Recommender Systems

IN 60 SECONDS

GO-Globe
CUSTOM WEB DEVELOPMENT

44,000,000 MESSAGES PROCESSED

486,000 PHOTOS

26 NEW REVIEWS POSTED ON YELP

120 NEW ACCOUNTS OPENED ON LINKEDIN

MORE THAN **140** SUBMISSIONS ON REDDIT

MORE THAN **2,315,000** SEARCHES

3,125,000

243,055

MORE THAN **21,000,000** MESSAGES SENT

70,000 VIDEO MESSAGES SHARED

MORE THAN **195,000** MINUTES OF **AUDIO CHATTING** ON WECHAT

Share

MORE THAN **3,000,000** ITEMS ARE **SHARED**

MORE THAN **18,000** MATCHES MADE

972,000 DAILY SWIPES ON TINDER

MORE THAN **69,500** HOURS OF **VIDEO WATCHED** ON NETFLIX

MORE THAN **150,000,000** E-MAILS ARE SENT

MORE THAN **48,000** APPS DOWNLOADED ON IPHONE

MORE THAN **95,000** APPS DOWNLOADED ON ANDROID

MORE THAN **430,000** TWEETS SENT

AROUND **56,000** PHOTOS **UPLOADED**

9,800 ARTICLES PINNED ON PINTEREST

MORE THAN **280,000** SNAPS SENT ON SNAPCHAT

MORE THAN **100** NEW DOMAINS REGISTERED

MORE THAN **39,300** HOURS OF MUSIC **LISTENED**

14 NEW SONGS ADDED ON SPOTIFY

MORE THAN **2,700,000** VIDEO VIEWS AND **139,000** HOURS OF VIDEO **WATCHED**

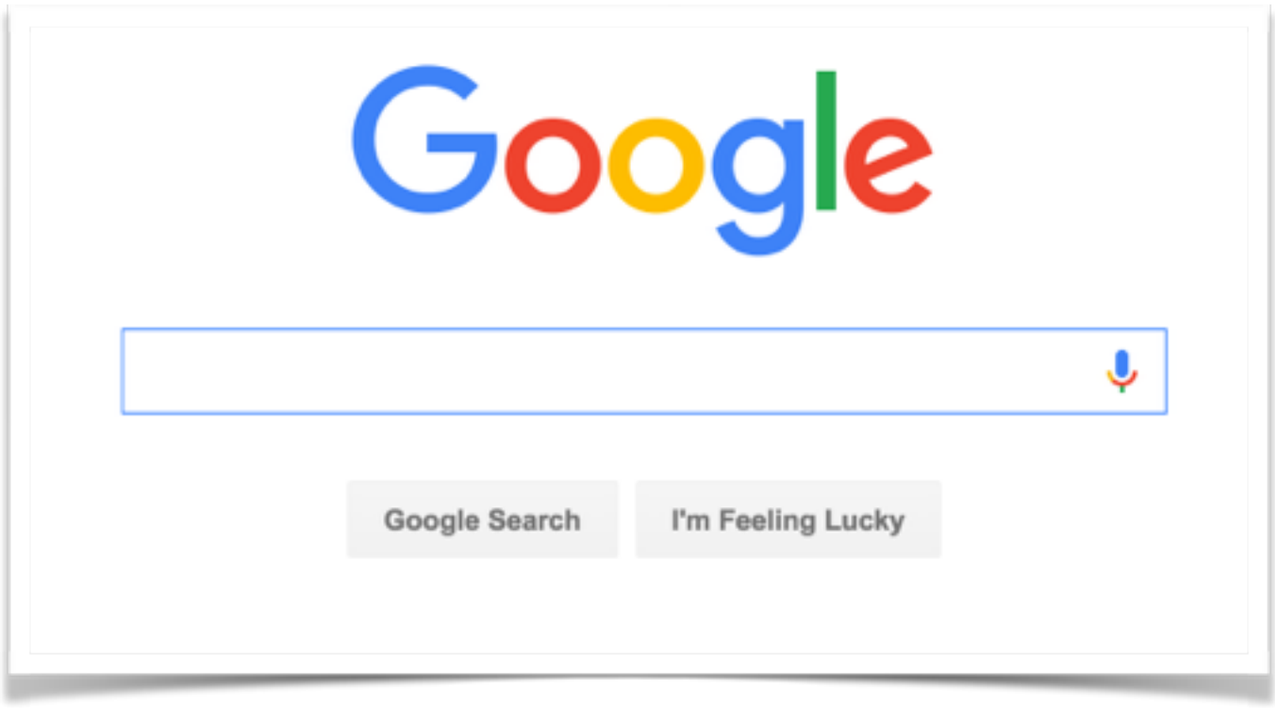
MORE THAN **300** HOURS OF VIDEO ARE **UPLOADED**

Getting information off the Internet is like taking a drink from a fire hydrant.

Mitchell Kapor

Adapted from http://www.flickr.com/photos/josephnebenzahl/127758523

Can Google help?



- Yes, but only when we really know what we are looking for
- What if I just want some interesting music tracks?
  - Btw, what does it mean by “interresting”?

Can Facebook help?

- Yes, I tend to find my friend's stuffs interesting
- What if I had only few friends, and what they like do not always attract me?

Can experts help?

- Yes, but it won't scale well
  - Everyone receives exactly the same advice!
- It is what they like, not me!
  - Like movies, what get expert approaval does not guarantee attention of the mass

