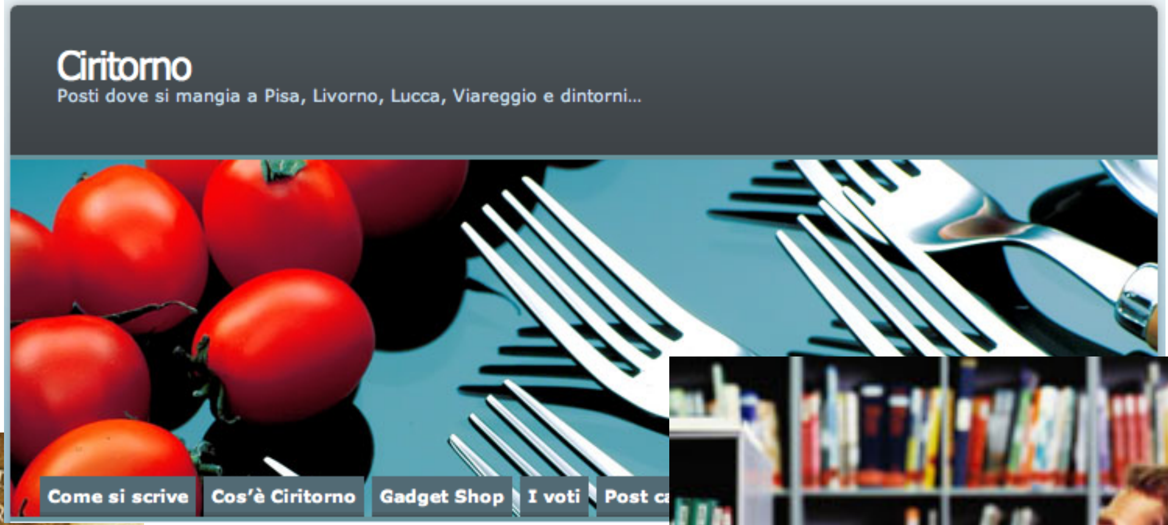
E-COMMERCE 3: RECOMMENDER SYSTEMS

Claudio Silvestri

Goal

- To recommend interesting stuff:
  - Songs, movies, web-pages, (queries…), …
- Popular items vs. the Long Tail
  - There is a lot of money in the tail
  - Recommending popular stuff may not be sufficient
  - Really understand what a user is looking for
- Exploit every fragment of knowledge available
  - Wisdom of the crowds

Some general approaches (different knowledge models/bases)



Quality Measures

- Efficiency in building the model
- Efficiency in generating suggestions
- Serendipity of recommendations
- Cold-start problem

Content-based

- A user is represented by the set of items s/he purchased
  - Define a description of each item
  - Sum/Avg those descriptions to model the user
- E-Commerce:
  - Title, brand, description, price, category, on-sale
- Web Pages:
  - Every single word ! (at least)

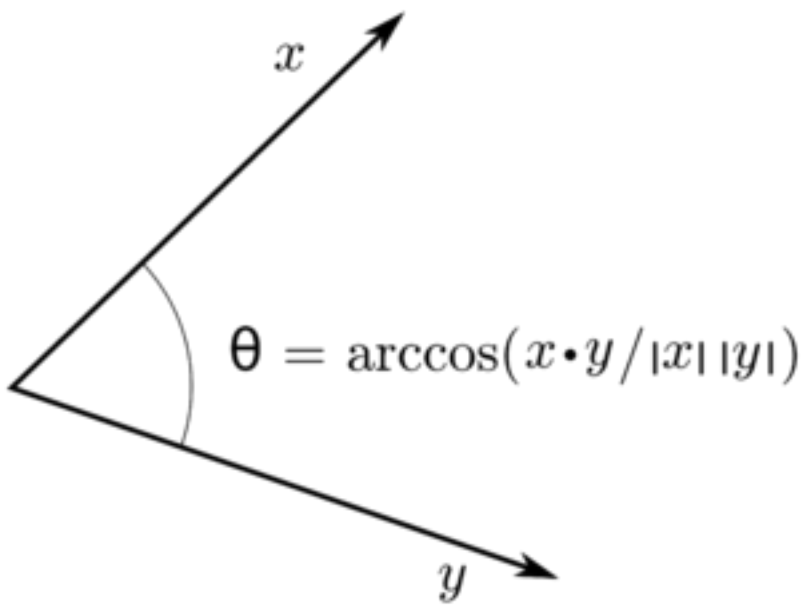
Modeling free text

- A document is a set of words
- In the vector-space model:
  - Each document is a vector *x* of size *N*, where N is the number of words in the lexicon
  - *x[i]=m* if the *i*-th word occurs *m* times in *x*
  - Given two documents *x* and *y*, their similarity is measured with the cosine of the two vectors:

$$cos(x, y) = \frac{x \cdot y}{|x||y|}$$

Why the cosine distance ?

