Recommender Systems: Latent Factor Models

CS246: Mining Massive Datasets
Jure Leskovec, Stanford University
http://cs246.stanford.edu

The 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**)
  - Netflix Cinematch RMSE: 0.9514
- **Competition**
  - 2700+ teams
  - $1 million prize for 10% improvement on Cinematch

The Netflix Utility Matrix

17,700 movies

480,000 users

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

Utility Matrix: Evaluation

17,700 movies

480,000 users

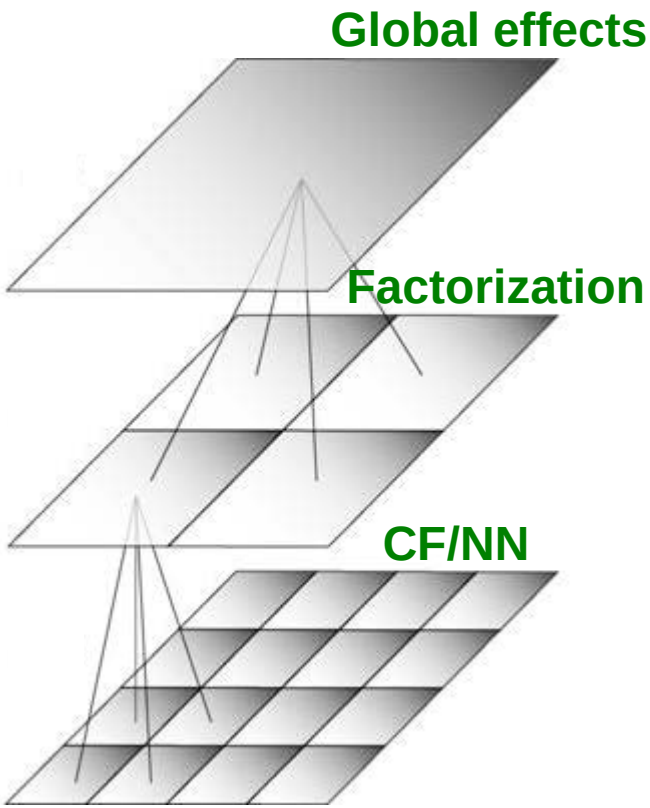
1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			?		?
				?	
	2	1			?
	3			?	
1					

Test Data Set

$$\textbf{SSE} = \sum_{(i,u)\in R}(r_{ui} - \hat{r}_{ui})^2$$

BellKor Recommender System

- **Basically the winner of the Netflix Challenge**
- Multi-scale modeling of the data: Combine top level, regional modeling of the data, with a refined, local view:
  - **Global:**
    - Overall deviations of users/movies
  - **Factorization:**
    - Addressing regional effects
  - **CF** (k-NN):
    - Extract local patterns



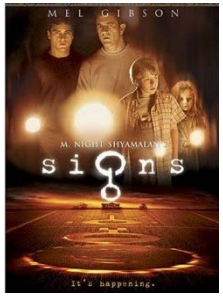
Modeling Local & Global Effects

- **Global:**
  - Mean movie rating: 3.7 stars
  - The Sixth Sense is 0.5 stars above avg.
  - Joe rates 0.2 stars below avg.

    $\Rightarrow$ Baseline estimation:

    Joe will rate *The Sixth Sense* 4 stars
- **Local neighborhood (CF/NN):**
  - *Joe* didn't like related movie *Signs*
  - $\Rightarrow$ Final estimate:

    Joe will rate *The Sixth Sense* 3.8 stars



Recap: Collaborative Filtering (CF)

- **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; u):*** items most similar to *i* that were rated by *u*

$$\hat{r}_{ui} = \frac{\sum_{j \in N(i;u)} s_{ij} r_{uj}}{\sum_{j \in N(i;u)} s_{ij}}$$

s_{ij}… similarity of items *i* and *j*
r_{uj}…rating of user *u* on item *j*
N(i;u)… set of similar items

Modeling Local & Global Effects

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

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in N(i;u)} s_{ij} \left(r_{uj} - b_{uj} \right)}{\sum_{j \in N(i;u)} s_{ij}}$$

baseline estimate for r_{ui}

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

μ = overall mean rating
b_u = rating deviation of user u
= *avg. rating of* user $u - \mu$
b_i = *avg. rating of* movie $i - \mu$

Problems:

1) Similarity measures are arbitrary
2) Pairwise similarities neglect interdependencies among neighbors
3) Taking a weighted average is restricting

Idea: Interpolation Weights

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

$$\hat{r}_{ui} = b_{ui} + \sum_{j \in N(i;u)} w_{ij} \left(r_{uj} - b_{uj} \right)$$

Allow:

$$\sum_{j \in N(i;u)} w_{ij} \neq 1$$

- **How to set w_{ij}?**
  - Remember, error metric is **SSE**: $\sum_{(i,u) \in R} (r_{ui} - \hat{r}_{ui})^2$
  - Find w_{ij} that minimize **SSE** on **training data**!

$$\min_{w} \sum_{v} \left(r_{vi} - \left[b_{vi} + \sum_{j \in N(i;v)} w_{ij} \left(r_{vj} - b_{vj} \right) \right] \right)^2$$

    - Models relationships between item *i* and its neighbors *j*
  - w_{ij} can be learnt through **gradient decent** based on *u* and all other users *v* that rated *i*

Interpolation Weights

- Find w_{ij} that minimize **SSE** on **training data**!

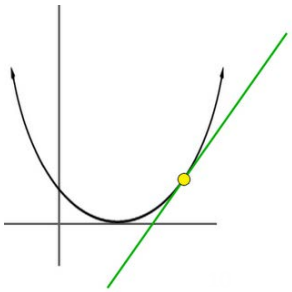
$$\min_{w} \sum_{v} \left(r_{vi} - \left[b_{vi} + \sum_{j \in N(i;v)} w_{ij} \left(r_{vj} - b_{vj} \right) \right] \right)^2$$

- **Gradient decent**
  - **Iterate until convergence:** $w \leftarrow w - \eta \cdot \nabla w$

    η … learning rate
  - Where:

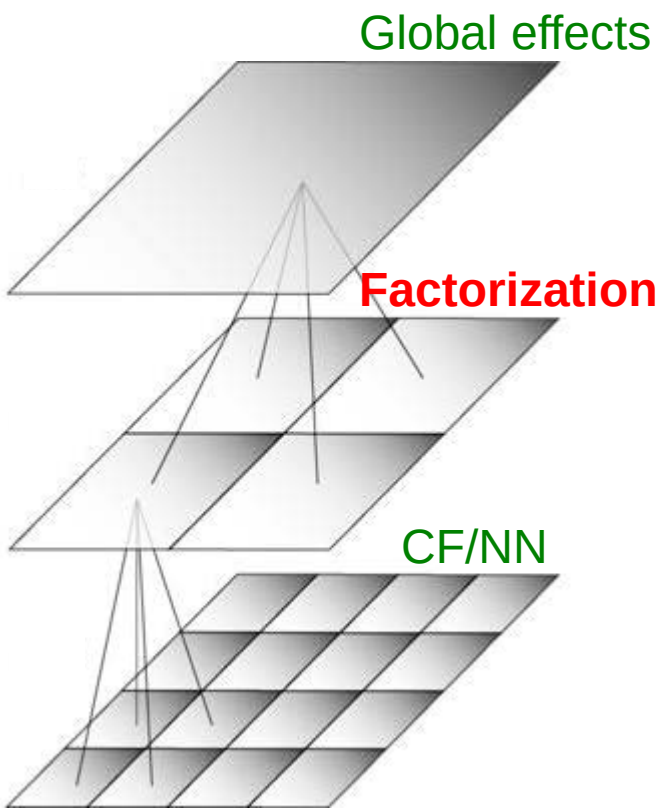
$$\nabla w_{ij} = 2 \sum_{v} \left(r_{vi} - \left[b_{vi} + \sum_{k \in N(i;v)} w_{ik} \left(r_{vk} - b_{vk} \right) \right] \right) \left(r_{vj} - b_{vj} \right)$$

for $j \in N(i;v)$ (else $\nabla w_{ij} = 0$)

