Finding the Latent Factors

Mining of Massive Datasets
Leskovec, Rajaraman, and Ullman
Stanford University

Latent Factor Models

- **Our goal is to find P and Q such tat:**

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

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	

≈

factors

items

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

Q

users

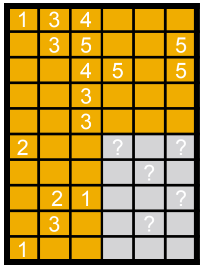
factors

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

$\boldsymbol{P^T}$

Dealing with Missing Entries

- **Want to minimize SSE (that is RMSE) for unseen test data**



- **Idea: Minimize SSE on training data:**

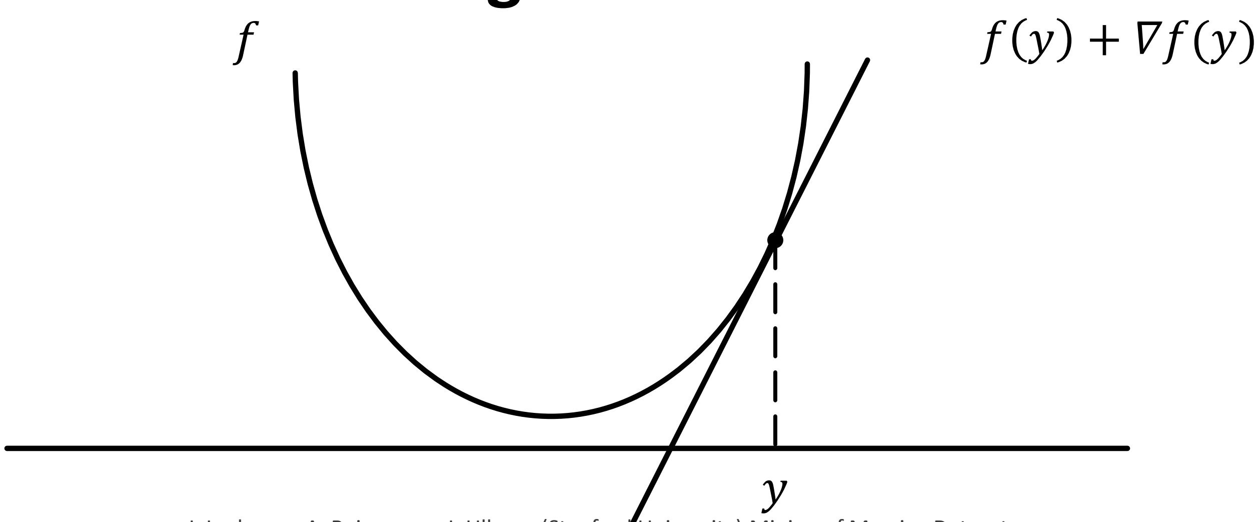
$$f(P,Q) = \sum_{(i,x)\in R} \left(r_{xi} - q_i \cdot p_x^T\right)^2$$

  - Want large k (# of factors) to capture all the signals

- **How to minimize our error function?**

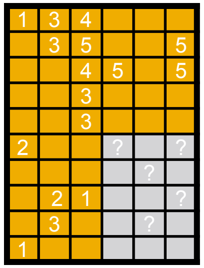
Detour: Minimizing a function

- **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**

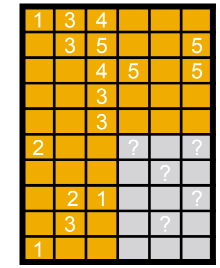


Back to Our Problem

- **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



Dealing with Missing Entries



- **To solve overfitting we introduce regularization:**
  - Allow rich model where there are sufficient data
  - Shrink aggressively where data are scarce

