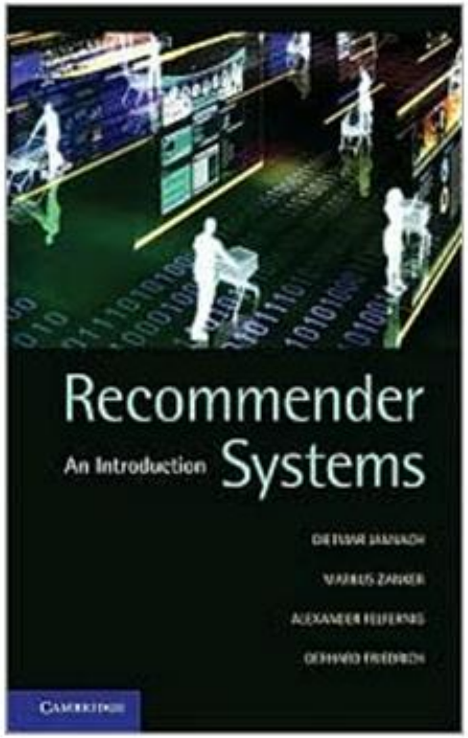
Inspire…Educate…Transform.

# Collaborative Filtering

**Dr. Manoj Chinnakotla**
Senior Applied Scientist, Microsoft
Adjunct Professor, IIIT Hyderabad




## Recommender Systems: An Introduction

by Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich

**AVERAGE CUSTOMER RATING:**

( Be the first to review )

Gefällt mir  Registrieren, um sehen zu können, was deinen Freunden gefällt.

**FORMAT:**
Hardcover

NOOKbook (eBook) - not available

Tell the publisher you want this in NOOKbook format

**NEW FROM BN.COM**

~~$65.00~~ List Price

**$52.00 Online Price**

**(You Save 20%)**

**Add to Cart**

**NEW & USED FROM OUR**

**New starting at $56.46**(You S

**Used starting at $51.98**(You S

**See All Prices**

Table of Contents

### Customers who bought this also bought





CSE 7306c

# Recommender Systems

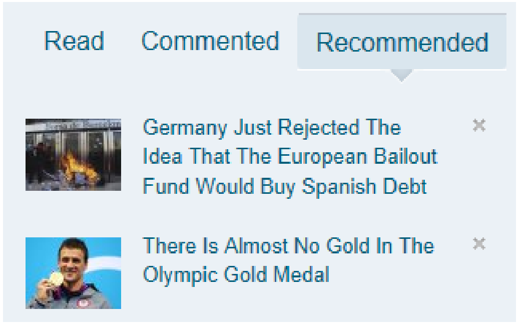
- Application areas


ALTERNATIVE PRODUCTS

Beko Washing Machine
Code: WMB81431LW
**£269.99**

Zanussi Washing Machine
Code: ZWH6130P
**£269.99**

Blomberg Washing Machine
Code: WNF6221
**£299.99**



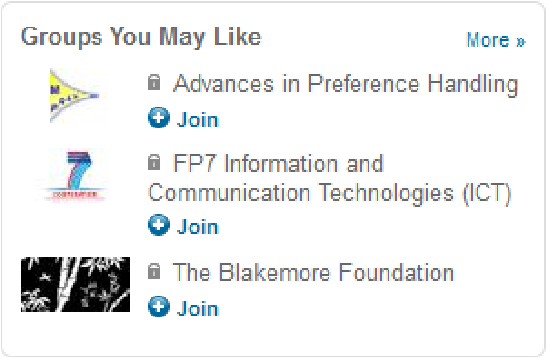
MOST POPULAR | RECOMMENDED

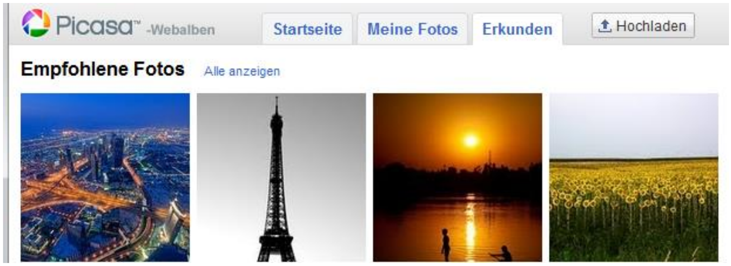
How to Break NRA's Grip on Politics: Michael R. Bloomberg