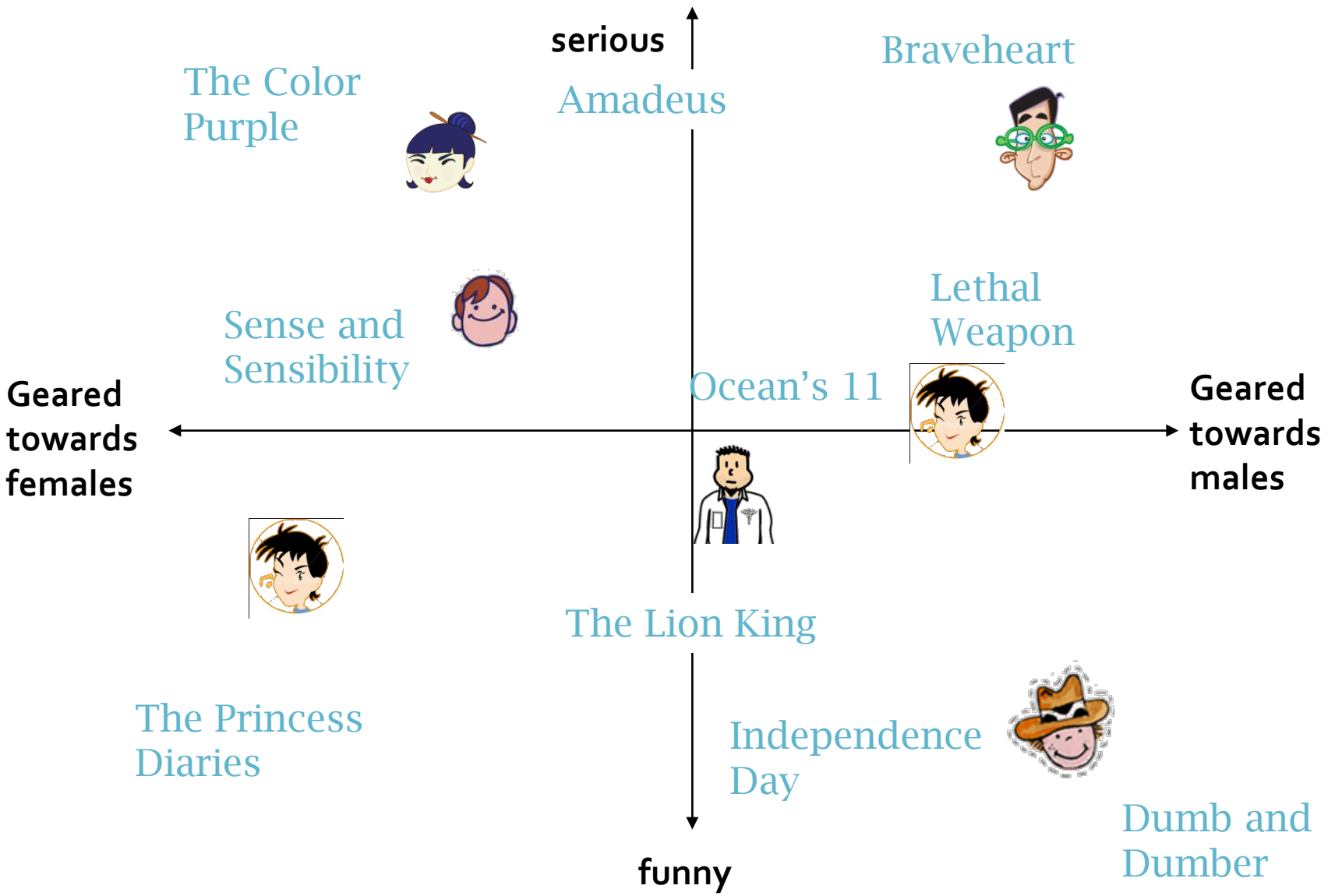


Interpolation Weights

- **So far:** $\hat{r}_{ui} = b_{ui} + \sum_{j \in N(i;u)} w_{ij} \left(r_{uj} - b_{uj} \right)$
  - 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 Models (e.g., SVD)



Latent Factor Models

- "SVD" on Netflix data: $\mathbf{R} \approx \boldsymbol{Q} \cdot \boldsymbol{P}^T$

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

$\boldsymbol{Q}$

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

$\boldsymbol{P}^T$

- For now let's assume we can approximate the rating matrix $\boldsymbol{R}$ as a product of "thin" $\boldsymbol{Q} \cdot \boldsymbol{P}^T$
  - There are important differences between "SVD" and the real SVD. We will get to them later

Ratings as Products of Factors

- How to estimate the missing rating?

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	

≈

$$\hat{r}_{iu} = q_i \cdot p_u^T$$

items

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

Q

•

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

P^T

Ratings as Products of Factors

- How to estimate the missing rating?

$$\hat{r}_{iu} = q_i \cdot p_u^T$$

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

Q

•

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

P^T

Ratings as Products of Factors

- How to estimate the missing rating?

