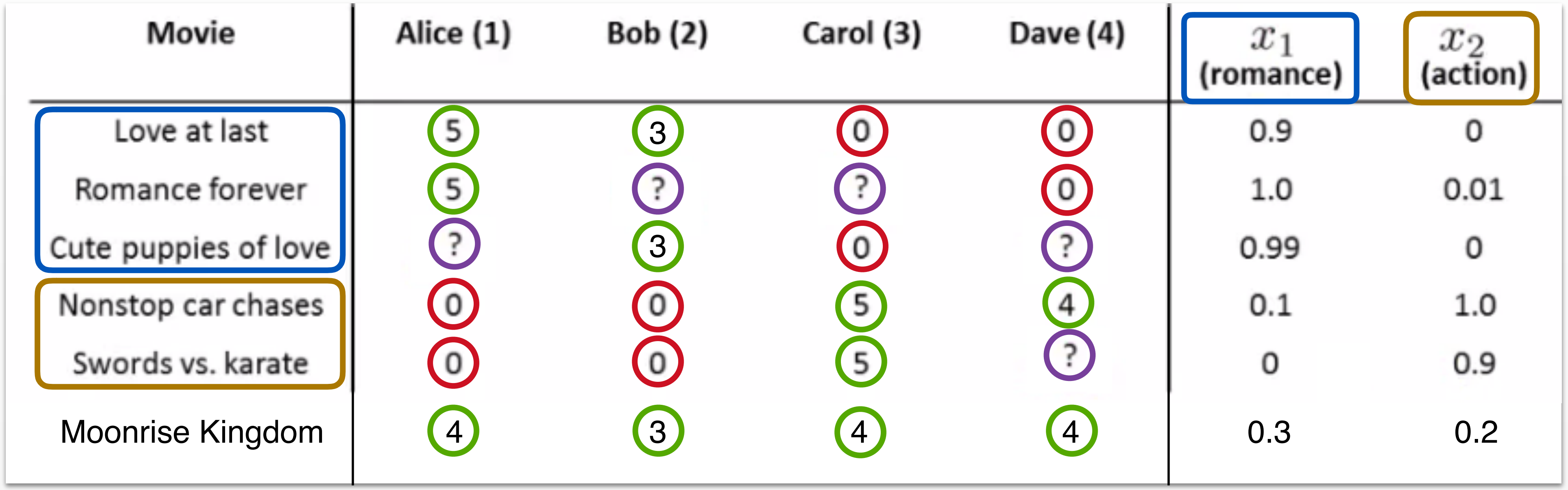
Bias



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	3	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	3	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9
Moonrise Kingdom	4	3	4	4	0.3	0.2

Problem: Some movies are universally loved / hated
some users are more picky than others

Bias



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	3	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	3	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9
Moonrise Kingdom	4	3	4	4	0.3	0.2

Problem: Some movies are universally loved / hated some users are more picky than others

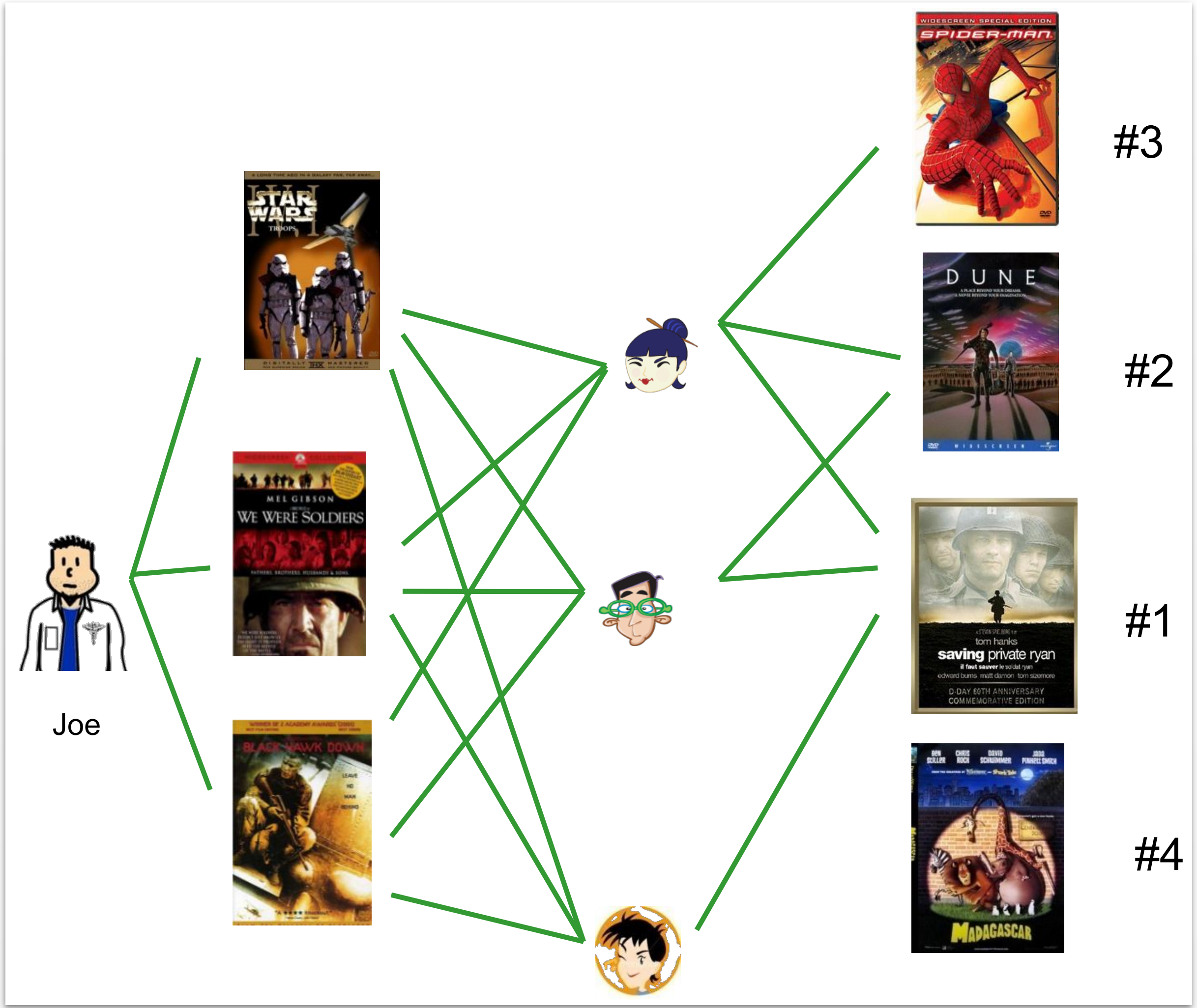
Solution: Introduce a per-movie and per-user bias

$$\hat{r}_{ui} = \mu + b_u + b_i + \boldsymbol{x}_i^\top \boldsymbol{w}_u$$

Collaborative Filtering

Connectivity-based recommendation

Neighborhood Based Methods



Users and items form a bipartite graph (edges are ratings)

