

Seminars in Database

Prerequisites – 2nd

F.Ricci

Content

- Information Retrieval
- Recommender Systems
 - Collaborative Filtering

Basic Concepts in Information Retrieval

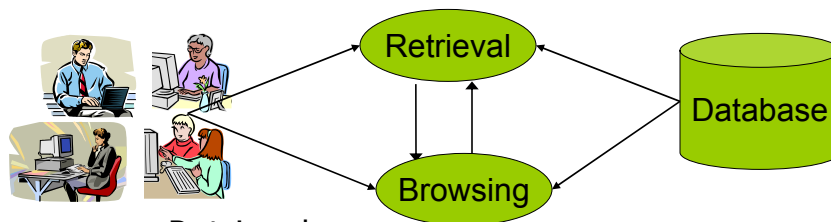
- Information Retrieval (IR) deals with the representation, storage and organization of **unstructured** data
- Its mission is to assist in information “discovery” (more exactly information search)
- 2 main discovery paradigms:

Search and Browse

- *Thanks to Aya Sofer and David Carmel (IBM Haifa) for these IR slides.*

Basic Concepts

- The User Task



- Retrieval
 - Search for particular information
 - Usually focused and purposeful
- Browsing
 - General looking around for information
 - For example: Asia-> Thailand -> Phuket -> Tsunami

Data- vs Information- Retrieval (from CJ Risjbergen)

	DR	IR
Matching	exact	partial/best
Inference Model	deduction	induction
Query Language	artificial	natural
Query Specification	complete	incomplete
Items wanted	matching	relevant
Error Response	sensitive	insensitive

Web search

- The Web is not a DB
 - no structure
 - can be seen as a flat collection of documents (What search engines did at the beginning, e.g., AltaVista)
- Yet the Web has a meta-structure:
 - it is a graph
 - using the information embedded in the graph led to a major breakthrough in IR (e.g., PageRank → Google)

Search: The Basic Concepts

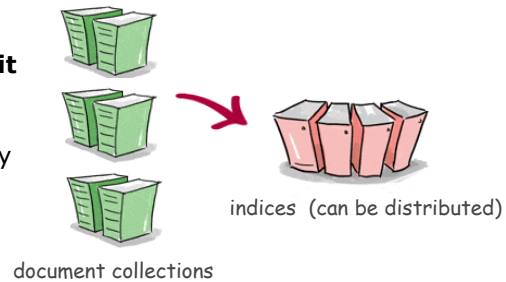
- The user has an information need, that is expressed as a free-text query
- The query is a “document”, to be compared to a collection of documents
- **Effectiveness vs Efficiency**
- How to compare documents? Similarity metrics needed!
- How to avoid doing a sequential search? Can we search in parallel in a set of servers?

Basic Concepts

- Two Main Stages
 - Indexing process (build knowledge base)
 - Involves pre-processing and storing of information in a repository
 - Retrieval process (search knowledge base)
 - Involves issuing a query, accessing the repository to find candidate documents most similar to query, and browse them
 - Basic object is a document

Search Preprocessing Stage - Indexing

- Analyze the distribution of indexing units (words) within a collection of documents
- The **inverted index**, maps every **indexing unit** (word) to a **list of postings** (documents ids associated with frequency of occurrence, location information, etc...)

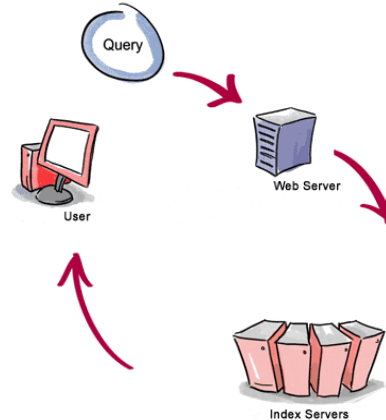


Inverted Index

- Given the texts $T_0 = \text{"it is what it is"}$, $T_1 = \text{"what is it"}$ and $T_2 = \text{"it is a banana"}$
- We have the following *full inverted index*:
 - "a": $\{(2, 2)\}$
 - "banana": $\{(2, 3)\}$
 - "is": $\{(0, 1), (0, 4), \mathbf{(1, 1)}, (2, 1)\}$
 - "it": $\{(0, 0), (0, 3), \mathbf{(1, 2)}, (2, 0)\}$
 - "what": $\{(0, 2), \mathbf{(1, 0)}\}$
- In bold face the postings associated to document T_1

The Actual Search Process

- At retrieval time, the query is indexed. Postings are fetched, and analyzed to return the most similar documents
- Effectivity constraint: *"The entire process should not take more than a second per query"*



Classical IR Models

- **Boolean model** (Does the term occur in document?)
 - Simple model based on set theory
 - Queries are specified as Boolean expressions
- **Vector space model** (Similarity between vectors)
 - Query, Documents are represented as vectors in a N-dimensional space, N is the number of terms in the corpus
 - Find documents most similar to the query in that space
- **Probabilistic model** (Find *probabilistically* best answer)
 - Goal: model IR with probabilistic framework
 - Given a user's query find out the document with the higher probability to be relevant to the query.