$$\hat{r}_{iu} = q_i \cdot p_u^T$$

users

items

1		3			5			5		4	
		5	4	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

Q

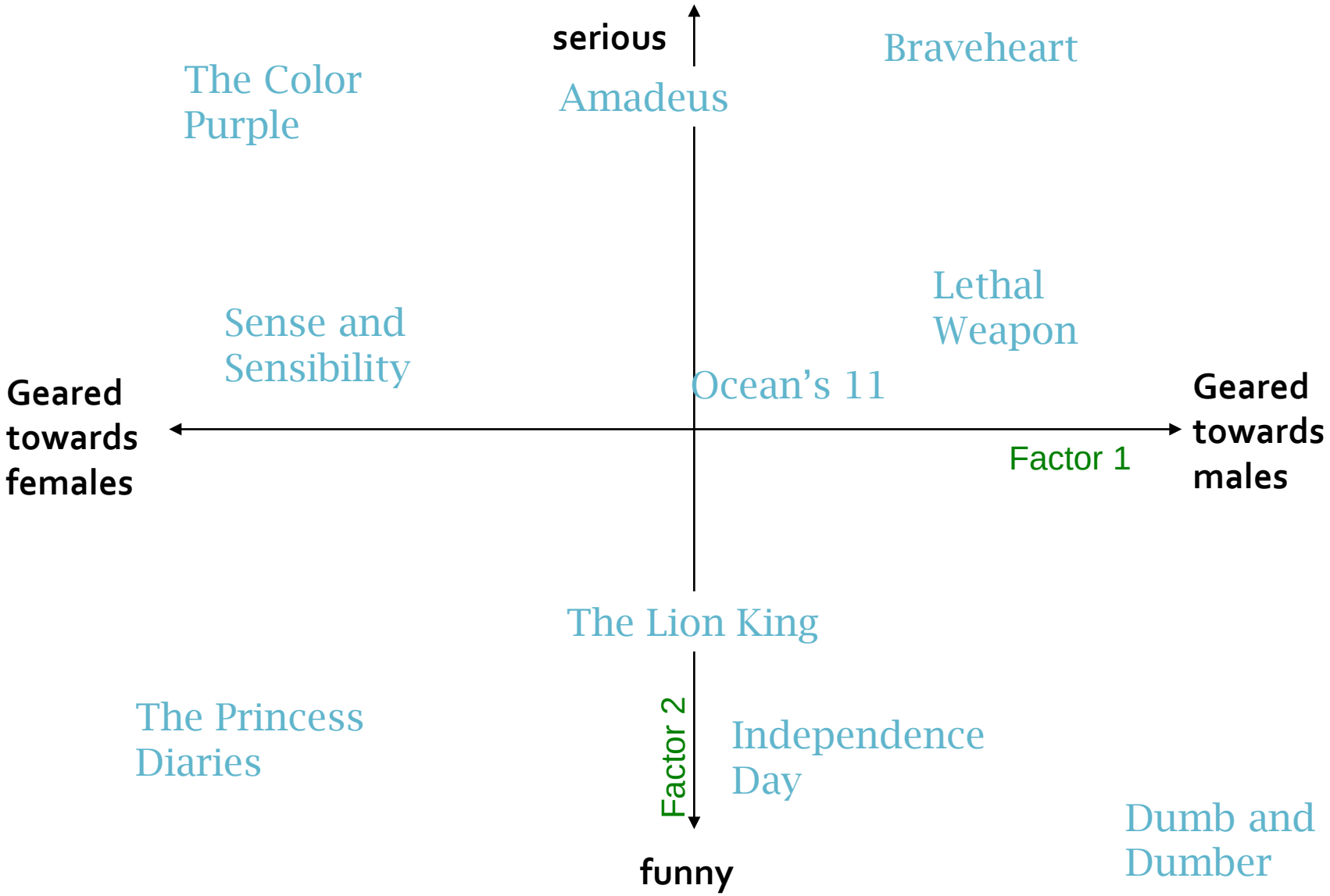
•

users

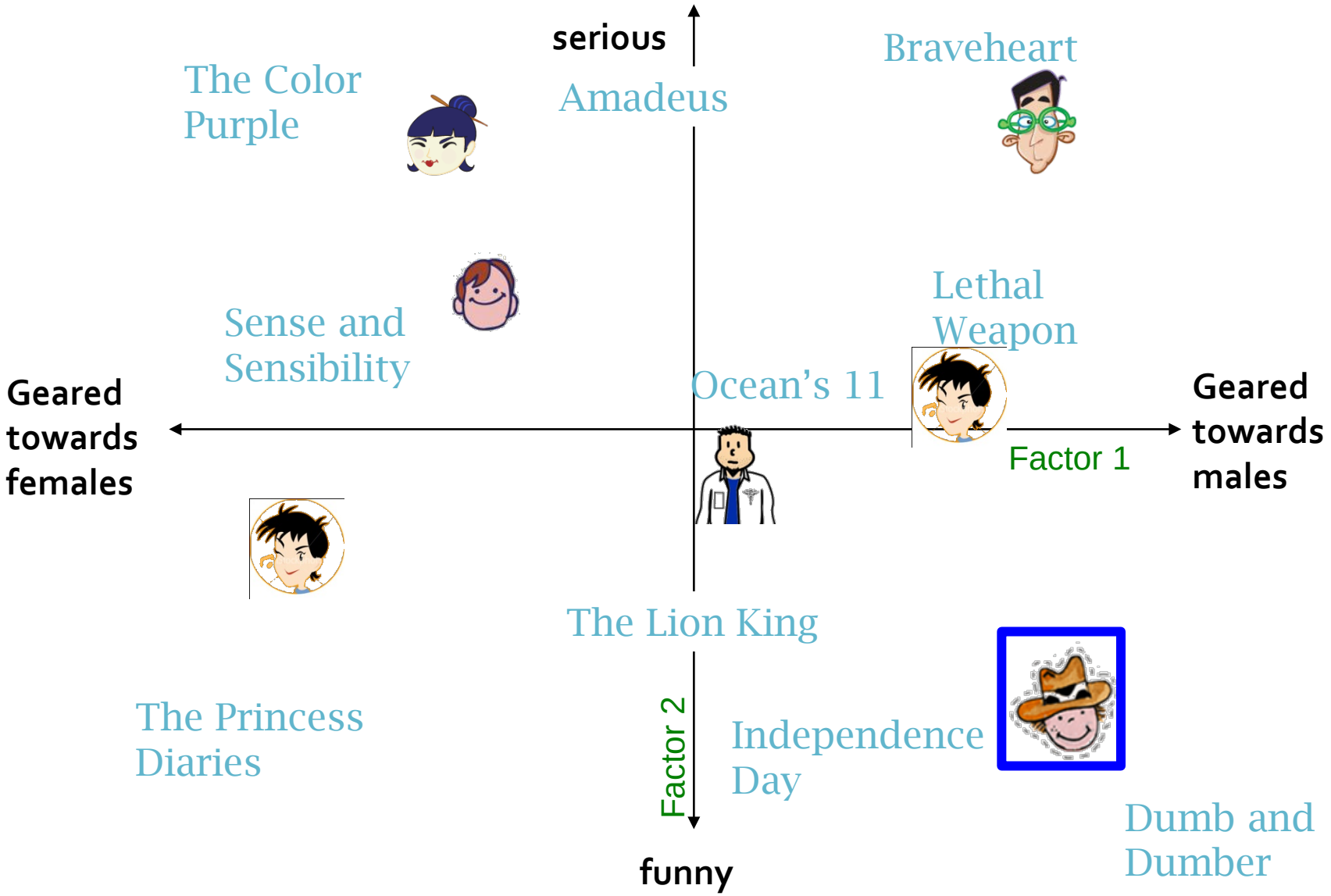
1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

P^T

Latent Factor Models

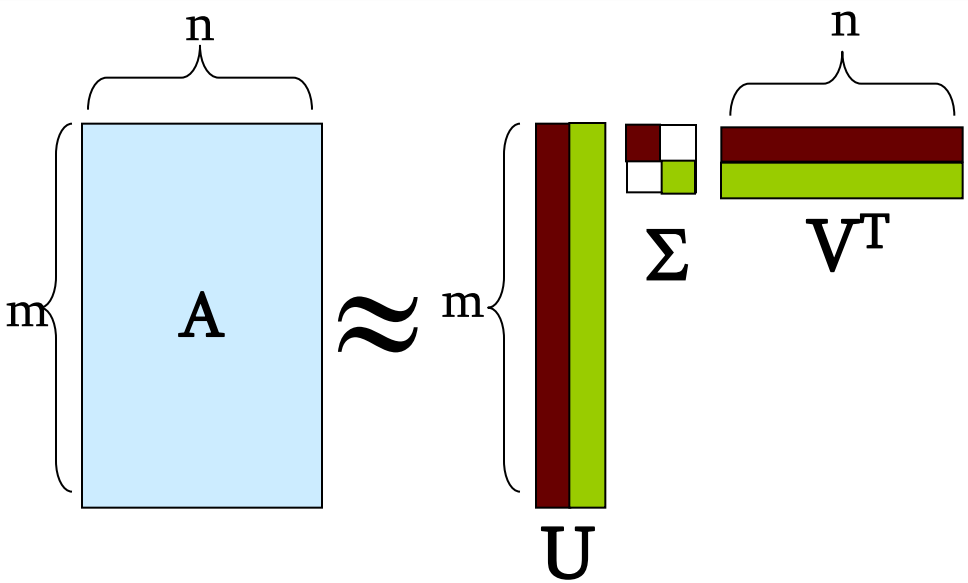


Latent Factor Models



Recap: SVD

- **Remember SVD:**
  - **A**: Input data matrix
  - **U**: Left singular vecs
  - **V**: Right singular vecs
  - **Σ**: Singular values
  - **SVD gives minimum reconstruction error (MSE!)**



$$\min_{U,V,\Sigma} \sum_{ij} \left(A_{ij} - [U\Sigma V^{\mathrm{T}}]_{ij}\right)^2$$

The sum goes over all entries.
Our *A*/R has missing entries!

- So in our case, "SVD" on Netflix data: $\boldsymbol{R \approx Q \cdot P^T}$

$\boldsymbol{A = R,\ \ Q = U,\ \ P^{\mathrm{T}} = \Sigma\, V^{\mathrm{T}}}$

$$\hat{r}_{iu} = q_i \cdot p_u^T$$

  - **But, we are not done yet! *R* has missing entries!**

[Bellkor Team]

Latent Factor Models

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

Q

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

$\boldsymbol{P}^T$

- SVD isn't defined when entries are missing
- **Use specialized methods to find *P, Q***
  - $\min_{P,Q} \sum_{(i,u)\in R} (r_{iu} - q_i \cdot p_u^T)^2$ $\qquad \hat{r}_{iu} = q_i \cdot p_u^T$
  - Don't require cols of ***P, Q*** to be orthogonal/unit length
  - ***P, Q*** map users/movies to a latent space
  - The most popular model among Netflix contestants

Dealing with Missing Entries

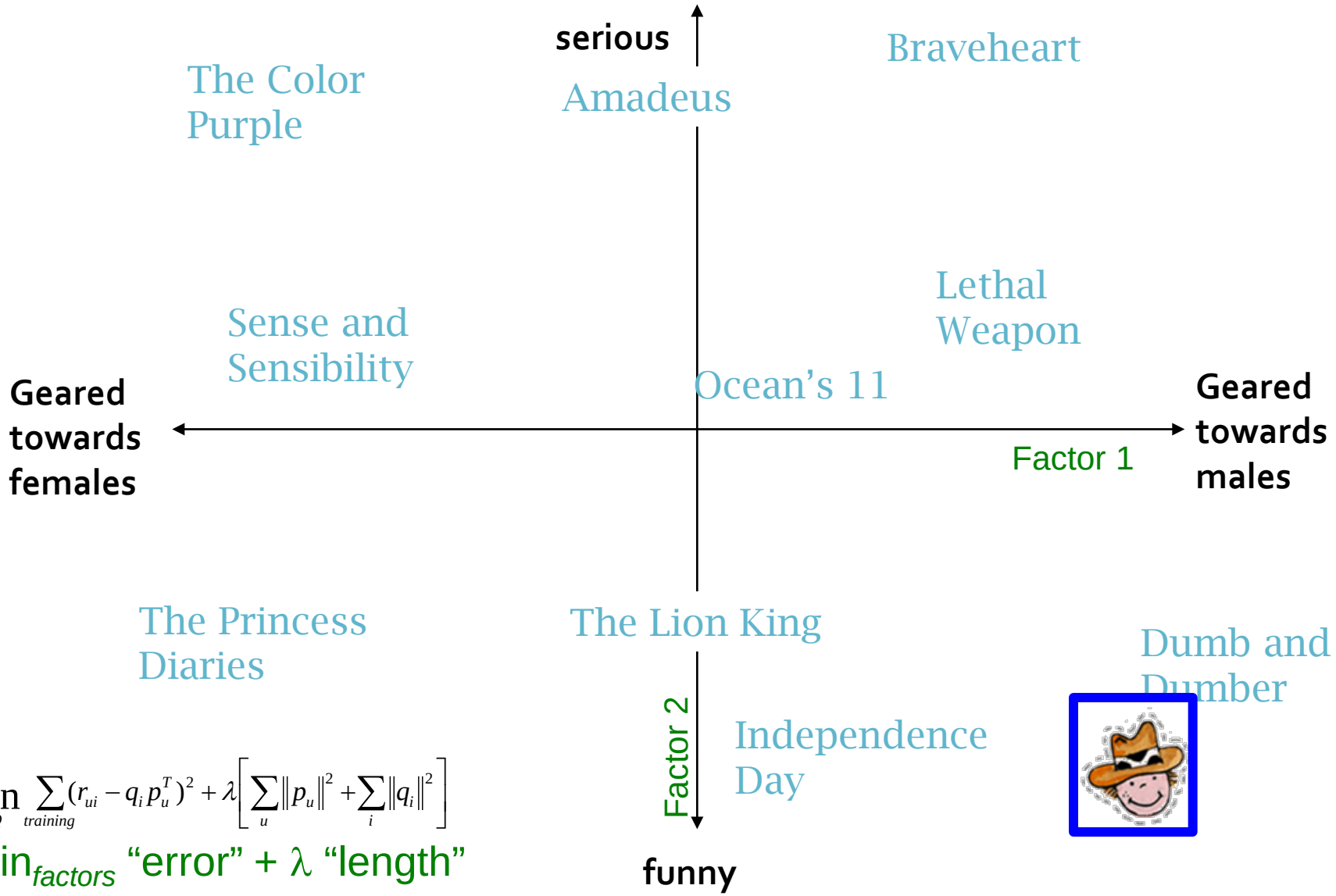
1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			?		?
				?	
	2	1			?
	3			?	
1					