Neighborhood Based Methods

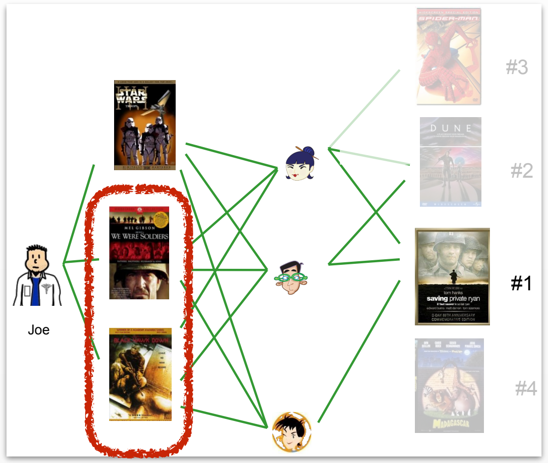
(user, user) similarity

- predict rating based on average from k-nearest users
- good if item base changes rapidly

(item,item) similarity

- predict rating based on average from k-nearest items
- good if user base changes rapidly

Parzen-Window Style CF



$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in \epsilon_k(i,u)} s_{ij}(r_{uj} - b_{uj})}{\sum_{j \in \epsilon_k(i,u)} |s_{ij}|}$$

$$b_{ui} = \mu + b_u + b_i$$

- Define a similarity s_{ij} between items
- Find set $\varepsilon_k(i,u)$ of k-nearest neighbors to i that were rated by user u
- Predict rating using weighted average over set
- How should we define s_{ij}?

Pearson Correlation Coefficient

User ratings for item i:

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

User ratings for item j:

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

$$s_{ij} = \frac{\text{cov}(r_{\cdot i}, r_{\cdot j})}{\text{std}(r_{\cdot i}) \times \text{std}(r_{\cdot j})}$$

(item,item) similarity

Empirical estimate of Pearson correlation coefficient

$$\hat{\rho}_{ij} = \frac{\sum_{u \in U(i,j)} (r_{ui} - b_{ui})(r_{uj} - b_{uj})}{\sqrt{\sum_{u \in U(i,j)} (r_{ui} - b_{ui})^2 \sum_{u \in U(i,j)} (r_{uj} - b_{uj})^2}}$$

U(i, j): set of users who have rated both *i* and *j*

Regularize towards 0 for small support

$$s_{ij} = \frac{|U(i,j)| - 1}{|U(i,j)| - 1 + \lambda} \hat{\rho}_{ij}$$

Regularize towards baseline for small neighborhood

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in \epsilon_k(i,u)} s_{ij}(r_{uj} - b_{uj})}{\lambda + \sum_{j \in \epsilon_k(i,u)} |s_{ij}|}$$

Similarity for binary labels

Pearson correlation not meaningful for binary labels
(e.g. Views, Purchases, Clicks)

Jaccard similarity

$$s_{ij} = \frac{m_{ij}}{\alpha + m_i + m_j - m_{ij}}$$

Observed / Expected ratio

$$s_{ij} = \frac{\text{observed}}{\text{expected}} \approx \frac{m_{ij}}{\alpha + m_i m_j / m}$$