Recommender Systems

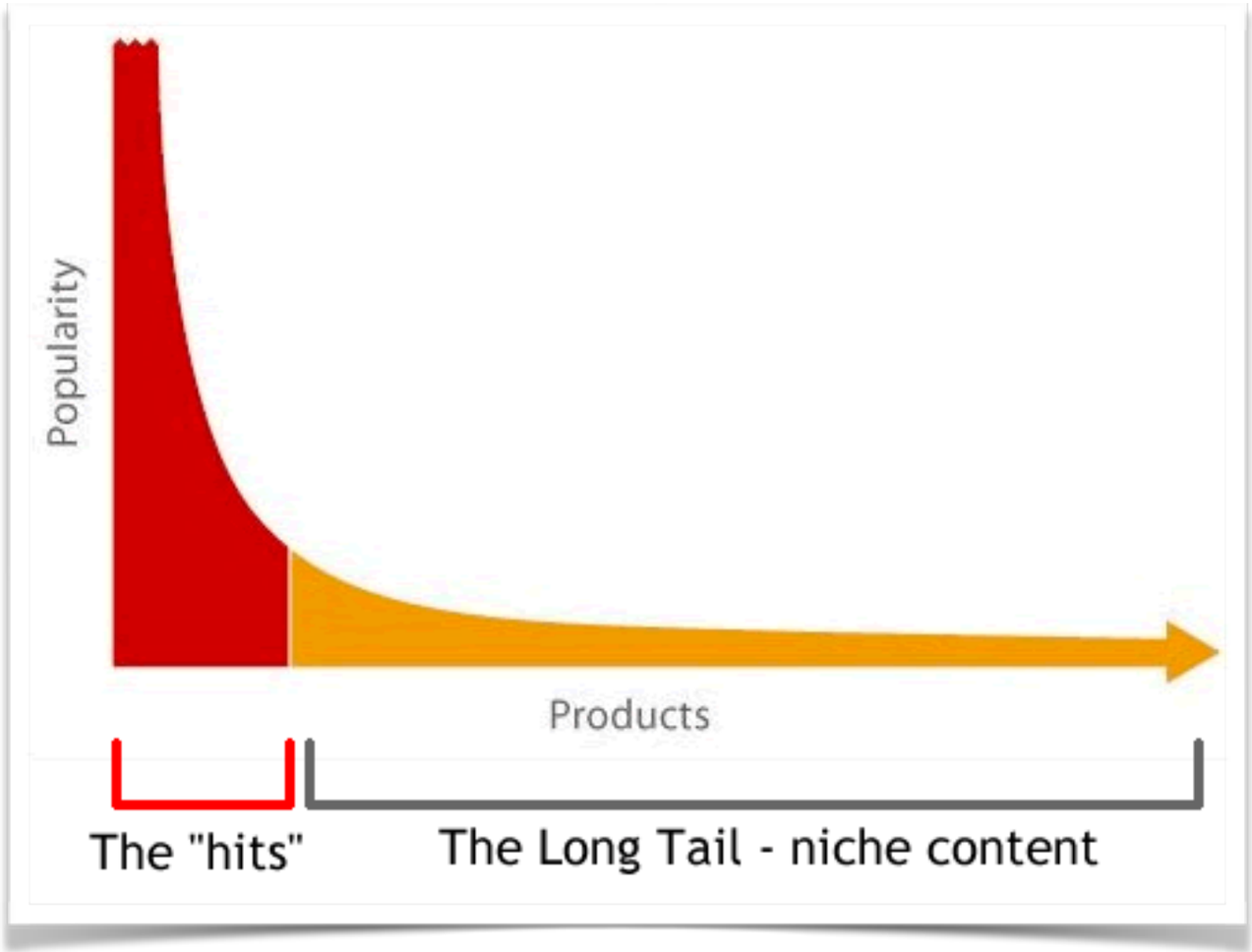
Help to match users with items

An idea called RecSys

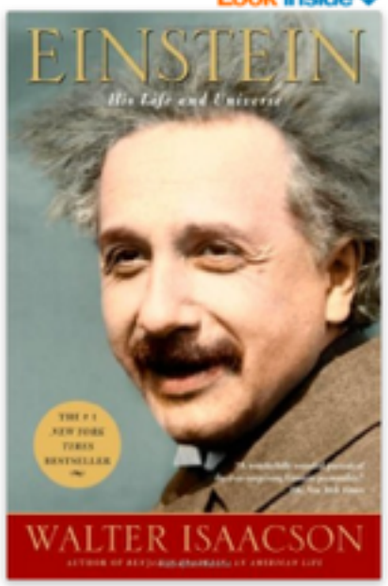
- To recommend to us something we may like
  - it may not be popular
  - the world is **long-tailed**
- How?
  - based on our **history** of using services
  - based on **other people** like us
  - among **other things**…

Hang on, what is long-tailed?

- Popularised by Chris Anderson, Wired 2004



Ever heard of



Einstein: His Life and Universe Paperback – May 13, 2008

by Walter Isaacson (Author)

4.5 out of 5 stars 796 customer reviews

See all 28 formats and editions

Kindle	Hardcover	Paperback	Audible
$13.59	from $0.01	$13.78	$0.00
Read with Our Free App	407 Used from $0.01 44 New from $7.45 34 Collectible from $8.99	278 Used from $2.00 111 New from $5.33 12 Collectible from $6.98	Free with your Audible trial

By the author of the acclaimed bestsellers *Benjamin Franklin* and *Steve Jobs*, this is the definitive biography of Albert Einstein.

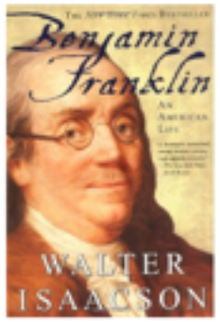
How did his mind work? What made him a genius? Isaacson's biography shows how his scientific imagination sprang from the rebellious nature of his personality. His fascinating story is a testament to the connection between creativity and freedom.

Based on newly released personal letters of Einstein, this book explores how an imaginative, impertinent patent clerk—a struggling father in a difficult marriage who couldn't get a teaching job or a doctorate—became the mind reader of the creator of the cosmos, the locksmith of the mysteries of the atom, and the

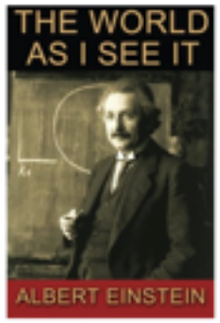
Read more

Customers Who Bought This Item Also Bought

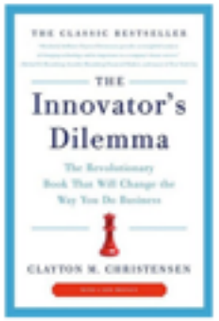
Page 1 of 17



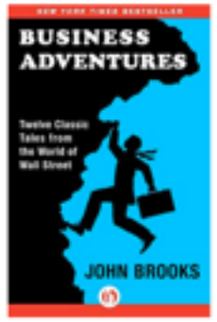
Benjamin Franklin: An American Life
Walter Isaacson
4.5 out of 5 stars 614
Paperback
$11.90 Prime



The World As I See It
Albert Einstein
4.5 out of 5 stars 21
Paperback
$5.99 Prime



The Innovator's Dilemma: The Revolutionary Book That Will Change the...
› Clayton M. Christensen
4.5 out of 5 stars 227
#1 Best Seller in Development & Growth Economics
Paperback
$10.11 Prime



