Matrix Factorization For Recommender Systems

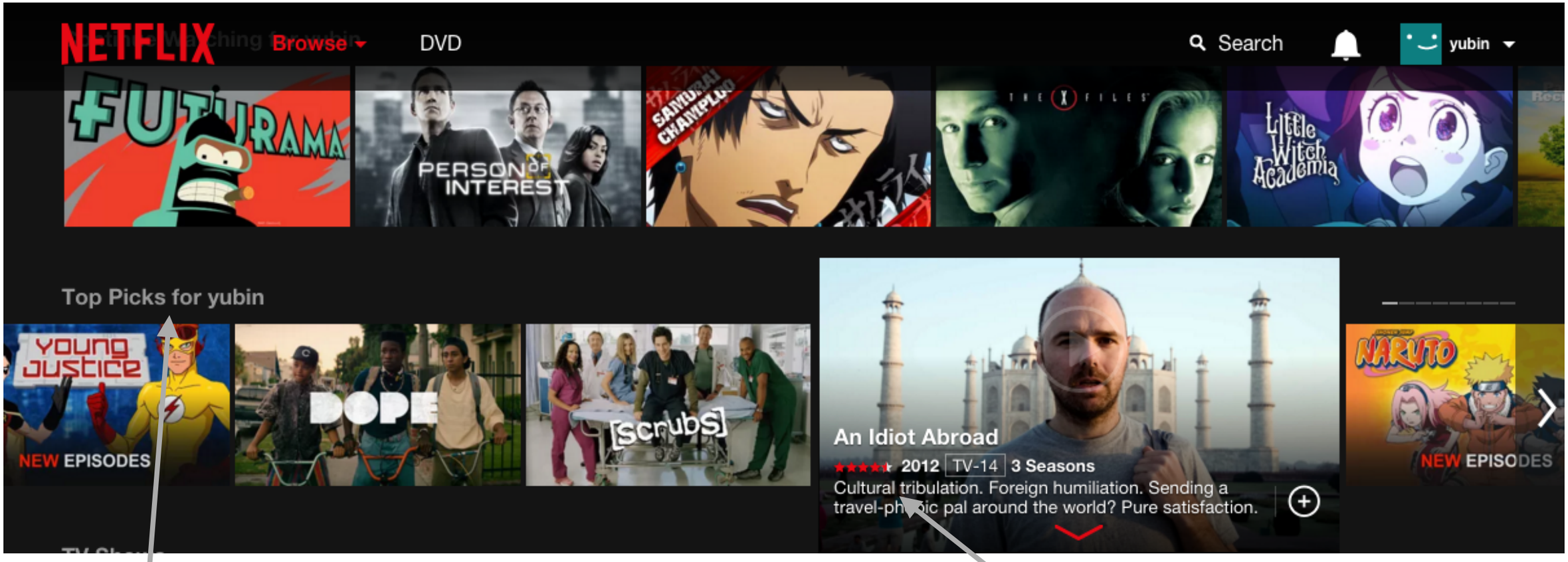
CS 584: Big Data Analytics

Material adapted from Alex Smola (http://alex.smola.org/teaching/berkeley2012/slides/8_Recommender.pdf)
& Lester Mackey (https://web.stanford.edu/~lmackey/papers/cf_slides-pml09.pdf)
& Yong Zheng (http://students.depaul.edu/~yzheng8/papers/
Slide_Talk_DePaul_2015_Matrix_Factorization_In_Recommender_Systems.pdf)

Recommender Systems are Everywhere

System that provides or suggests items to the end users

Recommender System Tasks



Task 1: Predict user rating

Task 2: Top N recommendation

Netflix Prize: $1 M (2006-2009)

Example: Netflix Movie Recommendation

- 480,000 users x 17,700 movies
- Test data: most recent ratings

Training data

user	movie	date	score
1	21	5/7/02	1
1	213	8/2/04	5
2	345	3/6/01	4
2	123	5/1/05	4
2	768	7/15/02	3
3	76	1/22/01	5
4	45	8/3/00	4
5	568	9/10/05	1
5	342	3/5/03	2
5	234	12/28/00	2
6	76	8/11/02	5
6	56	6/15/03	4

Test data

user	movie	date	score
1	62	1/6/05	?
1	96	9/13/04	?
2	7	8/18/05	?
2	3	11/22/05	?
3	47	6/13/02	?
3	15	8/12/01	?
4	41	9/1/00	?
4	28	8/27/05	?
5	93	4/4/05	?
5	74	7/16/03	?
6	69	2/14/04	?
6	83	10/3/03	?

Evaluation Metrics

Error on unseen test set Q, not on training error

- Root Mean Square Error

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

- Mean Absolute Error

$$\mathrm{MAE} = \frac{1}{|S|} \sum_{(i,u) \in S} |\hat{r}_{ui} - r_{ui}|$$

- Rank-based objectives (e.g., What fraction of true top-10 preferences are in predicted top 10?)

Recommendation System Challenges

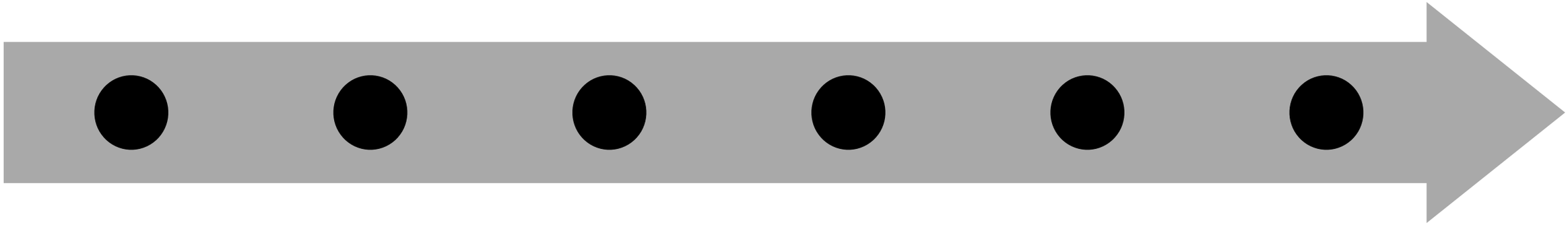
- Scalability - millions of objects and users
- Cold start
  - Changing user base
  - Changing inventory (movies, stories, goods)
  - Attributes
- Imbalanced dataset - user activity / item reviews are power law distributed

Evolution of Recommender Systems

Item hierarchy:
You bought a printer, will also need ink

Collaborative filtering & user-user similarity:
People like you who bought beer also bought diapers

Social + interest graph based:
Your friends like Lady Gaga so you will like Lady Gaga