m_i users acting on i

m_{ij} users acting on both i and j

m total number of users

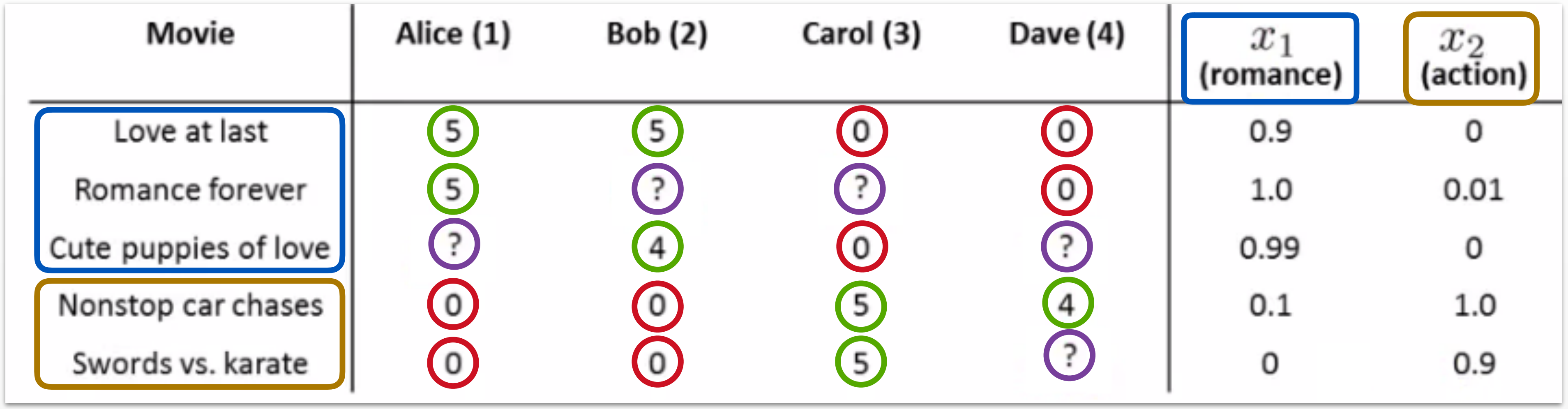
Recommender Systems

Shantanu Jain

Matrix Factorization Methods

Learning user and item features

Content based filtering



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

$$\hat{r}_{ui} = x_i^T w_u$$

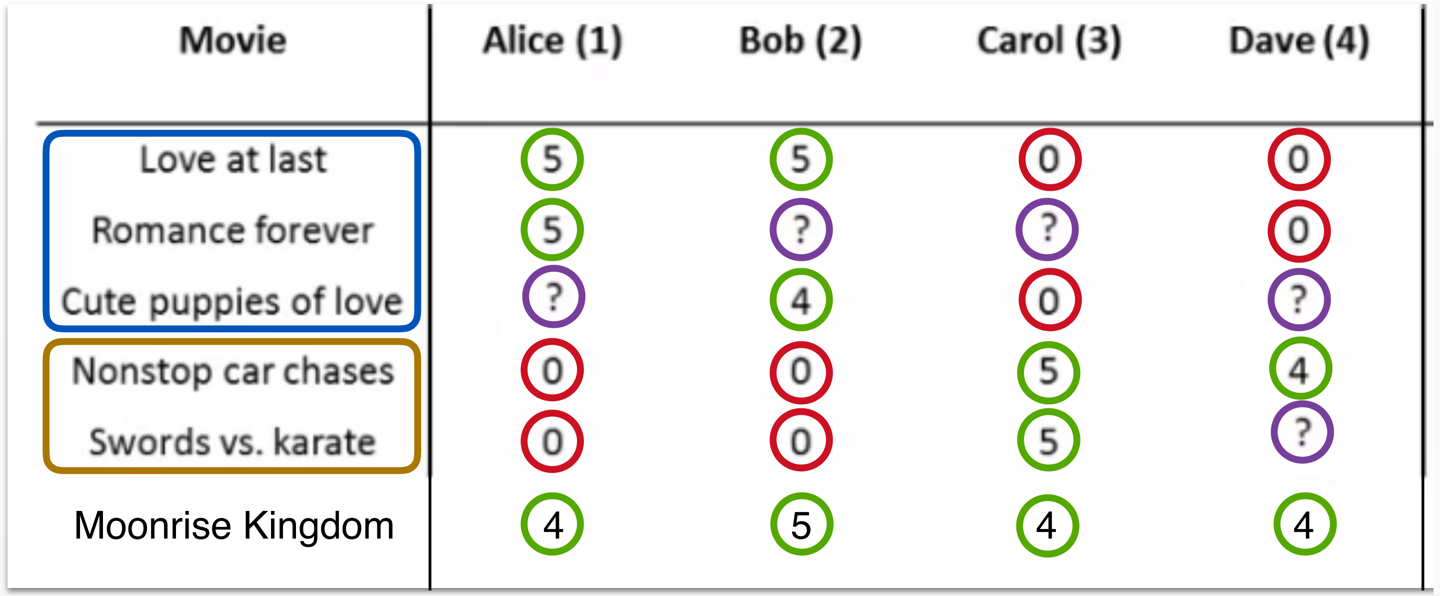
Learn a set of regression coefficients for each user separately

$$\boldsymbol{w}_u = \underset{\boldsymbol{w}}{\operatorname{argmin}} |\boldsymbol{r}_u - \boldsymbol{X}\boldsymbol{w}|^2$$

Solution: $w_u = (X^T X)^{-1} X^T r_u$

Each row of X contains an encoding for an item.

Matrix Factorization (SVD)



Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?
Moonrise Kingdom	4	5	4	4

This approach is referred to as SVD in the recommendation system literature, but it is not the SVD as defined in linear algebra

The predicted rating is an inner product of item and user embeddings

$$\hat{r}_{ui} = x_i^T w_u$$

In matrix form:

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

$$\hat{X}, \hat{W} = \text{argmin}_{X,W} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui})^2$$