Text Preprocessing Techniques

- Goal: construct a canonical form of the document text called *document profile*
- Stages
 - Sentence splitting – sentence separation
 - Tokenization – word separation
 - Normalization – changing terms to a standard form (e.g., lowercase)
 - Stop-word filtering: ignores frequent terms (e.g. *is, the, as, I, at, in, to...*)
 - Lemmatization – reducing terms to their base form
 - Map between documents and indexing units

Retrieving – Query Evaluation, Ranking

Query evaluation steps:

1. Input query
2. Parse query
3. Lexicon lookup
4. Get posting from inverted index
5. Weights accumulator
6. Sort by weights

Evaluation Criteria

□ **Effectiveness**

- How precise is the answer set to the information needs?

□ **Efficiency**

- Retrieval time, indexing time, index size?

□ **Usability and Personalization**

- Learnability, novice use, expert use

Web IR- IR on the Web

□ **First Generation**

- Classical approach
- Informational: IR/DB techniques on page content.
E.g., Lycos, Excite, AltaVista

□ **Second Generation**

- Web as a graph
- Navigational: use off-page Web specific data – links topology. E.g., Google

□ **Third Generation**

- Open research
- A lot of business potential, “monetization of infomediary role”, matching services

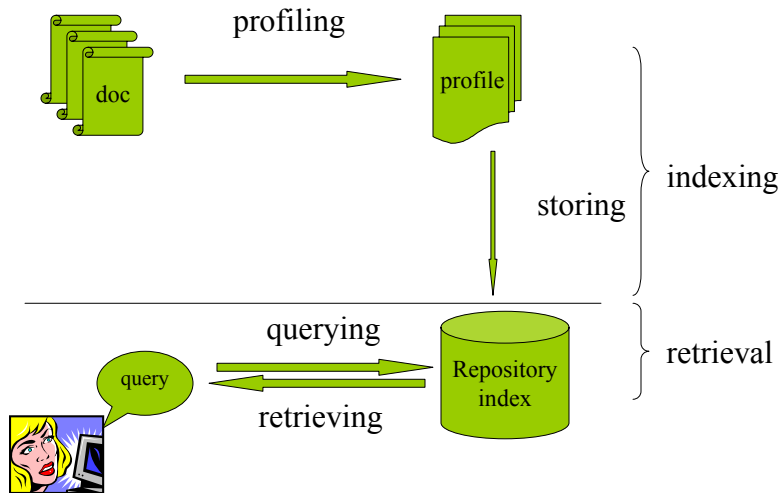
Problems with Using IR for Web

- Very large and heterogeneous collection
- Very short queries
- Unsophisticated users
- Difficult to judge relevance and to rank results
- Synonymy and ambiguity
- Authorship styles
- Search engine persuasion, keyword *stuffing* (a web page is loaded with keywords in the meta tags or in content)

Indexing/Retrieval

- Information Retrieval consists of 2 main stages
 - Indexing - pre-processing and storing of information into a repository (**an Index**)
 - Retrieval - issuing a query, accessing the index to find documents **RELEVANT** to the query

Typical IR system



Basic Concepts

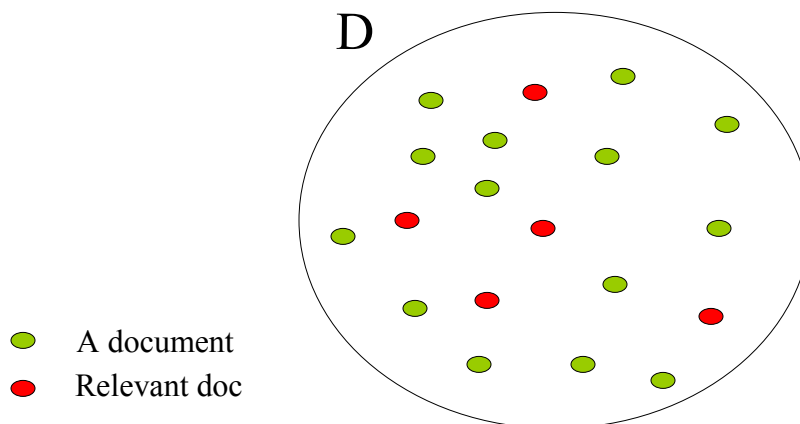
- **Document**
 - any piece of information (book, article, database record, image)
 - usually textual data
- **Query**
 - a text representing the user's information need
- **Relevance**
 - A relation (binary or numeric) between documents and queries $R(d,q)$