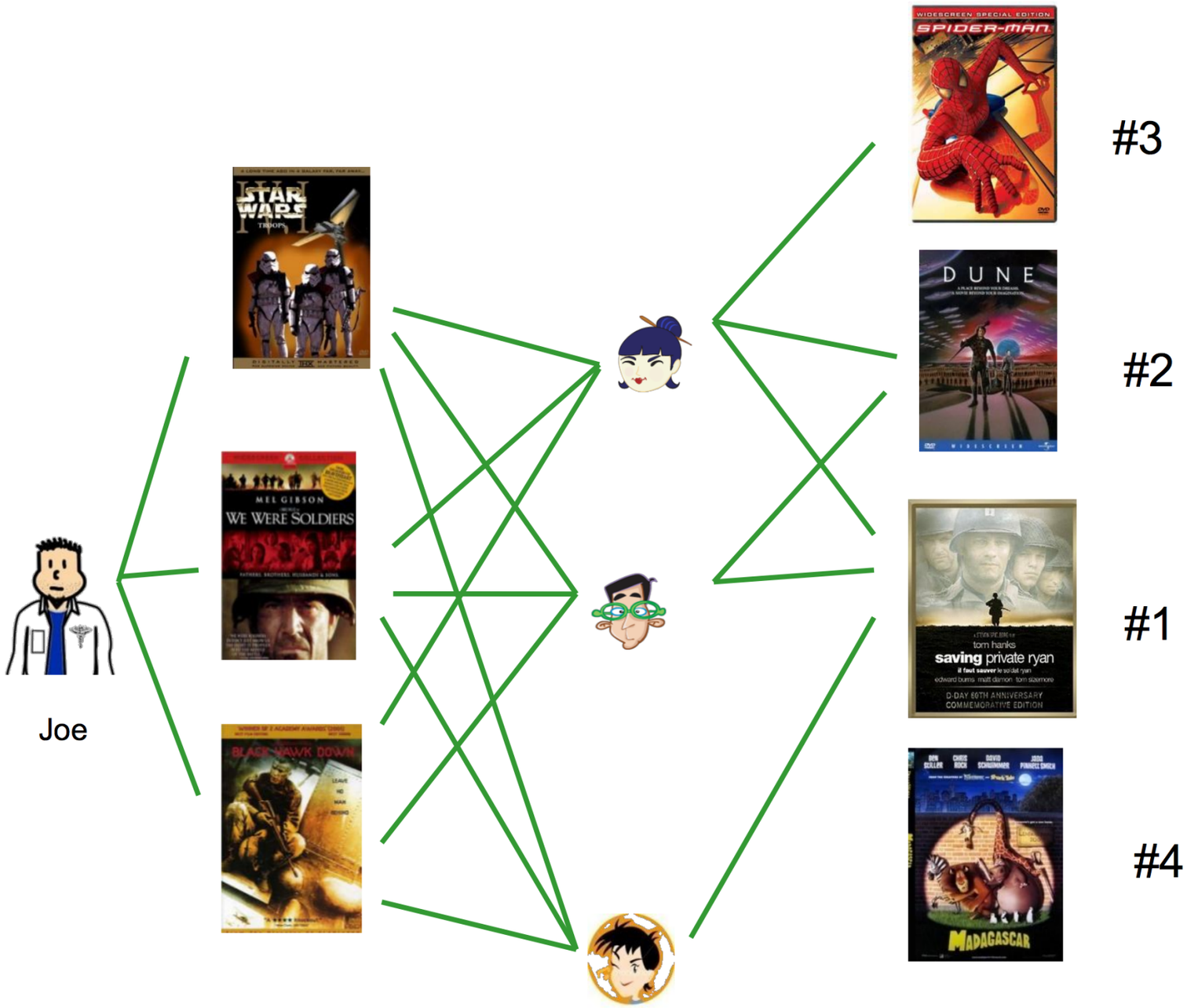


Attribute based:
You like action movies starring Clint Eastwood, you will also like Good, Bad, and Ugly

Collaborative filtering & item-item similarity:
You like Godfather so you will like Scarface

Model based:
Training SVM, LDA, SVD for implicit features

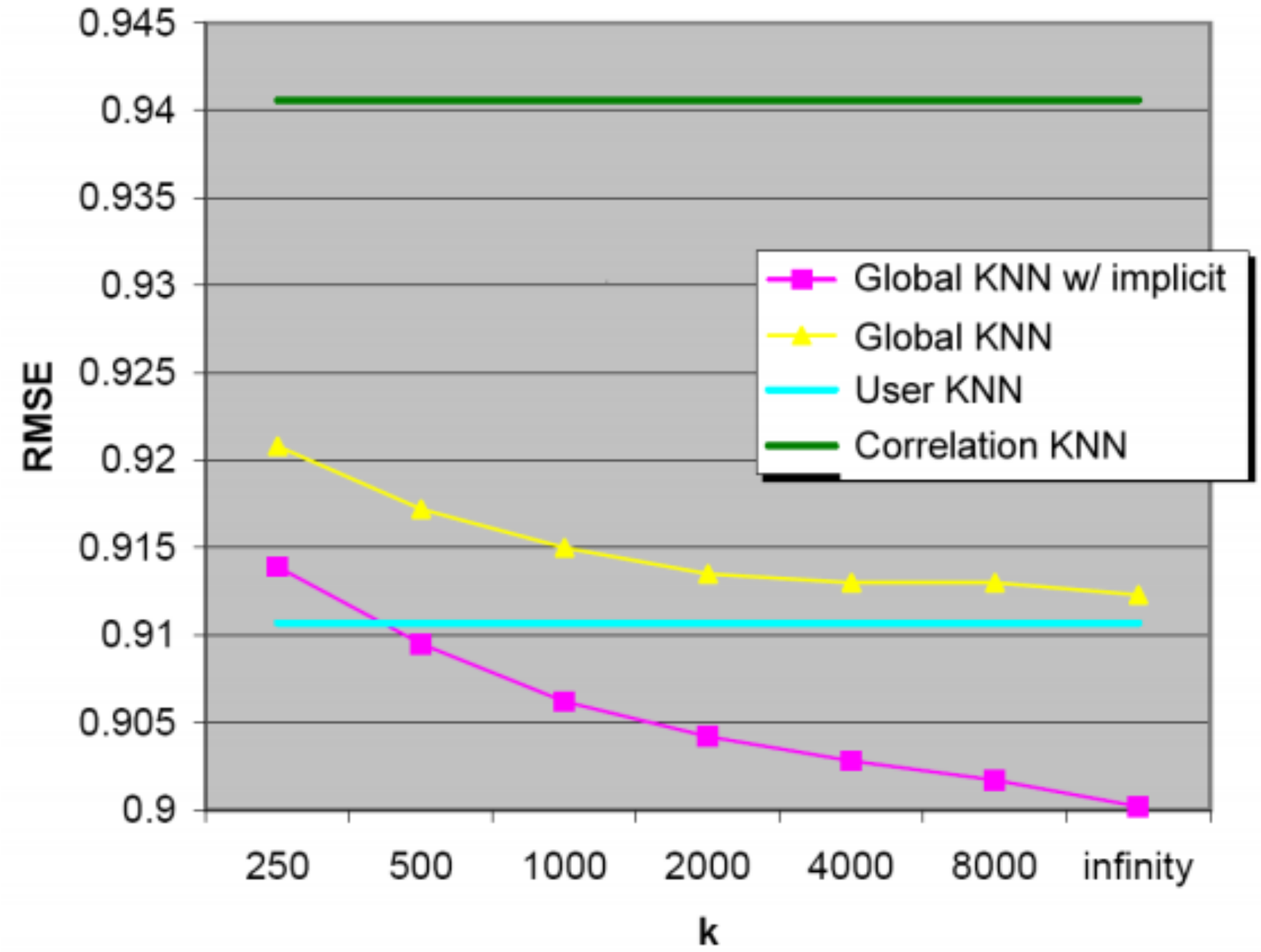
Neighborhood Methods: Basic Idea



kNN Methods

- Flavors: item-based and user-based
- Represent each item as incomplete vector of user ratings
- To predict a new rating for query user u and item i
  - Compute similarity between i and every other item
  - Find K items rated by u most similar to i
  - Predict weighted average of similar item's ratings
- Intuition: Users rate similar items similarly