- **Want to minimize SSE for test data**
- **Idea:** Minimize SSE on Training data
  - Want large f (# of factors) to capture all the signals
  - But, test **SSE** begins to rise for $f > 2$
- **Regularization is needed**
  - Allow rich model where there are sufficient data
  - Shrink aggressively where data are scarce

$$\min_{P,Q} \underbrace{\sum_{training} (r_{ui} - q_i p_u^T)^2}_{\text{“error”}} + \lambda \underbrace{\left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]}_{\text{“length”}}$$

λ… regularization parameter

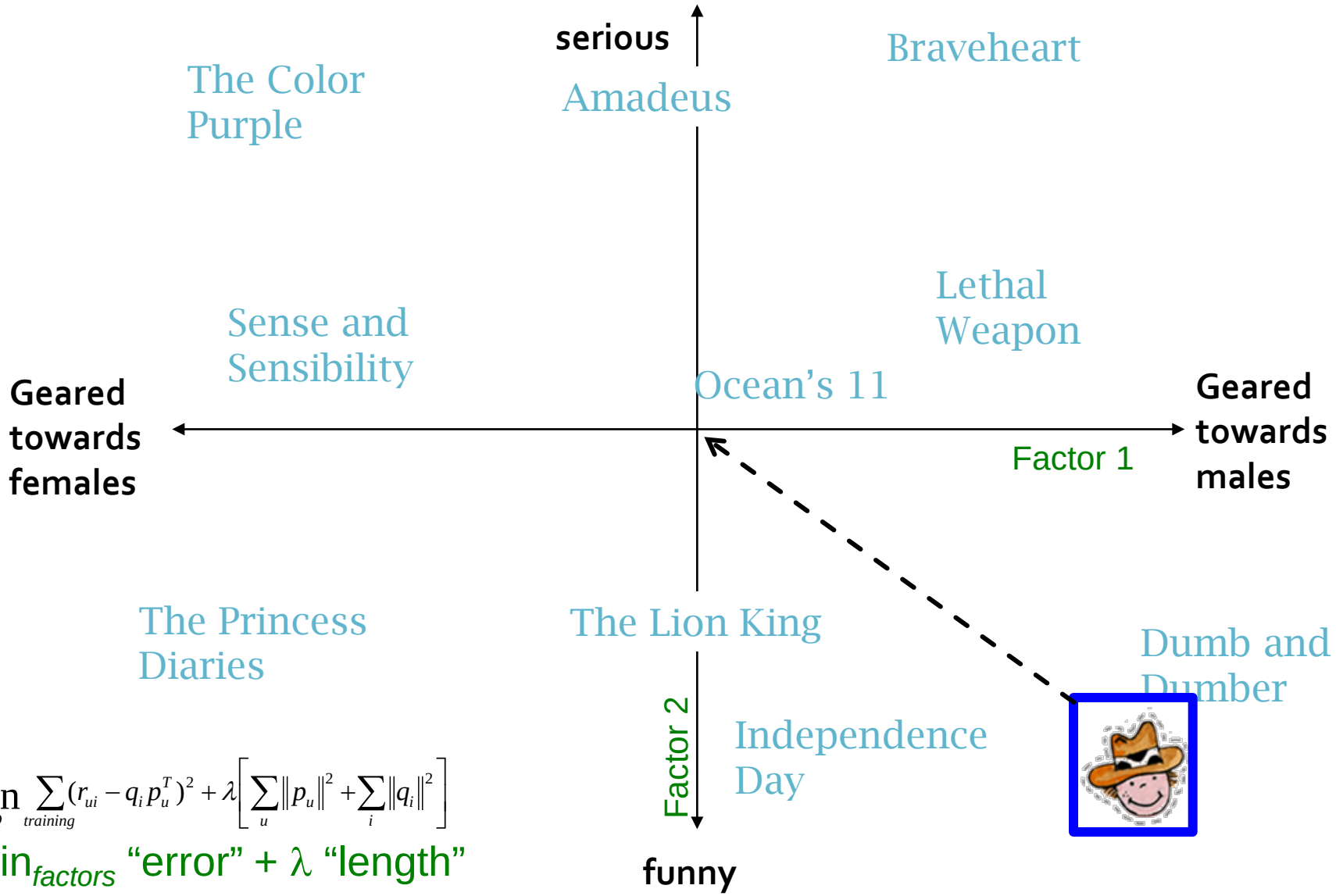
The Effect of Regularization



$$\min_{P,Q} \sum_{training} (r_{ui} - q_i p_u^T)^2 + \lambda \left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]$$

$\min_{factors}$ "error" + λ "length"

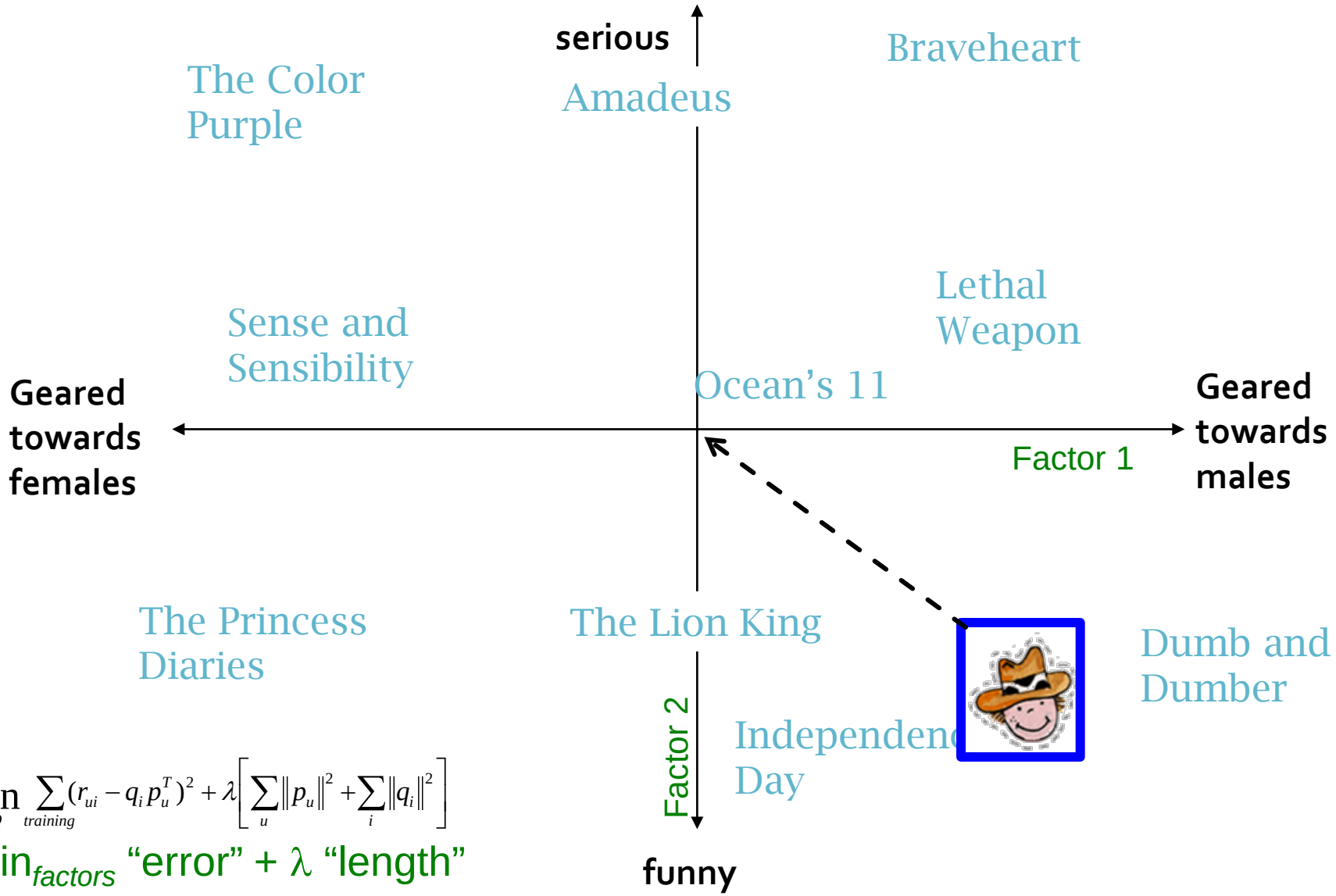
The Effect of Regularization



$$\min_{P,Q} \sum_{training} (r_{ui} - q_i p_u^T)^2 + \lambda \left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]$$