Growth in U.S. Slows as Consumers Restrain Spending

# In the Social Web



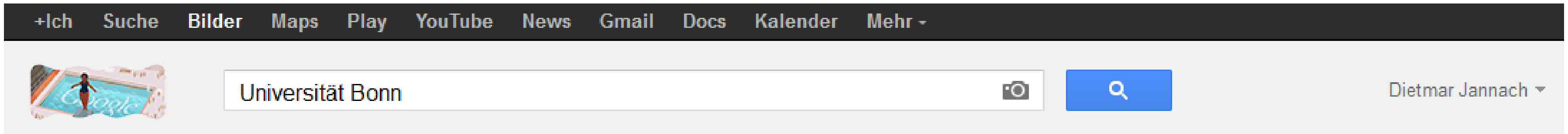



CSE 7306c

# Even more …

- Personalized search




- "Computational Advertising"


# Why use Recommender Systems?

- Value for the customer
  - Find things that are interesting
  - Narrow down the set of choices
  - Help me explore the space of options
  - Discover new things
  - Entertainment
  - ….
- Value for the provider
  - Additional and probably unique personalized service for the customer
  - Increase trust and customer loyalty
  - Increase sales, click trough rates, conversion etc.
  - Opportunities for promotion, persuasion
  - Obtain more knowledge about customers

# Real-World Check

- Myths from industry
  - Amazon.com generates X percent of their sales through the recommendation lists ($30 < X < 70$)
  - Netflix (DVD rental and movie streaming) generates X percent of their sales through the recommendation lists ($30 < X < 70$)
- There must be some value in it
  - See recommendation of groups, jobs or people on LinkedIn
  - Friend recommendation and ad personalization on Facebook
  - Song recommendation at last.fm
  - News recommendation at Forbes.com (plus 37% CTR)
- Academia
  - A few studies exist that show the effect
    - increased sales, changes in sales behavior

# Problem Domain

- Recommendation systems (RS) help to match users with items
  - Ease information overload
  - Sales assistance (guidance, advisory, persuasion,…)

*RS are software agents that elicit the interests and preferences of individual consumers […] and make recommendations accordingly.*

*They have the potential to support and improve the quality of the decisions consumers make while searching for and selecting products online.*


- Different system designs / paradigms
  - Based on availability of exploitable data
  - Implicit and explicit user feedback
  - Domain characteristics

# Recommender Systems

- RS seen as a function
- **Given:**
  - User model (e.g. ratings, preferences, demographics, situational context)
  - Items (with or without description of item characteristics)
- **Find:**
  - Relevance score. Used for ranking.
- **Finally:**
  - Recommend items that are assumed to be relevant
- **But:**
  - Remember that relevance might be context-dependent
  - Characteristics of the list itself might be important (diversity)

# Collaborative Filtering (CF)

- The most prominent approach to generate recommendations
  - used by large, commercial e-commerce sites
  - well-understood, various algorithms and variations exist
  - applicable in many domains (book, movies, DVDs, ..)

- Approach
  - use the "wisdom of the crowd" to recommend items
- Basic assumption and idea
  - Users give ratings to catalog items (implicitly or explicitly)
  - Customers who had similar tastes in the past, will have similar tastes in the future

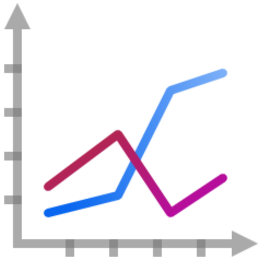
# User-based kNN CF (1)

- The basic technique:
  - Given an "active user" (Alice) and an item I not yet seen by Alice
  - The *goal is to estimate Alice's rating for this item*, e.g., by
    - find a set of users (peers) who liked the same items as Alice in the past **and** who have rated item I
    - use, e.g. the average of their ratings to predict, if Alice will like item I
    - do this for all items Alice has not seen and recommend the best-rated

| | Item1 | Item2 | Item3 | Item4 | Item5 |
|---|---|---|---|---|---|
| Alice | 5 | 3 | 4 | 4 | ? |
| User1 | 3 | 1 | 2 | 3 | 3 |
| User2 | 4 | 3 | 4 | 3 | 5 |
| User3 | 3 | 3 | 1 | 5 | 4 |
| User4 | 1 | 5 | 5 | 2 | 1 |

# User-based kNN CF (2)

- Some first questions
  - How do we measure similarity?
  - How many neighbors should we consider?
  - How do we generate a prediction from the neighbors' ratings?



| | Item1 | Item2 | Item3 | Item4 | Item5 |
|---|---|---|---|---|---|
| Alice | 5 | 3 | 4 | 4 | ? |
| User1 | 3 | 1 | 2 | 3 | 3 |
| User2 | 4 | 3 | 4 | 3 | 5 |
| User3 | 3 | 3 | 1 | 5 | 4 |
| User4 | 1 | 5 | 5 | 2 | 1 |

