Intelligent Interactive Systems: Recommender Systems

Mubashir Ali
m.ali.16@bham.ac.uk

Learning Outcomes

By the end of this lesson, you should be able to:

1. What is recommender system?
2. Difference between recommender system and information retrieval
3. Collaborative filtering
4. Content-based filtering
5. Hybrid methods
6. Knowledge-based system
7. Evaluation of recommender system
8. Challenges and limitations
9. Current trends and research areas

Recommender System

- A recommender system is any system that filters content for a user based on some information about the user.

- Systems for recommending items (e.g. books, movies, CD's, web pages, newsgroup messages) to users based on examples of their preferences.

- Recommender system aim to predict users' interests and recommend product items that quite likely are interesting for them.

Recommender System

- Recommender Systems are software agents that elicit the interests and preferences of individual users and make recommendations accordingly. They have the potential to support and improve the quality of the decisions users make while searching for and selecting products online.

(Xiao & Benbasat 2007)

Motivations

- Information overloaded
  - Many choices available
- Recommender system
  - Provide aid
- Personalization
  - Tailored recommendations
- Data-driven decision making
  - Valuable data on user preferences and behaviors
- Increased sales
  - E-commerce

Motivations

- What recommender systems do you know?
- What recommender system would you like to have?

Examples of Applications

- Movies, online videos
- Music
- Books
- Software (apps)
- Products in general
- People (friends)
- Services (restaurants, accommodation, ….)
- Research articles
- Jokes
- Many more

Good Recommendations

What are good recommendations?

Try to think about different criteria / aspects.

Purpose and Success Criteria (1)

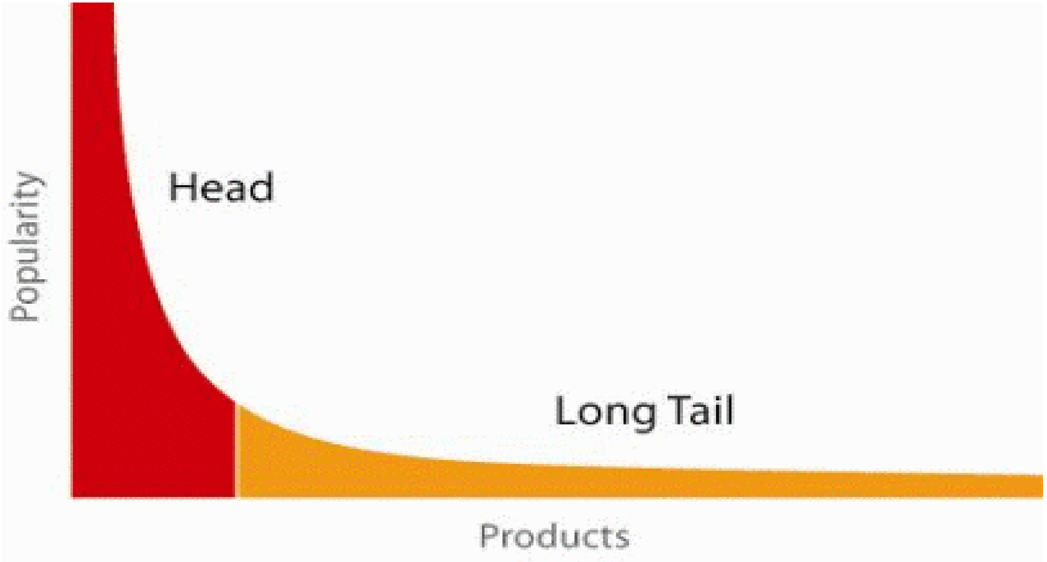
- Different perspectives / aspects
  - Depends on the domain and purpose
  - No holistic evaluation scenario exists

- Retrieval perspective
  - Reduce search cost
  - Provide “correct” proposals
  - 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 items from the long tail



Purpose and Success Criteria (2)

- Prediction perspectives
  - Predict to what degree users like an item
  - Most popular evaluation scenario in research
- Interaction perspective
  - Give user a “good feeling”
  - Educate users about the product domain
- Finally, conversion perspective
  - Commercial situations
  - Increase “hit”, “clickthrough” rates
  - Optimize sales margins and profit

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
  - Good items
  - Relevance score
  - Ranking
  - User interests

RecSys and Information Retrieval

- Information Retrieval (IR) is the activity of obtaining information resources relevant to an information need from a collection of information resources. *(Wikipedia)*

- The goal of a Recommender System is to generate meaningful recommendations to a collection of users for items or products that might interest them. *(Melville, Sindhwani)*

RecSys and Information Retrieval

- *RecSys and IR closely connected (many similar or analogical techniques)*
- *Different goals:*
    - *IR - I know what I'm looking for"*
    - *RecSys - I'm not sure what I'm looking for"*