kNN Results on Netflix Data



kNN: Summary

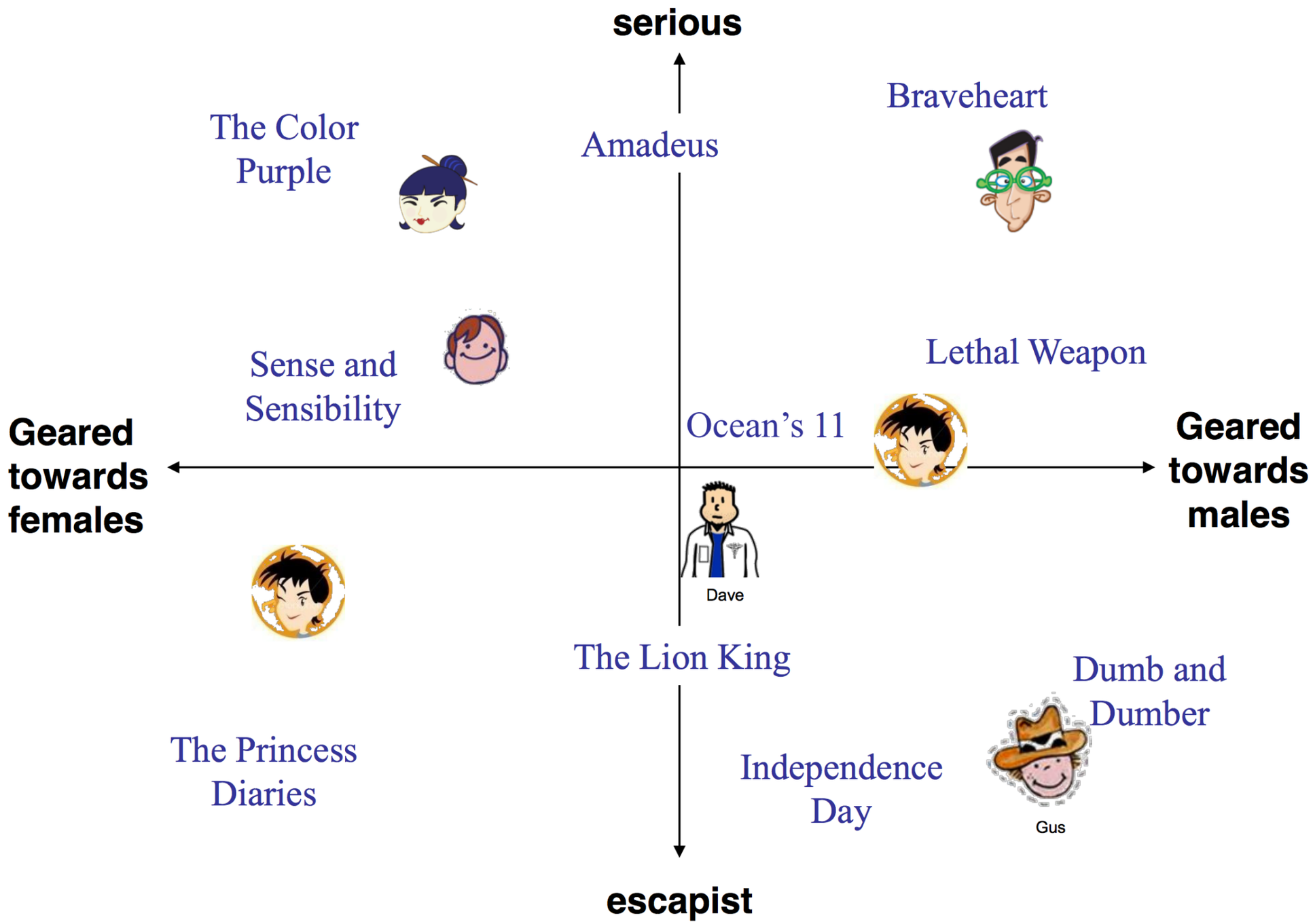
- (+) Intuitive interpretation: you will like what your neighbors like
- (+) Easy to implement and zero training time
- (-) Need to store all items or user vectors in memory
- (-) Recompute similarity scores at test time
- (-) High-dimensional similarity search is not easy

Dimensionality Reduction

- Generate a low-dimensional encoding of a high-dimensional space

- Purposes:
  - Data compression / visualization
  - Robustness to noise and uncertainty
  - Potentially easier to interpret

