# PREFER: Exploring Preference RElations as FEedback in Recommender Systems

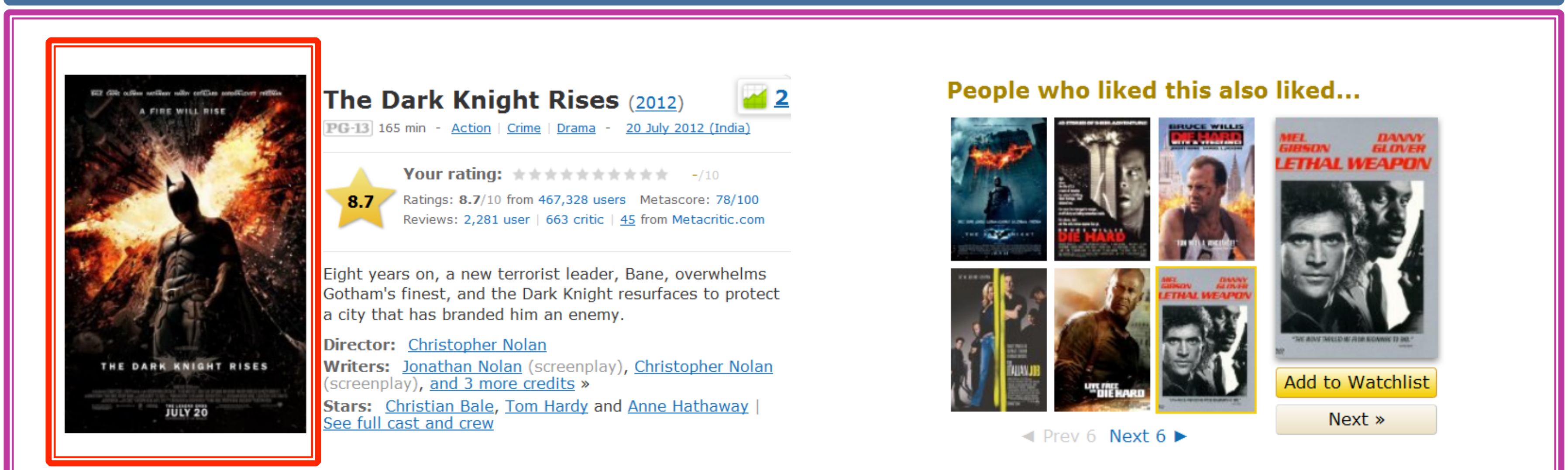
Workshop on Conformal Prediction for Reliable Machine Learning
Dec 10, 2015


भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Maunendra Sankar Desarkar

IIT Hyderabad

# Recommender Systems: Example

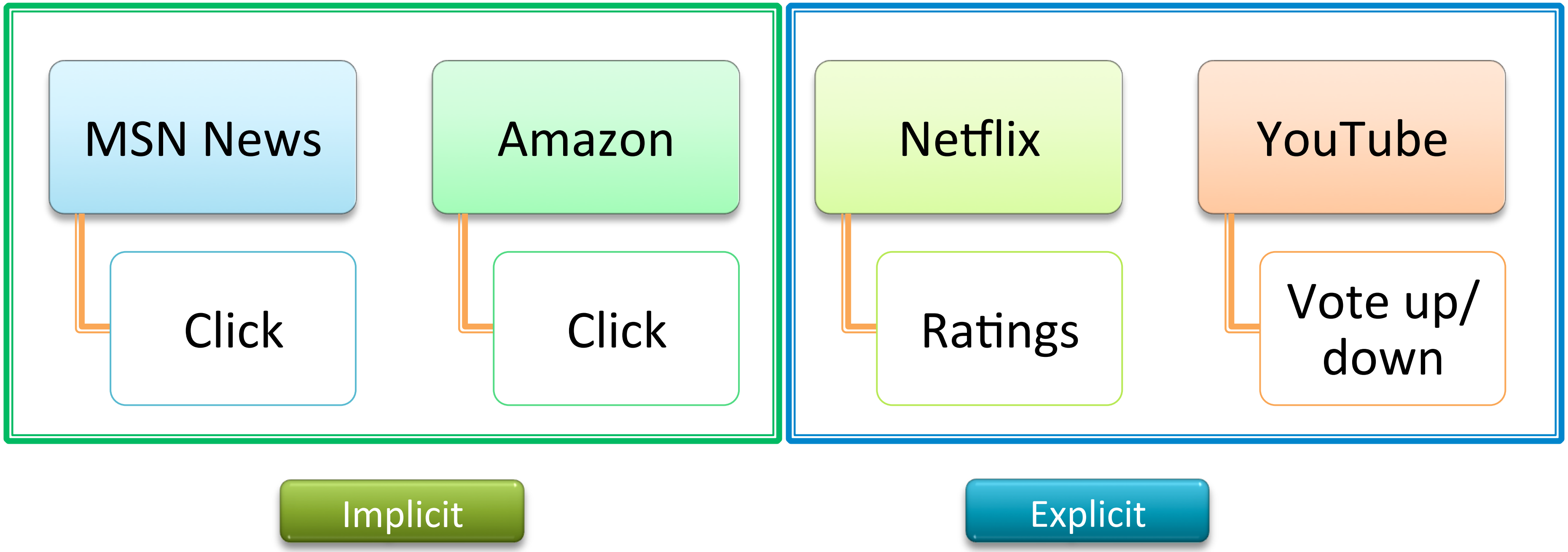



# Recommender Systems: Example

# Recommender Systems

- Recommend / suggest items to the users
- The recommendations are personalized
- System gathers user feedback
- Feedbacks: Ratings, Like/Dislike, Clicks etc.
- Uses these feedbacks to generate personalized recommendations