# Measuring user similarity

- A popular similarity measure in user-based CF: **Pearson correlation**

  a, b : users

  $r_{a,p}$ : rating of user a for item p

  P : set of items, rated both by a and b

  Possible similarity values between -1 and 1;

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r}_a)^2} \sqrt{\sum_{p \in P}(r_{b,p} - \bar{r}_b)^2}}$$

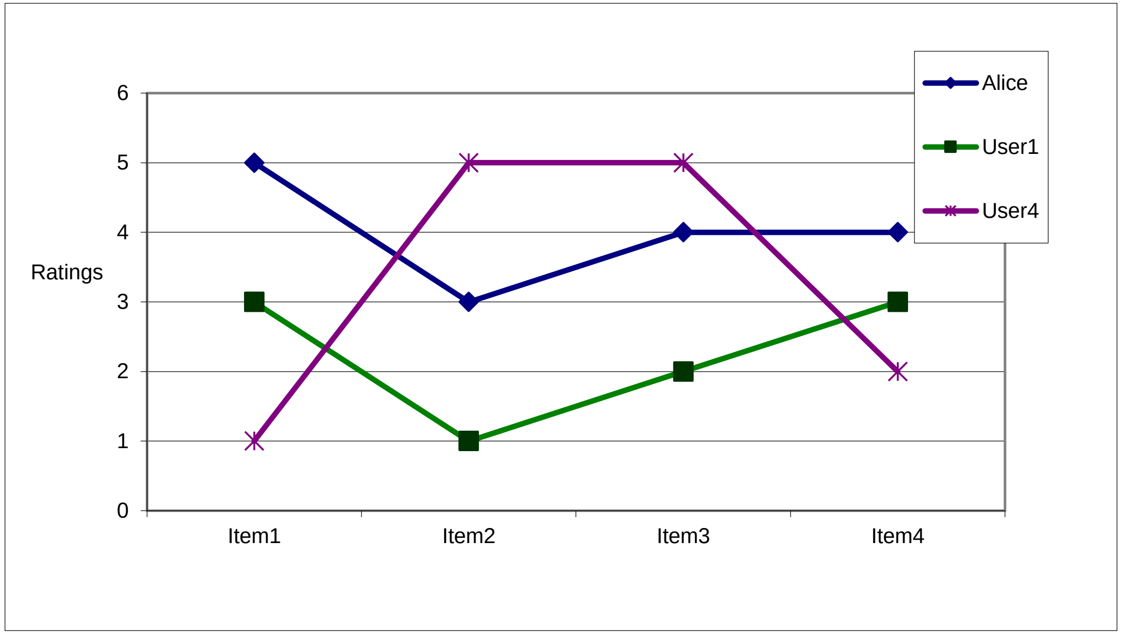
$\overline{r_a}, \overline{r_b}$ = user's average ratings

| | Item1 | Item2 | Item3 | Item4 | Item5 |
|---|---|---|---|---|---|
| Alice | 5 | 3 | 4 | 4 | ? |
| User1 | 3 | 1 | 2 | 3 | 3 |
| User2 | 4 | 3 | 4 | 3 | 5 |
| User3 | 3 | 3 | 1 | 5 | 4 |
| User4 | 1 | 5 | 5 | 2 | 1 |

sim = 0.85
sim = 0.70
sim = -0.79

# Pearson Correlation

- Takes differences in rating behavior into account




- Works well in usual domains, compared with alternative measures
  - such as cosine similarity

# Making Predictions

- A common prediction function:

$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$

- Calculate, whether the neighbors' ratings for the unseen item *i* are higher or lower than their average
- Combine the rating differences – use the similarity as a weight
- Add/subtract the neighbors' bias from the active user's average and use this as a prediction

# Making Recommendations

- Making predictions is typically not the ultimate goal
- Usual approach (in academia)
  - Rank items based on their predicted ratings
- However
  - This might lead to the inclusion of (only) niche items
  - **In practice also:** Take item popularity into account
- Approaches
  - "Learning to rank"
    - Optimize according to a given rank evaluation metric

# Improving the Prediction Function

- Not all neighbor ratings might be equally "valuable"
  - Agreement on commonly liked items is not so informative as agreement on controversial items
  - **Possible solution**: Give more weight to items that have a higher variance
- Value of number of co-rated items
  - Use "significance weighting", by e.g., linearly reducing the weight when the number of co-rated items is low
- Case amplification
  - Intuition: Give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1.
- Neighborhood selection
  - Use similarity threshold or fixed number of neighbors