The Impact of RecSys

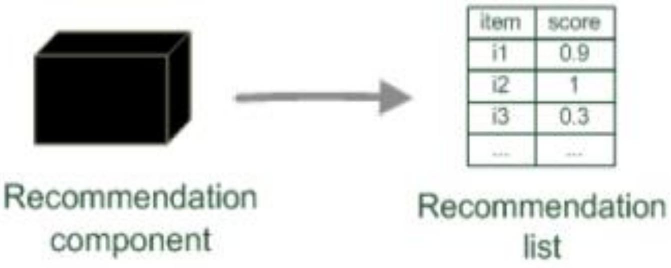
- 35% of the purchases on Amazon are the result of their recommender system, according to McKinsey.
- Recommendations are responsible for 70% of the time people spend watching videos on YouTube.
- 75% of what people are watching on Netflix comes from recommendations, according to McKinsey.
- Google News recommendations led to a 38% increase in clickthrough.

Netflix Prize

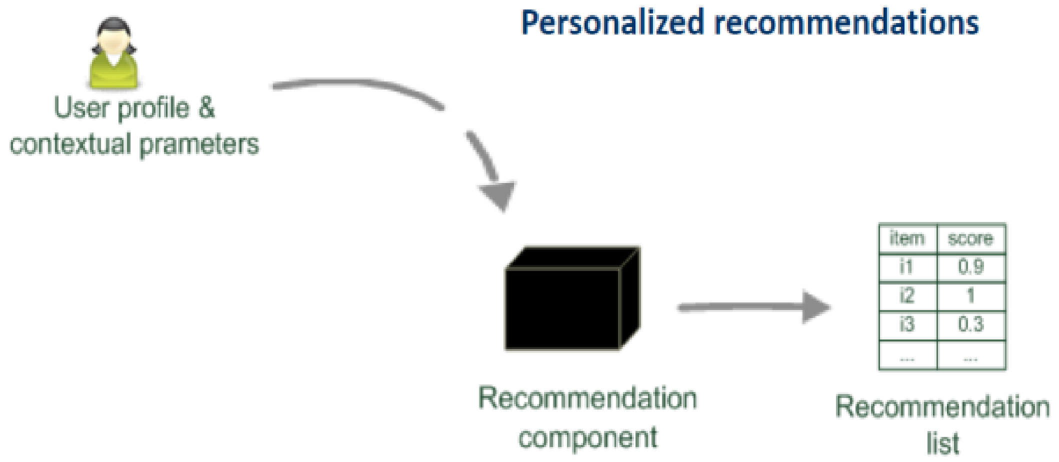
- Netflix - video rental company
- contest 2006: 10% improvement of the quality of recommendations
- collaborative filtering
- prize: 1 million dollars
- data: user ID, movie ID, time, rating

Types of Recommender Systems

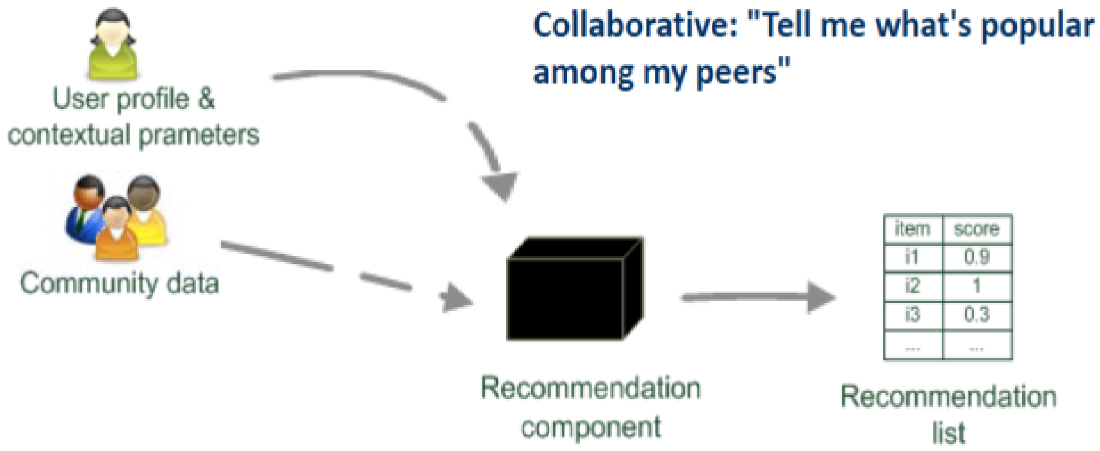
Recommender systems reduce information overload by estimating relevance