# Examples of feedback



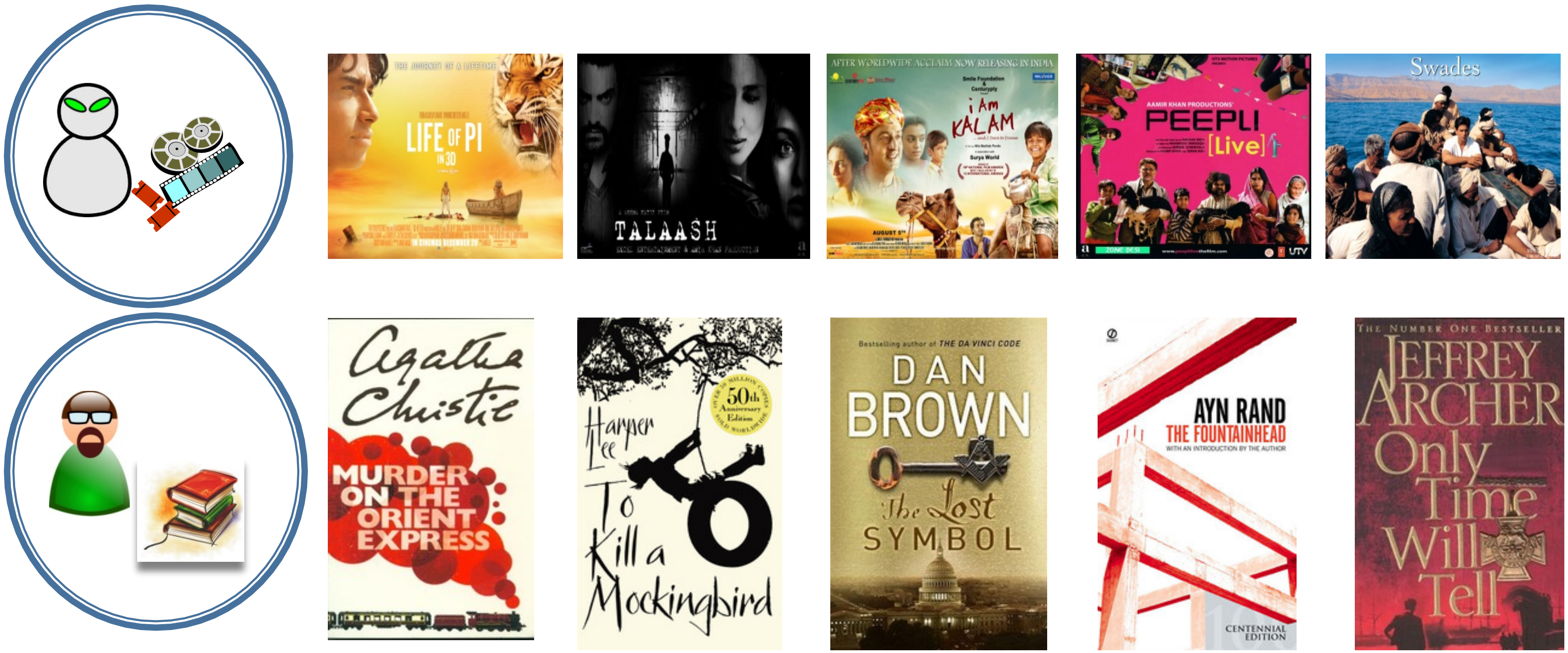
# Few Problems RS Domain

- Rating Prediction
  - Predict the rating that a user would give to an item that he has not rated in the past

| User ID | Item ID | Rating |
|---|---|---|
| 1 | 1 | ★★★ |
| 1 | 5 | ★★★★★ |
| 1 | 6 | ★★★ |
| 1 | 8 | ? |
| 2 | 1 | ★★★★★ |
| 2 | 3 | ★★ |
| 2 | 5 | ? |

# Few Problems RS Domain

- Item Recommendation
  - Suggest a list of items to a user



# Preference Relations as User Feedback

# Preference Relations: Easier to Give Feedbacks




# Preference Relations: No Choice Constraint due to Rating Scale

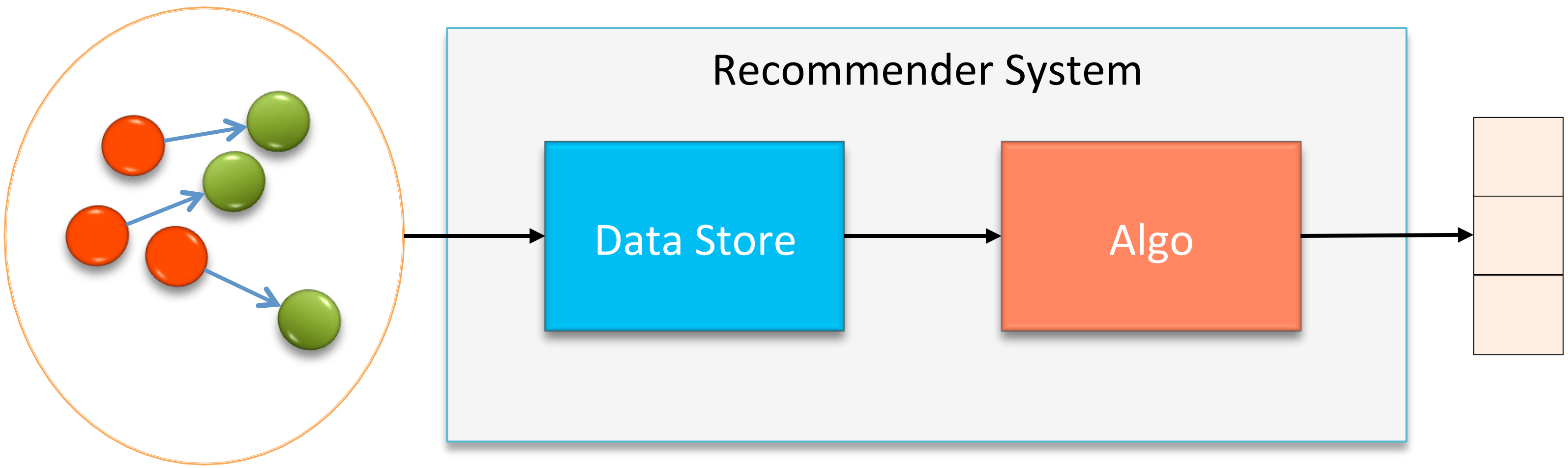

# Preference Relations: Address Rating Biases of the Users

| User ID | Movie ID | Rating |
|---|---|---|
| 1 | 10 | 5 |
| 1 | 20 | 5 |
| 1 | 30 | 4 |

| User ID | Movie ID | Rating |
|---|---|---|
| 2 | 10 | 4 |
| 2 | 20 | 3 |
| 2 | 30 | 3 |