Business Adventures: Twelve Classic Tales from the World of Wall Street
› John Brooks
4.0 out of 5 stars 311
#1 Best Seller in Commerce
Paperback
$13.23 Prime

The Innovators: How a Group of Hackers, Geniuses, and Geeks...
› Walter Isaacson
4.5 out of 5 stars 940
Paperback
$16.19 Prime

Steve Jobs
› Walter Isaacson
4.5 out of 5 stars 5,075
Paperback
$11.90 Prime

YouTube BR

PSY - GANGNAM STYLE(강남스타일) M/V

PSY - GANGNAM STYLE(강남스타일) M/V

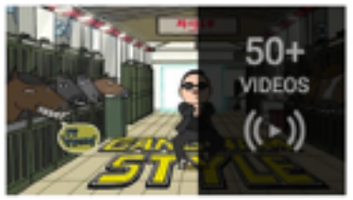
officialpsy

Subscribe 9,256,129

2,642,586,309 views

Add to Share More 11,271,747 1,589,155

LMFAO - Party Rock Anthem ft. Lauren Bennett, GoonRock
LMFAOVEVO
1,162,788,195 views



Mix - PSY - GANGNAM STYLE(강남스타일) M/V
YouTube

PSY - GENTLEMAN M/V
officialpsy
986,894,424 views

PSY - DADDY(feat. CL of 2NE1) M/V
officialpsy
192,574,916 views

PSY (ft. HYUNA) - 오빤 딱 내 스타일 M/V
officialpsy
652,729,010 views

The value of recommendation

- **Netflix**: 2/3 of the movies watched are recommended
- **Google News**: recommendations generate 38% more clickthrough
- **Amazon**: 35% sales from recommendations

Academic

- In 2007
  - ACM RecSys Conference
  - Tracks on major CS conferences
    - WWW, CIKM, WSDM, KDD, SIGIR, ICML, NIPS, …
  - and journals …

Problem characterization

- Given
  - The profile of the "target" user and possibly some situational context
- Compute
  - A relevance (ranking) score for each recommendable item
- The profile
  - … can include past user ratings (explict or implicit), demographics, …
- The problem
  - … is to learn a function that predicts the relevance score for a given (unseen) item

Recommender Systems

Recommender systems reduce information overload by estimating relevance

top-*N*

Recommender Function

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Recommender Systems

User profile
& context

Recommender Function

Personalized recommendations

top-*N*

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Recommender Systems

User profile & context

Community data

Recommender Function

Collaborative: "tell me what's popular among my peers"

top-*N*

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Recommender Systems

User profile
& context

Content-based: "show me more of the same what I've liked"

Recommender Function

top-*N*

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Item features

Title	Genre	Actors	...

Recommender Systems

User profile & context

Community data

Item features

Title	Genre	Actors	...

Recommender Function

Hybrid: combinations of various inputs and/or composition of different mechanisms

top-*N*

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Collaborative filtering

Collaborative filtering

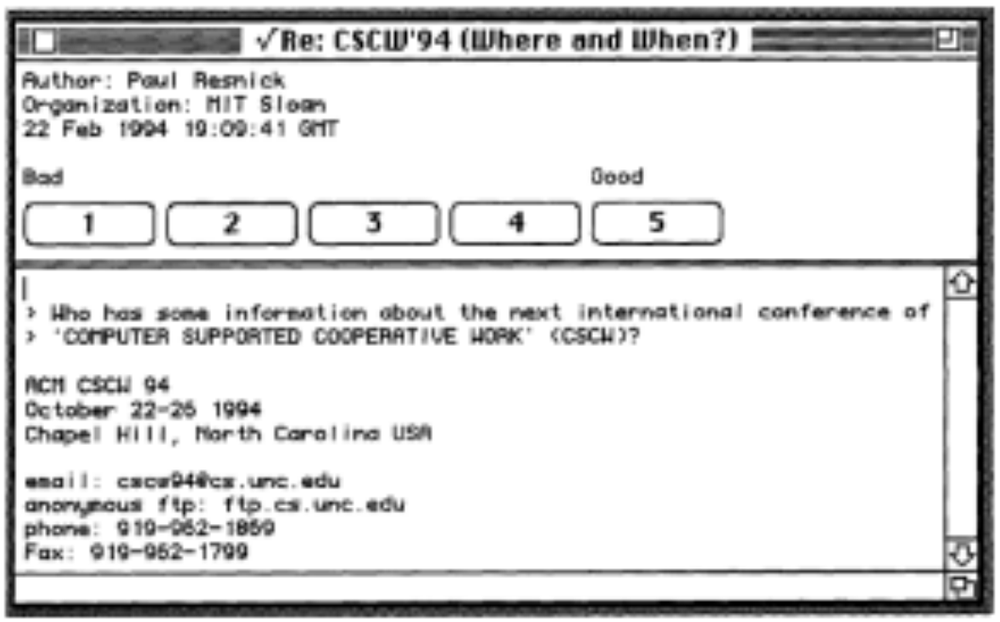
- The **most prominent** approach
- Use the preferences of a community to recommend items
- Basic assumption and idea
  - users give ratings to catalog items (implicitly or explicitly)
  - patterns in the data help me predict the ratings of individuals, i.e., fill the missing entries in the rating matrix
    - there are users with similar preference structures
    - there are latent characteristics of items that influence the ratings by users

1992: Using collaborative filtering to weave an information tapestry (D. Goldberg et. al., Comm. of the ACM)

- Eager readers read all docs immediately, casual readers wait for the eager readers to annotate
- Experimental mail system at Xerox Parc
  - Records reactions of users when reading a mail
- Users are provided with personalized mailing list filters instead of being forced to subscribe
  - content-based filters (topics, …)
  - collaborative filters
    - "mails to [all] which were replied by [Rob] and which received positive ratings from [X] and [Y]"

1994: Groplens: an open architecture for collaborative filtering of netnews (P. Resnick et al., ACM CSCW)

- Tapestry system does not aggregate ratings and requires knowing each other
- Basic idea of GroupLens:
  - **People who agreed in their subjective evaluation in the past are likely to agree again in the future**
- Builds on newsgroup browsers with rating functionality



Nearest-neighbors (kNN)

- A "pure" CF approach and traditional baseline
- Solution approach
  - Given an "target user" (Alice) and an item *i* not yet seen by Alice
    - Estimate Alice's rating for this item based on *like-minded users* (peers)
- Assumptions
  - If users had similar tastes in the past they will have similar tastes in the future
  - User preferences remain stable and consistent over time

Questions…

1. How to determine the similarity of two users?
2. How do we combine the ratings of the neighbors to predict Alice's rating?
3. Which/how many neighbors' opinions to consider?