- Cosine is high if the angle is small
- Angle is not biased by the document length
  - x="apple", y="apple apple apple apple" have cosine equal to 1 (Θ=0)



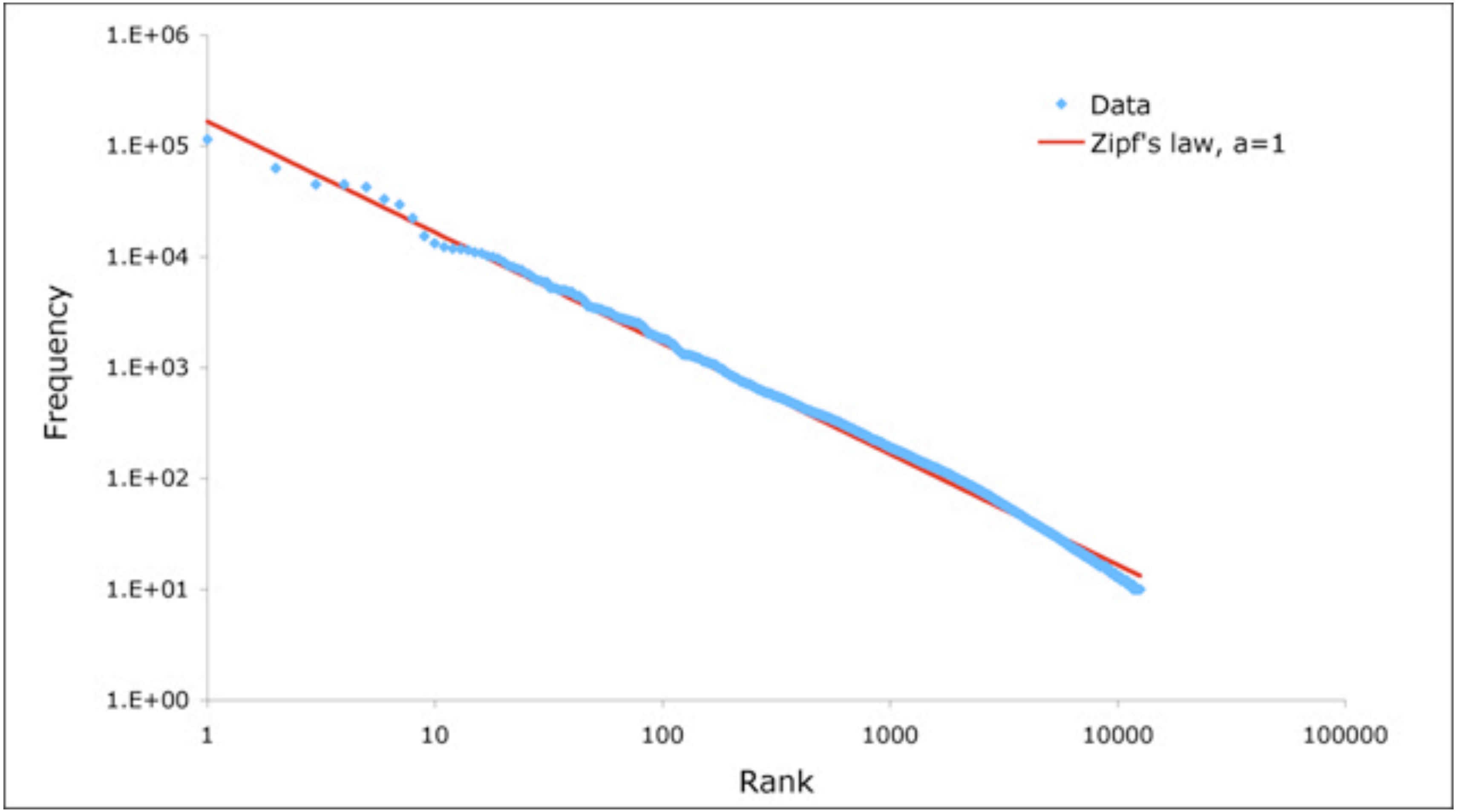
Is frequency sufficient ?

- x = [la:100, divina:1, commedia:1]
- y = [nella:100, divina:2, commedia:2]

- w = [che:1, bel:2, tempo:3, oggi:4]
- z = [the:1, Riemann: 1, zeta:1, function:1]

- Different words have different specificity, they must have different weight

Frequency distribution



TF x IDF

- Context is the corpus
- Promote rare words, demote common words

$$x[i] = \mathsf{tf}_{i,x} \cdot \log\left(\frac{N}{\mathsf{df}_i}\right)$$

where

- *N* is the total number of documents
- df: documents containing term i
- Tf: i term freq in document x

Content-based recommendation

- User profile U, given the set V of visited documents:

$$U = \frac{1}{|V|} \sum_{x \in V} x$$

- Recommendation:
  - The nearest document y in the corpus

Content-based – Wrap up

- Efficiency in building the model
  - No model for the corpus
  - Cheap model for the user
- Efficiency in generating suggestions
  - K-NN search among the collection of documents
- Serendipity of recommendations
  - small
- Cold-start problem
  - Partial
- Ageing effect:
  - Which documents to use when building the model ?

Collaborative Filtering

- In many cases, users rate items
  - Explicit: stars
  - Implicit: time on a web page, clicks on a result page

- **Rather then finding similar items, find similar users!**
  - Greater serendipity !

- A user is modeled by a vector U:
  - $U[i] = r$ if the user U gave r stars to the item i.

Similarity between users

- What do we need to measure ?
  - Do they have the same votes ??
  - Euclidean ??
  - Cosine ??