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

λ   user set regularization parameter

The Effect of Regularization

serious

Braveheart

The Color Purple

Amadeus

Lethal Weapon

Sense and Sensibility

Ocean's 11

Geared towards females

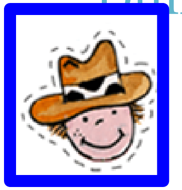
Geared towards males

Factor 1

The Princess Diaries

The Lion King

Dumb and Dumber



Factor 2

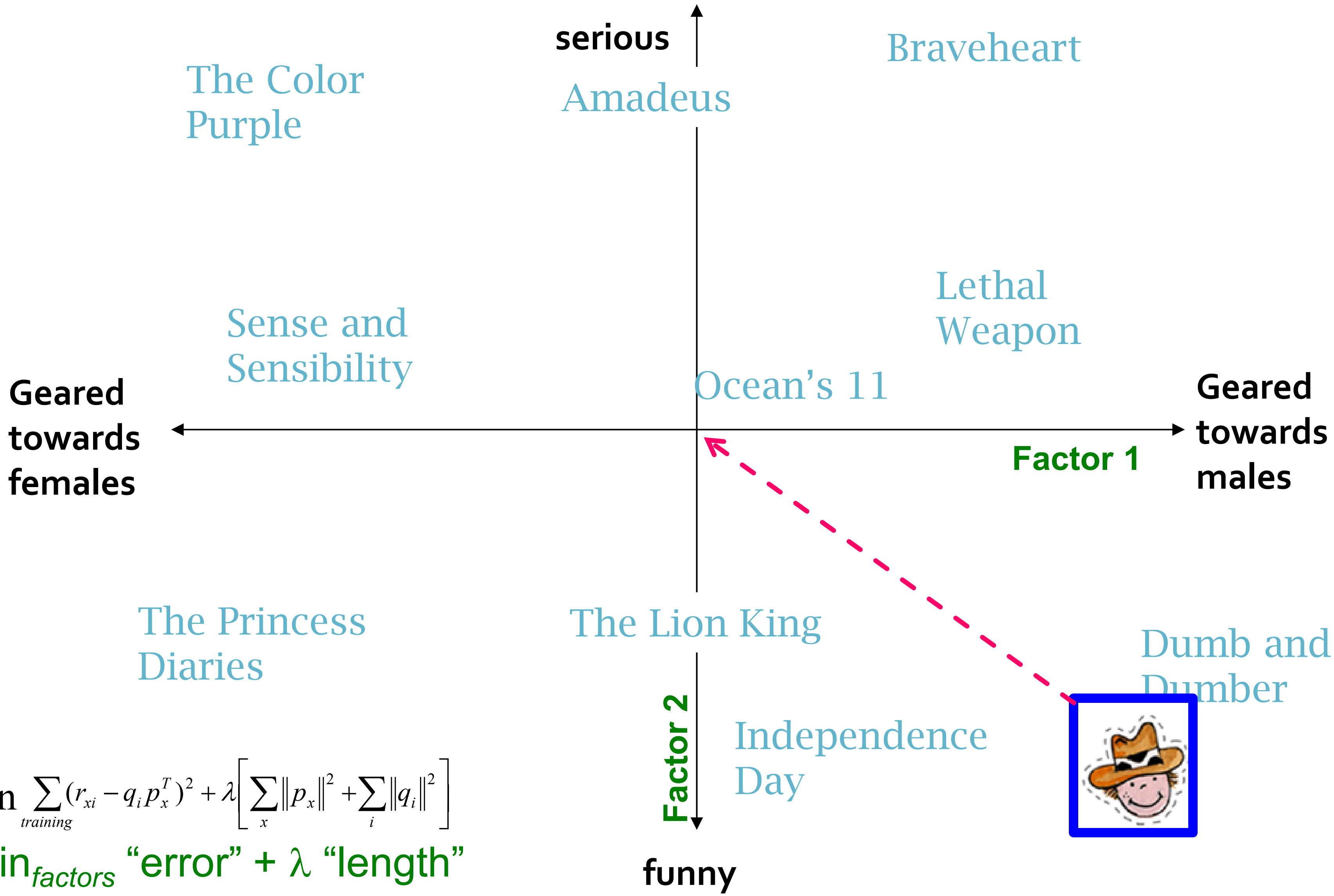
Independence Day

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

min$_{factors}$ "error" + λ "length"