Matrix Factorization

users

items

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

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

X

$\bullet$

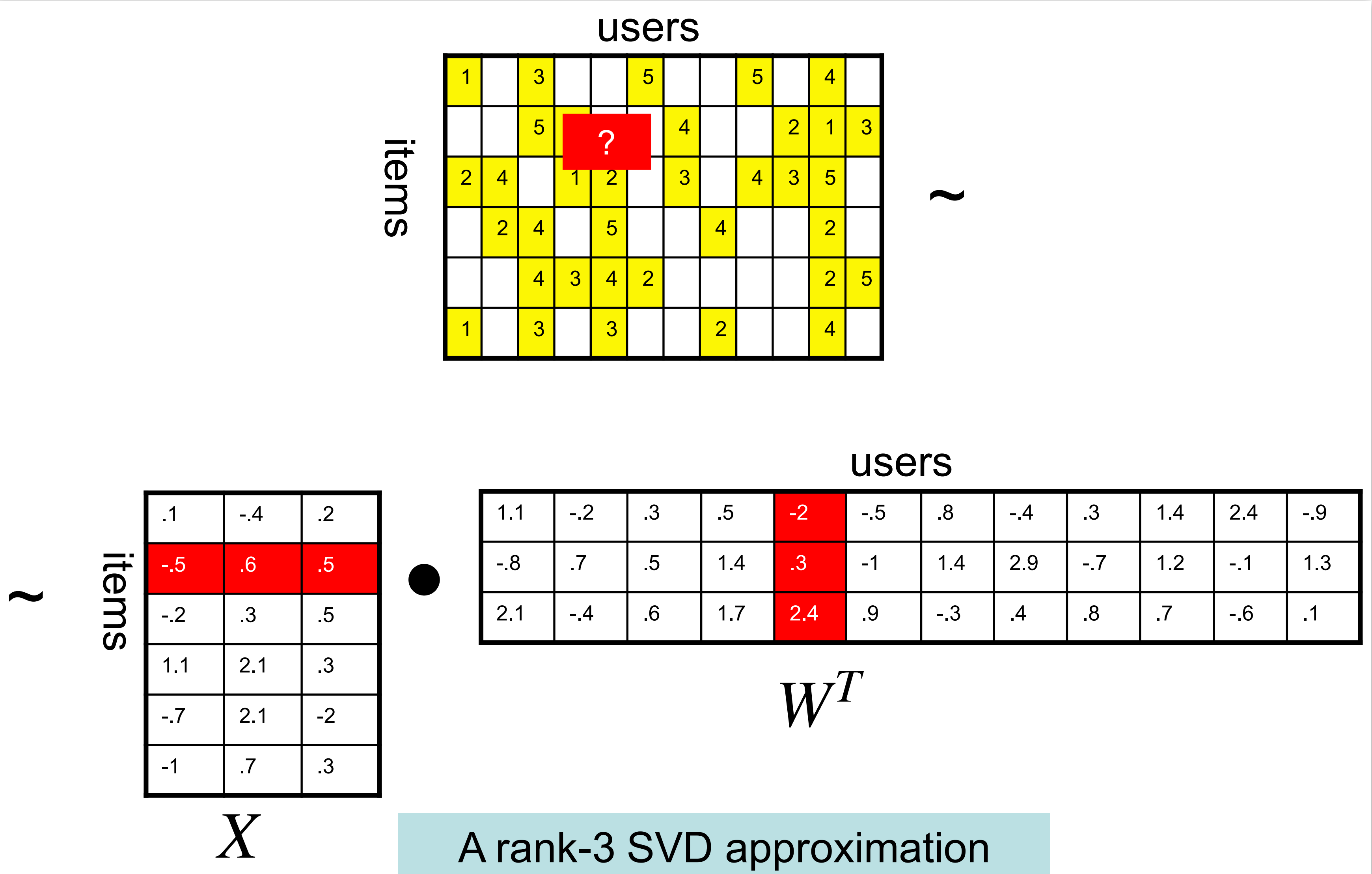
users

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

W^T

A rank-3 SVD approximation

Prediction



$$
\begin{array}{c|cccccccccccc}
 & \multicolumn{12}{c}{\text{users}} \\
\hline
\text{items} & 1 & & 3 & & & 5 & & & 5 & & 4 & \\
 & & & 5 & & ? & & 4 & & & 2 & 1 & 3 \\
 & 2 & 4 & & 1 & 2 & & 3 & & 4 & 3 & 5 & \\
 & & 2 & 4 & & 5 & & & 4 & & & 2 & \\
 & & & 4 & 3 & 4 & 2 & & & & & 2 & 5 \\
 & 1 & & 3 & & 3 & & & 2 & & & 4 &
\end{array}
\sim
$$

$$
\sim
\underbrace{
\begin{array}{c|ccc}
\text{items} & .1 & -.4 & .2 \\
 & -.5 & .6 & .5 \\
 & -.2 & .3 & .5 \\
 & 1.1 & 2.1 & .3 \\
 & -.7 & 2.1 & -2 \\
 & -1 & .7 & .3
\end{array}
}_{X}
\bullet
\underbrace{
\begin{array}{cccccccccccc}
\multicolumn{12}{c}{\text{users}} \\
\hline
1.1 & -.2 & .3 & .5 & -2 & -.5 & .8 & -.4 & .3 & 1.4 & 2.4 & -.9 \\
-.8 & .7 & .5 & 1.4 & .3 & -1 & 1.4 & 2.9 & -.7 & 1.2 & -.1 & 1.3 \\
2.1 & -.4 & .6 & 1.7 & 2.4 & .9 & -.3 & .4 & .8 & .7 & -.6 & .1
\end{array}
}_{W^T}
$$

A rank-3 SVD approximation

Prediction

users

items

1		3			5			5		4	
		5	[illegible]	2.4		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

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

X

●

users

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

W^T

A rank-3 SVD approximation

Matrix Factorization

$$\hat{X}, \hat{W} = \text{argmin}_{X,W} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui})^2$$

Initialize X and W randomly

Iterate till convergence

1. $W^{(t+1)} = \text{argmin}_{W} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

2. $X^{(t+1)} = \text{argmin}_{X} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

Matrix Factorization

1. $W^{(t+1)} = \text{argmin}_W \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{w}_u = (X_{S_u}^T X_{S_u})^{-1} X_{S_u}^T r_{S_u}$$

X_{S_u}: contains only those items rated by u. Each row is an item encoding

r_{S_u}: vector of ratings observed for user u

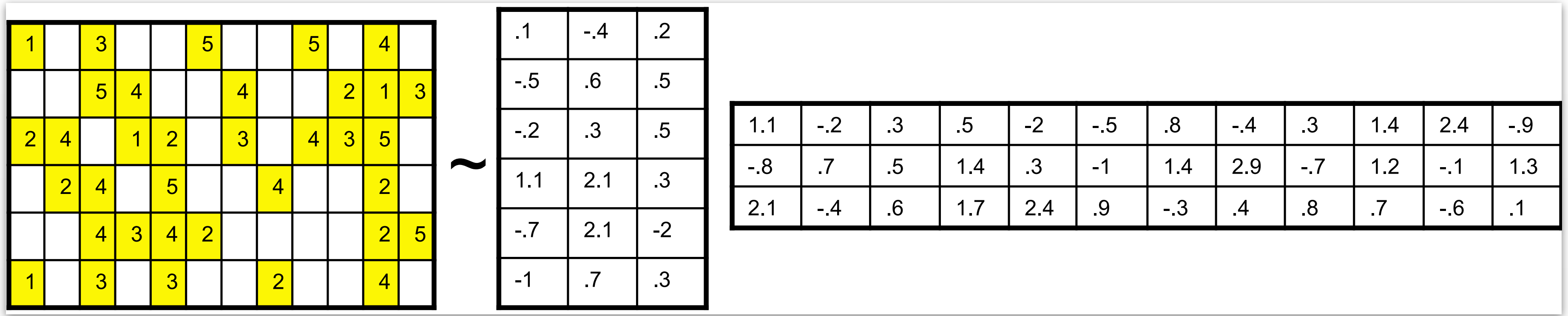
2. $X^{(t+1)} = \text{argmin}_X \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{x}_i = (W_{S_i}^T W_{S_i})^{-1} W_{S_i}^T r_{S_i}$$

W_{S_i}: contains only those users that have rated item i. Each row is a user encoding