- Pearson-correlation:

$$\rho(U,V) = \frac{\mathrm{cov}(U,V)}{\sigma_U \sigma_V} = \frac{\sum_i (U[i] - \overline{U})(V[i] - \overline{V})}{\sqrt{\sum_i (U[i] - \overline{U})^2}\sqrt{\sum_i (V[i] - \overline{V})^2}}$$

Rank items

- Find a set *N(U)* of neighbors
- Average their scores and take the best

$$S[i] = \overline{U} + \frac{\sum_{V \in N(U)} (V[i] - \overline{V}) \cdot \rho(U,V)}{\sum_{V \in N(U)} \rho(U,V)}$$

  - Average its weighted by user similarity

- *S[i]* is not only a score, but a prediction of *U*'s rate

Collaborative Filtering – Wrap up

- Efficiency in building the model
  - User similarity is expensive
  - Done off-line
- Efficiency in generating suggestions
  - K-NN search not needed if pre-computed off-line
- Serendipity of recommendations
  - Great !
- Cold-start problem
  - Present !
- Sparsity:
  - Little votes and little shared votes

Item-based Collaborative Filtering

- Search for similar items, but…
  Measure similarity on the basis of users' rates

- An item x is modeled as:
  - $x[U] = r$ if the user U rates the item x with a score r

- Items x and y are similar if they received similar votes
  - Which measure to use ?

Adjusted Cosine Similarity

- Pearson correlation coefficient ?
  - Measures linear dependence
- Cosine similarity ?
  - measures the angle between *x* and *y*

$$\text{a-cos}(x, y) = \frac{\sum_U (x[U] - \overline{U})(y[U] - \overline{U})}{\sqrt{\sum_U (x[U] - \overline{U})^2}\sqrt{\sum_U (y[U] - \overline{U})^2}}$$

Quality measure

- Mean Absolute Error

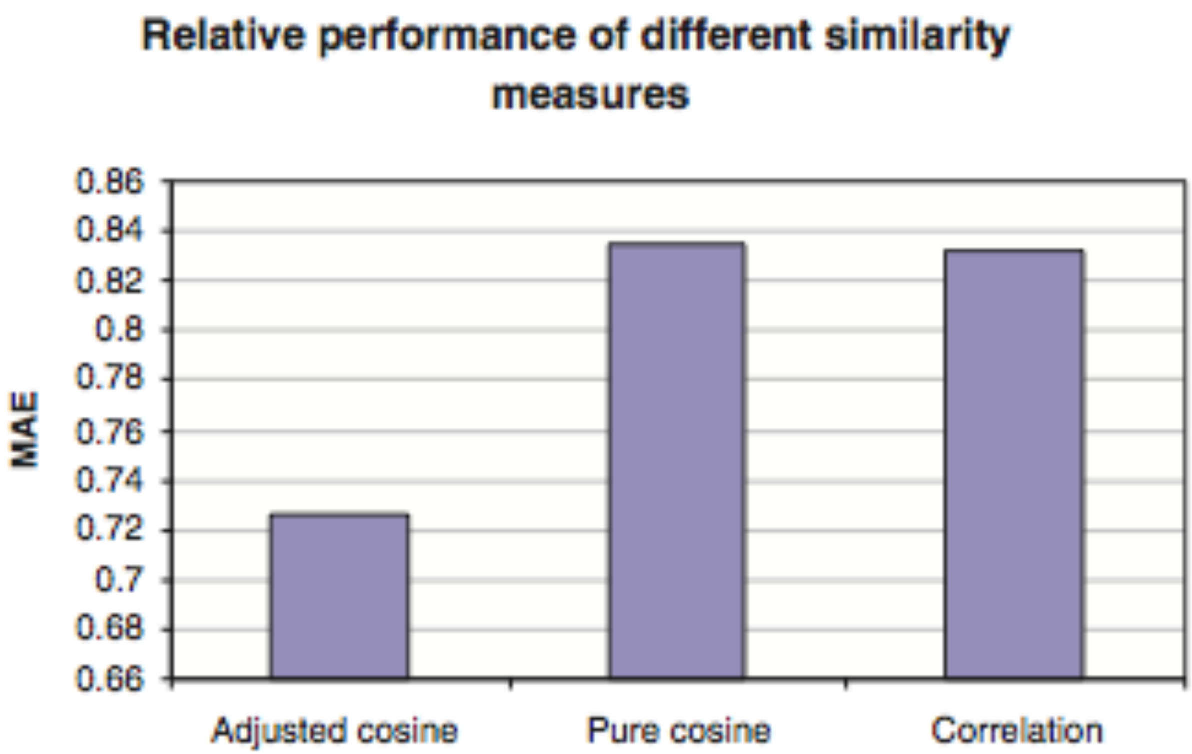
$$\text{MAE} = \frac{\sum_{i=1}^{N} |rate[i] - score[i]|}{N}$$

  - where *rate* is the actual vote and *score* is the predicted one

Experiments on

- Movie lens dataset:
  - 3500 movies
  - 43000 users
- A subset was used:
  - 943 users (with at least 20 ratings)
  - 1682 movies
  - 100,000 ratings
  - 94% of the users-movies matrix is empty

Does it make any difference ?



- Lower is better

Item-based C.F. – Wrap up

- Efficiency in building the model
  - Expensive offline computation
- Efficiency in generating suggestions
  - They are actually pre-computed
- Serendipity of recommendations
  - Great !
- Cold-start problem
  - Absent
- Used by Amazon !