1 - Computing similarity

- A popular measure: Pearson's correlation coefficient

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

a, b : users
$r_{a,p}$: rating of user a for item p
P : set of items, rated both by a and b
$\overline{r_a}, \overline{r_b}$: user's average ratings

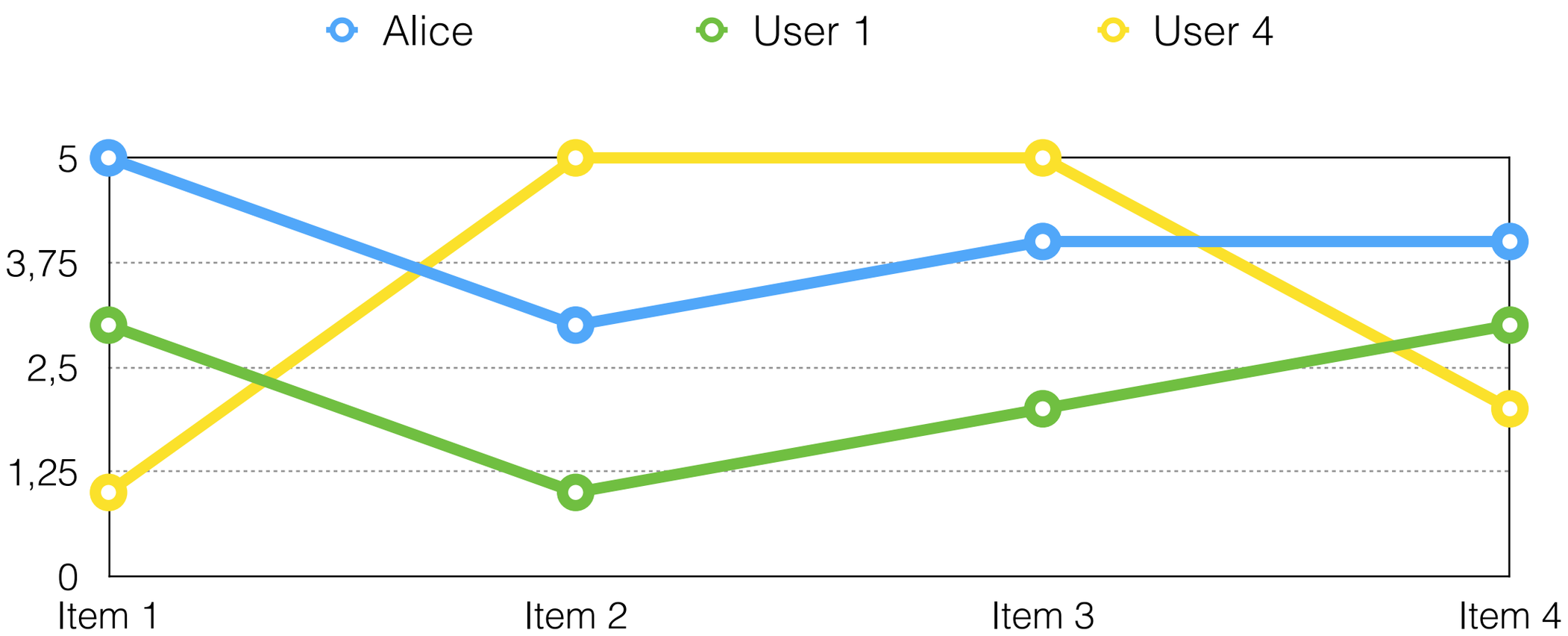
Possible similarity values between -1 and 1;

	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 with as a weight
    - add/subtract the neighbors' bias from the target user's average and use this as a prediction
- How many neighbors?
  - only consider positively correlated neighbors (or higher threshold)
  - can be optimized based on data set
  - often, between 50 and 200

kNN considerations

- Very simple scheme leading to quite accurate recommendations
  - still today often used as a baseline scheme
- Possible issues
  - scalability
    - thinking of millions of users and thousands of items
    - pre-computation of similarities possible but potentially unstable
    - clustering techniques are often less accurate
  - coverage
    - problem of finding enough neighbors
    - users with preferences for niche products

Memory- and model-based approaches

- kNN methods are often said to be "memory-based"
  - the rating matrix is directly used to find neighbors and make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have 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

Model-based techniques

- Variety of approaches proposed in recent years, e.g.,
  - matrix factorization techniques
    - singular value decomposition, principal component analysis
  - probabilistic models
    - clustering models, Bayesian networks, topic models, …
  - various other machine learning approaches
    - regression-based techniques, deep neural networks, …
- Costs of pre-processing
  - usually not discussed
  - incremental updates possible = algorithms exist

Matrix factorization

- Informally, the SVD theorem (Golub and Kahan 1965) states that a given matrix *M* can be decomposed into a product of three matrices as follows

$$M = U \times \Sigma \times V^T$$

  - where *U* and *V* are called *left* and *right singular vectors* and the values of the diagonal of Σ are called the *singular values*

- We can approximate the full matrix
  - by observing only the most important features - those with the largest singular values
- In the example,
  - we calculate *U, V, and* Σ (with the help of some linear algebra software)