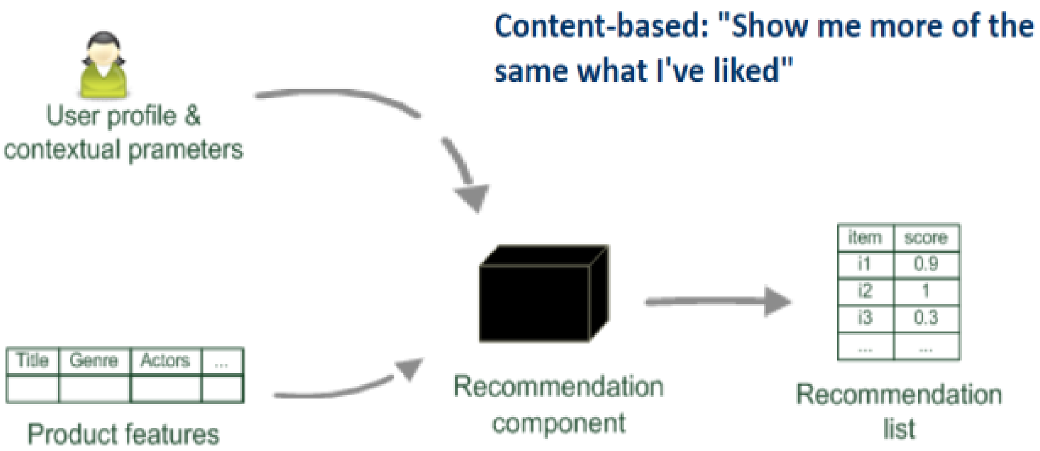
Types of Recommender Systems



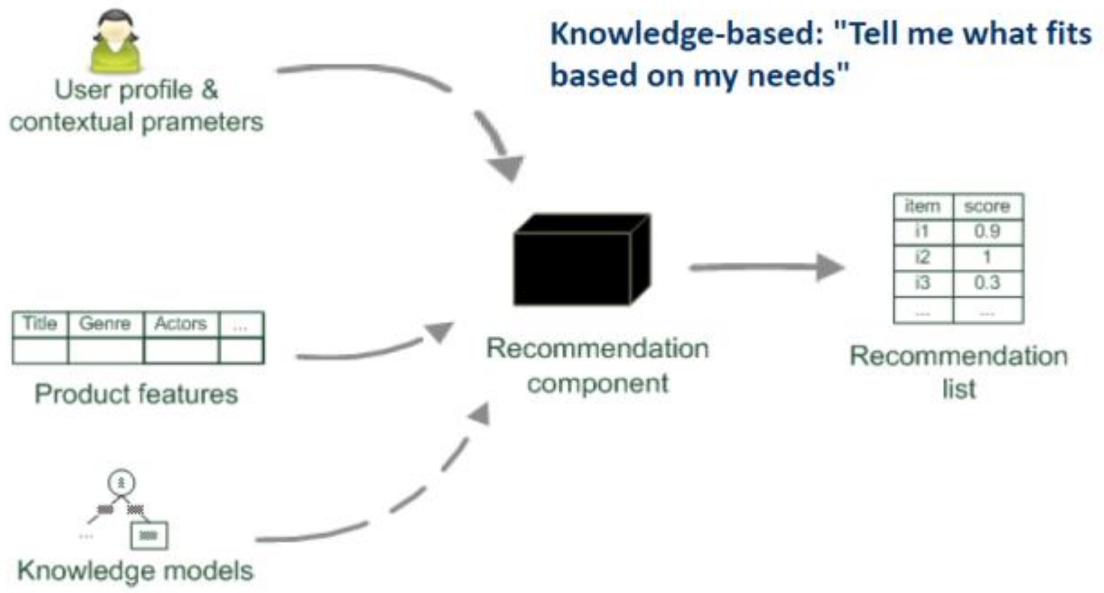
Types of Recommender Systems



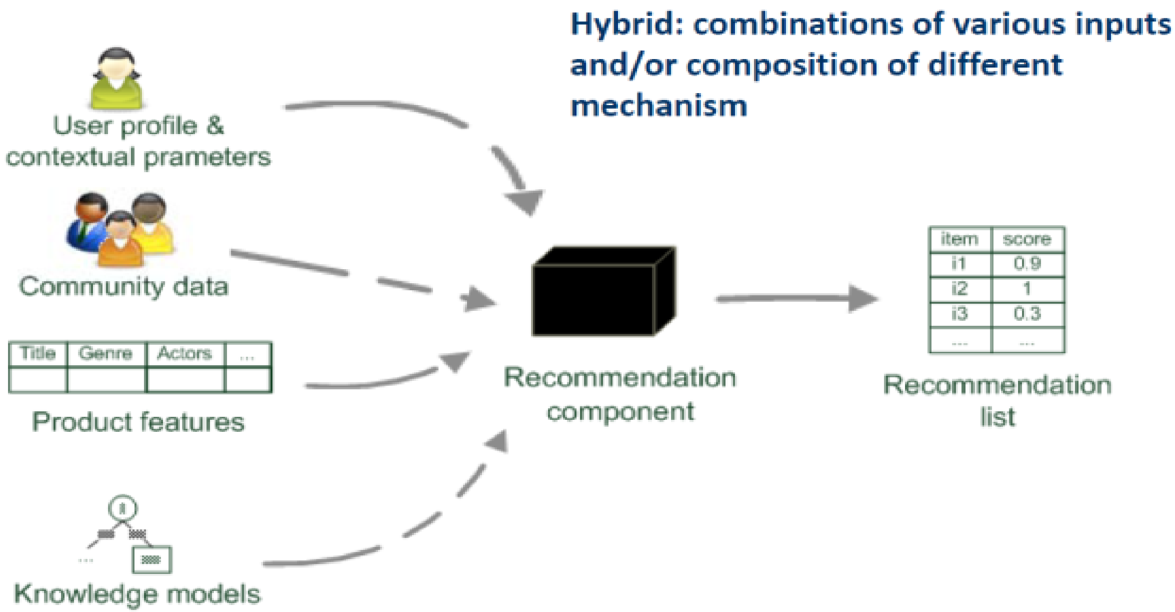
Types of Recommender Systems



Types of Recommender Systems



Types of Recommender Systems



Collaborative Filtering

Collaborative Filtering (CF)

- The most popular 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

Collaborative Filtering (CF)

- Input
  - Only a matrix of given user–item ratings
- Output types
  - A (numerical) prediction indicating to what degree the current user will like or dislike a certain item
  - A top-N list of recommended items

Collaborative Filtering (CF)

- There are two main approaches to collaborative filtering:
  - User-based collaborative filtering
    - This approach looks for users who are similar to the active user based on their ratings of items.