How to measure Document Relevance

- Although Relevance is a basic concept in IR, *there is still no unique definition*
- **Difficulties**
 - User dependent (experienced vs. novice)
 - Time dependent (e.g., news topics)
 - Geography dependent (Paris-FR vs. Paris-USA)
- **Simplified (binary) assumption**
 - D – the set of all documents in the world,
 - Q- the set of all queries in the world,
 - R: $D \times Q \rightarrow \{0,1\}$ is well defined:
d is “relevant” to q if $R(d,q) = 1$

IR main task

- retrieve the “relevant” documents to the user’s query



Index building: Text Profiling

- Both documents and queries are profiled to generate a canonical representation
- Profile is usually based on the set of indexing units in the text
- Indexing units are generally representative terms in the text
 - How to select representative terms?
 - For the moment, let's say all the words in the given document/query

In the beginning
God created the
heaven and the
earth. And the
earth was without
form and void.

...,and,,beginning,,created,,earth,,form,,god,,heaven,,in,,the,,void,,was,,without
.., 3 ,, 1 ,, 1 ,, 2 ,, 1 ,, 1 ,, 1 ,, 1 ,, 3 ,, 1 ,, 1 ,, 1

Let us formalize a bit

- Given a collection of documents (a corpus)
 - all the terms in the collection are labeled as: $T = \{t_1, t_2, \dots, t_N\}$.
 - profile of a document d_i is a vector of size N : $d_i \rightarrow (w_{i1}, w_{i2}, \dots, w_{iN})$

$$w_{ij} = \begin{cases} 0 & t_i \text{ does not appear in } d_j \\ \text{num. of appears of } t_i \text{ in } d_j & \text{otherwise} \end{cases}$$

Index Representation - sparse matrix

A(t,d)	d ₁	d ₂	...	d _M
t ₁	w ₁₁	w ₁₂		w _{1M}
t ₂				
t _N	w _{N1}			w _{NM}

Using the index for Relevance Retrieval

- **Assumption:** a document not containing any query term is not relevant (*think about that! Is it true?*)
- Given a simple query of one term $q = \{t_i\}$
- To find the relevant documents
 - 1. Retrieve all the documents d_j with $w_{ij} > 0$
 - 2. Sort them in decreasing order
 - 3. Return the sorted list of "relevant" documents to the user
- In general: given a user's query $q = \{t_1, t_2, \dots, t_k\}$ return the sorted list of documents that contain at least one of the query terms.

The Boolean Model

- Simple model based on set theory
- Queries are specified as Boolean expressions
 - Indexing units are words
 - Boolean Operator: OR, AND, NOT
 - Example:
q = "java" AND "compilers" AND ("unix" OR "linux")
- **Relevance:** A document is relevant to the query if it satisfies the Boolean expression of the query.

Boolean Model- Example

A(t,d)	d ₁	d ₂	d ₃	d ₄	d ₅
a	1	1	1	0	1
b	0	1	0	1	1
c	0	0	1	0	1

$$q = a \text{ AND } (b \text{ OR } (\text{NOT } c))$$

Search the other way around

□ a -> D1, D2, D3, D5

□ b -> D2, D4, D5

□ c -> D3, D5

□ a -> 1, 1, 1, 0, 1

□ b -> 0, 1, 0, 1, 1

□ NOT c -> 1, 1, 0, 1, 0

} 1, 1, 0, 1, 1 }
OR }
1, 1, 0, 0, 1 }
AND

$q = a \text{ AND } (b \text{ OR } (\text{NOT } c))$

Results: d1, d2, d5

Boolean Model – pros & cons

□ **Pros:**

- Fast (bitmap vector operations)
- Binary decision (doc is “relevant” or not)

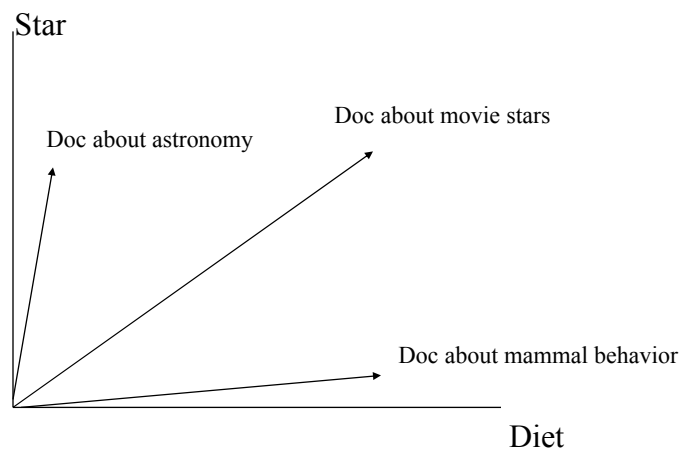
□ **Cons:**

- Binary decision – How to rank?
 - Query can fail (no result)
 - Who speaks Boolean?
- Still very popular by commercial IR systems