# Research Question

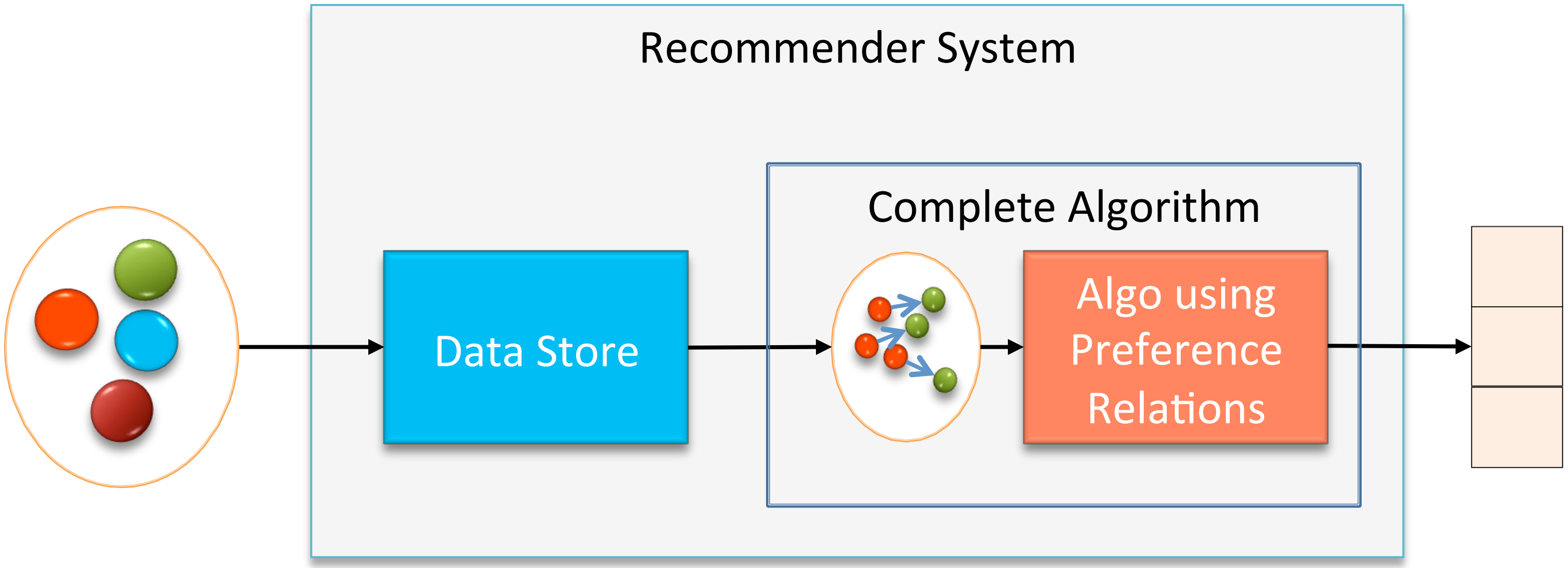
Develop recommender system that uses preference relations as feedback




*Not attempting to answer this design question in this talk.

# Research Question

Induce preference relations from existing feedbacks and use them in recommender systems?




* Focus of the next part of the talk

# Aggregating preference graphs for collaborative rating prediction

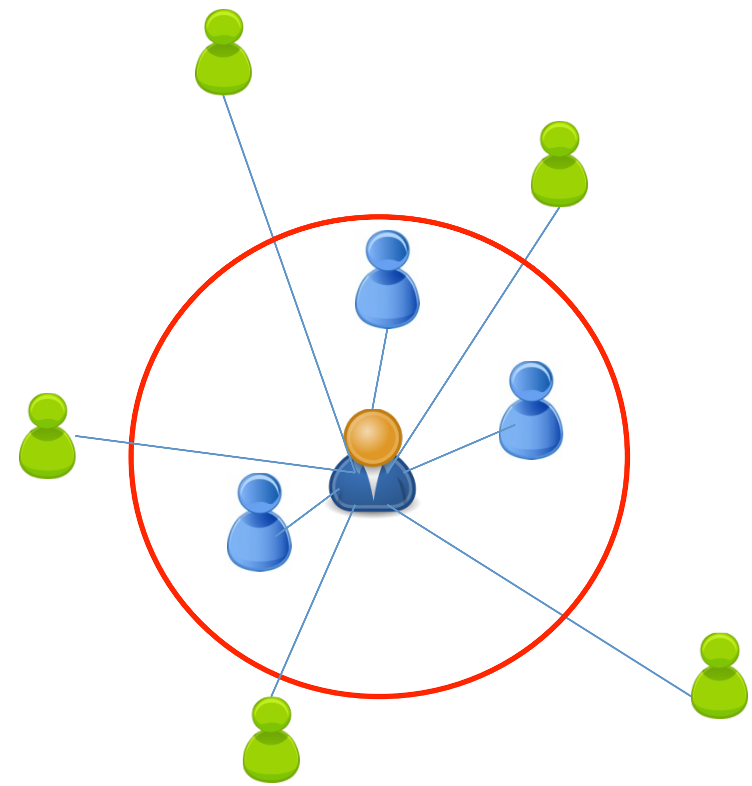
Presented in

RecSys 2010

Barcelona

# Neighborhood based CF for Rating Prediction

- Find rating for <test user, test item> pair
- Similar users rate items similarly
- For each *test user*, pick neighbors / experts
- Use ratings given by those users to predict ratings for the test user

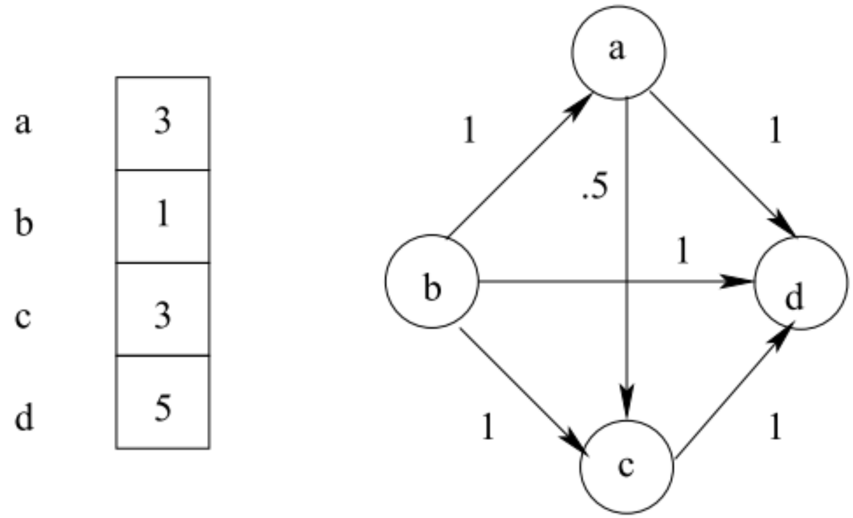


# Inducing Preference Relations

- If $r_{ui} > r_{uj}$, we assume that User $u$ prefers $i$ over $j$

- Why do this?
  - $r_{ui}$, $r_{uj}$ etc. can be noisy. But if many people say that $r_{ui} > r_{uj}$, then that information can be useful
  - Allows to connect different items. Helpful for *sparse* items.