r_{S_i}: vector of ratings observed for item i

SVD with missing values



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	

~

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

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

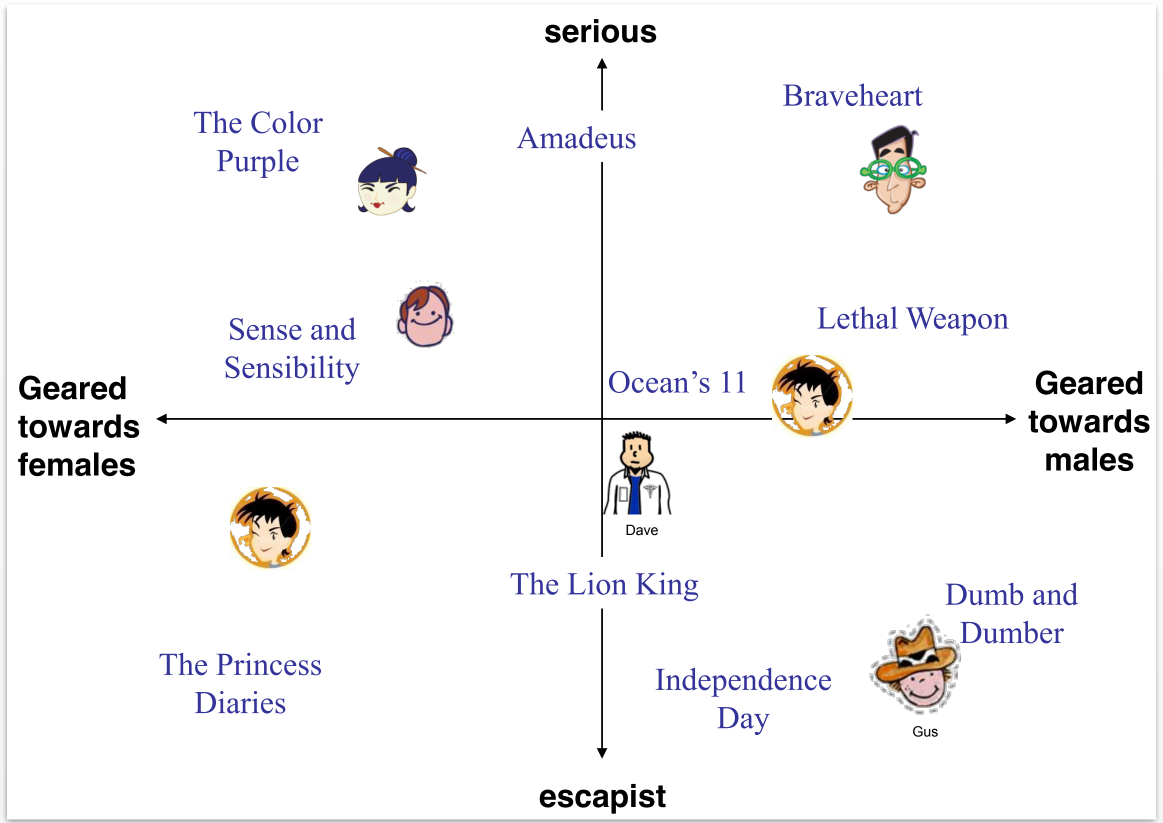
Pose as regression problem

$$\hat{X}, \hat{W} = \text{argmin}_{X,W} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\| X \|_F^2 + \| W \|_F^2)$$

Regularize using Frobenius norm

$$||A||_{\mathrm{F}}^2 = \sum_{ij} |A_{ij}|^2$$

Need for regularization



Matrix Factorization

1. $W^{(t+1)} = \text{argmin}_W \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{w}_u = (\lambda I + X_{S_u}^T X_{S_u})^{-1} X_{S_u}^T r_u$$

X_{S_u}: contains only those items rated by u. Each row is an item encoding

r_u: vector of ratings observed for user u

2. $X^{(t+1)} = \text{argmin}_X \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{x}_i = (\lambda I + W_{S_i}^T W_{S_i})^{-1} W_{S_i}^T r_i$$

W_{S_i}: contains only those users that have rated item i. Each row is a user encoding

r_i: vector of ratings observed for item i

Regularized matrix Factorization

1. $W^{(t+1)} = \text{argmin}_W \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{w}_u = (\lambda I + X_{S_u}^T X_{S_u})^{-1} X_{S_u}^T r_{S_u}$$

X_{S_u}: contains only those items rated by u. Each row is an item encoding

r_{S_u}: vector of ratings observed for user u

2. $X^{(t+1)} = \text{argmin}_X \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{x}_i = (\lambda I + W_{S_i}^T W_{S_i})^{-1} W_{S_i}^T r_{S_i}$$

W_{S_i}: contains only those users that have rated item i. Each row is a user encoding

r_{S_i}: vector of ratings observed for item i

How to pick λ and k

- Split the set of observed ratings into training set and a validation set
- Train with multiple values of (λ, k) pairs on the training set.
- Pick the one with the best performance on the validation set.

SVD: Adding the bias terms

$$\hat{r}_{ui} = \mu + b_u + b_i + \boldsymbol{x}_i^\top \boldsymbol{w}_u$$

$$\hat{\boldsymbol{R}} = \boldsymbol{B} + \boldsymbol{X}\boldsymbol{W}^\top$$

μ: (global) average of all ratings across all users and movies.
b_u: average difference between users $u's$ ratings and μ
b_i: average difference between item $i's$ ratings and μ

$$b_{ui} = \mu + b_u + b_i$$

$$B = [b_{ui}]$$

$$\hat{X}, \hat{W} = \text{argmin}_{X,W} \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\| X \|_F^2 + \| W \|_F^2)$$

Regularized matrix Factorization

1. $W^{(t+1)} = \text{argmin}_W \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{w}_u = (\lambda I + X_{S_u}^T X_{S_u})^{-1} X_{S_u}^T (r_{S_u} - b_{S_u})$$

B_{S_u}: vector of baseline ratings $\mu + b_u + b_i$ for those items where user u gives a rating.

2. $X^{(t+1)} = \text{argmin}_X \sum_{(u,i)\in S} (r_{ui} - \hat{r}_{ui}^{(t)})^2$

$$\hat{x}_i = (\lambda I + W_{S_i}^T W_{S_i})^{-1} W_{S_i}^T (r_{S_i} - b_{S_i})$$

b_{S_i}: vector of baseline ratings $\mu + b_u + b_i$ on item i, only for those users that indeed rate item i.