Dimensionality Reduction

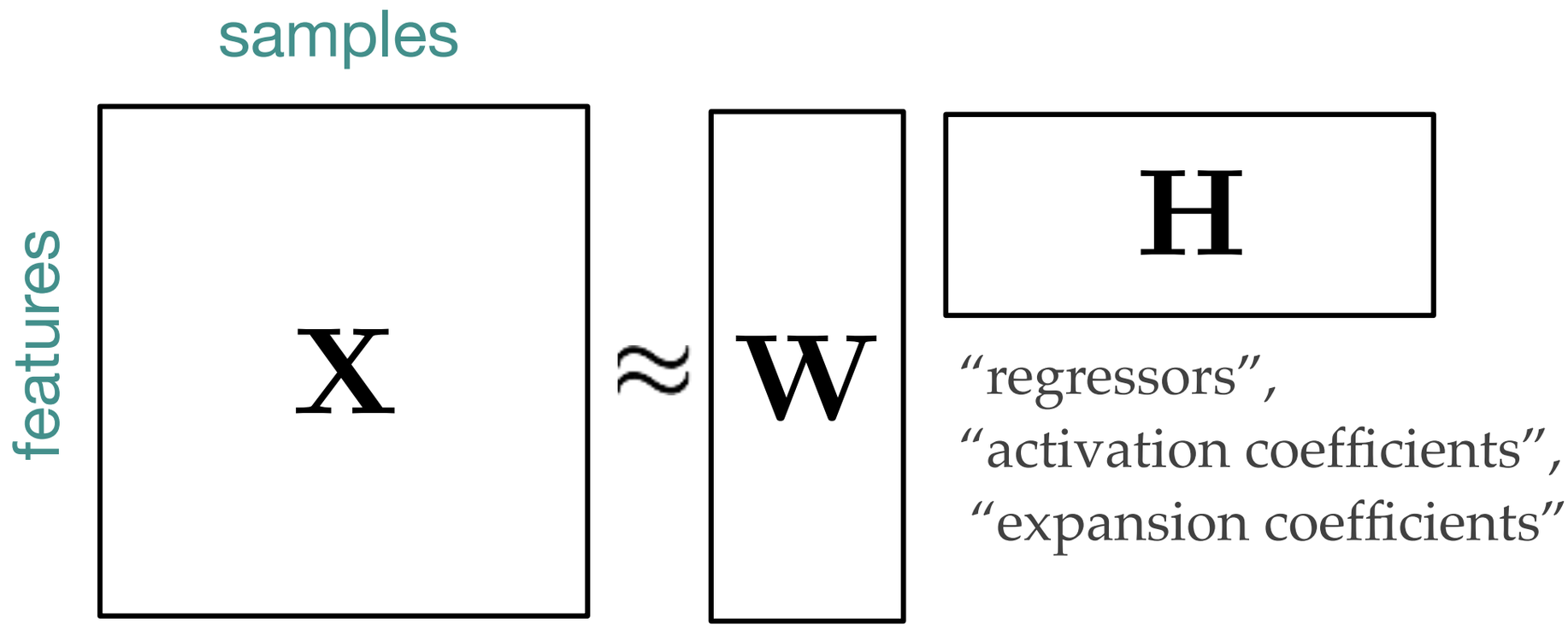


Matrix Factorization

- Low rank approximation to original matrix
- Generalization of many methods (e.g., SVD, QR, CUR, Truncated SVD, etc.)
- Basic Idea: Find two (or more) matrices whose product best approximate the original matrix

$$X \approx \underbrace{W}_{M \times R} \underbrace{H^\top}_{N \times R}, \; R << N$$

Matrix Factorization (Pictorially)

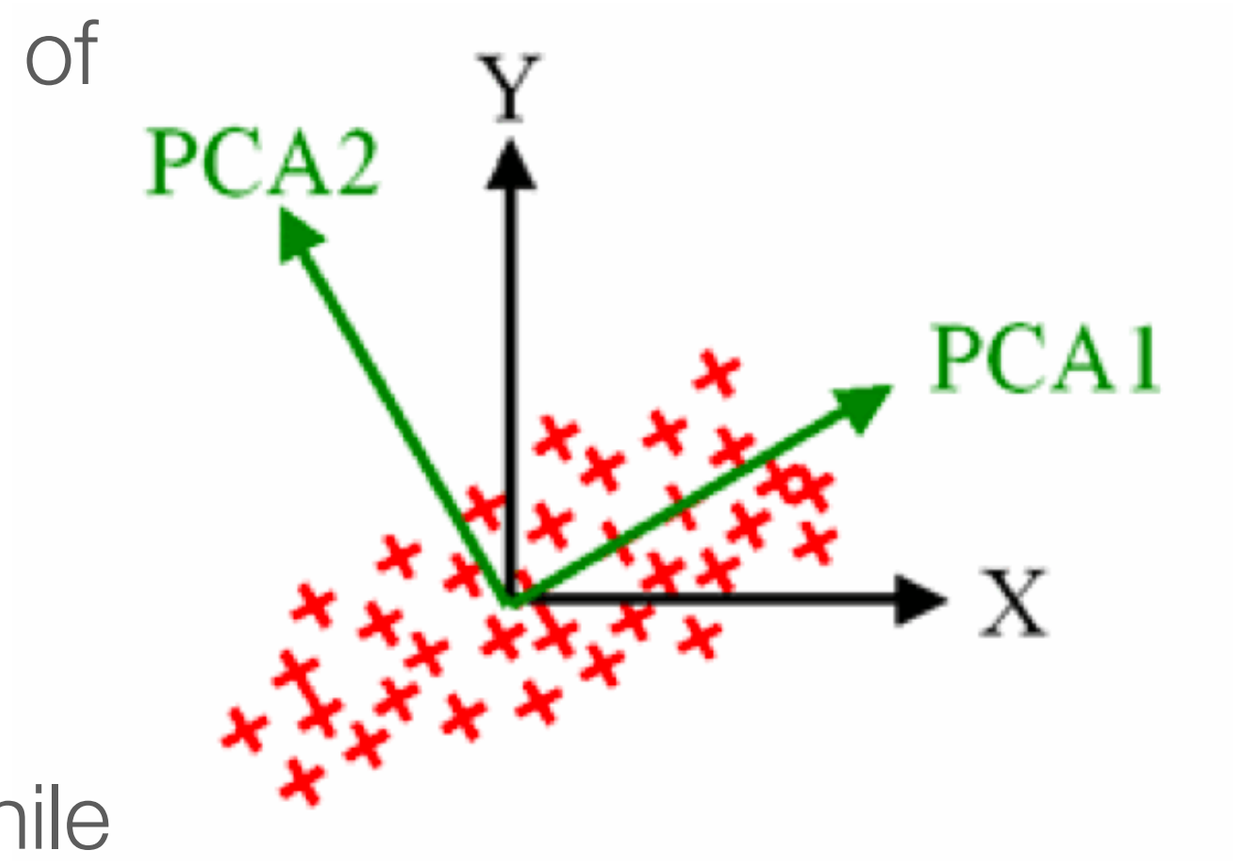


Principal Component Analysis (PCA)

- Each principal component (PC) is linear combination of uncorrelated attributes / features

- k^{th} PC is orthogonal to all previous PCs

- Reduce dimensionality while maintaining as much variance



https://prateekvjoshi.files.wordpress.com/2014/10/2-pca.png

Singular Value Decomposition (SVD)

- Any m x n matrix can be decomposed in this way

$$\underbrace{A}_{m\times n} = \underbrace{U}_{m\times r}\underbrace{\Sigma}_{r\times r}\underbrace{V^\top}_{r\times n}$$