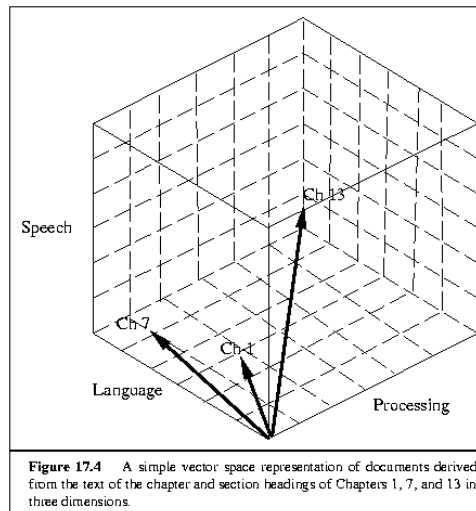
Vector Model

- Documents are represented as vectors in a N-dimensional space
 - N is the number of terms in the corpus
- Query is a document like any other document
- Relevance – measured by **similarity** between vectors:
 - A document is relevant to the query if its vector is similar to the vector of the query.

Documents as Vectors



Documents in 3D Space



Vector-space Model

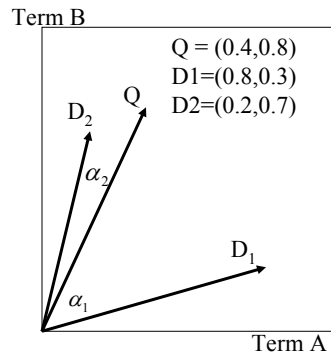
- ▣ Represent the query as vector in the same document vector-space
- ▣ Relevance is measured by similarity
- ▣ Measure the cosines of the angle between doc-vectors and the query vector

$$Sim(q, d) = \frac{q \cdot d}{|q| \cdot |d|} = \frac{\sum_i w_{iq} w_{id}}{\sqrt{\sum_i w_{iq}^2} \sqrt{\sum_i w_{id}^2}}$$

for N=2

$$= \frac{q_x d_x + q_y d_y}{\sqrt{(q_x^2 + q_y^2)(d_x^2 + d_y^2)}}$$

Example



$$Sim(q, d) = \frac{q \bullet d}{|q| \bullet |d|} = \frac{\sum_i w_{iq} w_{id}}{\sqrt{\sum_i w_{iq}^2 \sum_i w_{id}^2}}$$

$$sim(q, d_1) = \frac{(0.4 \cdot 0.8) + (0.8 \cdot 0.3)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.8)^2 + (0.3)^2]}}$$

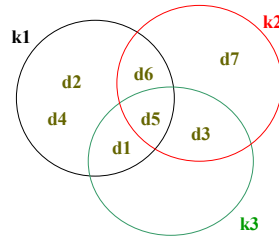
$$= \frac{0.56}{\sqrt{0.58}} = 0.74$$

$$sim(q, d_2) = \frac{(0.4 \cdot 0.2) + (0.8 \cdot 0.7)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.2)^2 + (0.7)^2]}}$$

$$= \frac{0.64}{\sqrt{0.42}} = 0.98$$

Example 2

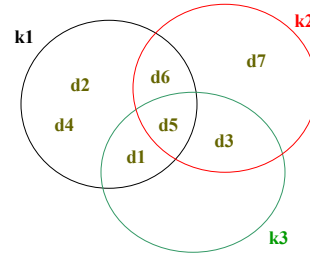
Query: (k1 k2 k3)



	k1	k2	k3	q • dj
d1	1	0	1	2
d2	1	0	0	1
d3	0	1	1	2
d4	1	0	0	1
d5	1	1	1	3
d6	1	1	0	2
d7	0	1	0	1
q	1	1	1	

Binary weights for the sake of the simplicity

Example 3



	k1	k2	k3	q • d _j
d1	1	0	1	4
d2	1	0	0	1
d3	0	1	1	5
d4	1	0	0	1
d5	1	1	1	6
d6	1	1	0	3
d7	0	1	0	2
q	1	2	3	

How to determine the $w(t,d)$ weights?

- Binary weights:
 - $w_{ij} = 1$ if document d_j contains term t_i , otherwise 0.
 - Actually it is the Boolean model
- Term Frequency (tf):
 - $w_{ij} =$ (number of occurrences of t_i in d_j)
- Inverse Document Frequency (idf):
 - E.g., $q =$ "galaxy in space".
 - Is the occurrence of the query term "in" in a document should contribute the same as the occurrence of the query term "galaxy"?

How to determine the $w(t,d)$ weights?