# Memory-based and Model-based Approaches

- User-based CF is said to be "memory-based"
  - the rating matrix is directly used to find neighbors / make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have tens of millions of customers and millions of items
- Model-based approaches
  - based on an offline pre-processing or "model-learning" phase
  - at run-time, only the learned model is used to make predictions
  - models are updated / re-trained periodically
  - large variety of techniques used
  - model-building and updating can be computationally expensive

# Item-based CF

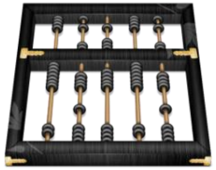
- Basic idea:
  - Use the similarity between items (and not users) to make predictions
- Example:
  - Look for items that are similar to Item5
  - Take Alice's ratings for these items to predict the rating for Item5

| | Item1 | Item2 | Item3 | Item4 | Item5 |
|---|---|---|---|---|---|
| Alice | 5 | 3 | 4 | 4 | ? |
| User1 | 3 | 1 | 2 | 3 | 3 |
| User2 | 4 | 3 | 4 | 3 | 5 |
| User3 | 3 | 3 | 1 | 5 | 4 |
| User4 | 1 | 5 | 5 | 2 | 1 |

# The cosine similarity measure

- Produces better results in item-to-item filtering
  - for some datasets, no consistent picture in literature
- Ratings are seen as vector in n-dimensional space
- Similarity is calculated based on the angle between the vectors

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$



- Adjusted cosine similarity
  - take average user ratings into account, transform the original ratings
  - U: set of users who have rated both items a and b

$$sim(a, b) = \frac{\sum_{u \in U}(r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U}(r_{u,a} - \overline{r_u})^2}\sqrt{\sum_{u \in U}(r_{u,b} - \overline{r_u})^2}}$$

# Pre-processing for Item-based CF

- Item-based filtering does not solve the scalability problem itself
- Pre-processing approach by Amazon.com (in 2003)
  - Calculate all pair-wise item similarities in advance
  - The neighborhood to be used at run-time is typically rather small, because only items are taken into account which the user has rated
  - Item similarities are supposed to be more stable than user similarities
- Memory requirements
  - Up to $N^2$ pair-wise similarities to be memorized (N = number of items) in theory
  - In practice, this is significantly lower (items with no co-ratings)
  - Further reductions possible
    - Minimum threshold for co-ratings (items, which are rated at least by *n* users)
    - Limit the size of the neighborhood (might affect recommendation accuracy)

# More on Ratings…

- Pure CF-based systems only rely on the rating matrix
- Explicit ratings
  - Most commonly used (1 to 5, 1 to 7 Likert response scales)
  - Research topics
    - "Optimal" granularity of scale; indication that 10-point scale is better accepted in movie domain
    - Multidimensional ratings (multiple ratings per movie)
  - Challenge
    - Users not always willing to rate many items; sparse rating matrices
    - How to stimulate users to rate more items?
- Implicit ratings
  - Clicks, page views, time spent on some page, demo downloads …
  - Can be used in addition to explicit ones; question of correctness of interpretation

# Data Sparsity Problems

- Cold start problem
  - How to recommend new items? What to recommend to new users?
- Straightforward approaches
  - Ask/force users to rate a set of items
  - Use another method (e.g., content-based, demographic or simply non-personalized) in the initial phase
- Alternatives
  - Use better algorithms (beyond nearest-neighbor approaches)
  - Example:
    - In nearest-neighbor approaches, the set of sufficiently similar neighbors might be to small to make good predictions
    - Assume "transitivity" of neighborhoods

# Example Algorithms for Sparse Datasets

- Recursive CF
  - Assume there is a very close neighbor ***n*** of u who however has not rated the target item ***i*** yet.
  - Idea:
    - Apply CF-method recursively and predict a rating for item ***i*** for the neighbor
    - Use this predicted rating instead of the rating of a more distant direct neighbor

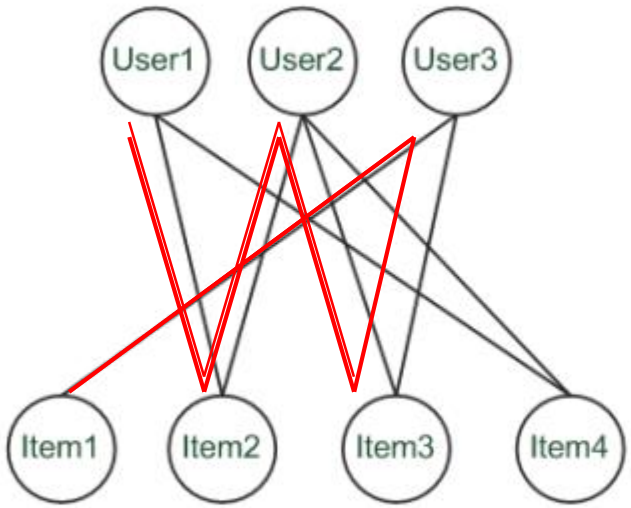
| | Item1 | Item2 | Item3 | Item4 | Item5 |
|---|---|---|---|---|---|
| Alice | 5 | 3 | 4 | 4 | ? |
| User1 | 3 | 1 | 2 | 3 | ? |
| User2 | 4 | 3 | 4 | 3 | 5 |
| User3 | 3 | 3 | 1 | 5 | 4 |
| User4 | 1 | 5 | 5 | 2 | 1 |