- r is the rank of matrix A

- U is a column-orthonormal matrix

- V is a column-orthonormal matrix

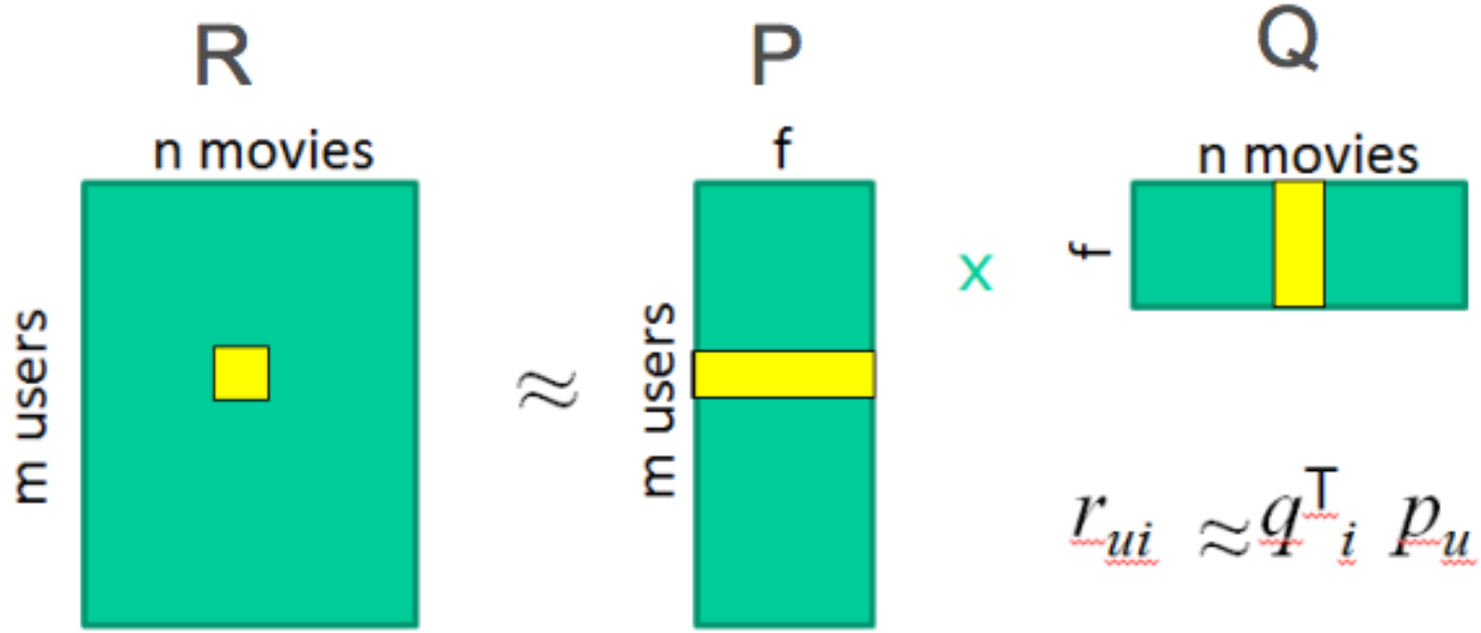
- Σ is a diagonal matrix with the singular values sorted in descending order

SVD: A “Master” Algorithm

- Solve a linear system or any least squares problem
- Compute other factorizations: LU, QR, eigenvectors, etc.
- Data analysis: PCA, LSI, etc
- Standard algorithms are very stable, have only $O(n^3)$ asymptotic complexity and provide double precision accuracy

SVD to MF

- Create two new matrices (user and item matrices) where the square root of the singular values are distributed to each matrix U and V



- Interpretation:
  - p_u indicates how much user likes each latent factor f
  - q_i means the contribution of item to each of the latent factors f

SVD with Missing Values

- Conventional SVD is undefined for missing entries
- One idea: Expectation maximization as form of imputation
  - Fill in unknown entries with best guess
  - Apply SVD
  - Repeat
- Can be expensive and inaccurate imputation can distort data

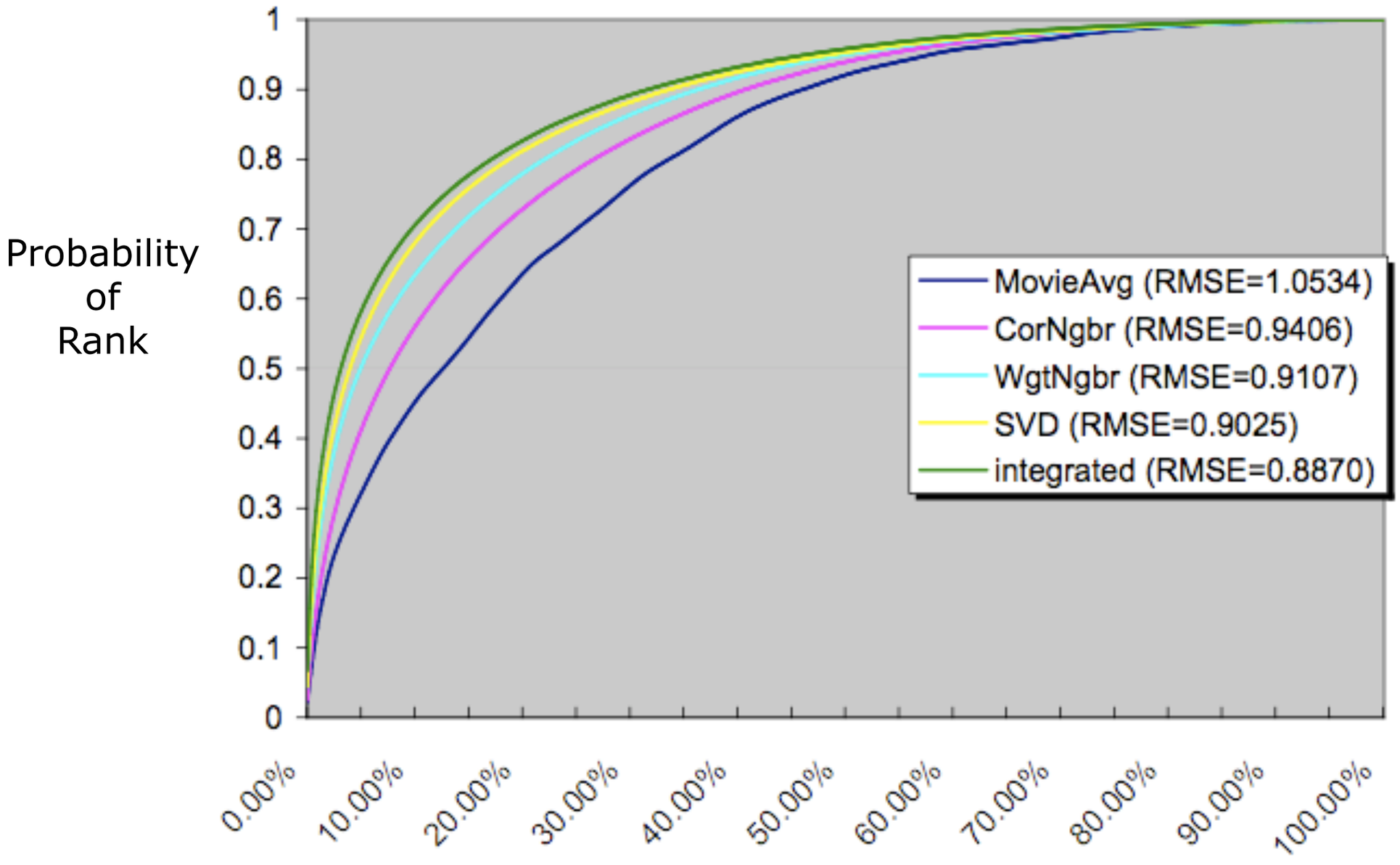
SVD with Missing Values

- New idea: Model only the observed entries and avoid overfitting via regularization

$$\min_{q_\cdot, p_\cdot} \sum_{(u,i)\in K} (r_{ui} - q_i^T p_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2)$$

- Two methods for solving the new model
  - Stochastic gradient descent
  - Alternating least squares - easier to parallelize as each q_i is independent and more memory efficient