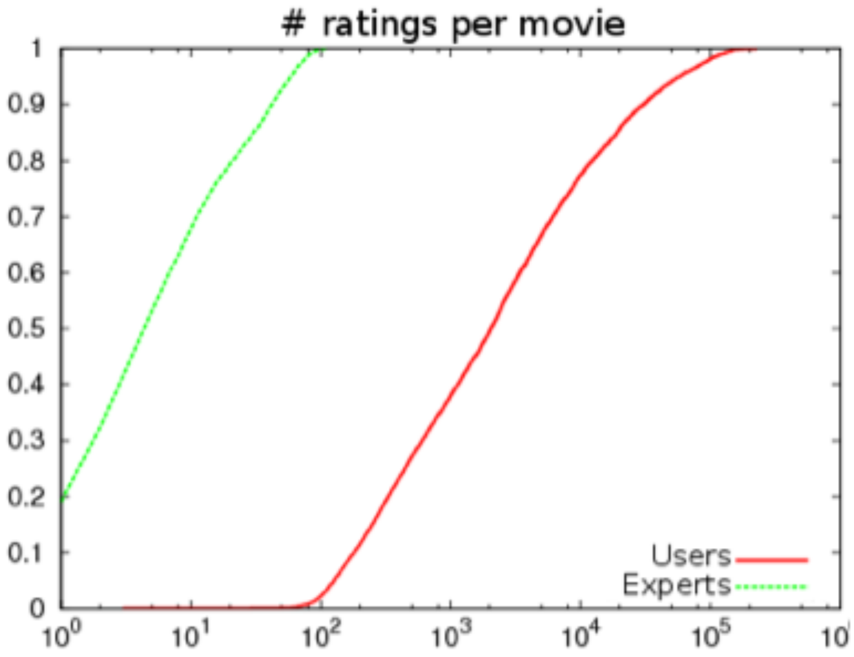
Some papers

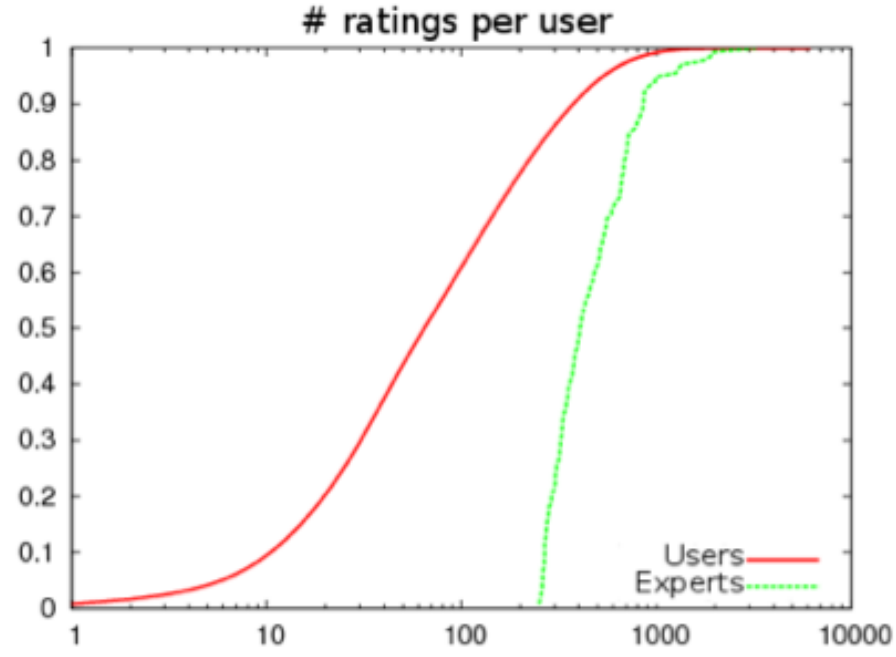
- Linden, G.; Smith, B.; York, J. . **Amazon.com recommendations: item-to-item collaborative filtering.** IEEE Internet Computing 2001.
- Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. **Item-Based Collaborative Filtering Recommendation Algorithms.** WWW 2001.

The wisdom of the few [sigir'09]

- User-based collaborative filtering
  - But, only a set of expert is selected
- Who are the users ?
  - Netflix database
- Who are the experts ?
  - Rotten Tomatoes website
  - Intersect movies from the two sources
  - Expert if at least 250 ratings