- $tf * idf$ weighting scheme (Salton 73)
 - tf – **term frequency**: a monotonic function of the term frequency in the document,
 - e.g., $tf(t,d) = \log(\text{freq}(t,d) + 1)$
 - e.g., $tf(t,d) = \text{freq}(t,d) / (\text{Max}_{u \in d} \{\text{freq}(u,d)\})$
 - idf – the **inverse document frequency of a term**: a decreasing function of the term total frequency N_t = the number of documents in the corpus where t occurs
 - e.g., $idf(t) = \log(N / N_t)$
 - $w_{ij} = tf(t_i, d_j) * idf(t_i)$

Computing TF-IDF -- An Example

- Given a document D containing terms (a, b, and c) with given frequencies:
 - $\text{freq}(a,D)=3, \text{freq}(b,D)=2, \text{freq}(c,D)=1$
- Assume collection contains 10,000 documents and the term total frequencies of these terms are:
 - $N_a=50, N_b=1300, N_c=250$
- Then:
 - a: $tf = 3/3; idf = \log(10.000/50) = 5.3; tf-idf = 5.3$
 - b: $tf = 2/3; idf = \log(10.000/1300) = 2.0; tf-idf = 1.3$
 - c: $tf = 1/3; idf = \log(10.000/250) = 3.7; tf-idf = 1.2$

Vector-Space pros & cons

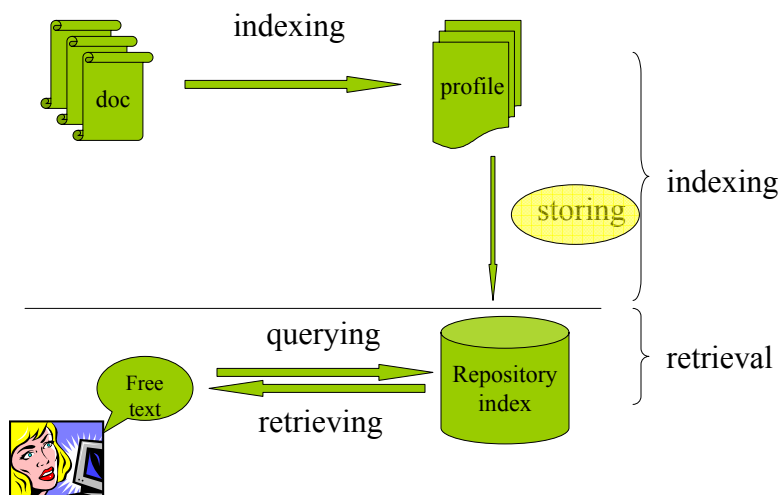
□ **Pros:**

- Terms weighting scheme improves retrieval effectiveness
- Approximate query matching
- Cosine similarity is a good ranking measure

□ **Cons:**

- Terms are not independent of all other terms
- Normalization makes incrementally difficult as recomputation is needed for each added term
- Considered superior to other models due to its simplicity and elegance.

Reminder



Inverted Index

- Purpose: to support efficient retrieval, avoid scanning the entire collection sequentially
- Retrieval should be proportional to the size of the query rather than to size of collection.

Inverted Index

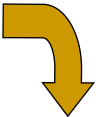
Inverted Index

- This is the primary data structure for text indexes
- Main Idea:
 - **Invert documents descriptions into a big index**
- Basic steps:
 - Make a "dictionary" of all the tokens in the collection
 - For each token, list all the docs it occurs in

Inverted Indexes

An Inverted File is a vector file
"inverted" so that rows become
columns and columns become rows

<i>docs</i>	<i>t1</i>	<i>t2</i>	<i>t3</i>
D1	1	0	1
D2	1	0	0
D3	0	1	1
D4	1	0	0
D5	1	1	1
D6	1	1	0
D7	0	1	0
D8	0	1	0
D9	0	0	1
D10	0	1	1



<i>Terms</i>	D1	D2	D3	D4	D5	D6	D7	...
<i>t1</i>	1	1	0	1	1	1	1	0
<i>t2</i>	0	0	1	0	1	1	1	1
<i>t3</i>	1	0	1	0	1	0	0	0

Evaluation Criteria

- **Effectiveness**
 - How precise is the answer set to the information need
- **Efficiency**
 - Retrieval time, indexing time, index size
- **Usability**
 - Learnability, novice use, expert use

Effectiveness Evaluation

□ **User-centered strategy**

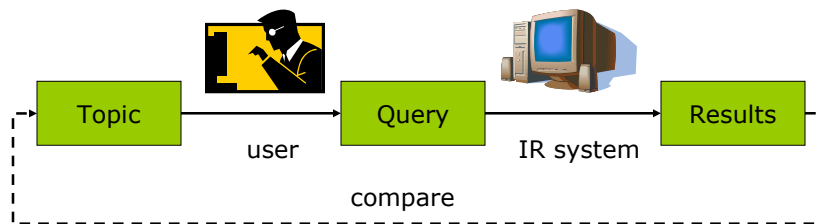
- Try several variations of the retrieval system
- Measure which variation works the "best"
- Given several users, and at least 2 retrieval systems
- Have each user try the same task on both systems (within groups)
- Or have each user try the same task on one system (between groups)
- Measure which system works the "best" (for the "average" user)

IR Test Collection

- A Collection of Documents
 - Representative sources and quantity
- A Set of Topics
 - Used to form queries
- Relevance judgments
 - For each document, with respect to each topic
 - This is the expensive part!