Netflix Results: SVD vs. Others

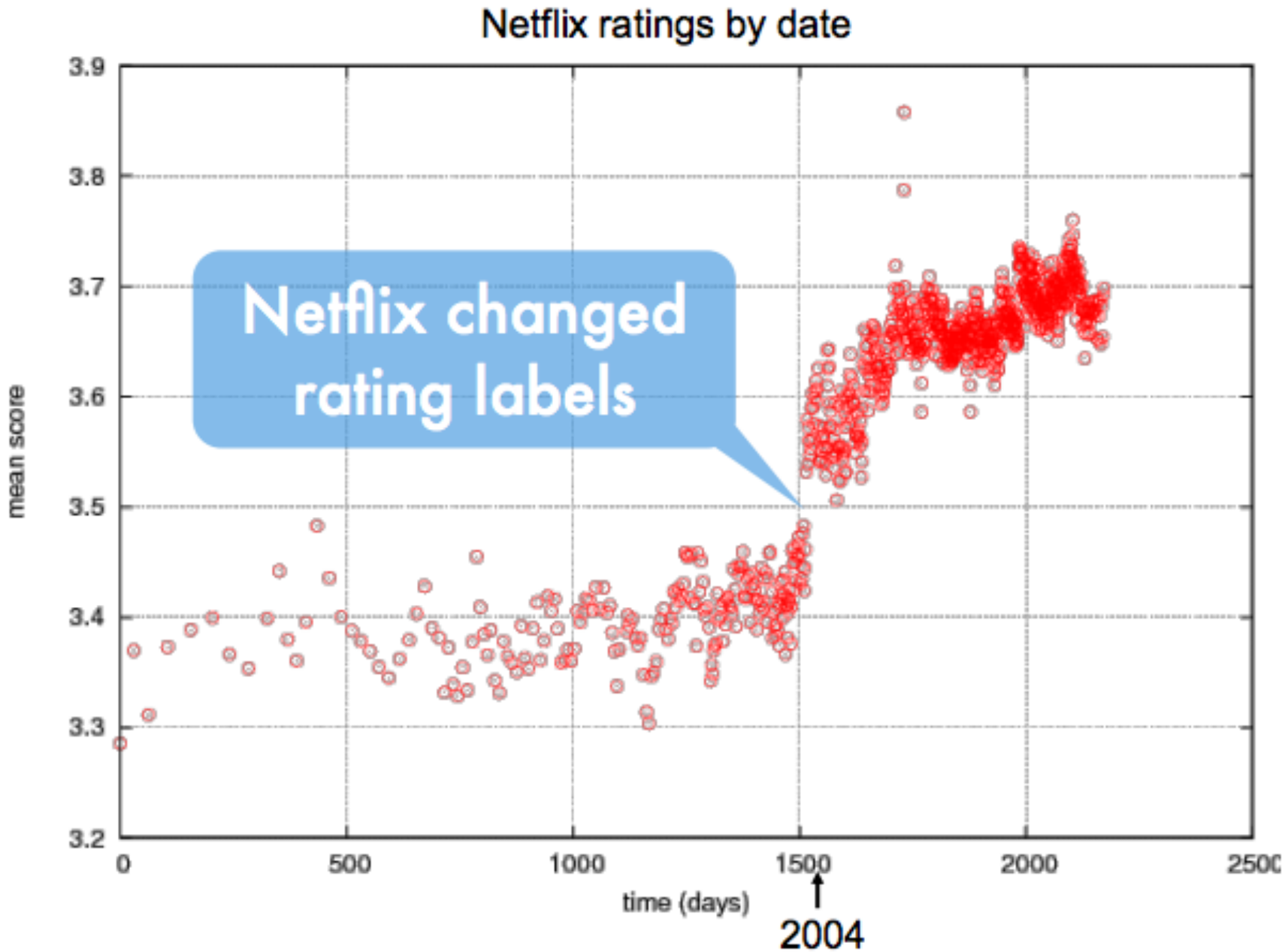


SVD with Bias

- Some users tend to give higher ratings than others
- Some items tend to receive higher rating than others
- Introduce three new components: global average, item bias, user bias

$$\underset{p,q}{\text{minimize}} \sum_{(u,i) \in S} (r_{ui} - (\mu + b_u + b_i + \langle p_u, q_i \rangle))^2 + \lambda \left[\|p\|_{\text{Frob}}^2 + \|q\|_{\text{Frob}}^2 + \|b_{\text{users}}\|^2 + \|b_{\text{items}}\|^2 \right]$$

SVD with Temporal Dynamics



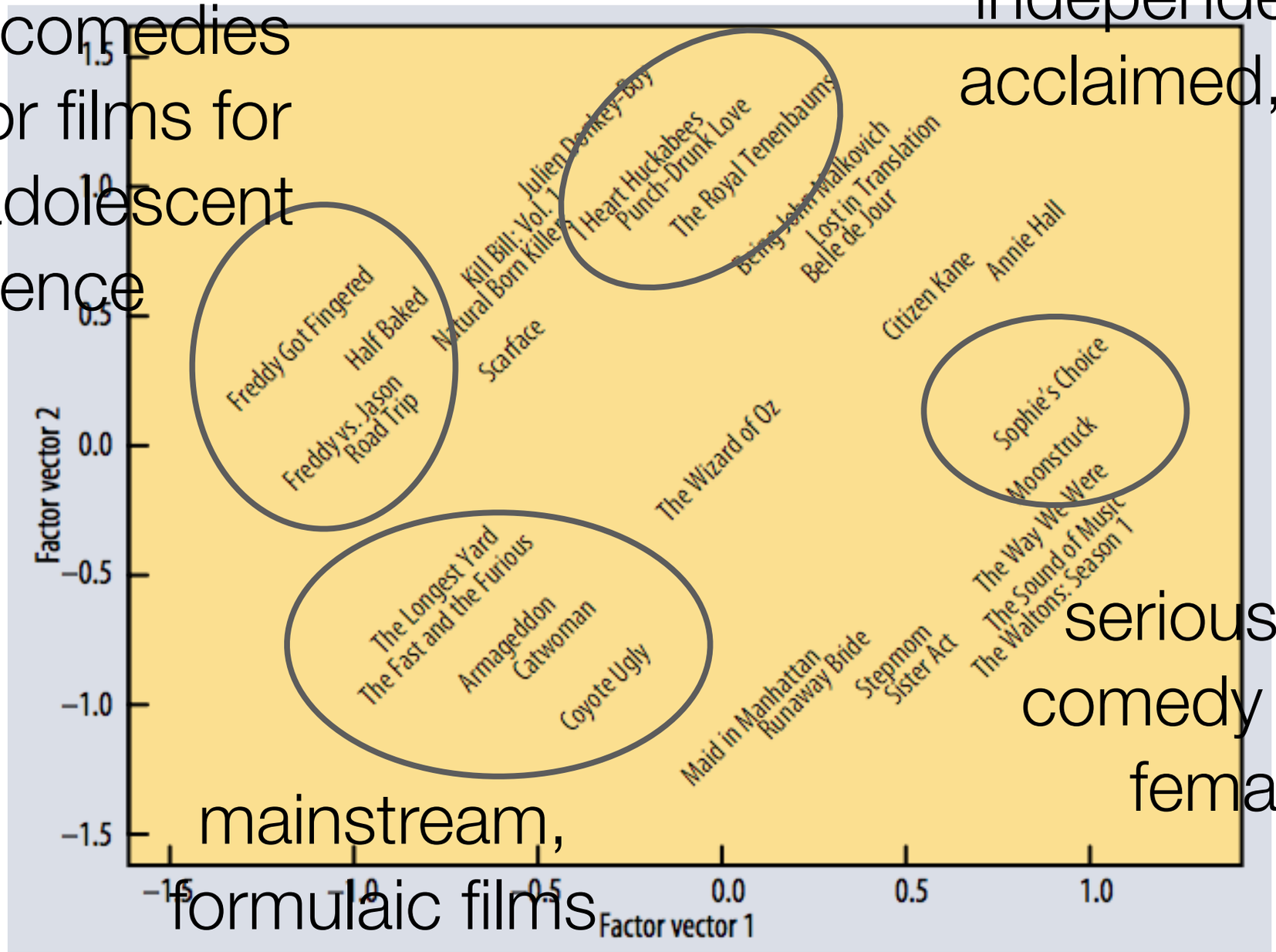
SVD with Temporal Dynamics

- Items
  - Seasonal effects (holidays, etc.)
  - Public perception of movies (Oscars, SAG, etc.)
- Users
  - Changed review labels
  - Anchoring (relative to previous movie)
  - Selection bias for time of viewing
  - …