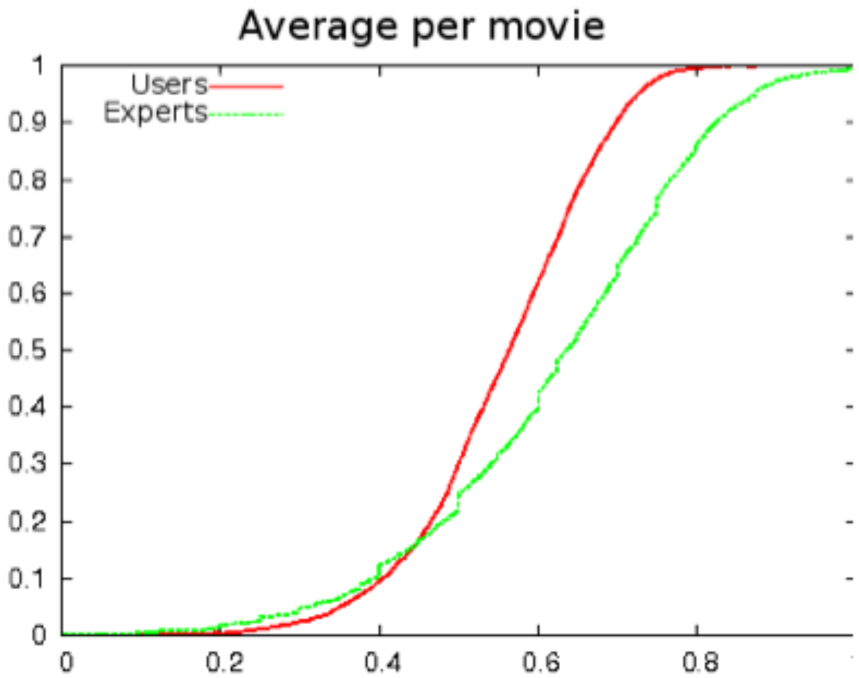
Expert vs. non-expert

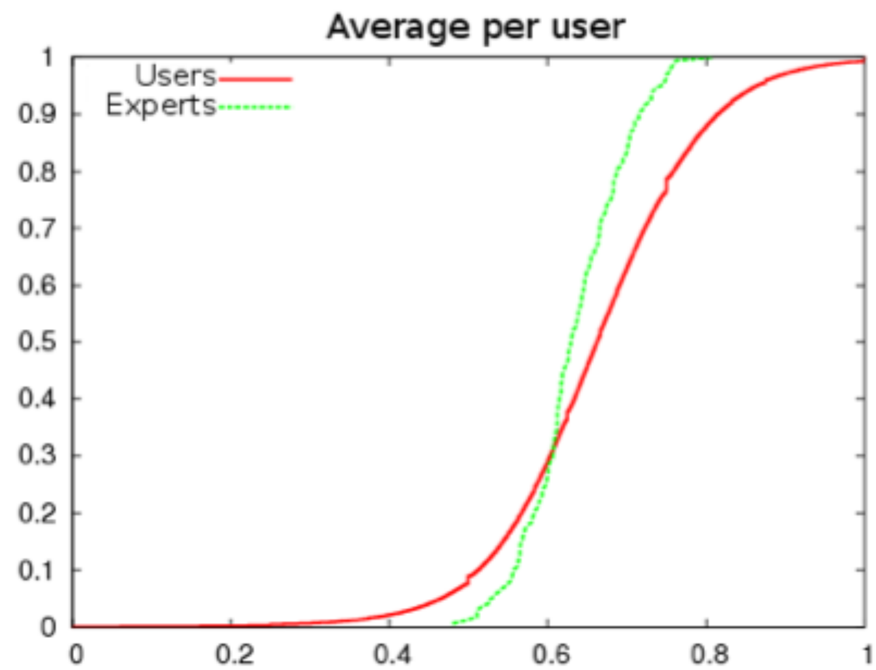




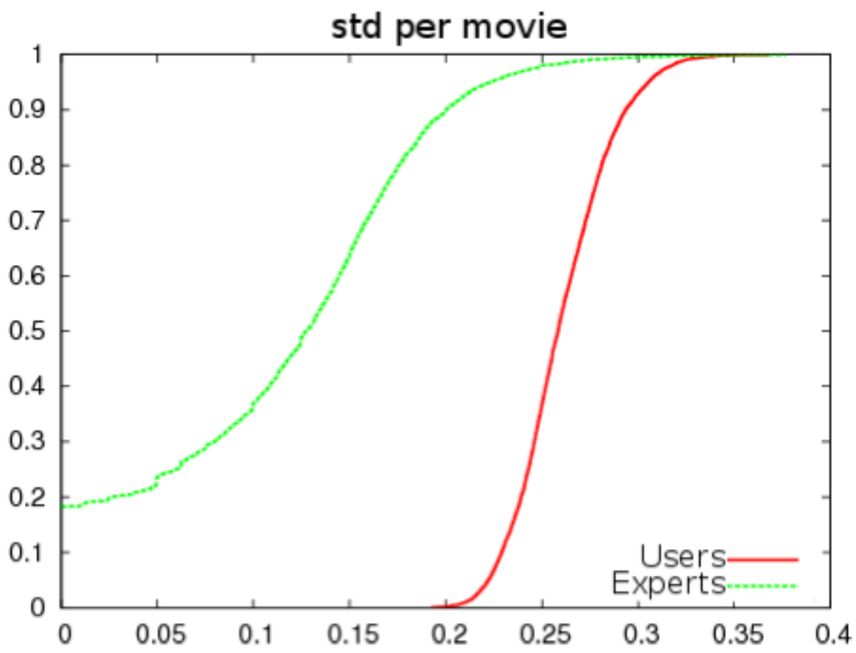
Netflix rating matrix has 1% non zero entries,
Experts rating matrix has 7%

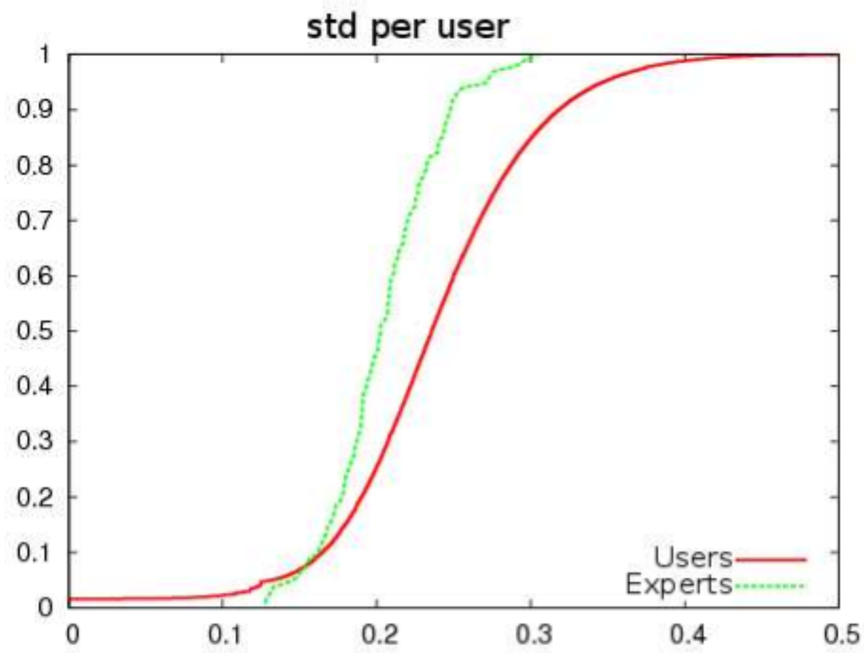
Expert vs. non-expert





Expert vs. non-expert





Expert vs. non-expert

- Experts use the full range of rates
- Experts rate good and bad movies
  - (not biased towards popular ones)
- Experts tend to agree