    - Assumption: users with similar preferences will like similar items
  - Item-based collaborative filtering
    - This approach looks for items that are similar to the items that the active user has already rated highly.
    - Assumption: users who have liked similar items in the past will also like similar items in the future

User-based collaborative filtering

User-Based Collaborative Filtering (UB-CF)

- The basic techniques
  - Given an "active user" (Alice) and an item i not yet seen by Alice
    - find a set of users (peers/nearest neighbors) 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
- Basic assumption and idea
  - If users had similar tastes in the past they will have similar tastes in the future
  - User preferences remain stable and consistent over time

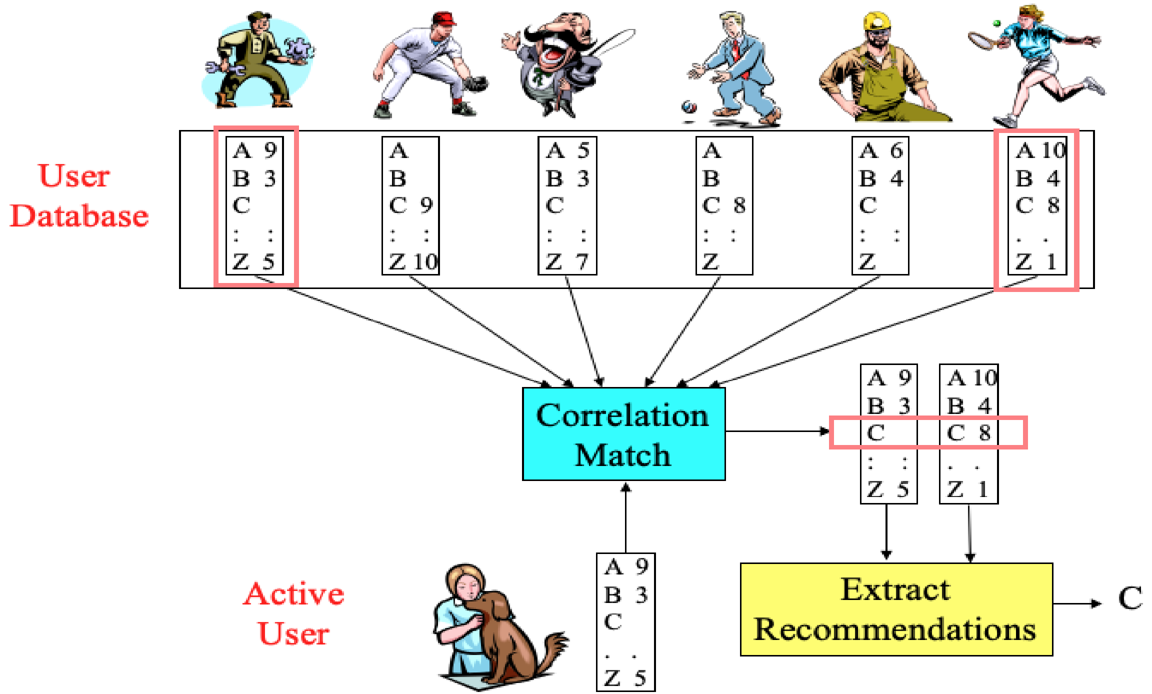
User-based Collaborative Filtering (UB-CF)