funny

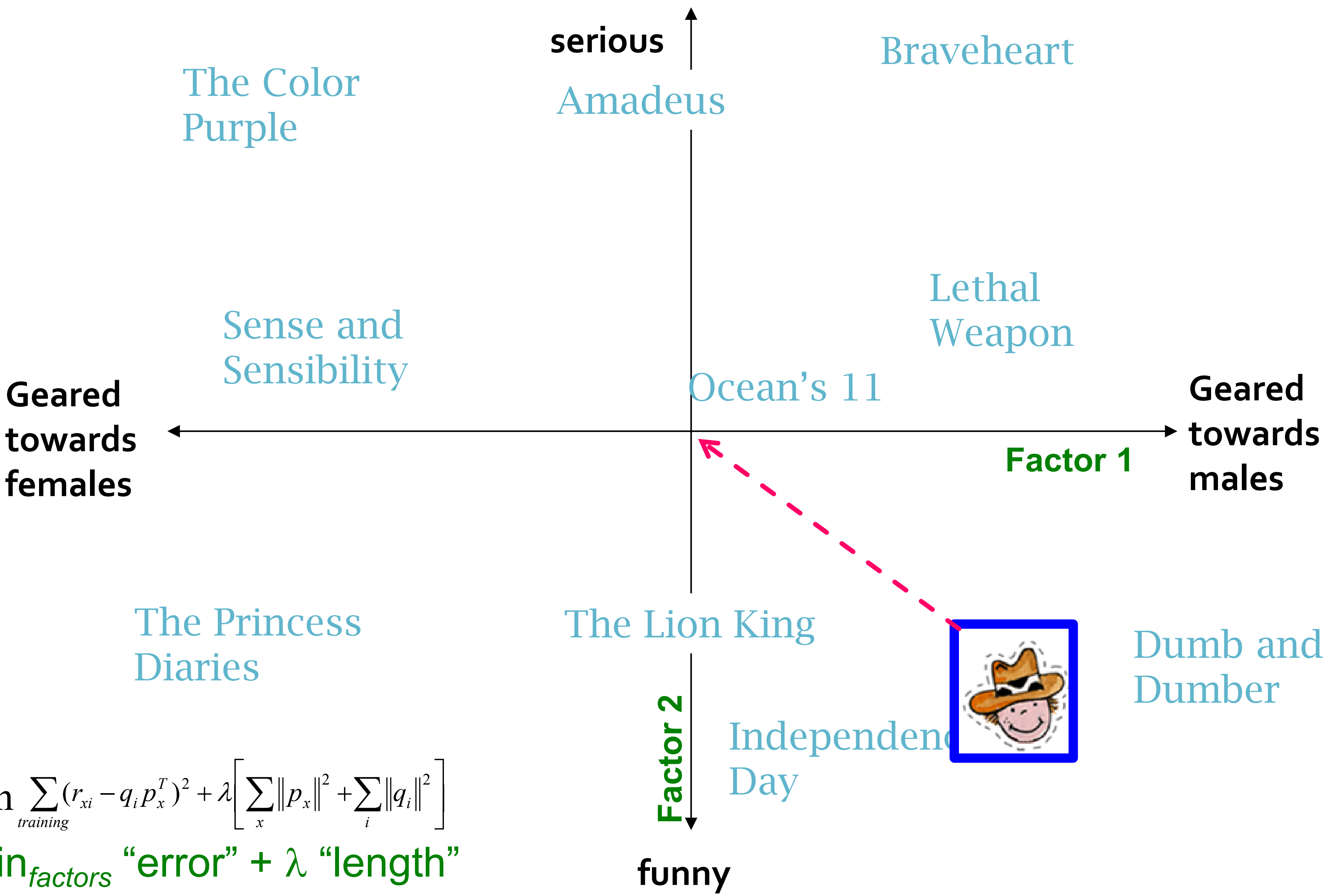
The Effect of Regularization



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

$\min_{factors}$ "error" + λ "length"