Building recommendations

- Compute score for item *i* and user *U*
- Search for the experts *E* such that $sim(U,E) > \delta$
- Take only the set of experts *E'* that rated *i*
  - If they are less τ than return no recommendations

- Then …

Building recommendations

-

- Score:

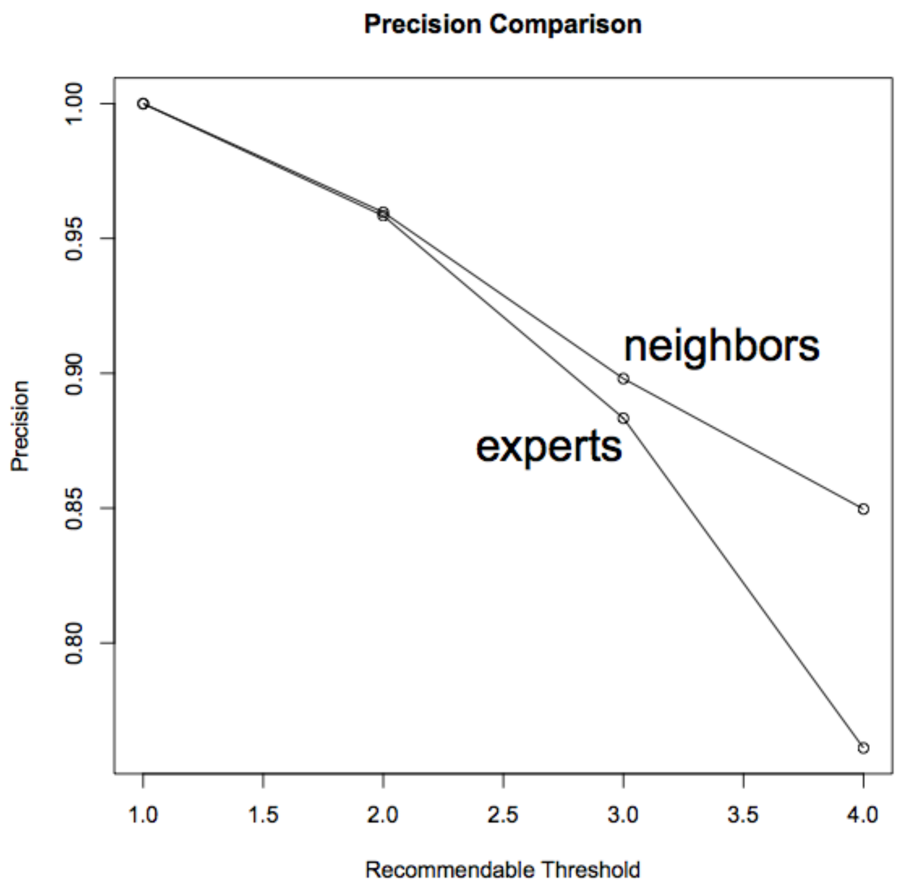
$$S[i] = \overline{U} + \frac{\sum_{E \in E'} (E[i] - \overline{E}) \cdot \mathsf{sim}(U, E)}{\sum \mathsf{sim}(U, E)}$$

Results

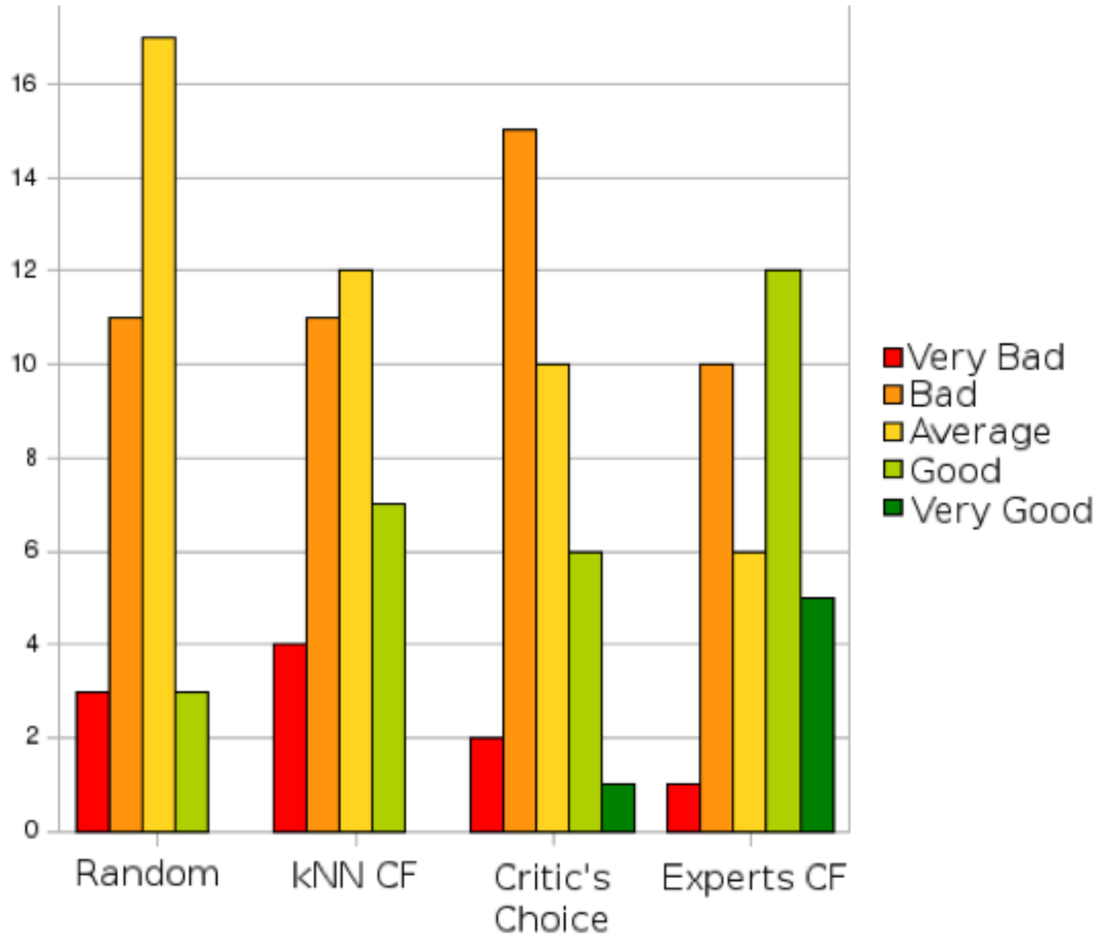
- CF:
  - MAE: 71%, Recall: 93%
- Expert-CF:
  - MAE: 78%, Recall: 98%