Goal of the evaluation

- Given a (generic) user and an information retrieval system
- We want to measure how good is a user, using the system, in retrieving documents relevant to a set of given topics



Relevance Table

	d1	d2	...	d _N
topic1	+	-		+
topic2	-	-		-
topic _M	-	+		+

Traditional MOE

- Precision
 - How much of what was found is relevant?
 - Particularly for interactive searching
- Recall
 - How much of what is relevant was found?
 - Particularly important for law and patent searches
- Fallout
 - How much of what was irrelevant was rejected?
 - Useful when different size collections are compared

The Contingency Table

Action Doc	Retrieved	Not Retrieved
Relevant	Relevant Retrieved	Relevant Rejected
Not relevant	Irrelevant Retrieved	Irrelevant Rejected

$$\text{Precision} = \frac{\text{Relevant Retrieved}}{\text{Retrieved}}$$

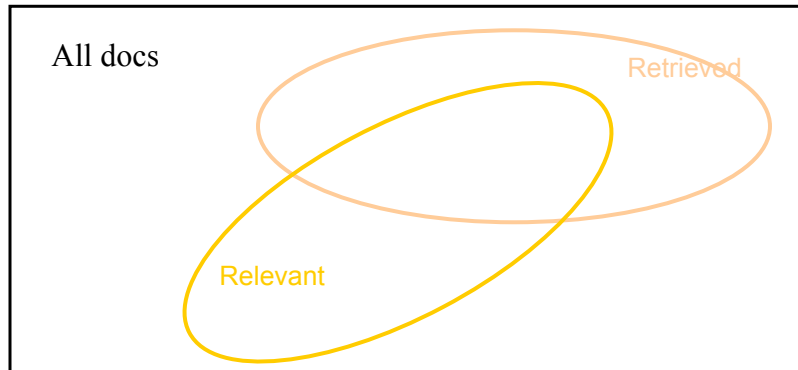
$$\text{Recall} = \frac{\text{Relevant Retrieved}}{\text{Relevant}}$$

$$\text{Fallout} = \frac{\text{Irrelevant Rejected}}{\text{Not Relevant}}$$

Precision vs. Recall

$$\text{Precision} = \frac{|\text{RelRetrieved}|}{|\text{Retrieved}|}$$

$$\text{Recall} = \frac{|\text{RelRetrieved}|}{|\text{Rel in Collection}|}$$



Why Precision and Recall?

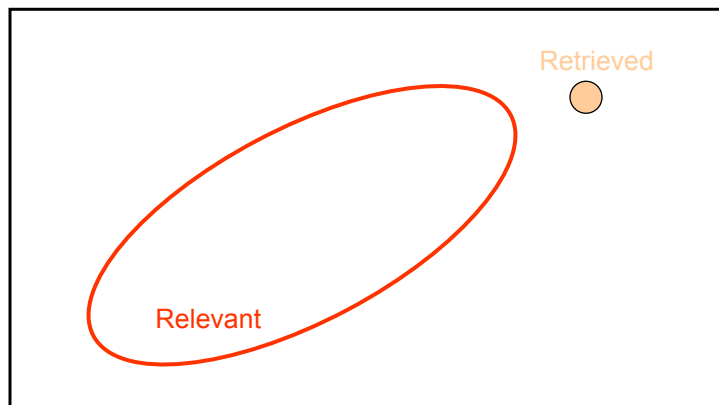
Get as much good stuff while at the same time getting as little junk as possible.