$\min_{factors}$ "error" + λ "length"

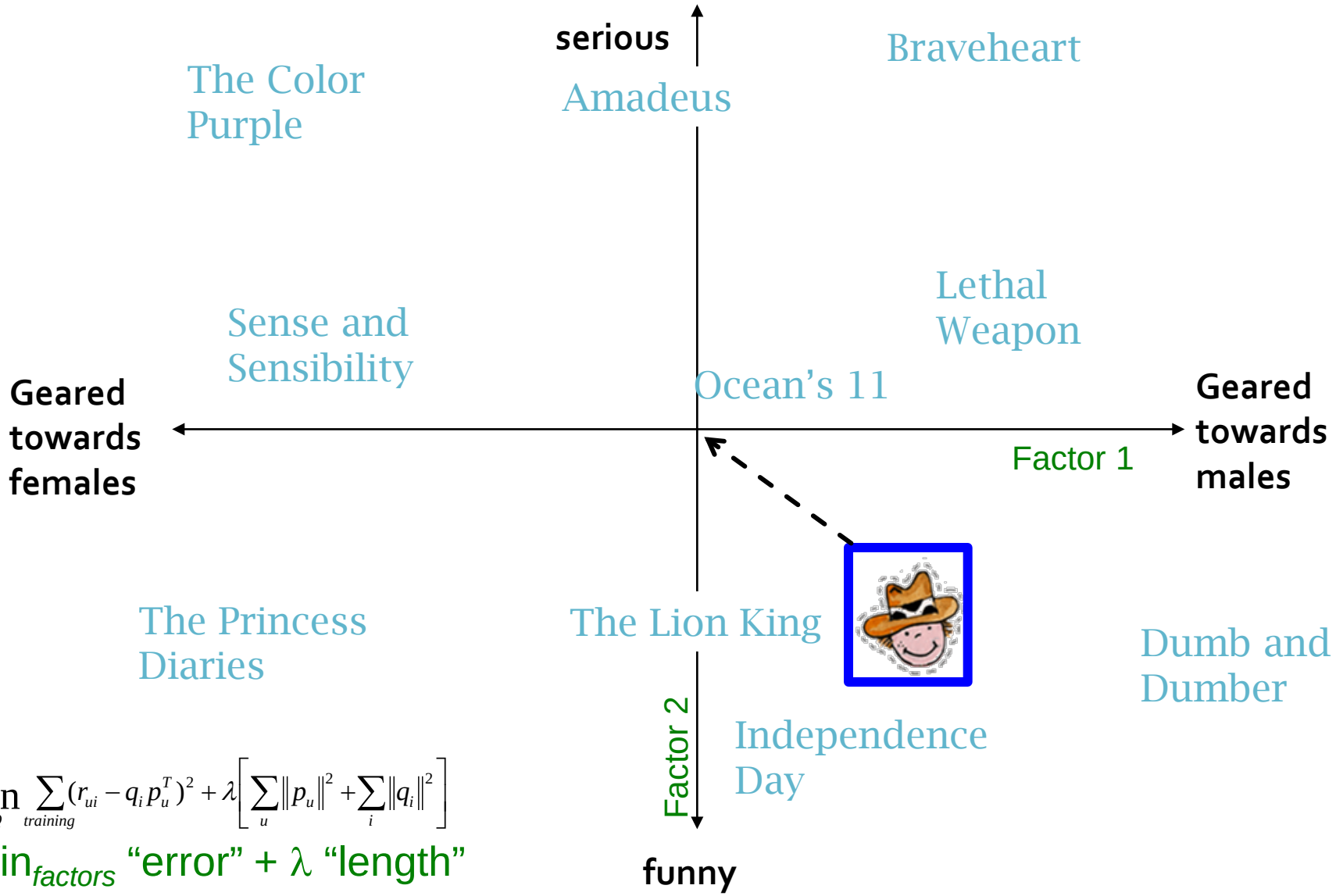
The Effect of Regularization



$$\min_{P,Q} \sum_{training} (r_{ui} - q_i p_u^T)^2 + \lambda \left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]$$

$\min_{factors}$ "error" + λ "length"

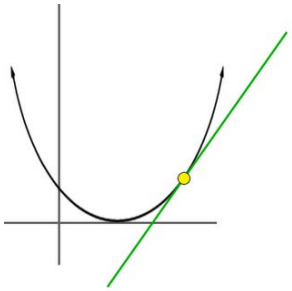
The Effect of Regularization



$$\min_{P,Q} \sum_{training} (r_{ui} - q_i p_u^T)^2 + \lambda \left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]$$

$\min_{factors}$ "error" + λ "length"

Stochastic Gradient Descent

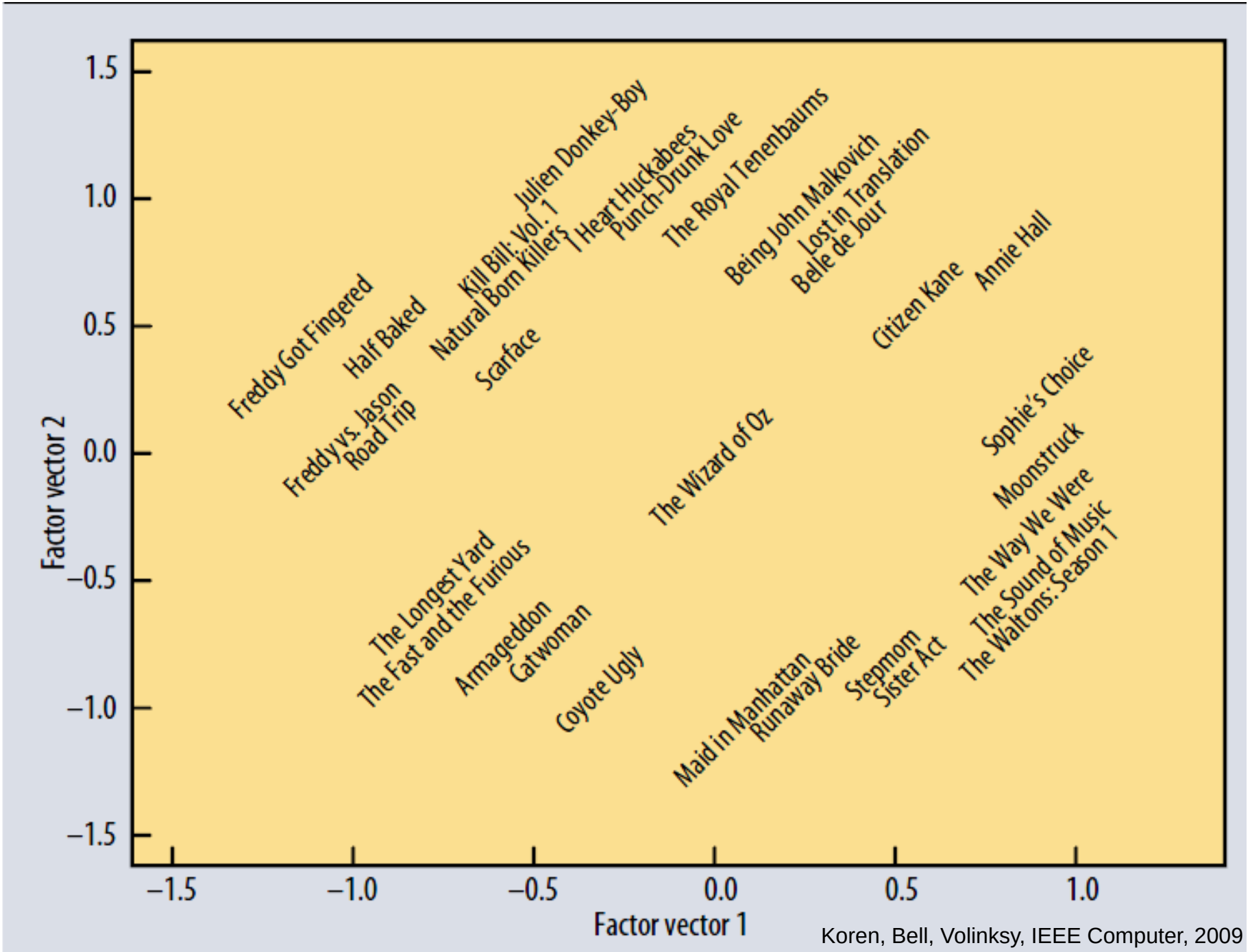


- **Want to find matrices *P* and *Q*:**

$$\min_{P,Q} \sum_{training} (r_{ui} - q_i \, p_u^T)^2 + \lambda \left[\sum_u \|p_u\|^2 + \sum_i \|q_i\|^2 \right]$$