Measuring only top recommendations

- Recommend only those items whose predicted score is at least σ
- Check whether the actual score of those items is greater than σ

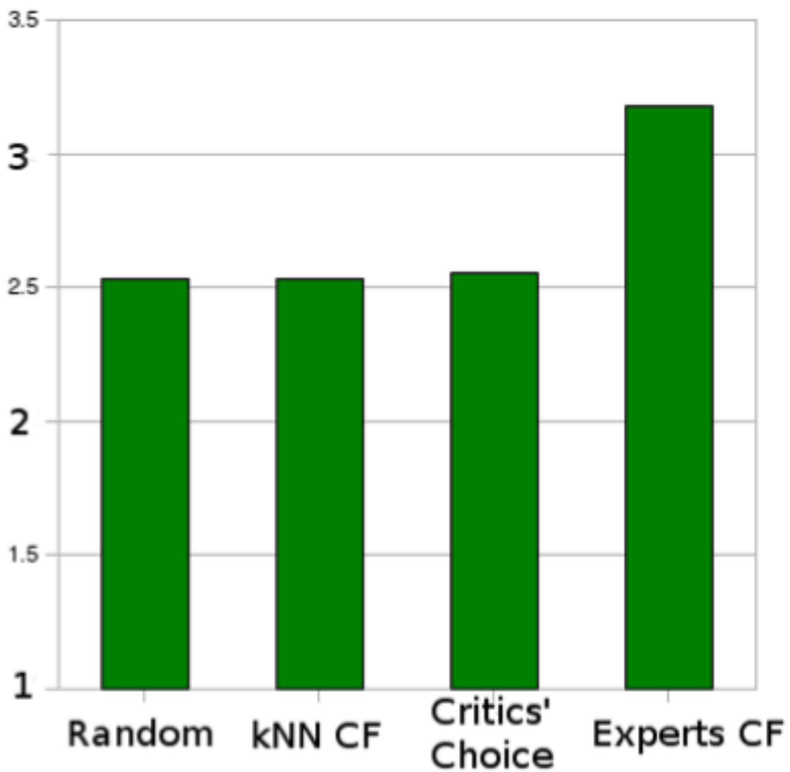


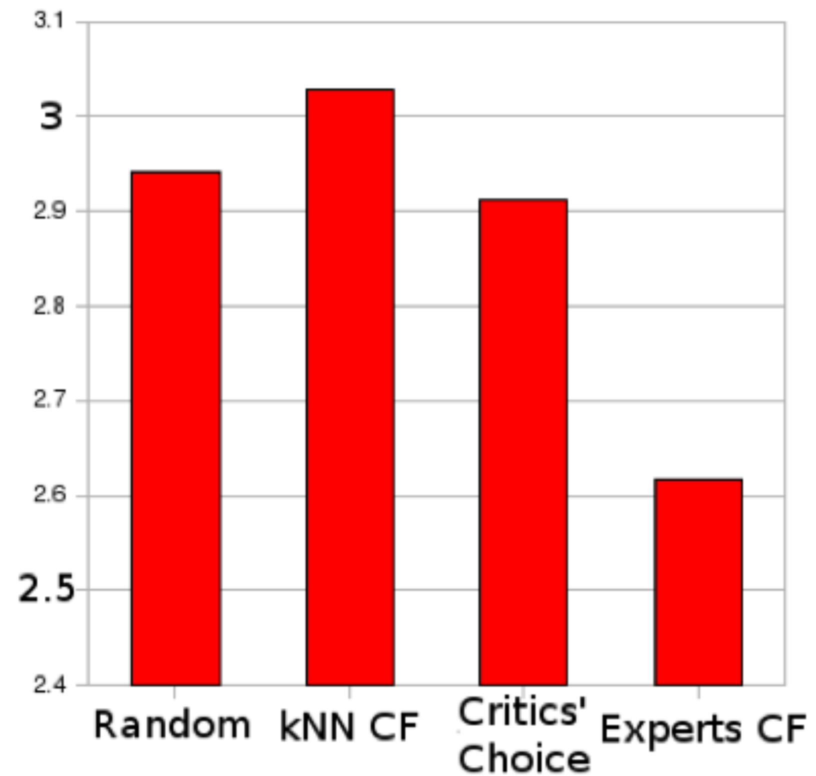
User study



16
14
12
10
8
6
4
2
0
Random
kNN CF
Critic's Choice
Experts CF
Very Bad
Bad
Average
Good
Very Good

User study


3.5
3
2.5
2
1.5
1
Random
kNN CF
Critics' Choice
Experts CF


3.1
3
2.9
2.8
2.7
2.6
2.5
2.4
Random
kNN CF
Critics' Choice
Experts CF

The wisdom of the few – Wrap-up

- A small set of experts by enclose as much knowledge as the big set of general users
  - (maybe more ?)