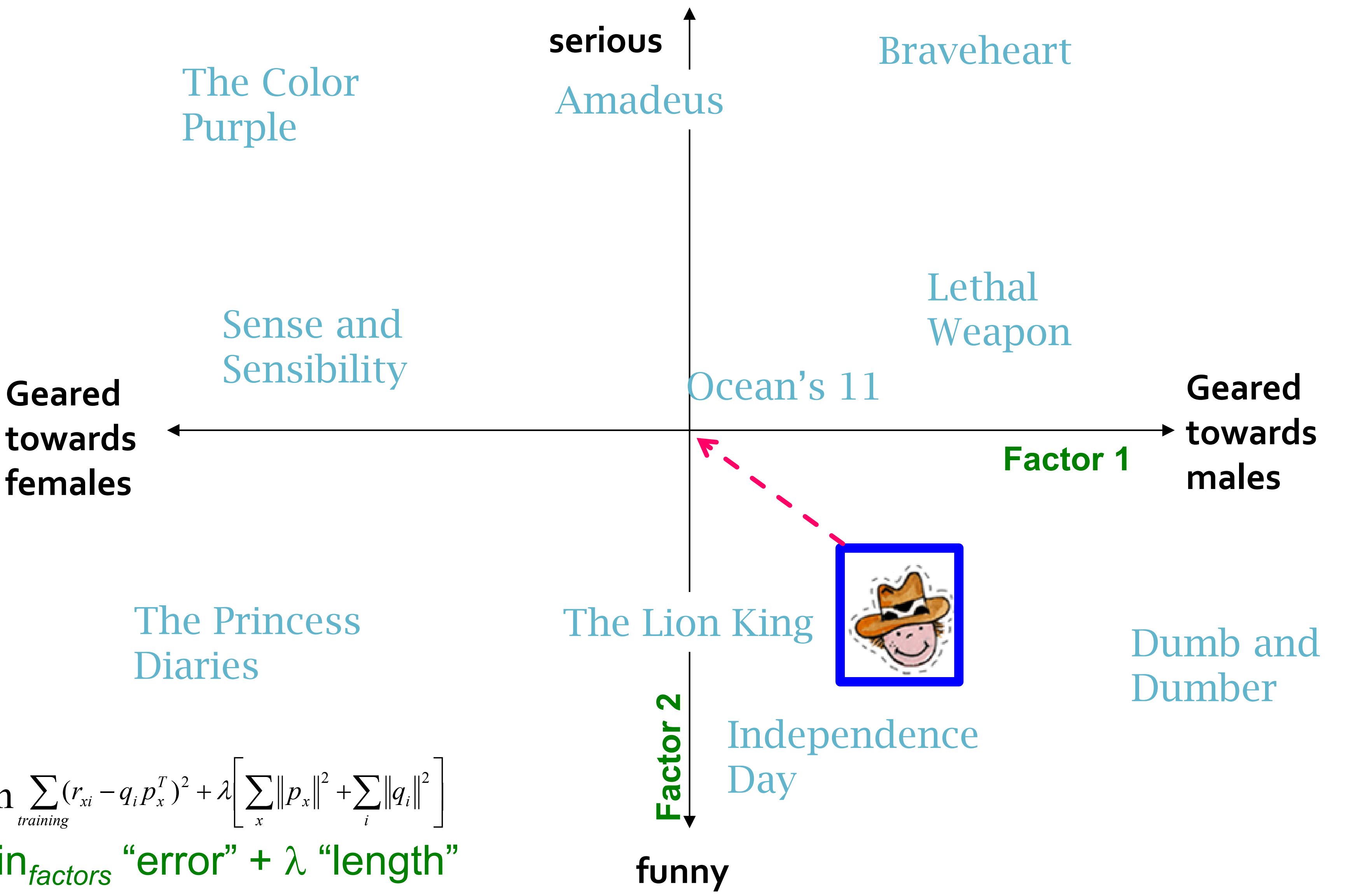
The Effect of Regularization



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

$\min_{factors}$ "error" + λ "length"

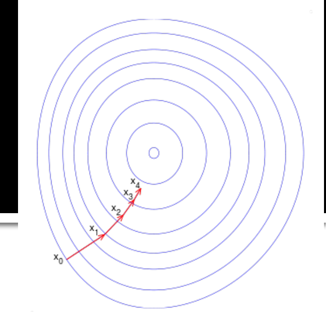
The Effect of Regularization



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

$\min_{factors}$ "error" + λ "length"

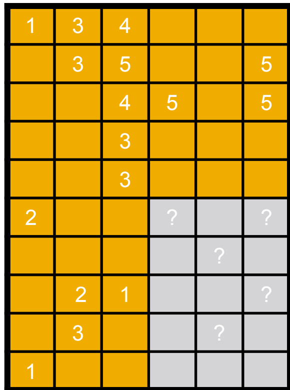
Stochastic Gradient Descent



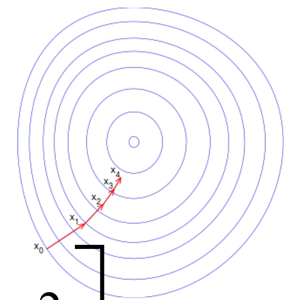
Want to find matrices *P* and *Q*:

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

- **Note:**
  - $\lambda \neq 0$ increases the value of the objective function
  - But we do not care about the value of the objective function but **P** and **Q** that minimize the value
  - And real our goal is to find **P** and **Q** on **seen ratings** so that we predict well the **unseen** ratings