sim = 0,85

Predict rating for User1

# Graph-based Methods

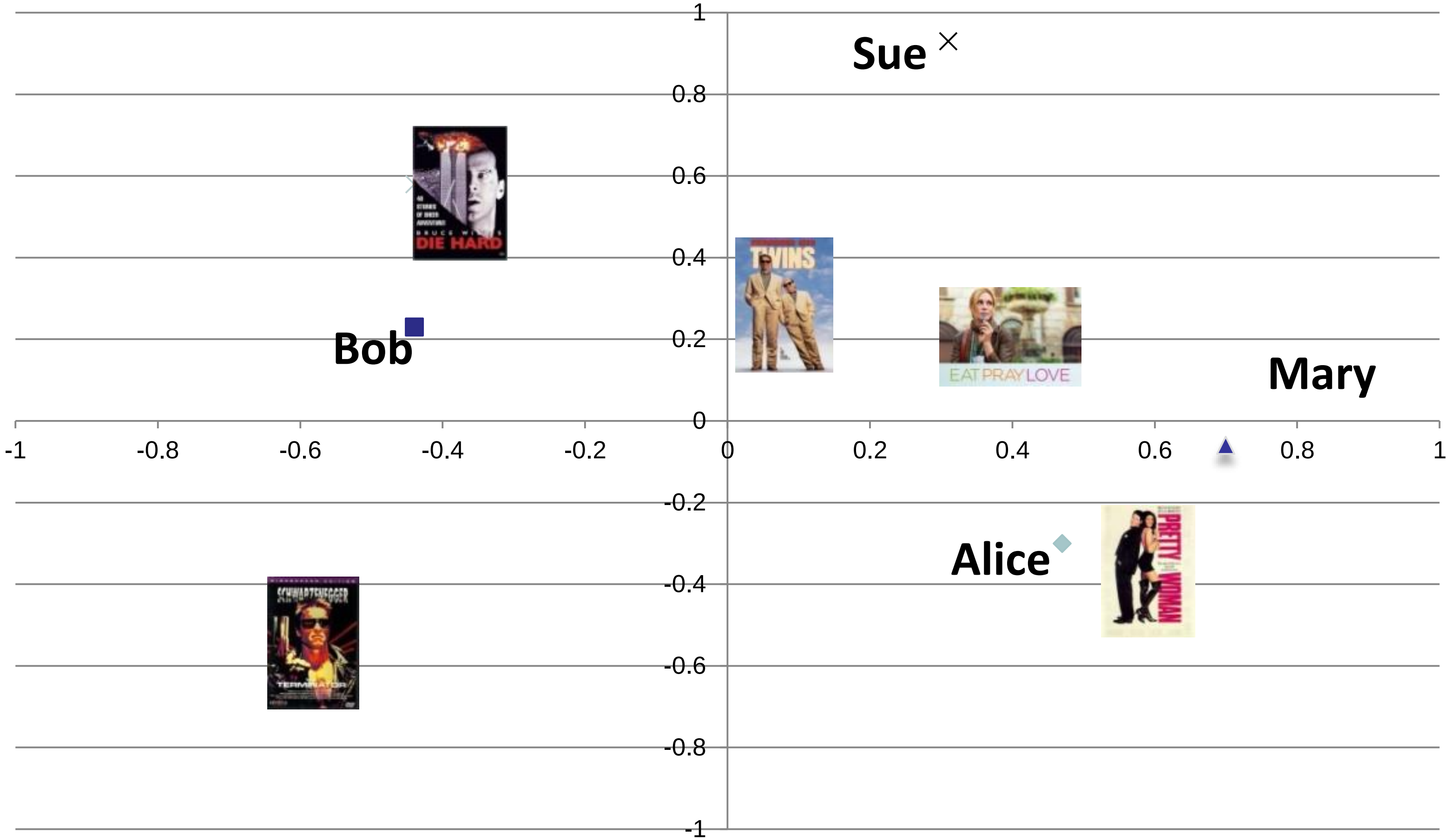
- "Spreading Activation" (sketch)
  - Idea: Use paths of lengths > 3 to recommend items
  - Length 3: Recommend Item3 to User1
  - Length 5: Item1 also recommendable