- Example
  - A database of ratings of the current user, Alice, and some other users is given:

	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

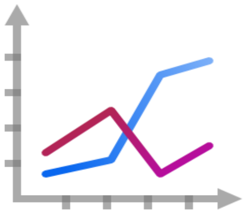
  - Determine whether Alice will like or dislike Item5, which Alice has not yet rated or seen

User-Based Collaborative Filtering (UB-CF)



User-Based Collaborative Filtering (UB-CF)

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



	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 very 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

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

Measuring user similarity

- A very 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

	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

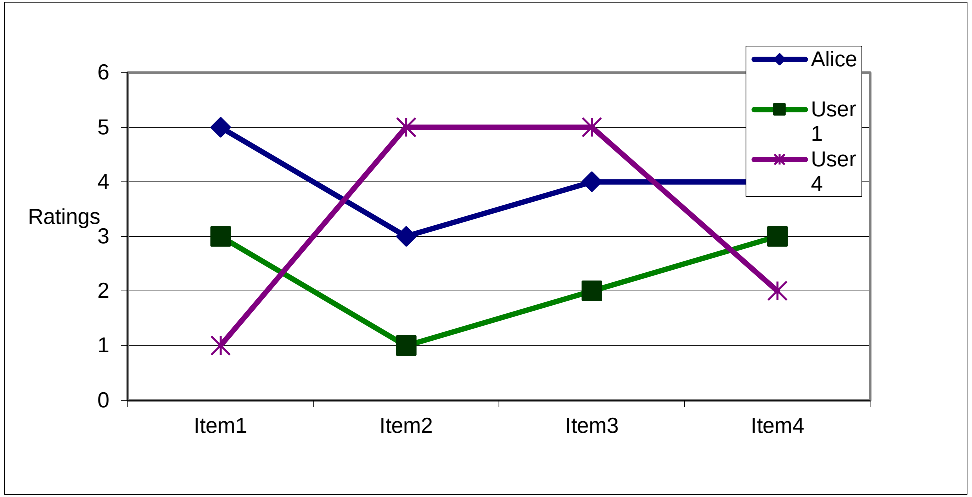
sim = 0.00

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:

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

  - Calculate the similarity between the active user ‘a’ and each user ‘b’ in the neighborhood ‘N’.
  - For each user ‘b’ in the neighborhood ‘N’, calculate the weighted sum of the differences between $\boldsymbol{r_{b,p}}$ and $\overline{\boldsymbol{r_b}}$ based on the sim(a, b).
  - Add the average rating $\overline{\boldsymbol{r_a}}$ to the weighted sum from step 2 to get the predicted rating for item ‘p’ by user ‘a’.

Item-based collaborative filtering

Item-based Collaborating Filtering

- 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
- 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

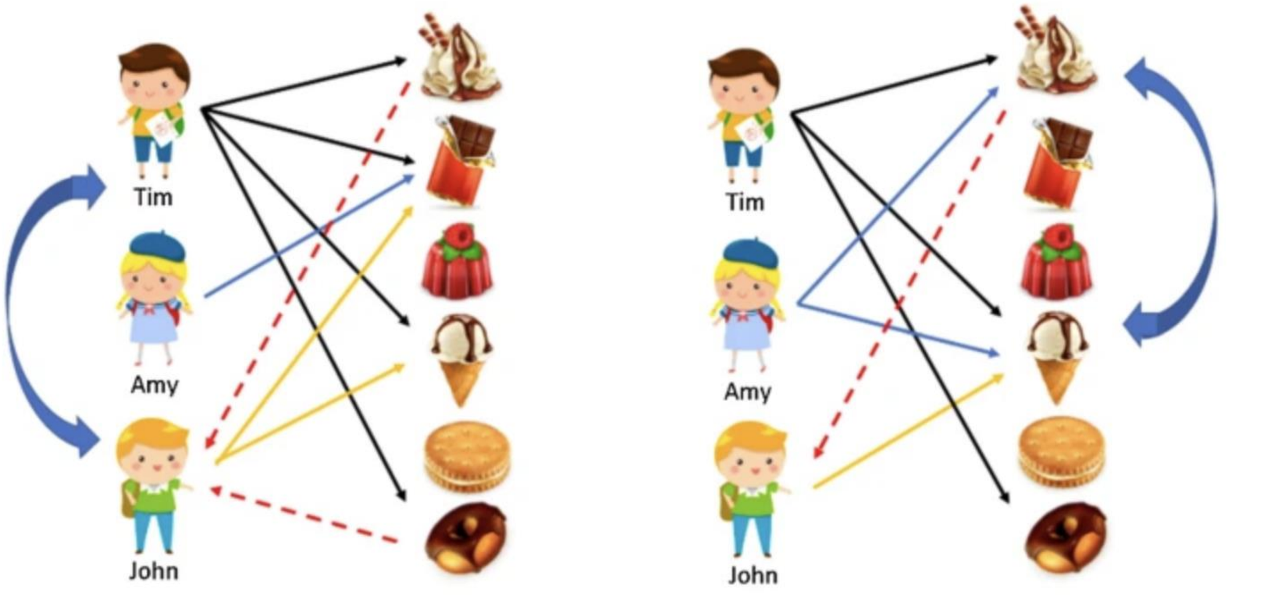
$$sim(\vec{a}, \vec{b}) = \frac{\sum_{u \in U}(r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U}(r_{u,a} - \overline{r_u})^2}\sqrt{\sum_{u \in U}(r_{u,b} - \overline{r_u})^2}}$$