# Rating profile as preference graph




- Each user's rating profile is considered as a preference graph.
- Nodes are the items rated by the user
- Edges denote preference relations over the item pairs

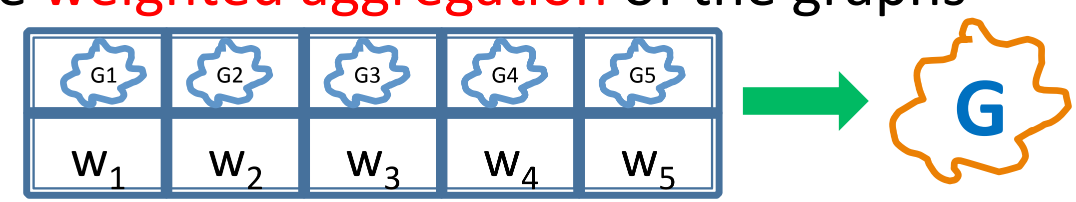
# Outline of our approach

- Given a user (u) and a set of items (I) predict the ratings.
- **Phase 1 (Aggregation Phase):**
  - Represent ratings from each user as preference graph


  - Assign weights to the users (or their preference graphs – algorithm motivated by online learning)

| | | | | |
|---|---|---|---|---|
| $w_1$ | $w_2$ | $w_3$ | $w_4$ | $w_5$ |

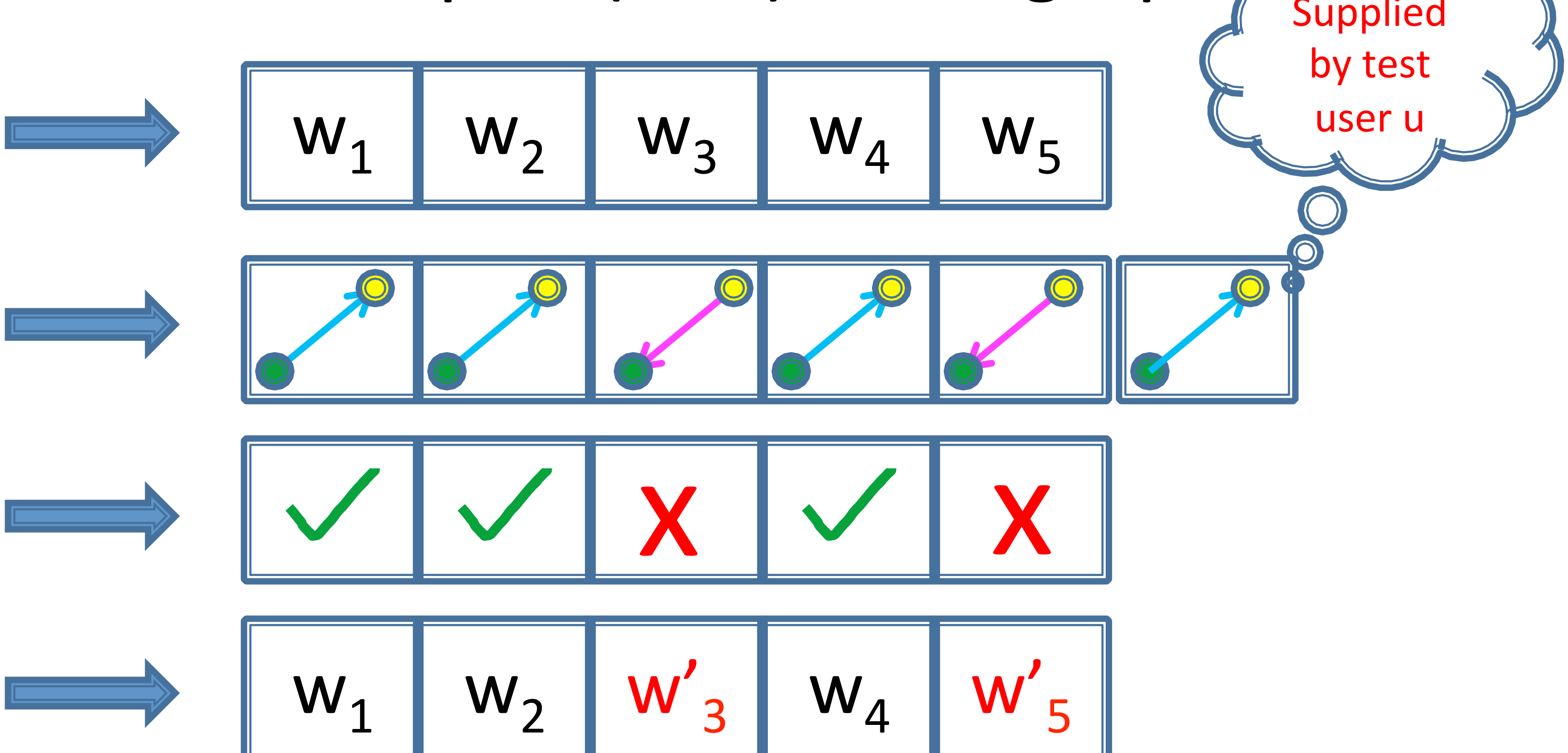
  - Compute weighted aggregation of the graphs



# Weight Assignment Algorithm

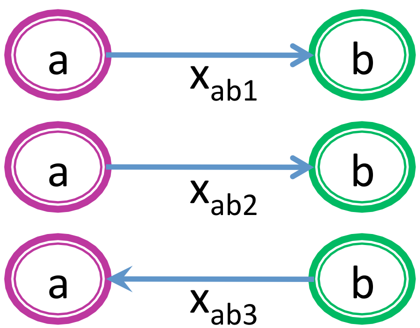
Input:

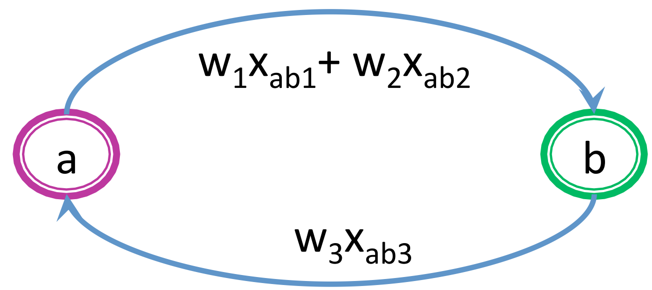

For each item pair (A B) in u's graph



# Create Aggregate Graph

- Aggregate Graph: Weighted combination of the individual preference graphs




Weight(a,b) = *weighted votes* in favor of the relation {b is better than a}

Weight(b,a) = *weighted votes* in favor of the relation {a is better than b}

# Recover Ratings from Aggregate Graph


Possible ratings "consistent" with the edge direction: 3, 4, 5


Back edge

- Assign ratings so as to minimize the total weight of back edges (W)

- This is the first rating prediction algorithm that works by looking at preference relations only, completely ignoring absolute rating information.

# Our algorithm: Phase 2

$$\text{minimize } Z = W + CX$$

$$\text{where}$$

$$W = \sum_{\substack{i,j \in M \\ k,l \in R}} x_{ik} x_{jl} (\delta_{lk} w_{ji} + \delta_{kl} w_{ij}) \; + \sum_{\substack{i \in M, a \in T \\ k \in R}} x_{ik} (\delta_{r_a k} w_{ai} + \delta_{k r_a} w_{ia})$$

$$X = \sum_{\substack{i \in M \\ k \in R}} x_{ik} (k - \mu)^2$$

$$\delta_{\alpha\beta} = \begin{cases} 1 & \text{if } \alpha > \beta \\ c & \text{if } \alpha = \beta \\ 0 & \text{if } \alpha < \beta \end{cases}$$

$$\text{subject to} \quad \sum_k x_{ik} = 1, \forall 1 \le i \le m$$

# Results (on Movielens dataset)

| Ratings given (Test User) | <= 10 | <= 20 | <= 30 | <= 40 |
|---|---|---|---|---|
| Pref-GrAgg | 1.144 | 1.122 | 1.115 | 1.109 |
| Somers [2] | 1.616 | 1.355 | 1.295 | 1.302 |
| UPCC [1] | 1.342 | 1.216 | 1.174 | 1.173 |
| IPCC [1] | 1.816 | 1.468 | 1.324 | 1.234 |
| RWR [3] | 1.263 | 1.255 | 1.250 | 1.248 |

RMSE corresponding to item sparsity 40
(Maximum number of available ratings for the test items is 40)

**Improvements vary from 5% to 9%**

# Preference Relation Based Matrix Factorization for Recommender Systems

Presented in

UMAP 2012

Montreal

# Latent Feature Model: Pictorially

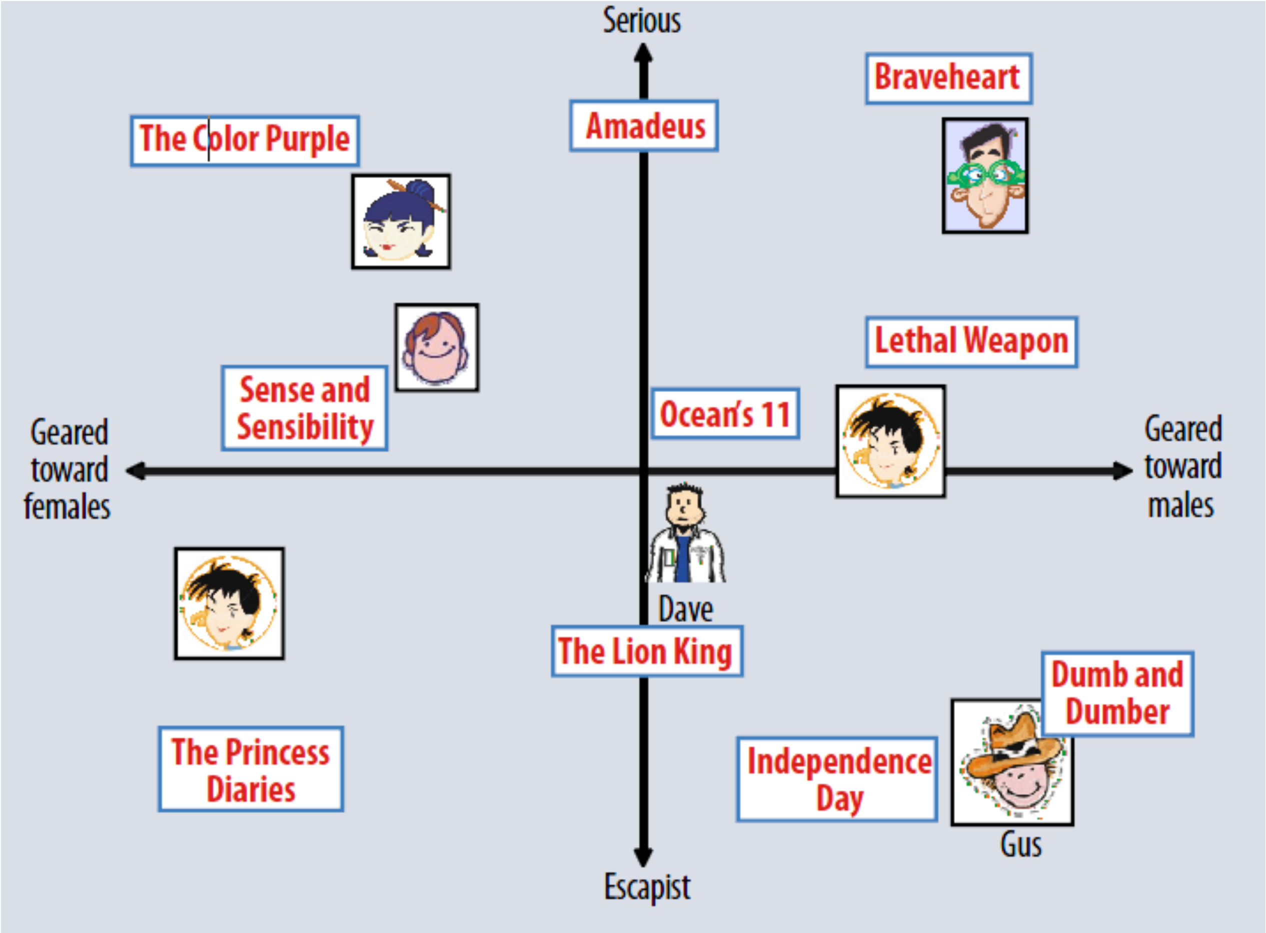



**Figure 2.** A simplified illustration of the latent factor approach, which characterizes both users and movies using two axes—male versus female and serious versus escapist.