# More Model-based Approaches

- Plethora of different techniques proposed in the last years, e.g.,
  - Matrix factorization techniques, statistics
    - singular value decomposition, principal component analysis
  - Association rule mining
    - compare: shopping basket analysis
  - Probabilistic models
    - clustering models, Bayesian networks, probabilistic Latent Semantic Analysis
  - Various other machine learning approaches
- Costs of pre-processing
  - Usually not discussed
  - Incremental updates possible?

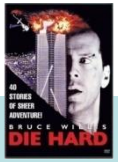
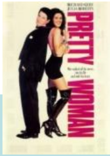
# A Picture Says …



# Matrix Factorization

- SVD: $$M_k = U_k \times \Sigma_k \times V_k^{\,T}$$

| $U_k$ | Dim1 | Dim2 |
|---|---|---|
| **Alice** | 0.47 | -0.30 |
| **Bob** | -0.44 | 0.23 |
| **Mary** | 0.70 | -0.06 |
| **Sue** | 0.31 | 0.93 |

| $V_k^T$ |  |  |  |  |  |
|---|---|---|---|---|---|
| **Dim1** | -0.44 | -0.57 | 0.06 | 0.38 | 0.57 |
| **Dim2** | 0.58 | -0.66 | 0.26 | 0.18 | -0.36 |

| $\Sigma_k$ | Dim1 | Dim2 |
|---|---|---|
| **Dim1** | 5.63 | 0 |
| **Dim2** | 0 | 3.23 |

- Prediction: $\hat{r}_{ui} = \bar{r}_u + U_k(Alice) \times \Sigma_k \times V_k^T(EPL)$
  = 3 + 0.84 = 3.84

# Association Rule Mining

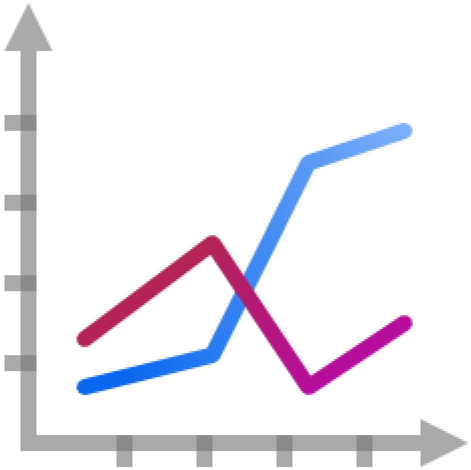
- Commonly used for shopping behavior analysis
  - aims at detection of rules such as
    *"If a customer purchases baby-food then he also buys diapers in 70% of the cases“*

- Association rule mining algorithms
  - can detect rules of the form X => Y (e.g., baby-food => diapers) from a set of sales transactions D = {$t_1$, $t_2$, … $t_n$}
  - measure of quality: support, confidence

# Probabilistic Methods

- Basic idea (simplistic version for illustration):
  - given the user/item rating matrix
  - determine the probability that user Alice will like an item i
  - base the recommendation on such these probabilities
- Calculation of rating probabilities based on Bayes Theorem
  - How probable is rating value "1" for Item5 given Alice's previous ratings?
  - Corresponds to conditional probability P(Item5=1 | X), where
    - X = Alice's previous ratings = (Item1 =1, Item2=3, Item3= … )
  - Can be estimated based on Bayes' Theorem
- Usually more sophisticated methods used
  - Clustering
  - pLSA …

# Evaluation of Recommender Systems



# What is a good Recommendation?

What are the measures in practice?

- Total sales numbers
- Promotion of certain items
- …
- Click-through-rates
- Interactivity on platform
- …
- Customer return rates
- Customer satisfaction and loyalty