- **Online "stochastic" gradient decent:**
  - Initialize *P* and *Q* (random, using SVD)
  - Then iterate over the ratings and update factors:
  - **For each r_{ui}:**
    - $\varepsilon_{ui} = r_{ui} - q_i \cdot p_u^T$ (derivative of the "error")
    - $q_i \leftarrow q_i + \eta \, (\varepsilon_{ui} \, p_u - \lambda \, q_i)$ (update equation)
    - $p_u \leftarrow p_u + \eta \, (\varepsilon_{ui} \, q_i - \lambda \, p_u)$ (update equation)

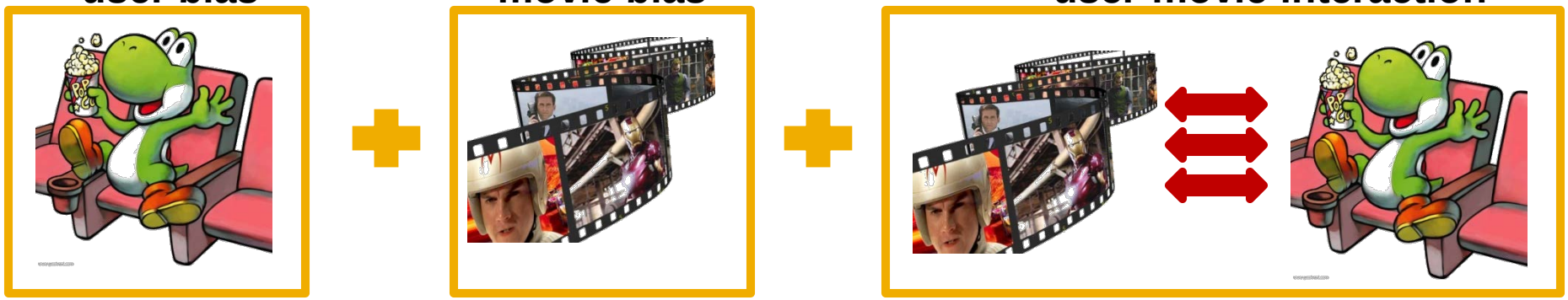
η … learning rate



Koren, Bell, Volinksy, IEEE Computer, 2009

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_u = bias of user u
- b_i = bias of movie i

Baseline Predictor

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

- Rating scale of user *u*
- 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

$$r_{ui} = \mu + b_u + b_i + q_i \cdot p_u^T$$

Overall mean rating | Bias for user *u* | 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: $b_u = -1$
  - 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:**

$$\min_{Q,P} \sum_{(u,i)\in R} \left(r_{ui} - (\mu + b_u + b_i + q_i\, p_u^T)\right)^2$$

goodness of fit

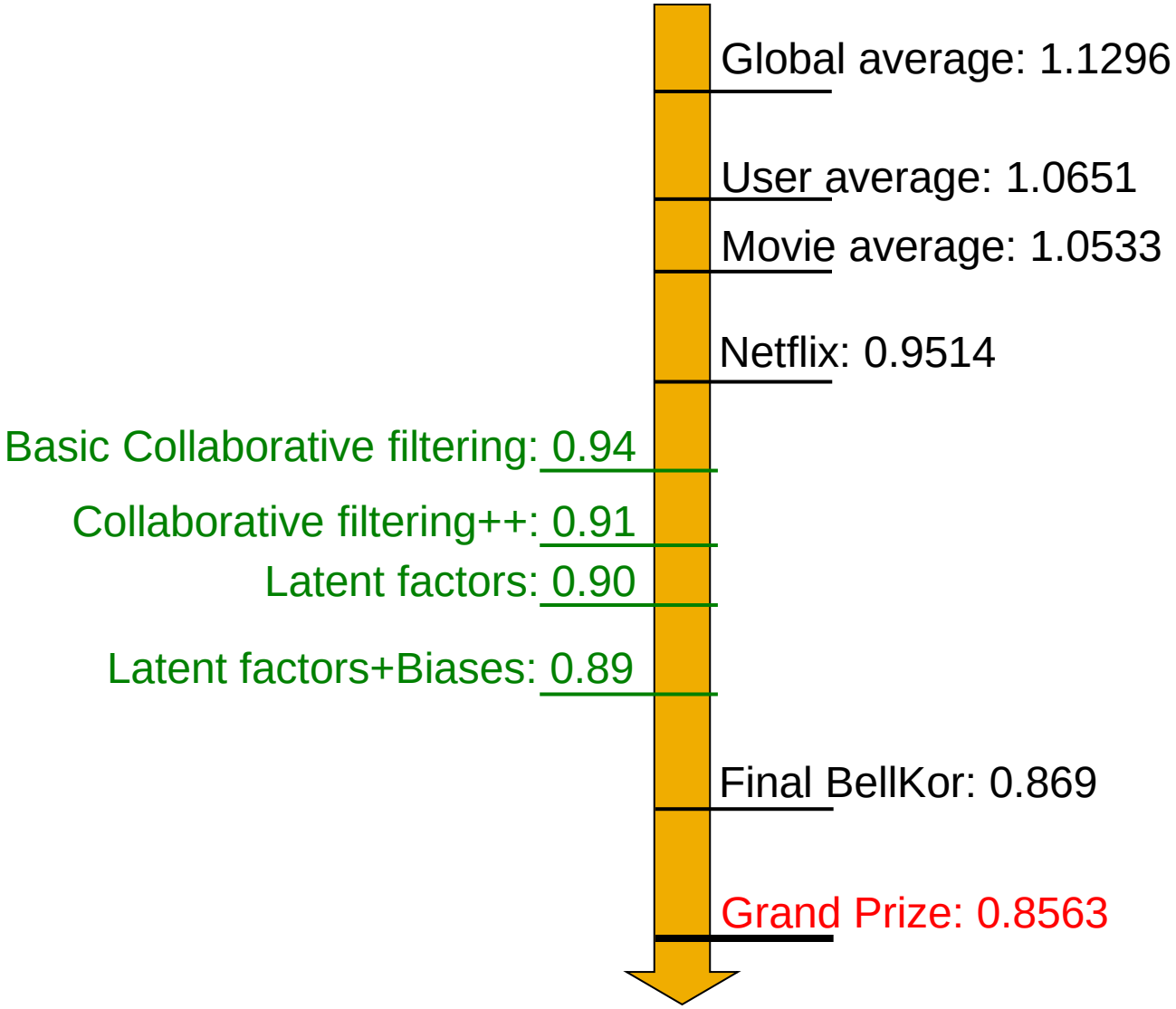
$$+ \lambda \left(\|q_i\|^2 + \|p_u\|^2 + \|b_u\|^2 + \|b_i\|^2 \right)$$