Example for SVD-based recommendation

- U and V correspond to the latent user and item factors

- 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

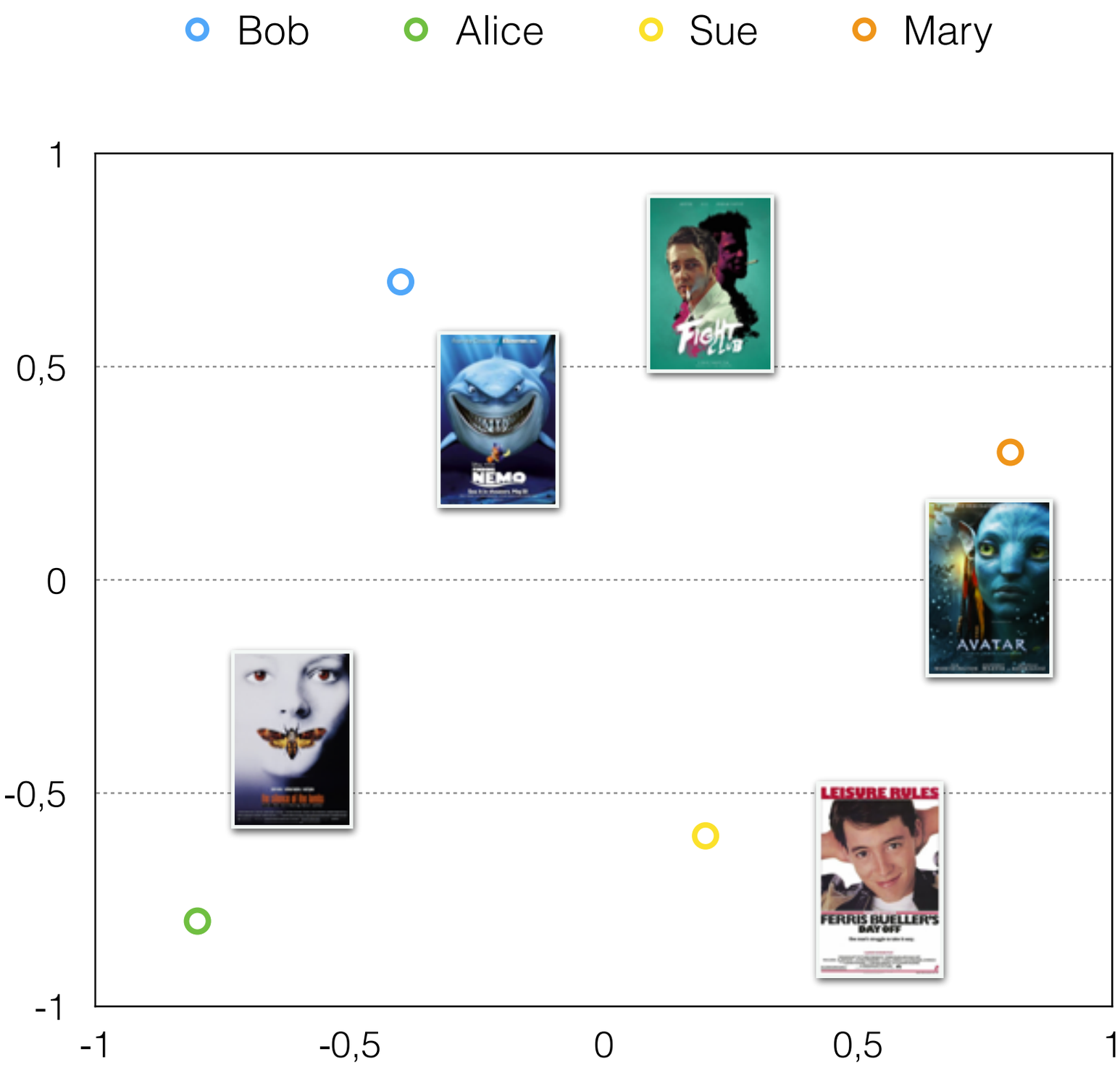
V_k^T	Terminator	Die Hard	Twins	Eat Pray Love	Pretty Woman
Dim1	-0.44	-0.57	0.06	0.38	0.57
Dim2	0.58	-0.66	0.26	0.18	-0.36

Σ_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

The “latent factor space”



2006: "Funk-SVD" and the Netflix prize

- Netflix announced a million dollar prize
  - Goal:
    - Beat their own "Cinematch" system by 10 percent
    - Measure in terms of the Root Mean Squared Error (RMSE)
  - Effect:
    - stimulated lots of research
- Idea of SVD and matrix factorization picked up again
  - S. Funk
    - use fast gradient descent optimization procedure
    - http://sifter.org/~simon/journal/20061211.html

Learn the weights iteratively

- Start with small initial weights
- Repeat
  - make prediction with current model
  - adapt the weights incrementally
    - learning rate as a hyperparameter
  - stop after *n* iterations

2008: Factorization meets the neighborhood: a multifaceted collaborative filtering model (Y. Koren, KDD)

- Combines neighborhood models with latent factor models
  - latent factor models
    - good to capture weak signals in the overall data
  - neighborhood models
    - good at detecting strong relationship between similar items
- combination in one prediction single function
  - includes user- and item bias, considers who rated what
  - add penalty (regularization) for high values to avoid over-fitting

$$\hat{r}_{ui} = \mu + b_u + b_i + p_u^T q_i$$

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

Generalization: a machine learning problem

- Recommendation is concerned with learning from noisy observations (x,y), where $f(x)=\hat{y}$ has to be determined such that $\sum_{\hat{y}}(\hat{y}-y)^2$ is minimal
- A variety of different learning strategies have been applied trying to estimate f(x)
  - non parametric neighborhood models
  - MF models, factorization machines, deep neural networks, …
  - Netflix prize winner:
    - combine a large number of predictors in ensemble method

Rating prediction and Item recommendation

- Making predictions is typically not the ultimate goal
- Usual approach
  - **rank items** based on their predicted ratings
- Ranking approaches
  - "Learning to rank"
    - recent interest in ranking techniques
    - optimize according to a (proxy of a) given rank evaluation metric

Explicit and implict ratings

- Explicit ratings
  - most commonly used (1 to 5, …)
  - typically only one rating per user and item, including time-stamp
- Some research topics
  - **data sparsity**
    - users not always willing to rate many items
    - how to stimulate users to rate more items?
  - which items have (not) been rated?
    - **ratings not missing at random**
  - **multidimensional ratings**
    - multiple ratings per hotel (location, service, …)

Explicit and implict ratings

- Implicit ratings (feedback)
  - clicks, page views, time spent on some page, …
  - multiple events over time
  - can be collected constantly and do not require additional efforts from the side of the user
- Research topics
  - correct interpretation of the (strength of the) action
    - buy something for a friend, accidental clicks
  - huge amounts of data to be processed
  - algorithmic questions
    - combination with explicit ratings (e.g., Koren's SVD++ method)
    - specific algorithms (e.g., Bayesian Personalized Ranking)

Data sparsity - cold start situations

- 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 or simply non-personalized) in the initial phase
  - default voting: assign default values to items that only one of the two users to be compared has rated
- Alternatives
  - use better algorithms (beyond nearest-neighbor approaches)
  - exploit additional information sources, e.g., Social Web data
- Example
  - in nearest-neighbor approaches, the set of sufficiently similar neighbors might be too small to make good predictions
  - assume "transitivity" of neighborhoods