Stochastic Gradient Descent

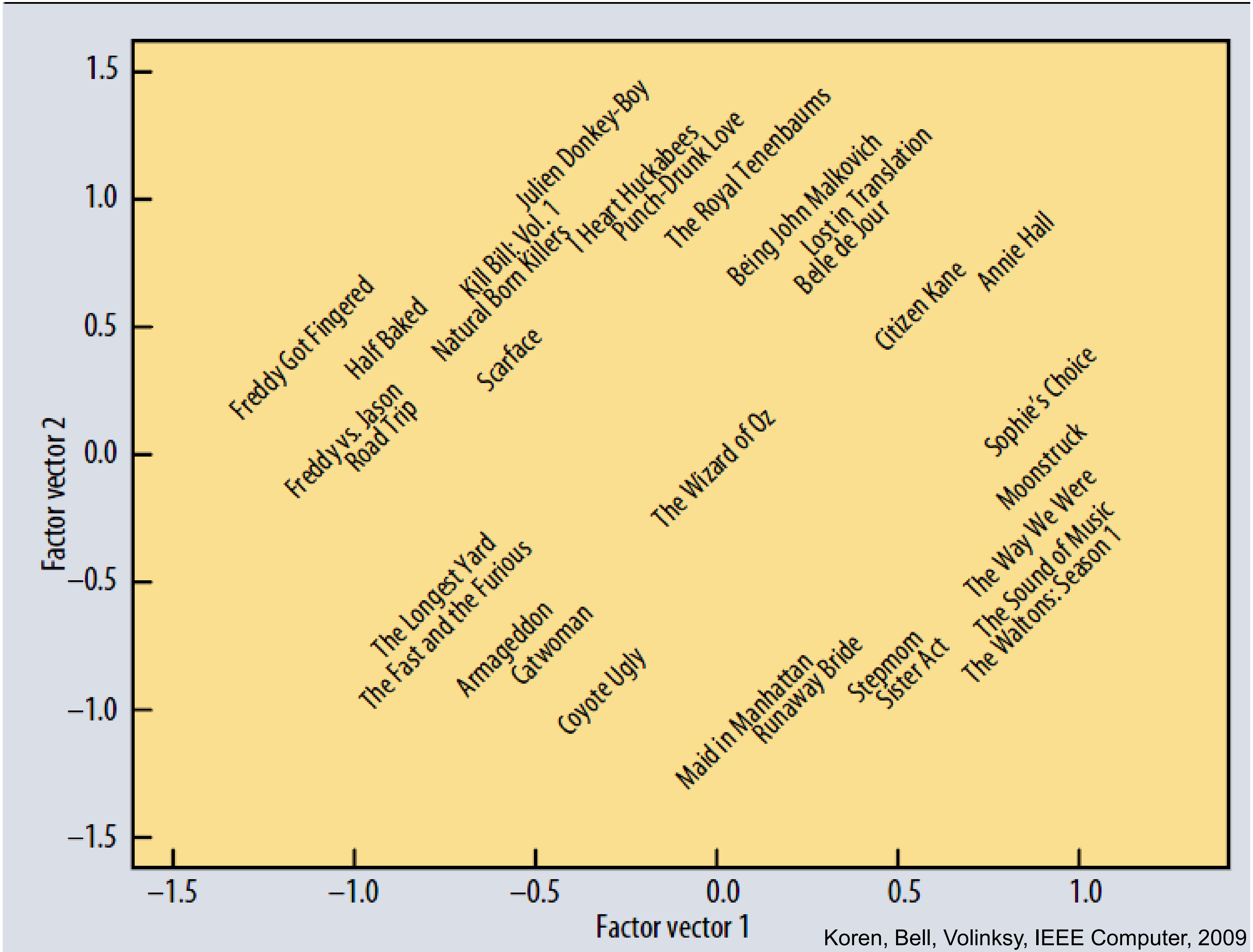


$$\min_{P,Q} \sum_{training} (r_{xi} - q_i \, p_x^T)^2 + \lambda \left[\sum_x \|p_x\|^2 + \sum_i \|q_i\|^2 \right]$$

- **Gradient decent:**
  - Initialize ***P*** and ***Q*** (using SVD, pretend missing ratings are 0)
  - Do gradient descent:
    - $\boldsymbol{P} \leftarrow \boldsymbol{P} - \eta \cdot \nabla \mathbf{P}$
    - $\boldsymbol{Q} \leftarrow \boldsymbol{Q} - \eta \cdot \nabla \mathbf{Q}$

      where $\nabla \boldsymbol{Q}$ is gradient/derivative of matrix ***Q***:

      $\nabla Q = [\nabla q_{ik}]$ and $\nabla q_{ik} = \sum_{xi} -2(r_{xi} - q_i p_x^T) p_{xk} + 2\lambda q_{ik}$
      - Here $\boldsymbol{q_{ik}}$ is entry ***k*** of row $\boldsymbol{q_i}$ of matrix ***Q***
      - **And similarly for $\nabla \boldsymbol{P}$**



Koren, Bell, Volinksy, IEEE Computer, 2009