regularization

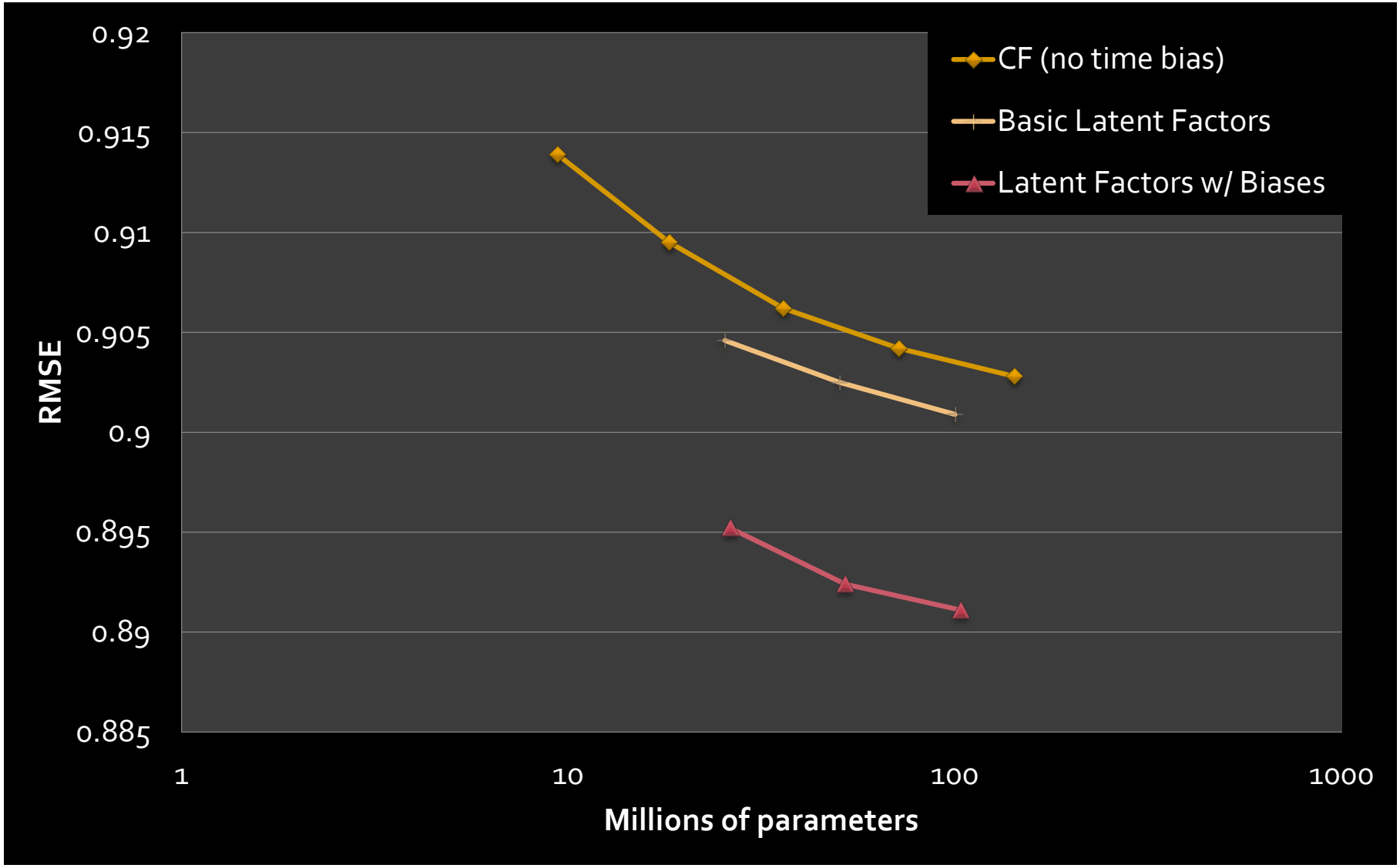
λ Is typically selected via grid-search on a validation set

- **Stochastic gradient decent to find parameters**
  - **Note:** Both biases (b_u, b_i) as well as interactions (q_i, p_u) are treated as parameters (we estimate them)

Performance of Various Methods



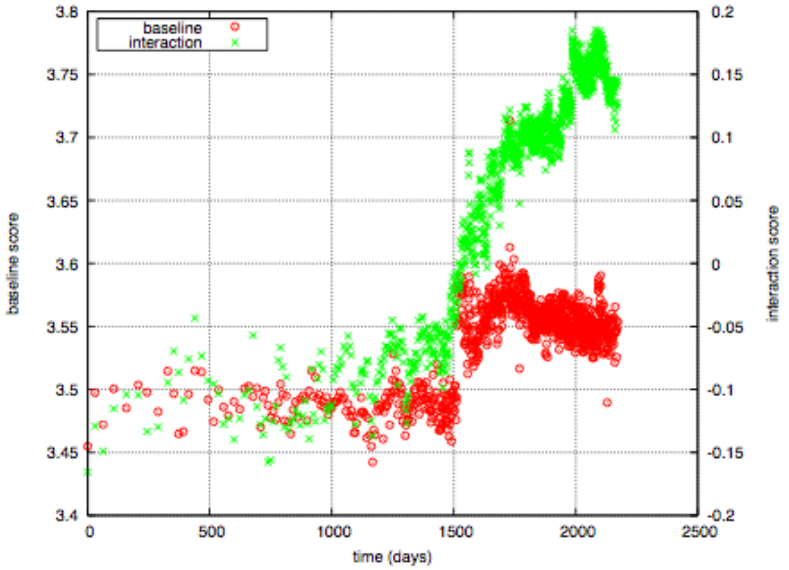
Performance of Various Methods

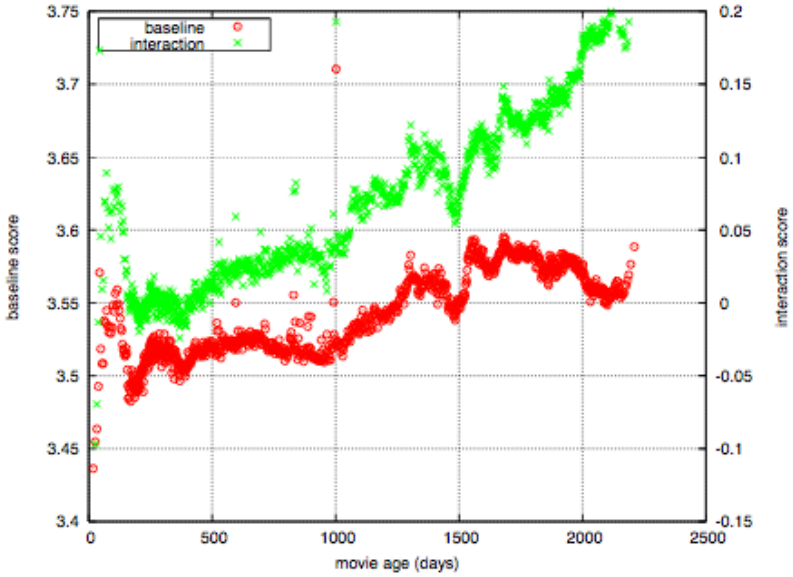


The Netflix Challenge: 2006-09

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





Y. Koren, Collaborative filtering with temporal dynamics, KDD '09

Temporal Biases & Factors

- **Original model:**

  $$r_{ui} = \mu + b_u + b_i + q_i \cdot p_u^T$$

- **Add time dependence to biases:**

  $$r_{ui} = \mu + b_u(t) + b_i(t) + q_i \cdot p_u^T$$

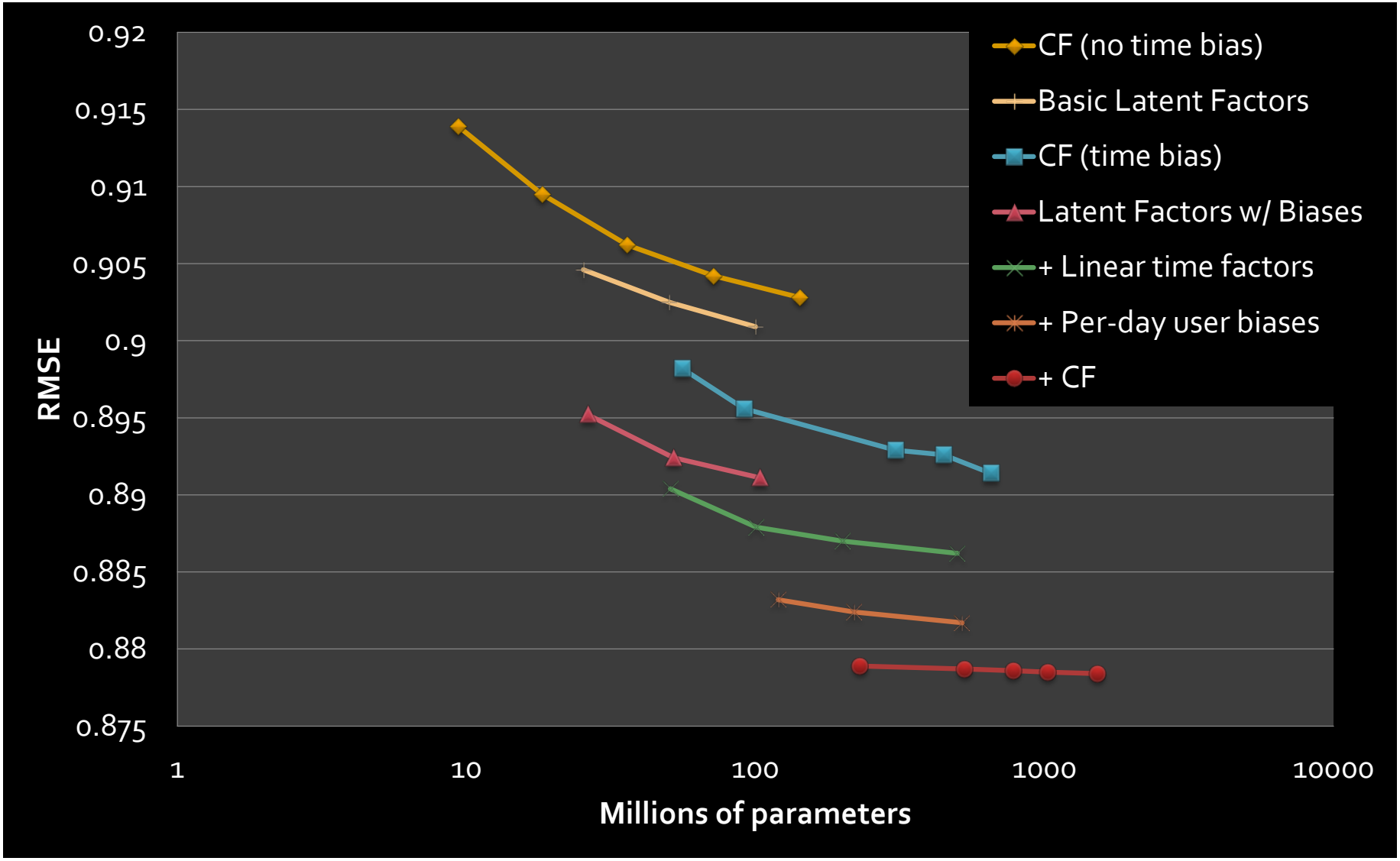
  - Make parameters b_u and b_i to depend on time
  - **(1)** Parameterize time-dependence by linear trends
    (2) Each bin corresponds to 10 consecutive weeks