Summary CF approaches

- Operate on the basis of explicit or implicit feedback of a user community
  - well-understood, lots of algorithms
  - works in practice
  - no information about the items required
- Challenges
  - **cold start** and data sparsity issues
  - **scalability** can be an issue
  - often no **explanations** possible

CF tools and libraries

- Some open source solutions exist
  - MyMediaLite (C#)
    - Implements wide range of modern algorithms
  - LensKit
    - modular framework built in Java
  - PREA
    - Java-based library of recent CF algorithms
  - Apache Mahout, RapidMiner, Apache Spark + MLib
    - implement learning algorithms usable for recommenders
    - Mahout: distributed algorithms on Hadoop
  - Recomender101
    - Java-based framework, several algorithms and metrics

Content-based filtering

Content-based filtering

User profile
& context

Content-based: "show me more of the same what I've liked"

Recommender Function

top-*N*

Item	score
i_3	0,87
i_8	0,65
i_2	0,29

Item features

Title	Genre	Actors	...

Content-based filtering

- Again:
  - determine preferences of user based on past behavior
- However,
  - look at what the target user liked (purchased, viewed, …)
  - estimate the user's taste for certain item features
    - e.g., genre, authors, release date, …
- alternative preference acquisition
  - ask the user, look at recently viewed items

What is the "content"?

- Most CB-recommendation techniques were applied to recommending text documents
  - like web pages or newsgroup messages
- Content of items can also be represented as text documents
  - with textual descriptions of their basic characteristics
    - **structured**: each item is described by the same set of attributes
    - **unstructured**: free-text description

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, murder, neo-Nazism

Content representation and item similarities

- Represent items and users in the same way

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, murder, neo-Nazism

Title	Genre	Author	Type	Price	Keywords
...	Fiction	Brunonia, Barry, Ken Follett	Paperback	25.65	Detective, murder, New York

- A simple method
  - compute the similarity of an unseen item with the user profile based on the keyword overlap
  - or use and combine multiple metrics

$$\frac{2 \times |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + |keywords(b_j)|}$$

Term-frequency - Inverse document frequency (TF-IDF)

- Simple keyword representation has its problems
  - specially when automatically extracted:
    - not every word has similar importance
    - longer documents have a higher chance to have an overlap with the user profile
- Standard measure: TF-IDF
  - encodes text documents in multi-dimensional Euclidian space
  - **TF**: measures how often a term appears (density in a document)
    - assuming that important terms appear more often
    - normalization has to be done in order to take document length into account
  - **IDF**: aims to reduce the weight of terms that appear in all documents

Improving the vector space model

- Vectors are usually long and sparse
- **Remove stop words**
  - they will appear in nearly all documents
  - e.g.: “a”, “the”, “on”, …
- Use **stemming**
  - aims to replace variants of words by their common stem
  - e.g.: “went” = “go”, “stemming” = “stem”, …
- Size **cut-offs**
  - only use top n most representative words to remove “noise” from data
  - e.g.: use top 100 words
- **Tuning** of **representation**
  - logarithmic instead of linear TF count

Improving the vector space model

- Use **lexical knowledge**, use more elaborate methods for feature selection
  - **remove** words that are **not relevant** in the **domain**
- Detection of phrases/n-grams
  - more descriptive for a text that single words
    - e.g.: "United Nations"
- Limitations
  - semantic meaning remains unknown
  - example: **usage of a word in a negative context**
    - **"there is nothing on the menu that a vegetarian would like…"**
    - the word "vegetarian" will receive a higher weight then desired
      - an unintended match with a user interested in vegetarian restaurants

Comparing the vectors (users/items)

- Usual similarity metric to compare vectors: cosine similarity
  - cosine similarity is calculated based on the angle between the vectors
  - compensates for the effect of different document lengths

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

Recommending items

- Item recommendation: nearest neighbors
  - Given a set of documents *D* already rated by the user (like/dislike)
  - Find the *n* nearest neighbors of a not-yet-seen item *i* in *D*
    - use similarity measures to capture similar documents
- Rating predictions
  - Take these neighbors to predict a rating for *i*
    - e.g.: *k* = 5 most similar items to *i*
    - 4 of *k* items were liked by current user = item *i* will also be liked by this user
  - Variations:
    - varying neighborhood size *k*
    - lower/upper similarity thresholds
  - Good to model short-term interests / follow-up stories
  - Used in combination with method to model long-term preferences

On feature selection

- Process of choosing a subset of available terms
- Different strategies exist for deciding which features to use
  - feature selection based on domain knowledge and lexical information from WordNet
  - frequency-based feature selection to remove words appearing "too rare" or "too often"
- Evaluate value of individual features (keywords) independently and
- Construct a ranked list of "good" keywords
  - typical measures for determining utility of keywords
    - e.g.: X^2, mutual information or Fisher's discrimination index

Limitations of content-based methods

- Keywords alone may not be sufficient to judge quality/ relevance of a document or web page
  - up-to-date-ness, usability, writing style
  - content may also be limited / **too short**
  - content may not be automatically extractable (multimedia)
- Overspecialization
  - algorithms tend to propose “more of the same”
  - or: too similar news items

Discussion and summary

- Content-based techniques do not require a user community
  - they however require content information
  - recetly, Wikipedia, Linked Data, Social Tags, …
- The presented approaches learn a model of the user's interest preferences based on explicit or implicit feedback
  - deriving implicit feedback from user behavior can be problematic
- Danger exists that recommendation lists contain too many similar items
  - all learning techniques require a certain amount of training data
  - some learning methods tend to overfit the training data
- Research focuses on CF methods, in practice, however
  - content-based methods work well in some domains

Evaluation

What is a good recommendation?

- This might lead to ..
- What is a good recommendation?
- What is a good recommendation strategy?
- What is a good recommendation strategy for my business?

What is a good recommendation?

- What are the measures in practice?
  - total number of sales
  - click-through rates
  - customer return rates
  - customer satisfaction and loyalty

How do we as researchs know?