Making predictions

- A common prediction function:

$$\boldsymbol{pred(u,p)} = \frac{\sum_{i \in ratedItem(u)} \boldsymbol{sim(i,p) * r_{u,i}}}{\sum_{i \in ratedItem(u)} \boldsymbol{sim(i,p)}}$$

    - $pred(u,p)$: The predicted rating of user u for item p.
    - $sim(i,p)$: The similarity between item i and item p.
    - $r(u,i)$: The rating that user u gave to item i.
    - $ratedItem(u)$: The set of items rated by user u.
- For each item i that user u has rated (i.e., $i \in ratedItem(u)$), calculate the similarity between item i and the target item p (i.e., $sim(i,p)$).
- Multiply each similarity $sim(i,p)$ by the rating $r(u,i)$ that user u gave to item i.
- Sum up the weighted ratings for all items that user u has rated.
- Divide the sum by the sum of the similarities for all items that user u has rated



(a) User-based filtering (b) Item-based filtering

Ratting

- recommender systems (particularly collaborative filtering) rely on user “ratings”
- rating of item ~ how much the user likes the item
- many different forms of ratings
- what kinds of ratings do you know (can you imagine)?
- what are their advantages and disadvantages?

Explicit Ratting

- Probably the most precise ratings
- Likert scale (5 stars), like/dislike
- Main problems:
  - users not always willing to rate many items
  - how to stimulate users to rate more items?

Implicit Ratting

- Click through rate, buying an item, visiting a page, viewing a video, dwell time.
- easier to collect, less precise

- Main problem:
  - One cannot be sure whether the user behavior is correctly interpreted
  - For example, a user might not like all the books he or she has bought; the user also might have bought a book for someone else

- Implicit ratings can be used in addition to explicit ones; question of correctness of interpretation

CF Issues

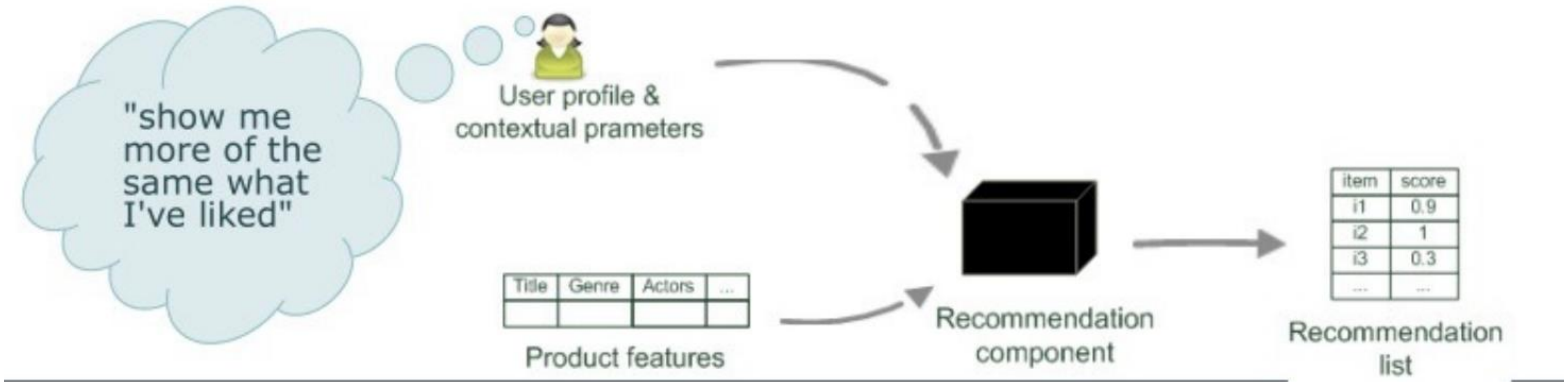
- Cold start problem
  - How to recommend new items? What to recommend to new users?
- Data sparsity
  - Users interacted with a small fraction of the available items
- Scalability
  - Computationally expensive when the numbers of users and items grows
- Popularity bias
  - Recommend popular items more frequently
- Demographic bias
  - Recommendations are biased towards certain groups of users (young adults)