Netflix Prize

Training data

- 100 million ratings, 480,000 users, 17,770 movies
- 6 years of data: 2000-2005

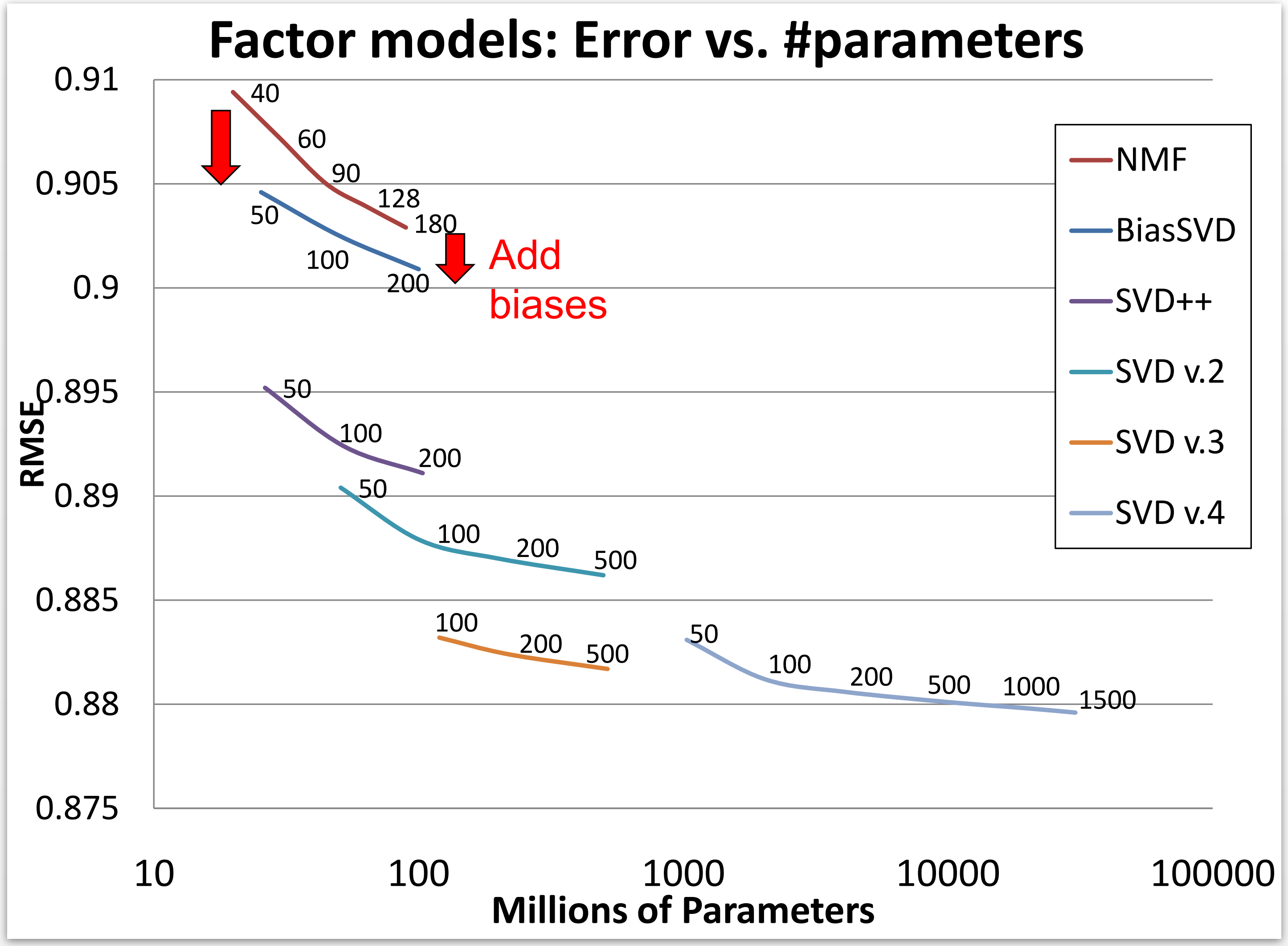
Test data

- Last few ratings of each user (2.8 million)
- Evaluation criterion: Root Mean Square Error (RMSE)

Competition

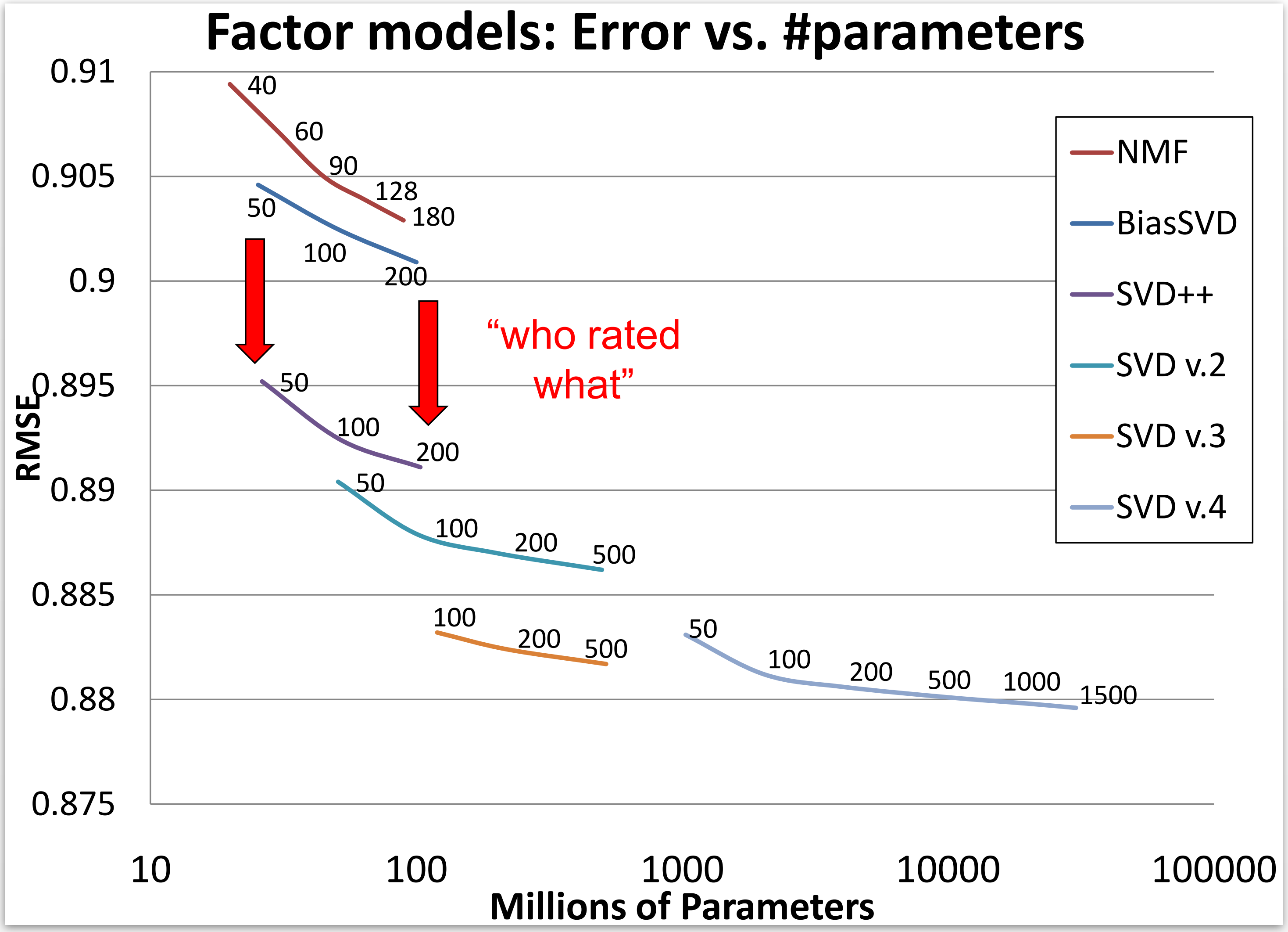
- 2,700+ teams
- Netflix's system RMSE: 0.9514
- $1 million prize for 10% improvement on Netflix