High Recall – all the truth

High Precision – nothing but the truth

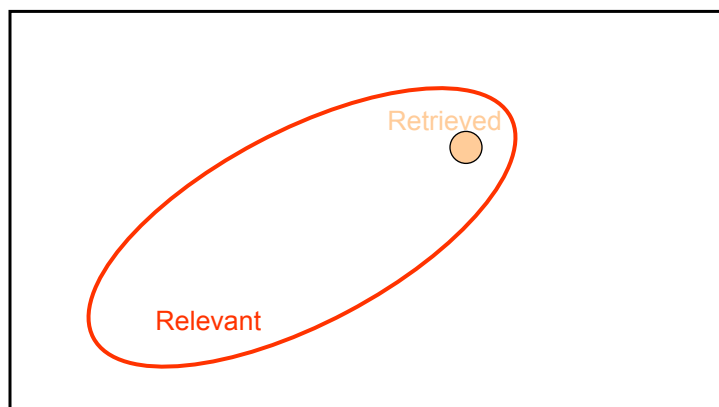
Retrieved vs. Relevant Documents

Very low precision, very low recall (0 in fact)



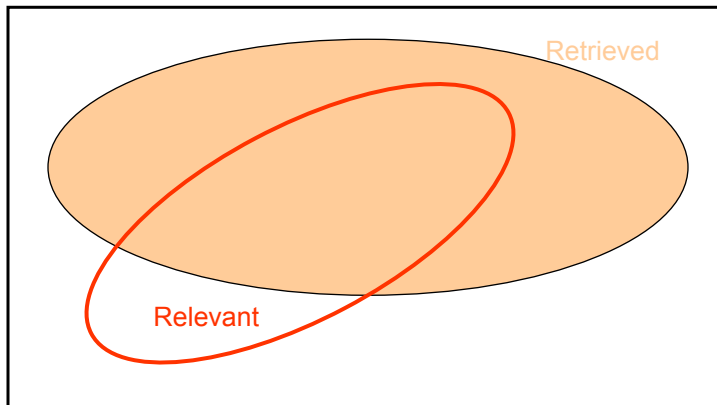
Retrieved vs. Relevant Documents

Very high precision, very low recall



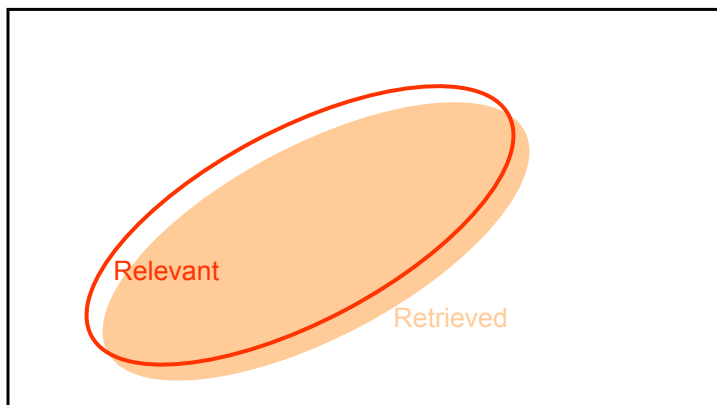
Retrieved vs. Relevant Documents

High recall, low precision

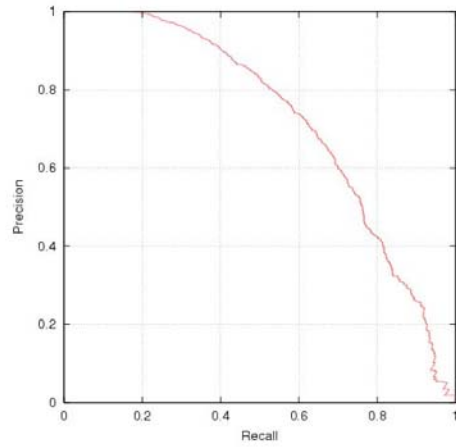


Retrieved vs. Relevant Documents

High precision, high recall (at last!)



Precision and recall



A typical precision and recall curve

Recommender Systems

Original Definition of RS

- In everyday life we rely on recommendations from other people either by word of mouth, recommendation letters, movie and book reviews printed in newspapers ...
- In a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients
 - Aggregation of recommendations
 - Match the recommendations with those searching for recommendations

[Resnick and Varian, 1997]

Recommender Systems

- A **recommender system** helps to make choices without sufficient personal experience of the alternatives
 - To **suggest products** to their customers
 - To provide consumers with **information to help them decide** which products to purchase
- They are based on a number of **technologies**: information filtering, machine learning, adaptive and personalized system, user modeling, ...
- Not clear separation from Information Retrieval – [Burke, 2002] claims that is the “individualized” and “interesting and useful” features that make the difference.

A Simplified Model of Recommendation

1. Two types of entities: Users and Items
2. A background knowledge:
 - A set of ratings: a map $R: \text{Users} \times \text{Items} \rightarrow [0,1] \cup \{?\}$
 - A set of "features" of the Users and/or Items
3. A **method** for eliminating all or part of the '?' values for some (user, item) pairs – substituting '?' with the true values
4. A method for selecting the items to recommend
 - Recommend to u the item i^* such that:
 - $i^* = \arg \max_{i \in \text{Items}} \{R(u,i)\}$

[Adomavicius et al., 2005]

"Core" Recommendation Techniques

U is a set of users

I is a set of items/products

Technique	Background	Input	Process
Collaborative	Ratings from U of items in I .	Ratings from u of items in I .	Identify users in U similar to u , and extrapolate from their ratings of i .
Content-based	Features of items in I	u 's ratings of items in I	Generate a classifier that fits u 's rating behavior and use it on i .
Demographic	Demographic information about U and their ratings of items in I .	Demographic information about u .	Identify users that are demographically similar to u , and extrapolate from their ratings of i .
Utility-based	Features of items in I .	A utility function over items in I that describes u 's preferences.	Apply the function to the items and determine i 's rank.
Knowledge-based	Features of items in I . Knowledge of how these items meet a user's needs.	A description of u 's needs or interests.	Infer a match between i and u 's need.