# Purpose and Success Criteria (1)

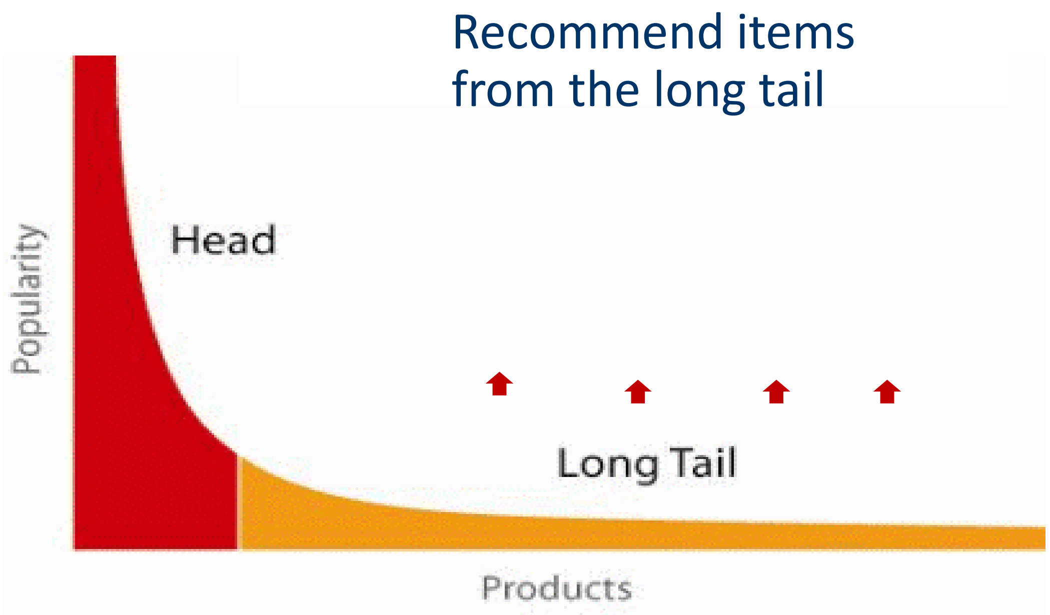
Different perspectives/aspects

- Depends on domain and purpose
- No holistic evaluation scenario exists

- Retrieval perspective
  - Reduce search costs
  - Provide "correct" proposals
  - Assumption: Users know in advance what they want

- Recommendation perspective
  - Serendipity – identify items from the Long Tail
  - Users did not know about existence

# When does a RS do its job well?




- "Recommend widely unknown items that users might actually like!"
- 20% of items accumulate 74% of all positive ratings

CSE 7306c

# Purpose and Success Criteria (2)

- Prediction perspective
  - Predict to what degree users like an item
  - Most popular evaluation scenario in research

- Interaction perspective
  - Give users a "good feeling"
  - Educate users about the product domain
  - Convince/persuade users - explain

- Finally, conversion perspective
  - Commercial situations
  - Increase "hit", "clickthrough", "lookers to bookers" rates
  - Optimize sales margins and profit

# Evaluation in Information Retrieval (IR)

- Recommendation is viewed as information retrieval task:
  - Retrieve (recommend) all items which are predicted to be "good" or "relevant".
- Common protocol :
  - Hide some items with known ground truth
  - Rank items or predict ratings -> Count -> Cross-validate
- Ground truth established by human domain experts

| | | Reality | |
|---|---|---|---|
| | | Actually Good | Actually Bad |
| Prediction | Rated Good | True Positive (tp) | False Positive (fp) |
| | Rated Bad | False Negative (fn) | True Negative (tn) |

# Metrics: Precision and Recall

- **Precision:** a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved
  - E.g. the proportion of recommended movies that are actually good

$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|\text{all recommendations}|}$$

- **Recall:** a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items
  - E.g. the proportion of all good movies recommended

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

# Dilemma of IR measures in RS

- **IR measures** are frequently applied, however:
  - Ground truth for most items actually unknown
  - What is a relevant item?
  - Different ways of measuring precision possible
- Results from offline experimentation may have limited predictive power for
  online user behavior.

# Metrics: Rank Score – Position Matters

For a user:

| Actually good |
|---|
| Item 237 |
| Item 899 |

hit

| Recommended (predicted as good) |
|---|
| Item 345 |
| Item 237 |
| Item 187 |

- **Rank Score** extends recall and precision to take the positions of correct items in a ranked list into account
  - Particularly important in recommender systems as lower ranked items may be overlooked by users
  - Learning-to-rank: Optimize models for such measures (e.g., AUC)

# Accuracy Measures

- Datasets with items rated by users
  - MovieLens datasets 100K-10M ratings
  - Netflix 100M ratings
- Historic user ratings constitute ground truth
- Metrics measure error rate
  - Mean Absolute Error (*MAE*) computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i|$$

  - Root Mean Square Error (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2}$$

# Offline Experimentation Example

- Netflix competition
  - Web-based movie rental
  - Prize of $1,000,000 for accuracy improvement (RMSE) of 10% compared to own Cinematch system.
- Historical dataset
  - ~480K users rated ~18K movies on a scale of 1 to 5 (~100M ratings)
  - Last 9 ratings/user withheld
    - Probe set – for teams for evaluation
    - Quiz set – evaluates teams' submissions for leaderboard
    - Test set – used by Netflix to determine winner
- Today
  - Rating  prediction only seen as an additional input into the recommendation process