Improvements



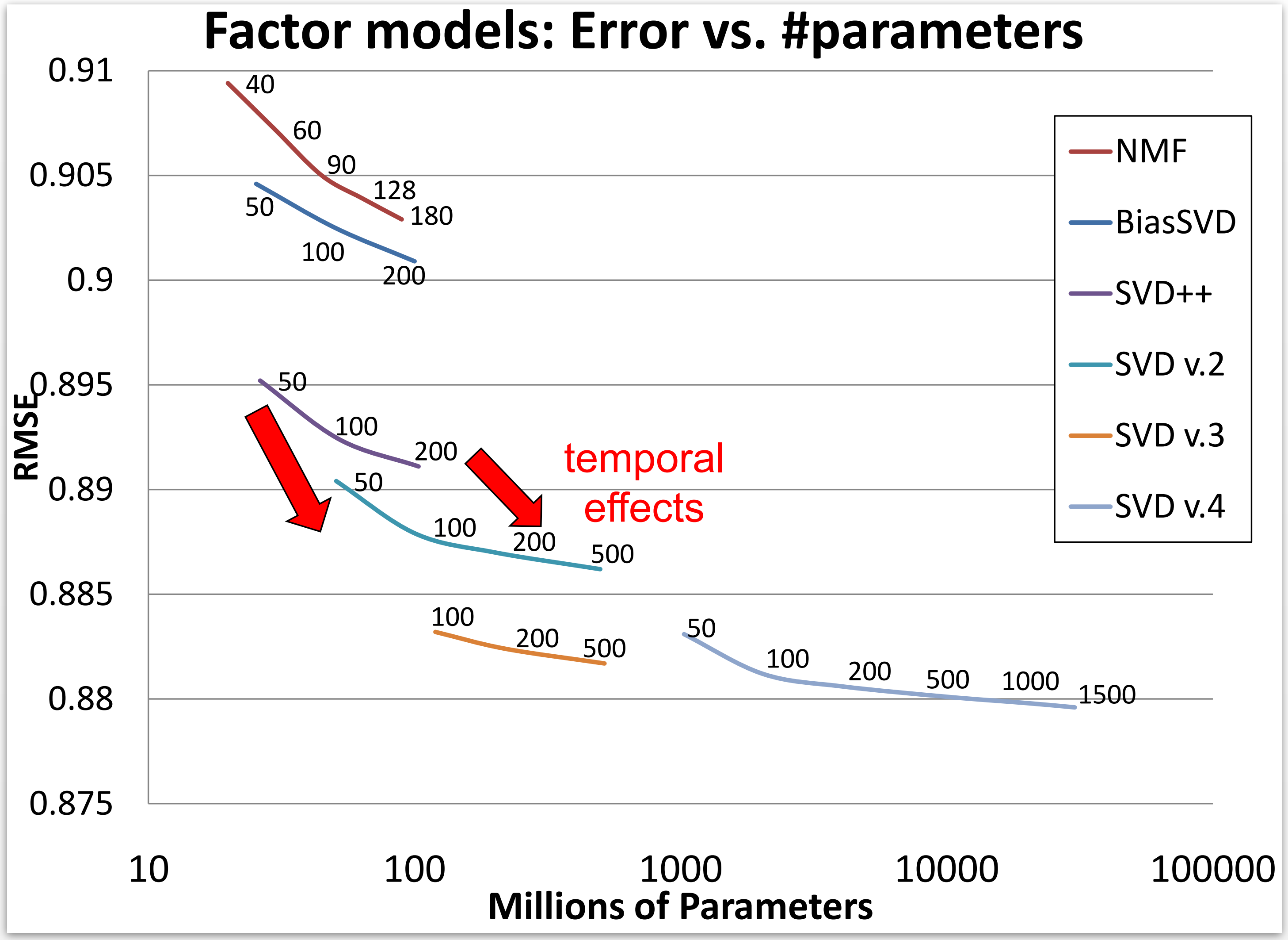
Do SGD, but also learn biases μ, b_u and b_i

Improvements



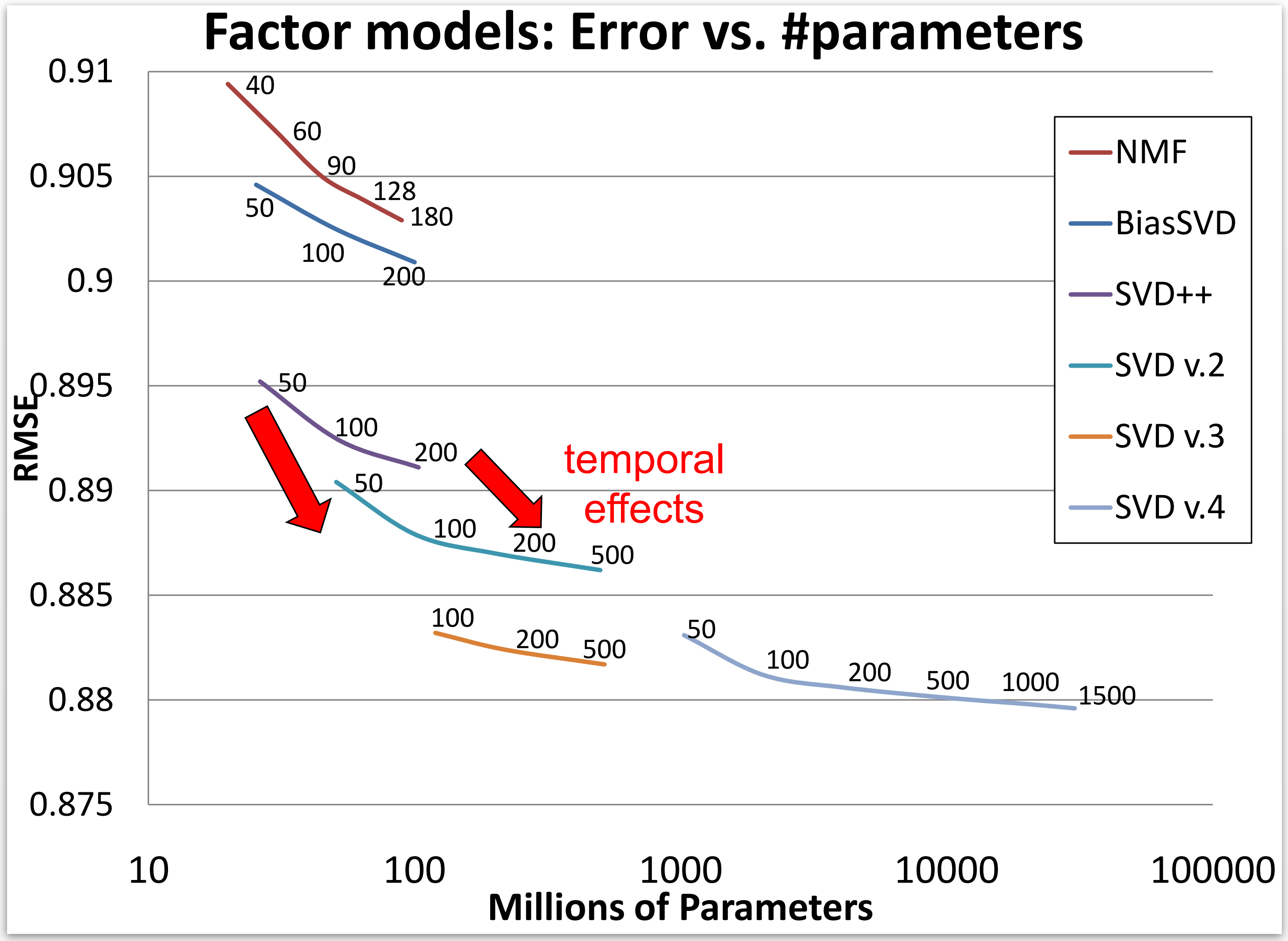
Account for fact that ratings are not missing at random.