- Test with real users
  - A/B tests
  - example measures: sales increase, click-through rates
- Laboratory studies
  - controlled experiments
  - example measures: satisfaction with the system
- Offline experiments
  - based on historical data
  - example measures: prediction accuracy, coverage, …

Metrics: Precision and Recall

- Recommendation is viewed as information retrieval tasks:
  - retrieve (recommend) all items which are predicted to be "good"
  - compare with "hidden" elements for which the ground truth is known
- **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

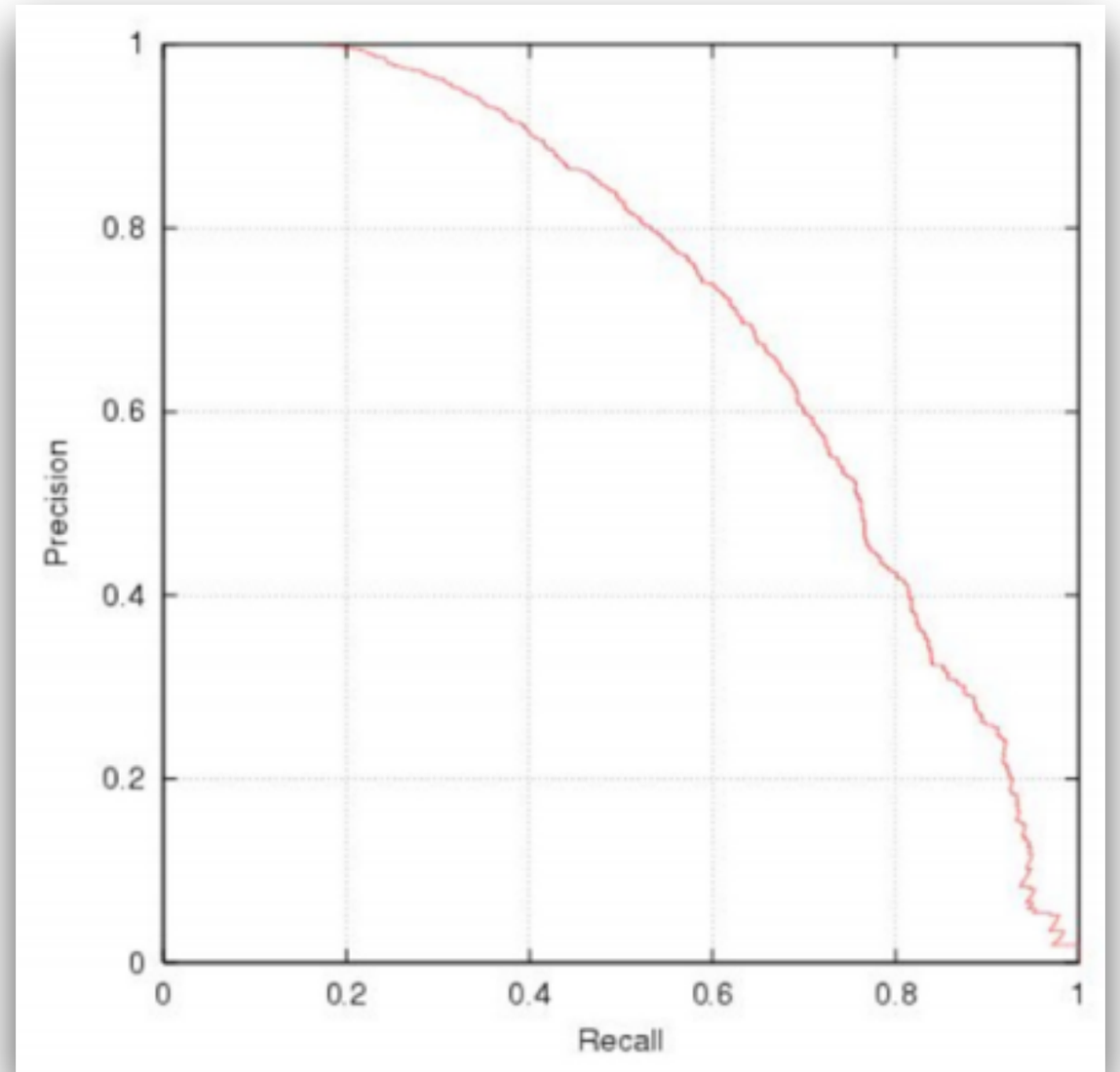
$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|\text{all recommendations}|}$$

- **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

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

Precision vs. Recall

- E.g. typically when a recommender system is tuned to increase precision, recall decreases as a result (or vice versa)



F1 metric

- The F1 metric attempts to combine Precision and Recall into a single value for comparison purposes
  - may be used to gain a more balanced view of performance

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

- the F1 metric gives equal weight to precision and recall
  - other F* metrics weight recall with a factor of *

Precision@k, recall@k, Mean Average Precision

- **Precision@k/Recall@k**
  - define a threshold (list length) and count the "hits" proportion
- **Mean average precision (MAP)**
  - determine the position of each hit (e.g., 2,3,5)
  - calculate the average for all hits in the list
  - average over all recommendations
- **Mean reciprocal rank (MRR)**
  - assume that there is only one relevant item or only the first is important
  - it its position is K, the MRR is 1/K

Average precision

- Average precision (AP) is a ranked precision metric that places emphasis on highly ranked correct predictions (hits)
- Essentially it is the average of precision values determined after each successful prediction, i.e.