Content-based recommendation

Content-based Recommendation

- While CF – methods do not require any information about the items,
  - it might be reasonable to exploit such information; and
  - recommend fantasy novels to people who liked fantasy novels in the past
- What do we need:
  - some information about the available items such as the genre ("content")
  - some sort of user profile describing what the user likes (the preferences)
- The task:
  - learn user preferences
  - locate/recommend items that are "similar" to the user preferences

Content-based Recommendation



What is the “Content”

- Most CB-recommendation techniques were applied to recommending text documents.
  - Like web pages or newsgroup messages for example.
- 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

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

  - Unstructured: free-text description

Content representation and item similarities

- **Item representation**

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

- **User profile**

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

$keywords(b_j)$ describes Book b_j with a set of keywords

- **Simple approach**
  - Compute the similarity of an unseen item with the user profile based on the keyword overlap (e.g. using the Dice coefficient)
  - 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
  - in particular when automatically extracted as
    - 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
    - weighted term vector
  - TF: Measures, how frequently 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: Measure, how important a term is across all the documents in a collection

TF-IDF II

- Given a keyword i and a document j
- $TF(i,j)$ $\quad TF = \frac{\text{number of times the term appears in the document}}{\text{total number of terms in the document}}$
  - term frequency of keyword i in document j
- $IDF(i)$
  - inverse document frequency calculated as $\boldsymbol{IDF(i) = log\frac{N}{n(i)}}$
    - N : number of all recommendable documents
    - $n(i)$: number of documents from N in which keyword i appears
- $TF - IDF$
  - is calculated as: $\boldsymbol{TF\text{-}IDF(i,j) = TF(i,j) * IDF(i)}$

Example TF-IDF representation

- Term frequency
  - Each document is a $\boxed{\text{count vector}}$ in $\mathbb{N}^{|v|}$

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	1.51	0	3	5	5	1
worser	1.37	0	1	1	1	0

Vector v with dimension $|v| = 7$

Example TF-IDF representation

- Combined TF-IDF weights
  - Each document is now represented by a real-valued vector of $TF\text{-}IDF$ weights $\in \mathbb{R}^{|v|}$

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4					
Caesar	23					
Calpurnia	0					
Cleopatra	57					
mercy	1.5					
worser	1.3					

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Improving the vector space model

- Vectors are usually long and sparse
  - 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. ”going” → "go", "stemming" → "stem", …
- size cut-offs
  - only use top n most representative words to remove "noise" from data
  - e.g. use top 100 words

Improving the vector space model II

- Use lexical knowledge, use more elaborate methods for feature selection
  - Remove words that are not relevant in the domain
- Detection of phrases as terms
  - More descriptive for a text than 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

Recommending items – nearest neighbors

- Simple method: nearest neighbors
  - Given a set of documents D already rated by the user (like/dislike)
    - Either explicitly via user interface
    - Or implicitly by monitoring user's behavior
  - Find the n nearest neighbors of a not-yet-seen item i in D
    - Use similarity measures (like cosine similarity) to capture similarity of two documents
  - 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 use
- Good to model short-term interests / follow-up stories
- Used in combination with method to model long-term preferences

Probabilistic methods

- Recommendation as classical text classification problem
  - long history of using probabilistic methods
- Simple approach
  - 2 classes: hot/cold
  - simple Boolean document representation
  - calculate probability that document is hot/cold based on Bayes theorem

Doc-ID	recommender	intelligent	learning	school	Label
1	1	1	1	0	1
2	0	0	1	1	0
3	1	1	0	0	1
4	1	0	1	1	1
5	0	0	0	1	0
6	1	1	0	0	?