- Scalability

Patterns of Influence in a Recommendation Network

- Objective:
  - Discover the most frequent patterns of recommendation propagation
  - Applicable to information propagation as well
    - Think about facebook…
- User study on a large on-line retailer:
  - Books, DVDs, music, videos.
  - After a purchase, users could send recommendations via email
  - If the receives buys the item, the sender gets some credit

The input data

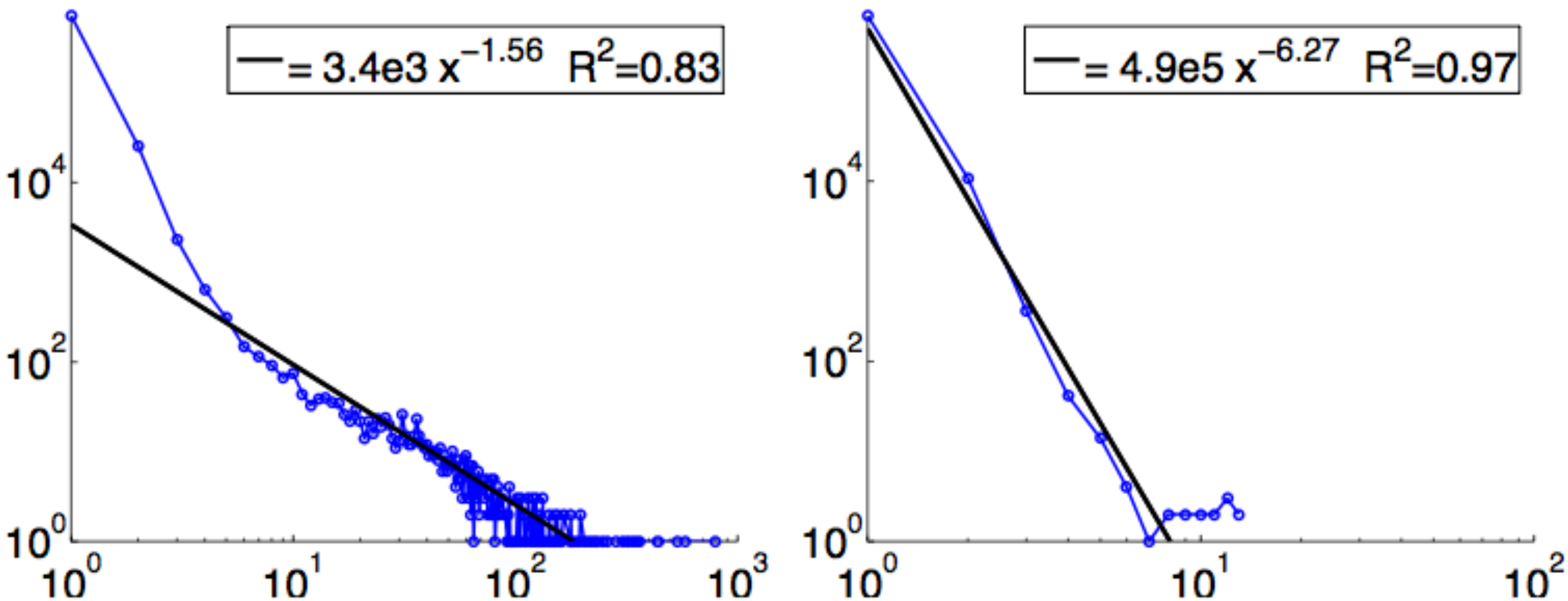
- 15,646,121 recommendations, 3,943,084 distinct users, 711 days, 542,719 products
- Represented as a labeled directed graph:
  - Nodes are costumers
  - a directed edge (v, w) with label (p, t) means that node v recommended product p to customer w at time t

- The goal is to identify recurrent sub-graphs

Strategies

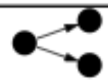
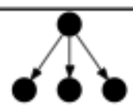
- Delete late recommendations:
  - If a use buys a product, remove all the incoming edges happening after the purchase
- Delete no-purchase nodes
- All connected components in the resulting graphs are potentially interesting
- “Cascade” enumeration:
  - For each node consider his predecessors up to distance *h*
  - Count the number of those graphs

Size distribution



(b) DVD (c) Music

The patterns

Id	Graph	Nodes	Edges	Book R	Book F	DVD R	DVD F	Music R	Music F	Video R	Video F
G_1		2	1	1	86,430	1	36,863	1	11,518	1	1,425
G_2		3	2	2	10,573	4	3,238	2	492	5	33
G_3		3	2	3	5,089	2	5,147	3	389	3	61
G_4		3	2	6	1,593	5	2,419	5	115	22	4
G_6		4	3	5	2,769	15	505	6	55	20	5
G_{13}		4	3	92	21	12	549	54	4	0	0

Other patterns