Rank	Hit?
1	
2	X
3	X
4	X
5	

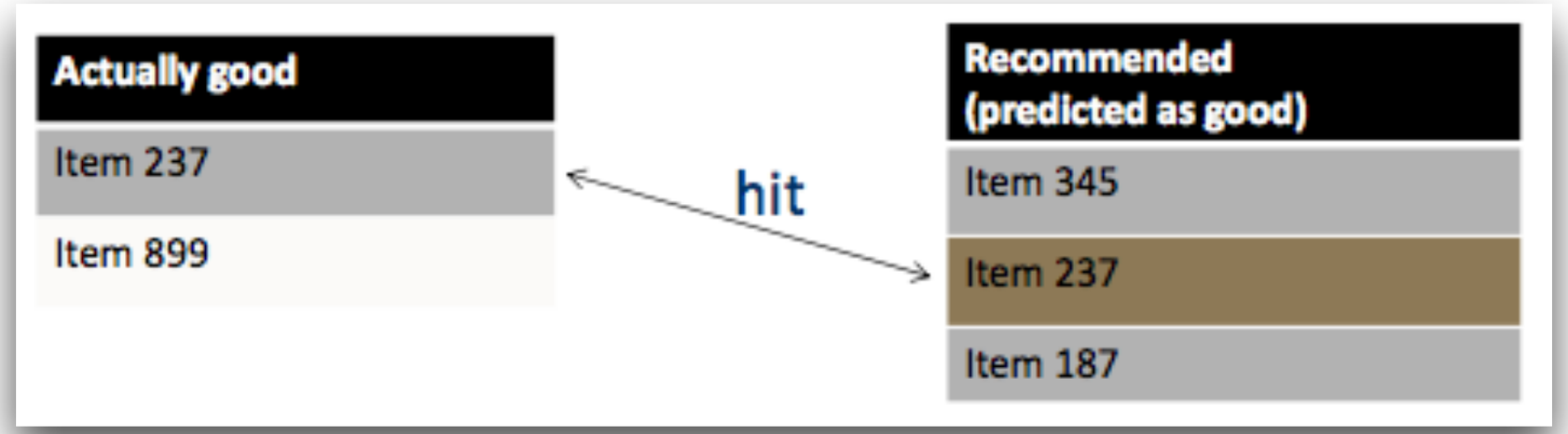
$$AP = \frac{1}{3}\left(\frac{1}{2} + \frac{2}{3} + \frac{3}{4}\right) = \frac{23}{36} \approx 0.639$$

$$AP = \frac{1}{3}\left(\frac{1}{1} + \frac{2}{4} + \frac{3}{5}\right) = \frac{21}{30} = 0.7$$

Rank	Hit?
1	X
2	
3	
4	X
5	X

Metrics: rank position matters

- For a user:



- **Rank metrics** extend recall and precision to take the positions of correct items in a ranked list into account
  - relevant items are more useful when they appear earlier in the recommendation list
  - nDCG, Lift index, Rank Score

Discounted cumulative gain (DCG)

- Concept of graded relevance
  - hits at the beginning count more (more "gain")
  - documents of higher relevance are more important
  - discounted gain at later positions
    - often an exponential decay (half life) is assumed
      - e.g., based on the log function
- given a rank position p, and the graded relevance "rel" of an item *i*

$$\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$$

- nDCG: Normalized value at length n
  - compare with "ideal" ranking

nDCG example

- There are 6 items to rank
- Relevance scores (0-3) scale:
  - 3,2,3,0,1,2
- DCG at 6:

$$\mathrm{DCG}_6 = rel_1 + \sum_{i=2}^{6} \frac{rel_i}{\log_2 i} = 3 + (2 + 1.892 + 0 + 0.431 + 0.774) = 8.10$$

- An ideal ordering IDCG:
  - 3,3,2,2,1,0 would lead to an DCG of 8.69
- The nDCG
  - DCG/IDCG: 8.10/8.69 = 0.932

Problem of the ground truth

- Often in Information Retrieval settings
  - set of target documents is labeled with ground truth
- In recommender systems
  - no rating available for most of the items
  - considering unrated items as irrelevant?
  - different ways of computing precision/recall
    - how to count the ranked elements with unknown ground truth

Error measures

- Mean absolute error (MAE)
- Root Mean Square Error (RMSE)

Dataset characteristics

- Natural datasets include historical interaction records of real users
  - explicit user ratings
  - datasets extracted from web server logs (implicit feedback)
- Sparsity of a dataset is derived from ratio of empty and total entries in the user-item matrix
  - Sparsity = 1 - |R| / (|I| x |U|)
  - R = ratings
  - I = items
  - U = users

The Netflix Prize setup

- Netflix competition
  - web-based movie rental and streaming
  - Prize of $1M 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 evaluation
    - quiz set - evaluates teams' submissions for leaderboard
    - test set - used by Netflix to determine winner

General methodology

- Setting to ensure internal validity
  - one randomly selected share of known ratings (**training set**) used as input to train the algorithm and build the model
  - model allows the system to compute recommendations at runtime
  - remaining share of withheld ratings (**testing set**) required as ground truth to evaluate the model's quality
  - to ensure the reliability of measurements the random split, model building and evaluation steps are repeated several times
- K-fold cross validation is a stratified random selection procedure
  - k disjunct fractions of known ratings with equal size are determined
  - k repetitions of the model building and evaluation steps, where each fraction is used exactly once as a testing set while the other fractions are used for training
  - setting k to 5 or 10 is popular

Analysis of results

- Are observed differences statistically meaningful or due to chance?
  - standard procedure for testing the statistical significance of two deviating metrics is the pairwise analysis of variance (ANOVA)
  - Null hypothesis H0: observed differences have been due to chance
  - if outcome of test statistics rejects H0, significance of findings can be reported

Beyond accuracy - more quality metrics for recommenders

- **Coverage**
  - for how many users can we make recommendations?
  - how many catalog items are ever recommended?
- **Diversity & novelty**
  - avoiding monotone lists, discover new (families of) items
- **Serendipity**
  - unexpected and surprising items might be valuable
- **Familiarity**
  - give the user the impression of understanding his/her needs
- **Biases**
  - does the recommender only recommend popular items and blockbusters?
- …

Hybrid recommenders

Hybrid RecSys

- Collaborative and content-based filtering
  - all pieces of information can be relevant in real-world advisory or recommendation scenarios
  - but all have their shortcomings
- Idea of crossing two (or more) species/implementations
  - hybrida: denotes an object made by combining two different elements
  - avoid some of the shortcomings
  - reach desirable properties not (or only inconsistently) present in parent individuals

Limitations and success of hybridization strategies

- Only few works that compare strategies from the meta-perspective
  - several datasets do not allow to compare different recommendation paradigms
- Netflix competition: "stacking" recomender systems
  - weighted design based on > 100 predictors - recommendation functions
  - adaptive switching of weights based on user model parameters

Recent research topics

- **Human decision making**
  - take into account insights from consumer psychology
  - phenomena like choice overload, (ir)rationality of human decision making processes, preference construction and stability
- **Sales psychology**
  - context effects
    - how to present items
    - primacy/recency effects (list positions matter)
    - decoy effects
  - trust
- **Behavioural patterns**
  - maximizer / satisfizer

Recent research topics

- Popularity and concentration biases of algorithms
- Short-term user interests
- Explanation interfaces for recommenders
- Multi-criteria recommender systems
- Music playlist generation (music recommendation)
  - discussion of limitations of current evaluation measures
  - analysis of what makes a good playlist
- Novel applications of recommender systems
  - process modeling, drug discovery

Recommender Systems