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

- **Add temporal dependence to factors**
  - $p_u(t)$… user preference vector on day t

Y. Koren, Collaborative filtering with temporal dynamics, KDD '09

Adding Temporal Effects

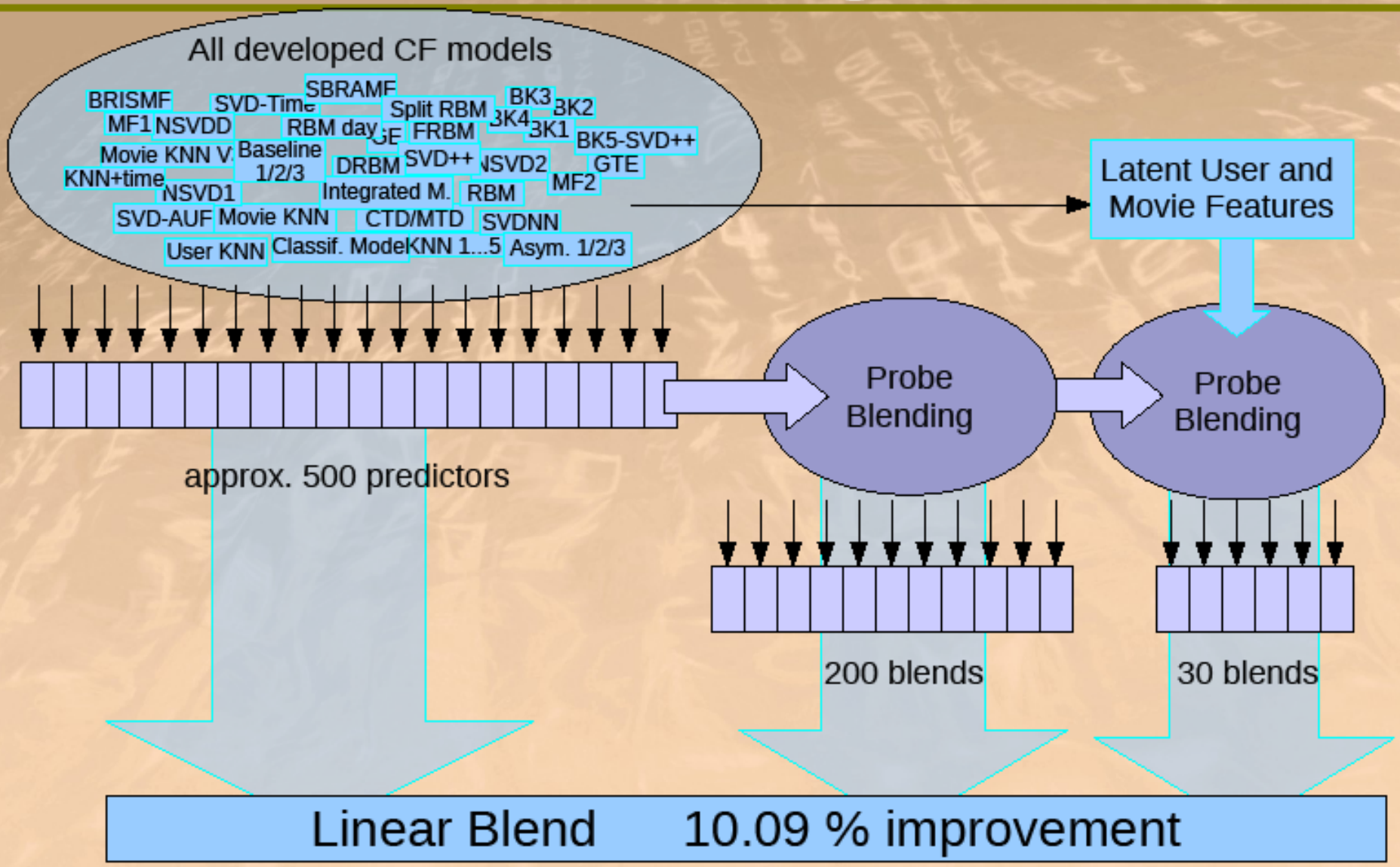


The Kitchen Sink Approach….

- **Many options for modeling**
  - Variants of the ideas we have seen so far
    - Different numbers of factors
    - Different ways to model time
    - Different ways to handle implicit information
  - Other models (not described here)
    - Nearest-neighbor models
    - Restricted Boltzmann machines
- Model averaging is useful….
  - Linear model combining

The big picture

Solution of BellKor's Pragmatic Chaos



Standing on June 26th 2009

NETFLIX

Netflix Prize

Home | Rules | Leaderboard | Register | Update | Submit | Download

Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	BellKor's Pragmatic Chaos	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	PragmaticTheory	0.8582	9.80	2009-06-25 22:15:51
3	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
4	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
5	Dace	0.8604	9.56	2009-04-22 05:57:03
6	BigChaos	0.8613	9.47	2009-06-23 23:06:52
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
7	BellKor	0.8620	9.40	2009-06-24 07:16:02
8	Gravity	0.8634	9.25	2009-04-22 18:31:32
9	Opera Solutions	0.8638	9.21	2009-06-26 23:18:13
10	BruceDengDaoCiYiYou	0.8638	9.21	2009-06-27 00:55:55
11	pengpengzhou	0.8638	9.21	2009-06-27 01:06:43
12	xlvector	0.8639	9.20	2009-06-26 13:49:04
13	xiangliang	0.8639	9.20	2009-06-26 07:47:34

June 26th submission triggers 30-day "last call"

The Last 30 Days

- **Ensemble team formed**
  - Group of other teams on leaderboard forms a new team
  - Relies on combining their models
  - Quickly also get a qualifying score over 10%
- **BellKor**
  - Continue to get small improvements in their scores
  - Realize that they are in direct competition with Ensemble
- **Strategy**
  - Both teams carefully monitoring the leaderboard
  - Only sure way to check for improvement is to submit a set of predictions
    - This alerts the other team of your latest score

24 Hours from the Deadline

- **Submissions limited to 1 a day**
  - Only 1 final submission could be made in the last 24h
- **24 hours before deadline…**
  - BellKor team member in Austria notices (by chance) that Ensemble posts a score that is slightly better than BellKor's
- **Frantic last 24 hours for both teams**
  - Much computer time on final optimization
  - run times carefully calibrated to end about an hour before deadline
- **Final submissions**
  - BellKor submits a little early (on purpose), 40 mins before deadline
  - Ensemble submits their final entry 20 mins later
  - **….and everyone waits….**

NETFLIX

Netflix Prize

COMPLETED

Home | Rules | Leaderboard | Update | Download

Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

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	[illegible]	[illegible]	[illegible]
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
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
16	Invisible Ideas	0.8653	9.15	2009-07-15 15:53:04
17	Just a guy in a garage	0.8662	9.06	2009-05-24 10:02:54
18	J Dennis Su	0.8666	9.02	2009-03-07 17:16:17
19	Craig Carmichael	0.8666	9.02	2009-07-25 16:00:54
20	acmehill	0.8668	9.00	2009-03-21 16:20:50
Progress Prize 2007 - RMSE = [illegible]				

Million $ Awarded Sept 21st 2009

Acknowledgments

- Some slides and plots borrowed from Yehuda Koren, Robert Bell and Padhraic Smyth

- **Further reading:**
  - Y. Koren, Collaborative filtering with temporal dynamics, KDD ’09
  - http://www2.research.att.com/~volinsky/netflix/bpc.html
  - http://www.the-ensemble.com/