Figure taken from [4]

# Non Negative Matrix Factorization

- **User Vector**: $p_u$
- **Item Vector**: $q_i$
- **Predicted utility:** $p_u q_i^T$
- **Objective function to optimize**:

$$\min_{p,q} \sum_{\langle u,i,r_{ui} \rangle} (r_{ui} - p_u q_i^T)^2 + \lambda_1 \sum_{u \in U} \|p_u\|^2 + \lambda_2 \sum_{i \in I} \|q_i\|^2.$$

  - $1^{st}$ term is the error on the training data
  - Remaining terms are for regularization

# Relative Ratings for Item Recommendation (PrefNMF)

$$\pi(u,i,j) = \begin{cases} 0 & \text{if } u \text{ prefers } j \text{ over } i, \\ 0.5 & \text{if } i \text{ and } j \text{ are equally preferable to } u, \\ 1 & \text{if } u \text{ prefers } i \text{ over } j. \end{cases}$$

# Modeling users and items using MF

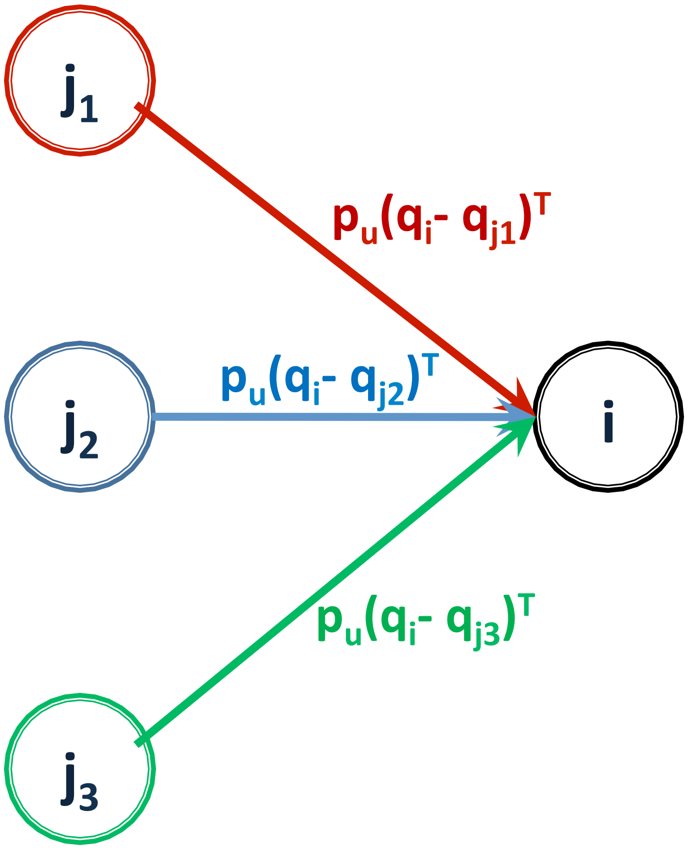
- **User representation**: $p_u$
- **Item representation**: $q_i$
- **Predicted preference relations:** Modeled using the inverse-logit function

$$\hat{\pi}(u,i,j) \stackrel{def}{=} \frac{e^{p_u(q_i-q_j)^T}}{1+e^{p_u(q_i-q_j)^T}}$$

- The features can be learned by optimizing

$$\min_{p,q} \sum_{\substack{\langle u,i,j,\pi(u,i,j)\rangle \\ \in S \\ \wedge (i<j)}} (\pi(u,i,j) - \hat{\pi}(u,i,j))^2 + \lambda_p \sum_{u \in U} ||p_u||^2 + \lambda_q \sum_{i \in I} ||q_i||^2$$

# Determining item scores




- Score of i:

  $x(u,i) = \sum p_u(q_i - q_j)^T$

  – O(nd) time to compute

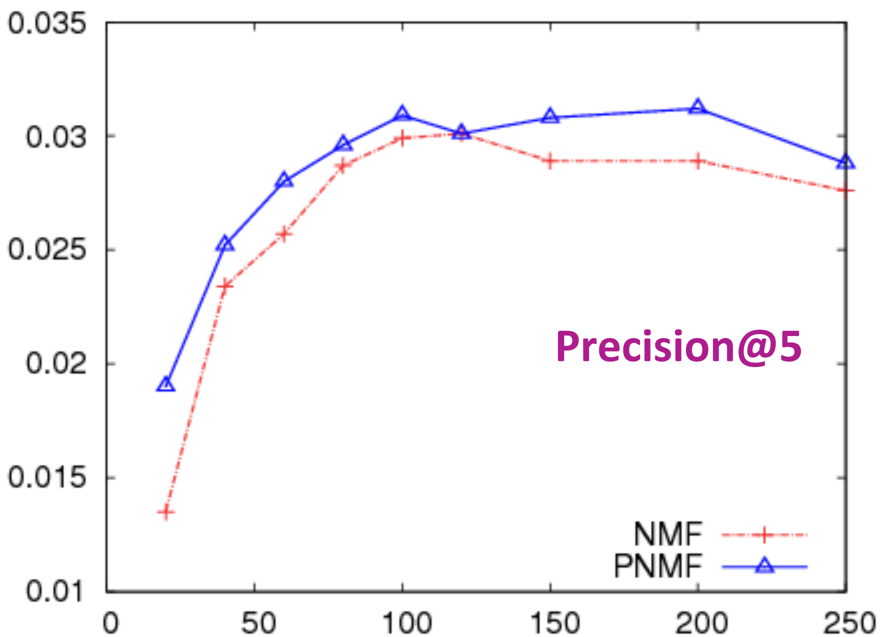
- Select Top-K items (according to scores) for recommendation