Improvements



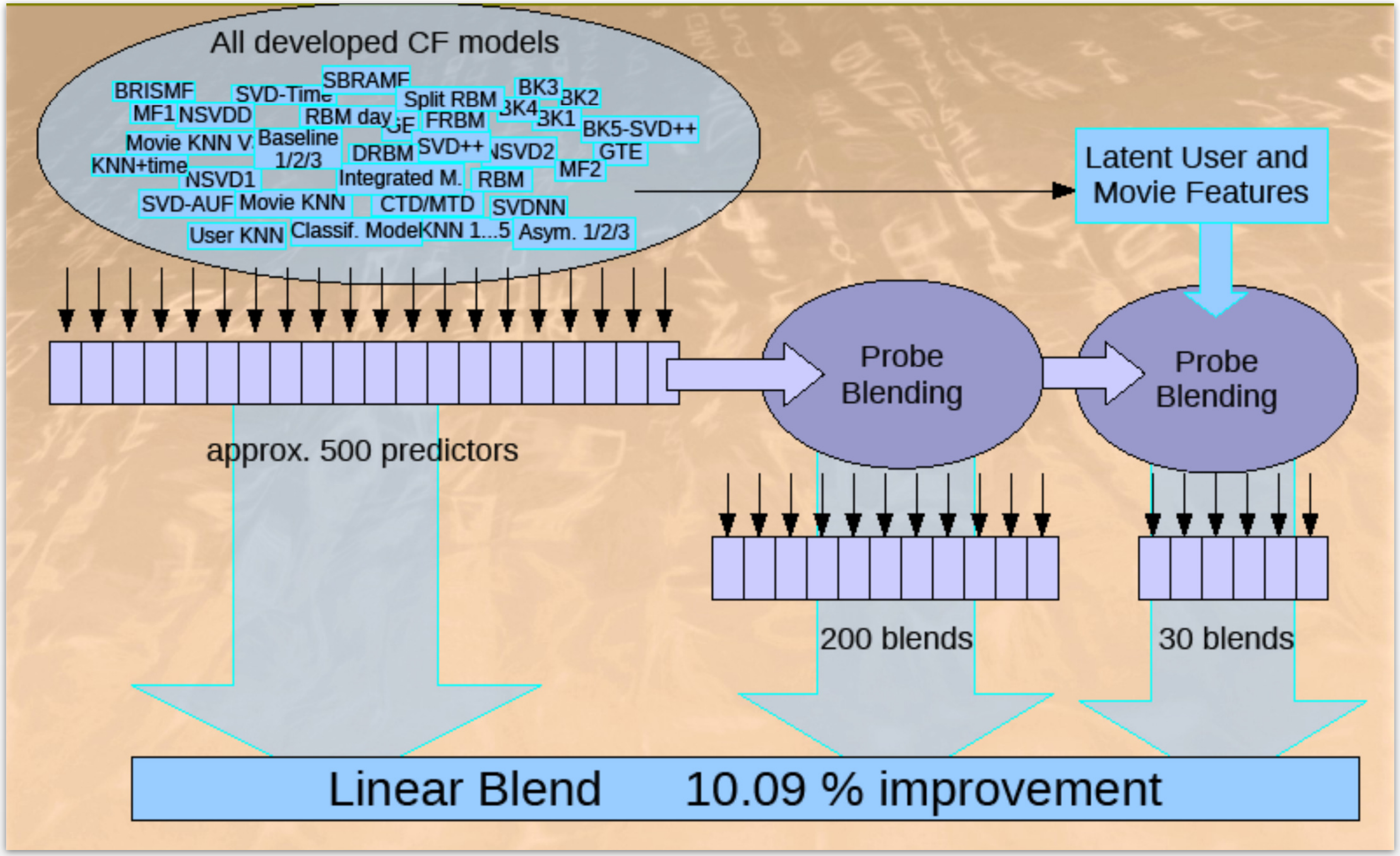
$$\hat{r}_{ui} = \mu(t) + b_u(t) + b_i(t) + \boldsymbol{x}_i^\top \boldsymbol{w}_u + \boldsymbol{c}_u(t)^\top \boldsymbol{v}_i$$

Improvements



Still pretty far from 0.8563 grand prize

Winning Solution from BellKor


All developed CF models
BRISMF
SVD-Time
SBRAMF
Split RBM
BK3
BK2
MF1
NSVDD
RBM day
GE
FRBM
BK4
BK1
BK5-SVD++
Movie KNN V.
Baseline 1/2/3
DRBM
SVD++
NSVD2
GTE
KNN+time
NSVD1
Integrated M.
RBM
MF2
SVD-AUF
Movie KNN
CTD/MTD
SVDNN
User KNN
Classif. Model
KNN 1...5
Asym. 1/2/3
Latent User and Movie Features
approx. 500 predictors
Probe Blending
Probe Blending
200 blends
30 blends
Linear Blend 10.09 % improvement