Netflix Results: Latent Factors

lowbrow comedies and horror films for male or adolescent audience

independent, critically acclaimed, quirky films



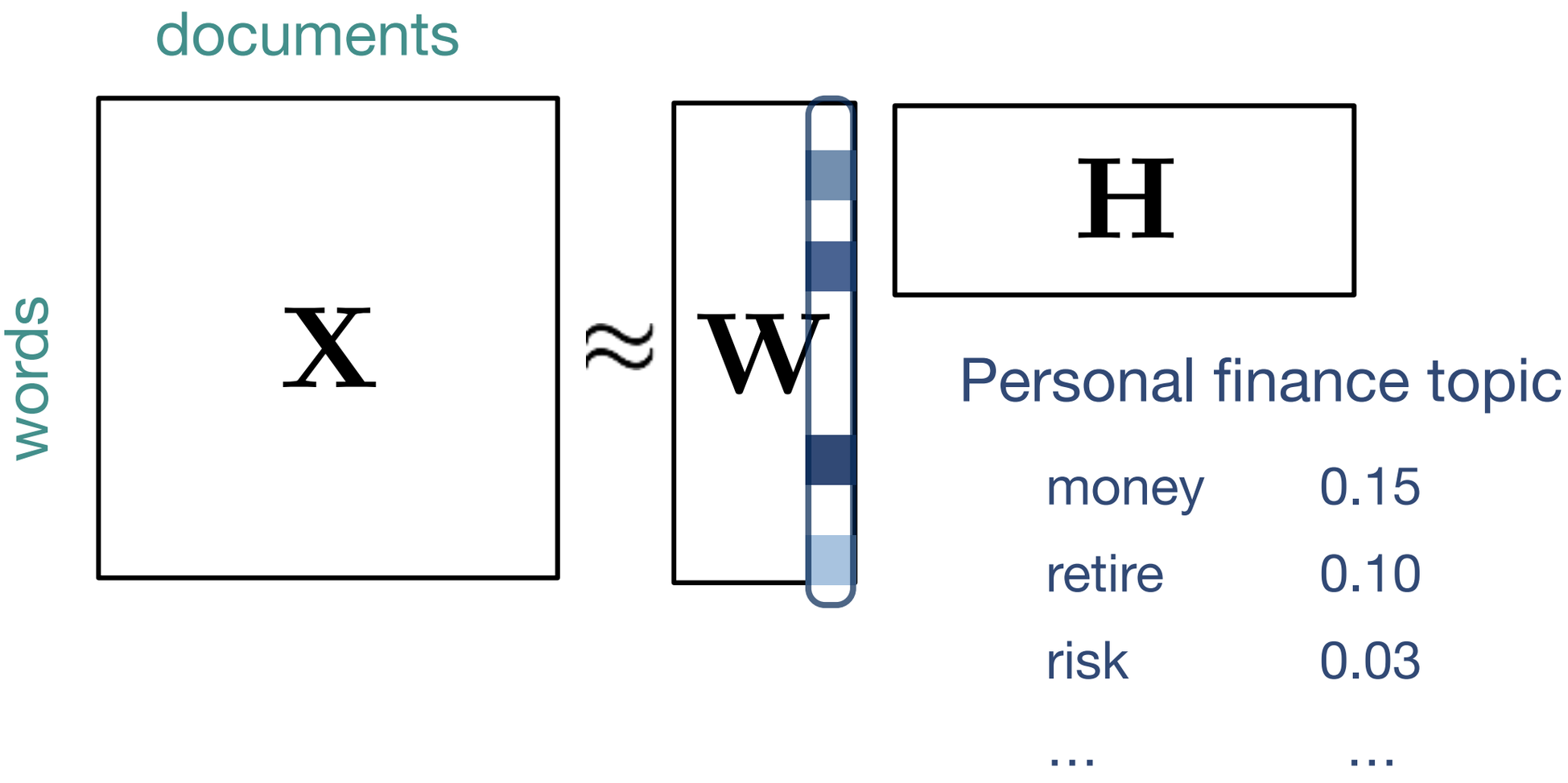
serious drama or comedy with strong female leads

mainstream, formulaic films

Nonnegative Matrix Factorization (NMF)