# Using PrefNMF for Item Recommendation

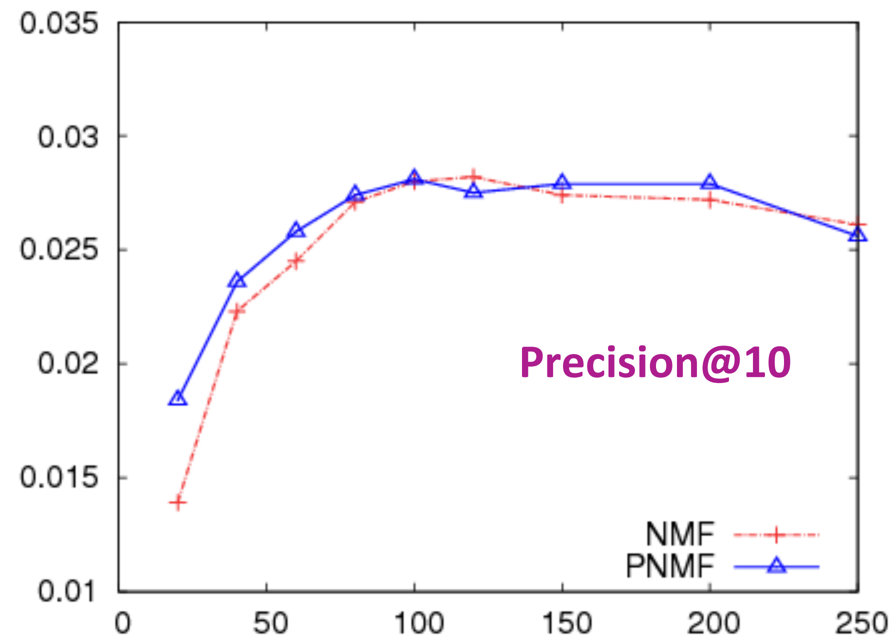
- PrefNMF gives better recommendation, specially for the dense users.

- First published algorithm that incorporates preference relations in the NMF framework for recommendation.

# Comparing NMF and PrefNMF: All users



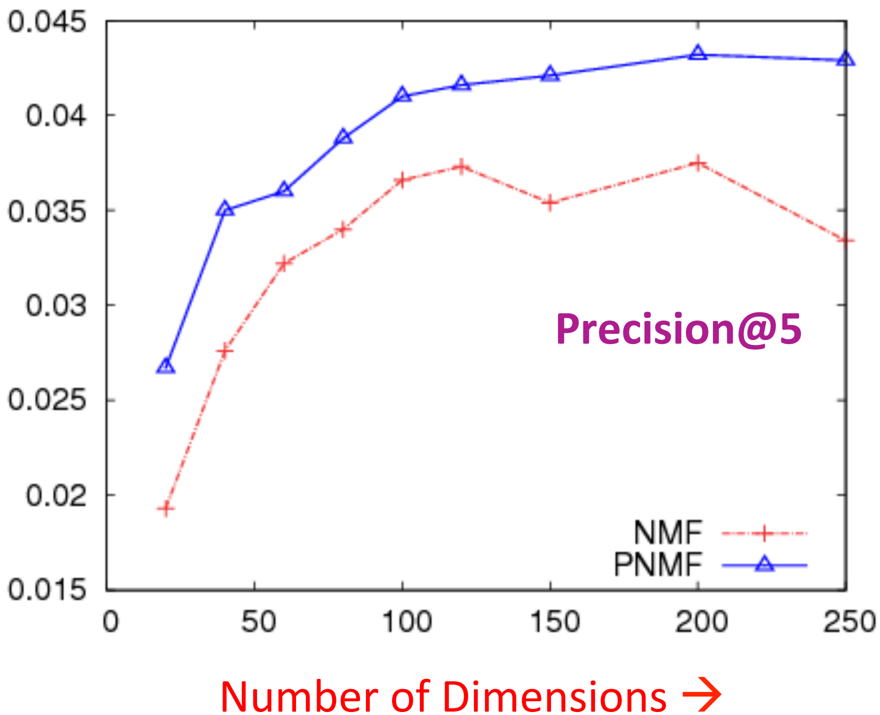

Number of Dimensions→

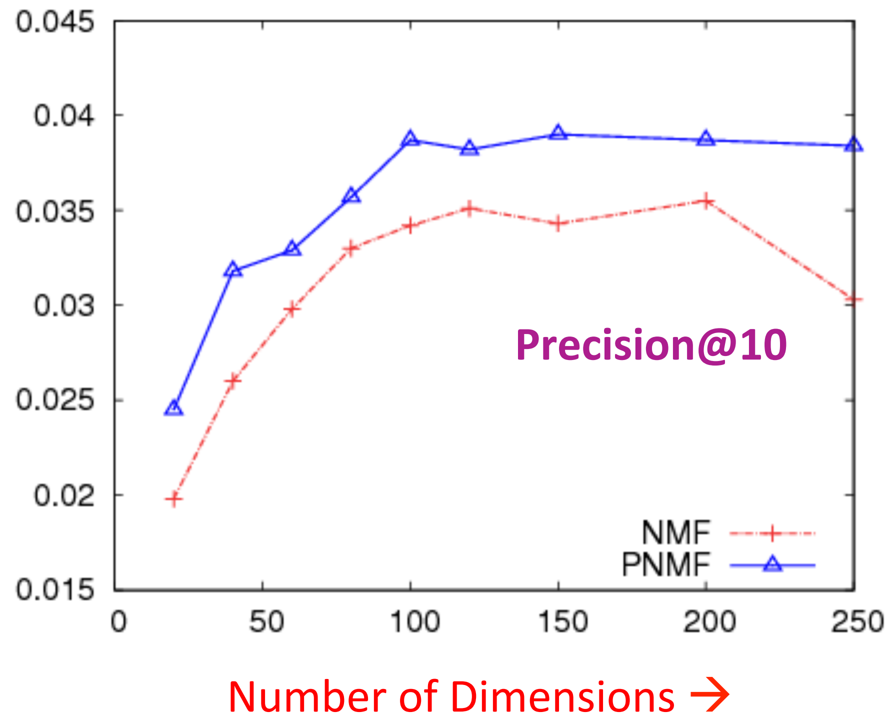



Number of Dimensions →

x-axis represents number of features. y-axis represents Precision@k.