# An Imperfect World…

- Offline evaluation is the cheapest variant
  - Still, gives us valuable insights
  - and lets us compare our results (in theory)
- Dangers and trends:
  - Domination of accuracy measures
  - Focus on small set of domains (40% on movies in CS)
- Alternative and complementary measures:
  - Diversity, Coverage, Novelty, Familiarity, Serendipity, Popularity, Concentration effects (Long tail)

# Online Experimentation Example

- Effectiveness of different algorithms for recommending cell phone games [Jannach, Hegelich 09]
- Involved 150,000 users on a commercial mobile internet portal
- Comparison of  recommender methods

# CF – Pros and Cons

- Pros:
  - well-understood, works well in some domains, no knowledge engineering required
- Cons:
  - requires user community, sparsity problems, no integration of other knowledge sources, no explanation of results
- What is the best CF method?
  - In which situation and which domain? Inconsistent findings; always the same domains and data sets; differences between methods are often very small (1/100)
- How to evaluate the prediction quality?
  - MAE / RMSE: What does an MAE of 0.7 actually mean?
  - Serendipity: Not yet fully understood
- What about multi-dimensional ratings?

# Summary

- CF is one of a handful of learning-related tools that have had broadly *visible* impact:
  - Google, TIVO, Amazon, personal radio stations, ...
- Critical tool for finding “consensus information” present in a large community (or large corpus of web pages, or large DB of purchase records, ....)
  - Similar in some respects to Q/A with corpora
- Science is relatively-well established
  - in certain narrow directions, on a few datasets
- Set of applications still being expanded

- Some resources:
  - http://www.sims.berkeley.edu/resources/collab/
  - http://www.cs.umn.edu/Research/GroupLens/
  - http://www.cis.upenn.edu/~ungar/CF/

# References


- [Adomavicius and Tuzhilin 2005] Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (2005), no. 6, 734–749
- [Breese et al. 1998] Empirical analysis of predictive algorithms for collaborative filtering, Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (Madison, WI) (Gregory F. Cooper and Seraf ´in Moral, eds.), Morgan Kaufmann, 1998, pp. 43–52
- [Gedikli et al. 2011] RF-Rec: Fast and accurate computation of recommendations based on rating frequencies, Proceedings of the 13th IEEE Conference on Commerce and Enterprise Computing - CEC 2011, Luxembourg, 2011, forthcoming
- [Goldberg et al. 2001] Eigentaste: A constant time collaborative filtering algorithm, Information Retrieval 4 (2001), no. 2, 133–151
- [Golub and Kahan 1965] Calculating the singular values and pseudo-inverse of a matrix, Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis 2 (1965), no. 2, 205–224
- [Herlocker et al. 2002] An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms, Information Retrieval 5 (2002), no. 4, 287–310
- [Herlocker et al. 2004] Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems (TOIS) 22 (2004), no. 1, 5–53

# References


- [Hofmann 2004] Latent semantic models for collaborative filtering, ACM Transactions on Information Systems 22 (2004), no. 1, 89–115
- [Huang et al. 2004] Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering, ACM Transactions on Information Systems 22 (2004), no. 1, 116–142
- [Koren et al. 2009] *Matrix factorization techniques for recommender systems*, Computer 42 (2009), no. 8, 30–37
- [Lemire and Maclachlan 2005] Slope one predictors for online rating-based collaborative filtering, Proceedings of the 5th SIAM International Conference on Data Mining (SDM ’05) (Newport Beach, CA), 2005, pp. 471–480
- [Sarwar et al. 2000a] Application of dimensionality reduction in recommender systems – a case study, Proceedings of the ACM WebKDD Workshop (Boston), 2000
- [Zhang and Pu 2007] A recursive prediction algorithm for collaborative filtering recommender systems, Proceedings of the 2007 ACM Conference on Recommender Systems (RecSys ’07) (Minneapolis, MN), ACM, 2007, pp. 57–64

## International School of Engineering

Plot 63/A, 1st Floor, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

| | |
|---|---|
| For Individuals: | +91-9502334561/63 or 040-65743991 |
| For Corporates: | +91-9618483483 |
| Web: | http://www.insofe.edu.in |
| Facebook: | https://www.facebook.com/insofe |
| Twitter: | https://twitter.com/Insofeedu |
| YouTube: | http://www.youtube.com/InsofeVideos |
| SlideShare: | http://www.slideshare.net/INSOFE |
| LinkedIn: | http://www.linkedin.com/company/international-school-of-engineering |