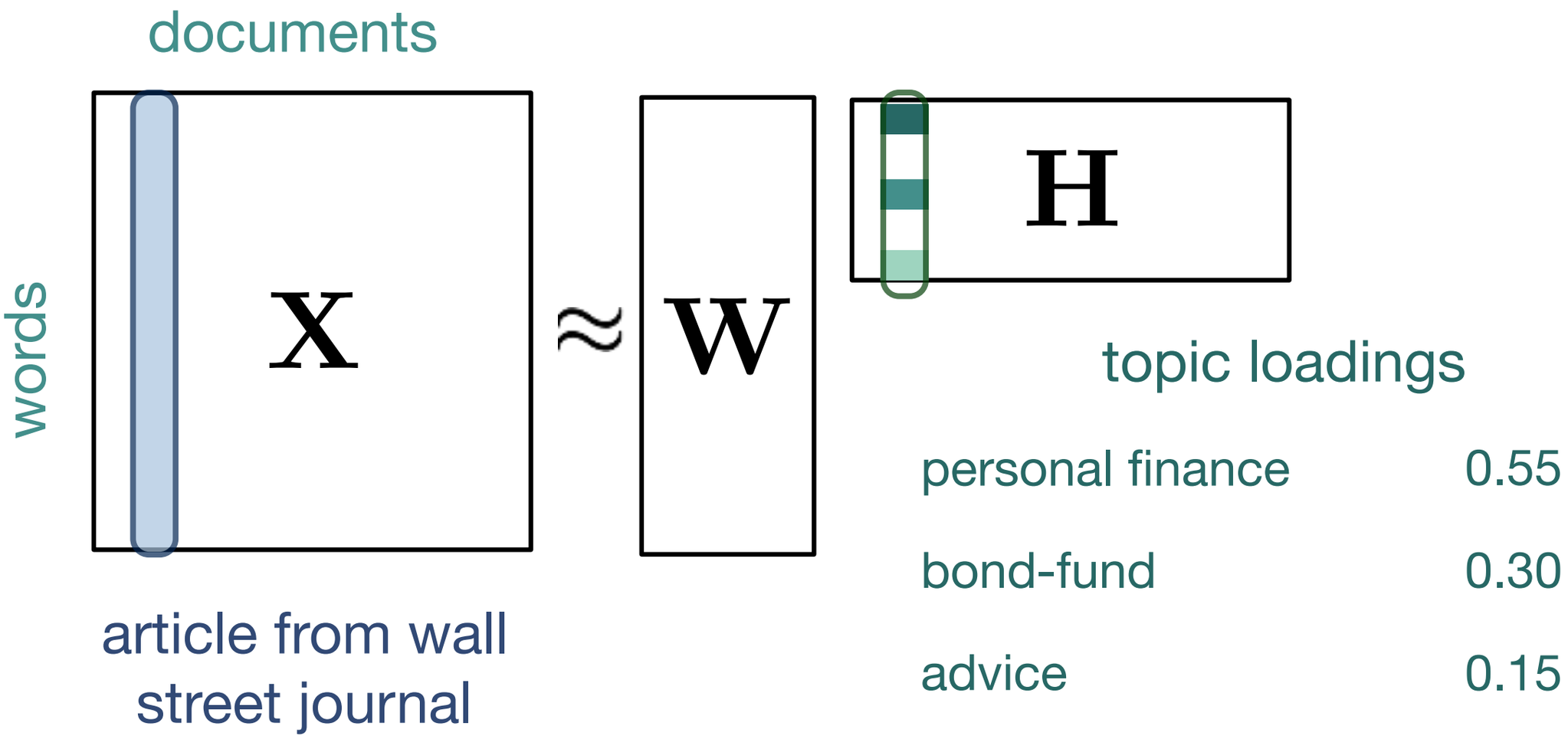
- Popularized by Lee and Seung (1999) for “learning the parts of objects”
- Both W and H are nonnegative
- Empirically induces sparsity
- Improved interpretability (sum of parts representation)
- Applications to text classification, information retrieval, collaborative filtering, etc.

NMF (Example)



WLOG, assume columns of **W** and **H** sum to 1

NMF (Example)



WLOG, assume columns of $\mathbf{W}$ and $\mathbf{H}$ sum to 1

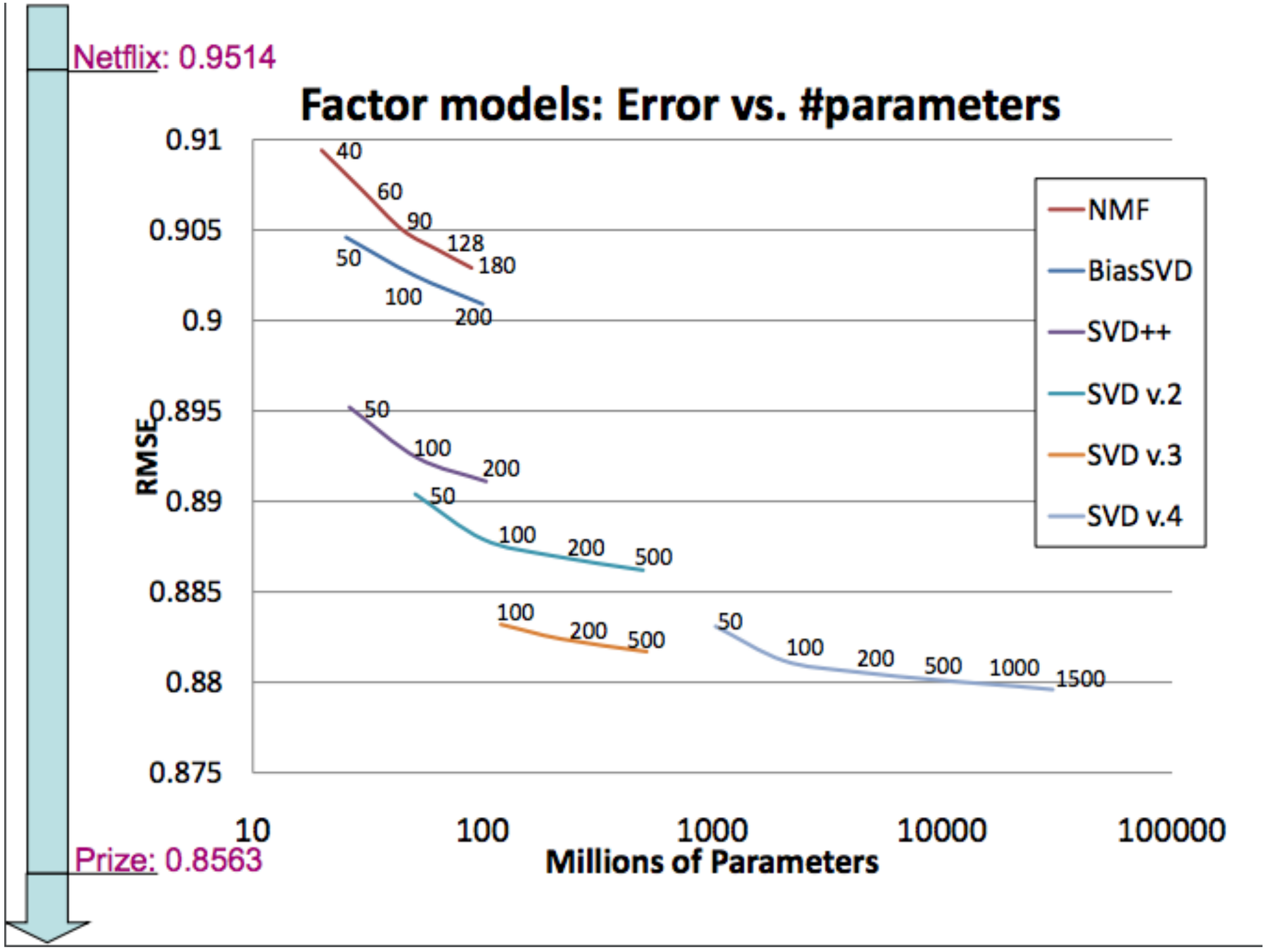
Algorithms for NMF

Local Search: Given W, compute H, compute W, …

- Known to fail on worst-case inputs (stuck in local optima)

- Highly sensitive to:
  - Cost function
  - Update procedure
  - Regularization

Netflix Results: RMSE



Many More Ideas

- Cold start (new users)
- Different regularization for different parameter groups and differs users
- Sharing of statistical strength between users
- Hierarchical matrix co-clustering / factorization
- Incorporate social network, user profiles, item profiles
- …

Challenges for Recommendation Systems

- Relevant objectives
  - Predicting actual rating may be useless!
  - May care more about ranking of items
- Missing at random assumption
  - How can our models capture information in choices of our ratings?
- Handling users and items with few ratings

Challenges for Recommendation Systems (2)

- Multiple individuals using the same account — individual preference
- Preference versus intention
  - Distinguish between liking and interested in seeing / purchasing
  - Worthless to recommend an item a user already has
- Scalability
  - Simple and parallelizable algorithms are preferred