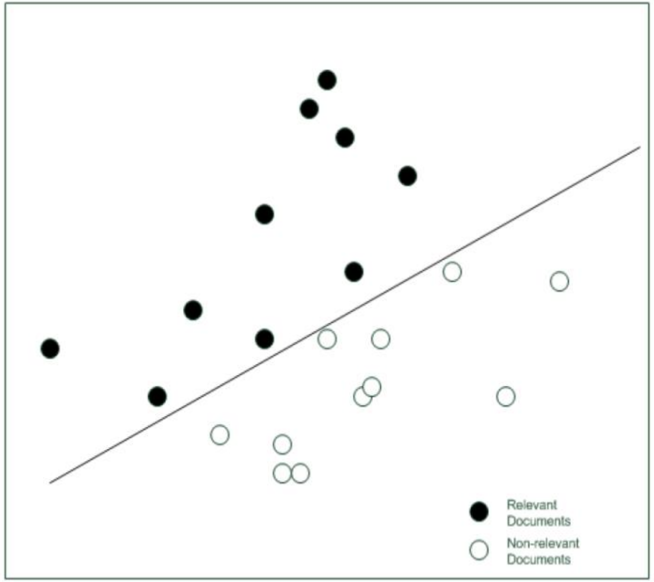
$$
\begin{aligned}
& P(X|Label = 1) \\
&= P(recommender = 1|Label = 1) \\
&\times P(intelligent = 1|Label = 1) \\
&\times P(learning = 0|Label = 1) \\
&\times P(school = 0|Label = 1) \\
&= {}^{3}/_{3} \times {}^{2}/_{3} \times {}^{1}/_{3} \times {}^{2}/_{3} \approx 0.149
\end{aligned}
$$

Linear classifier

- A simplified classifier with only two dimensions can be represented by a line
- The line has the form $\boldsymbol{w_1 x_1 + w_2 x_2 = b}$
  - x_1 and x_2 correspond to the vector representation of a document (using e.g. TF-IDF weights)
  - w_1, w_2 and b are parameters to be learned
  - Classification of a document based on checking

$$w_1 x_1 + w_2 x_2 > b$$

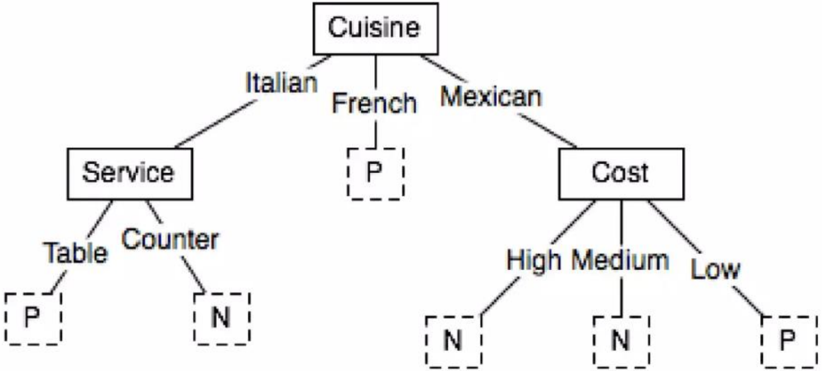
- In n-dimensional space the classification function is $\boldsymbol{\vec{w}^T \vec{x} = b}$
- Other classification learning algorithm
  - Naïve bayes, Rocchio method, Windrow-Hoff algorithms, Support Vector Machine



Decision Tree and Rule Induction

- Given the history of user's interests as training data, build a decision tree which represents the user's profile of interest.

Cuisine	Service	Cost	Rating
Italian	Counter	Low	Negative
French	Table	Med	Positive
French	Counter	Low	Positive
...	...	...	...



- Will the user like an inexpensive Mexican restaurant?

Decision Tree and Rule Induction

- Well-suited for structured data
- In unstructured data, the number of attributes become too enormous and consequently, the tree becomes too large to provide sufficient performance.
- Use meta features like author name, genre, …. Instead of TF-IDF representation
- Rule induction:
  - Built on RIPPER algorithm
- Main advantages of these decision models:
  - Inferred decision rules serve as basis for generating explanations or recommendation
  - Existing domain knowledge can be incorporated in models

Limitations of Content-based recommendation methods

- Keywords alone may not be sufficient to judge quality/relevance of a document or web page
  - up-to-date-ness, usability, aesthetics, writing style
  - content may also be limited / too short
  - content may not be automatically extractable (multimedia)
- Ramp-up phase required
  - Some training data is still required
  - Web 2.0: Use other sources to learn the user preferences
- Overspecialization
  - Algorithms tend to propose "more of the same"
  - Or: too similar news items

Current Trends in Recommender Systems

- Personalization
  - focusing on personalization, tailoring recommendations
- Explainable AI
  - recommender systems should be transparent and explainable
- Context-Aware Recommendations
  - incorporating contextual information such as location, time, and social context
- Multi-Stakeholder Recommendations
  - users, content creators, and platform owners
- Ethical Considerations
  - concerns around data privacy, biases, and algorithmic fairness

Research Areas in Recommender Systems

- Reinforcement Learning for Recommendations
- Long-Term User Modeling
- Hybrid Recommender Systems
- Novel Recommendation Paradigms
- Fairness and Bias in Recommendations