# Comparing NMF and PrefNMF: *Dense* users




x-axis represents number of features. y-axis represents Precision@k.

# Rating Prediction Using Preference Relations Based Matrix Factorization

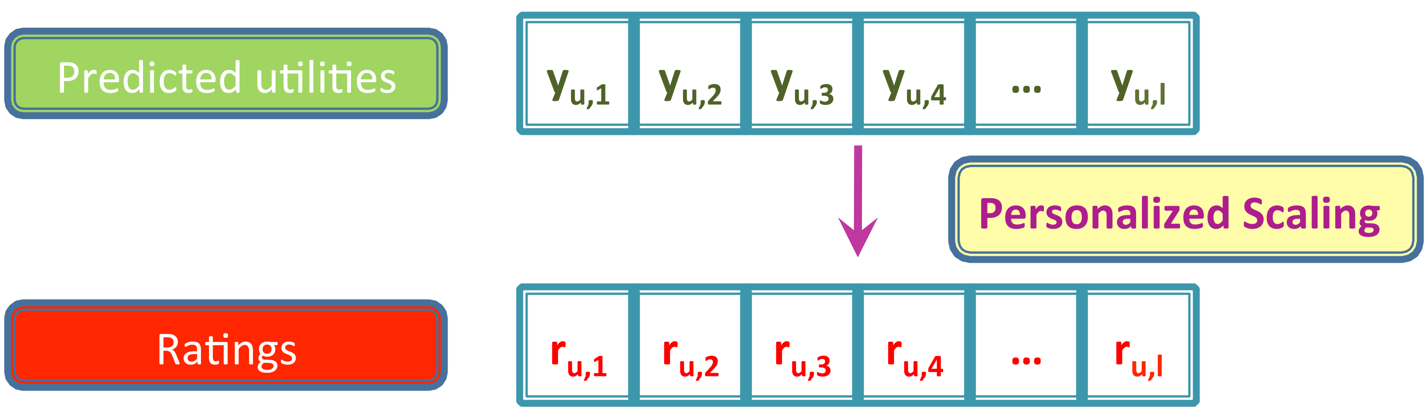
Presented in

FactMod Workshop in UMAP 2012

Montreal

# Rating Prediction using PrefNMF

- User and item representations are learned using the previous algorithm (PrefNMF)
- The score should be mapped to rating



# Personalized Scaling

- Suppose u has rated l different items: $I_u=\{i_1, i_2, i_3, \ldots, i_l\}$
- Corresponding ratings are: $R_u=\{r_{u1}, r_{u2}, \ldots, r_{ul}\}$
- Use this to learn a linear function:

$$r_{u,ik} = \alpha_u y(u,i_k) + \beta_u$$

- Can be achieved by solving the following optimization function

$$\min_{\alpha\beta} \ [\textstyle\sum_k (r_{u,ik} - \alpha_u y(u,i_k) - \beta_u)^2]$$

# Experimental Results

- Performed on two different samples (D1, D2) of Netflix data

| Statistics | D1 | D2 |
|---|---|---|
| #Ratings | 124,637 | 485,333 |
| #Users | 3229 | 22920 |
| #Items | 1255 | 1232 |
| Sparsity | 96.9% | 98.2% |
| Minimum #ratings by any user | 20 | 10 |
| Maximum #ratings by any user | 449 | 455 |
| Average #ratings for any user | 38 | 21 |
| Minimum #ratings for any item | 1 | 16 |
| Maximum #ratings for any item | 652 | 16, |
| Average #ratings for any item | 99 | 394 |

# Comparing Prediction Accuracies

**Results on D1**
**[Lower values are better]**

**Improvements**
**MAE:5.9%, RMSE: 3.2%**

| Algorithm | MAE | RMSE |
|---|---|---|
| PC-CF [1] | 1.0765 | 1.5543 |
| Som-CF [2] | 1.2068 | 1.6678 |
| Pref-CF [5] | 1.0579 | 1.4783 |
| Pref-GrAgg | 0.7650 | 1.0850 |
| NMF [4] | 0.8085 | 1.1278 |
| PrefNMF-RP | 0.7199 | 1.0505 |

**Results on D2**
**[Lower values are better]**

**Improvements**
**MAE: 6.1%, RMSE: 4.4%**

| Algorithm | MAE | RMSE |
|---|---|---|
| PC-CF [1] | 0.9602 | 1.4001 |
| Som-CF [2] | 1.0898 | 1.5300 |
| Pref-CF [5] | 0.9759 | 1.3665 |
| Pref-GrAgg | 0.7623 | 1.0738 |
| NMF [4] | 0.8525 | 1.1832 |
| PrefNMF-RP | 0.7153 | 1.0267 |

# Summary

- Use of preference relations as feedback eliminates some of the drawbacks of absolute ratings.
- Explained preference relations based algorithms in both the collaborative Filtering and NMF framework.
- The described methods work better than methods from literature on benchmark datasets.
- Need to understand the issues that may exist in a real system that supports preference relations based feedbacks.

# References


- [1] D. Heckerman, J. Breese, and C. M. Kadie. "**Empirical analysis of predictive algorithms for collaborative filtering**". **UAI 1998**.
- [2] N. Lathia, S. Hailes, and L. Capra. "**Private distributed collaborative filtering using estimated concordance measures**". In **RecSys 2007**, pages 1–8.
- [3] H. Yildirim and M. S. Krishnamoorthy. "**A random walk method for alleviating the sparsity problem in collaborative filtering**". In **RecSys 2008**, pages 131–138.
- [4] Y. Koren, R. Bell, and C. Volinsky. "**Matrix factorization techniques for recommender systems**". **IEEE Computer, 42:**30–37**, August 2009**.
- [5] A. Brun, A. Hamad, O. Buffet, and A. Boyer, "**Towards preference relations in recommender systems**," in **Preference Learning (PL-10) ECML/PKDD-10 Workshop, 2010**.
- [6] N. Kawamae, H. Sakano, and T. Yamada. "**Personalized recommendation based on the personal innovator degree**". In **RecSys 2009**, pages 329–332, ACM.
- [7] N. Zheng and Q. Li. "**A recommender system based on tag and time information for social tagging systems**". In **Expert Syst. Appl.**, **38:**4575–4587**, 2011**.

Thank you 

Questions??