[Burke, 2002]

The Collaborative Filtering Idea

- ❑ Trying to predict the opinion the user will have on the different items
- ❑ Be able to recommend the “best” items to each user based on the user’s previous likings and the opinions of other like minded users
- ❑ CF is a typical Internet application – it must be supported by a networking infrastructure
 - At least many users and one server
 - But we are thinking of using many servers
- ❑ There is no stand alone CF application.

Movie Lens

movielens
helping you find the right movies

Welcome to MovieLens!

Free, personalized, non-commercial, ad-free, great movie recommendations. Have questions? Take the [MovieLens Tour](#) for answers. Not a member? [Join MovieLens now](#).

Need a gift idea? Try [MovieLens QuickPick!](#)

New to MovieLens?

Join today!

You get **great recommendations** for movies while **helping us do research**. Learn more:

- Try out [QuickPick: Our Movie Gift Recommender](#)
- Take the [MovieLens Tour](#)
- Read our [Privacy Policy](#)
- See our [Browser Requirements](#)
- Learn about [Our Research](#)

Hello MovieLens Users!

Please log in:

Username:

Password:

Save login:

[Log into MovieLens](#)

[Forgot your password?](#)

[New member? Join now](#)

MovieLens is a free service provided by [GroupLens Research](#) at the [University of Minnesota](#). We sometimes study how our members use MovieLens in order to learn how to build better recommendation systems. We promise to never give your personal information to anyone; see our [privacy policy](#) for more information.

Welcome to the new MovieLens!

Existing MovieLens users: We'd like to welcome you back to MovieLens, and let you know we have a new MovieLens FAQ you might want to read. We hope you like what you will see!

[Take me to MovieLens!](#)

New MovieLens users: Thank you for joining MovieLens! In order to generate personalized movie recommendations, we need to know a little about what movies you have already seen. MovieLens will now display several lists of movies. If you have seen any of the listed movies, please rate them using the rating scale shown below.

Ratings are on a scale of 1 to 5:

★★★★★ = Must See
★★★★☆ = Will Enjoy
★★★☆☆ = It's OK
★★☆☆☆ = Fairly Bad
★☆☆☆☆ = Awful

Remember: the more movies you rate, the more accurate MovieLens' predictions will be.

To rate a movie, just click on the pulldown next to the title of a movie you have seen. Blue stars will appear to indicate that your rating has been received.

★☆☆☆☆ 1.5 stars Dude, Where's My Car? (2000)
DVD, VHS, info | imdb
Comedy

This image shows that the movie 'Dude, Where's My Car?' was rated 1.5 stars.

[I'm ready to start rating!](#)

So far you have rated 0 movies.
MovieLens needs at least 15 ratings from you to generate predictions for you.
Please rate as many movies as you can from the list below.

[next >](#)

Your Rating	Movie Information
★★★☆☆ 3.0 stars <input type="text" value="3.0 stars"/>	Austin Powers: International Man of Mystery (1997) Action, Adventure, Comedy
★★★★☆ 4.0 stars <input type="text" value="4.0 stars"/>	Contact (1997) Drama, Sci-Fi
??? Not seen <input type="text" value="Not seen"/>	Crouching Tiger, Hidden Dragon (Wu Hu Zang Long) (2000) Action, Adventure, Drama, Fantasy, Romance
??? Not seen <input type="text" value="Not seen"/>	Demolition Man (1993) Action, Comedy, Sci-Fi
??? Not seen <input type="text" value="Not seen"/>	Eraser (1996) Action, Drama, Thriller
??? Not seen <input type="text" value="Not seen"/>	Maverick (1994) Action, Comedy, Western
★★★★☆ 4.5 stars <input type="text" value="4.5 stars"/>	Philadelphia (1993) Drama
★★★☆☆ 3.5 stars <input type="text" value="3.5 stars"/>	Piano, The (1993) Drama, Romance
??? Not seen <input type="text" value="Not seen"/>	Toy Story 2 (1999) Adventure, Animation, Children, Comedy, Fantasy
★★★☆☆ 3.5 stars <input type="text" value="3.5 stars"/>	X-Men (2000) Action, Adventure, Sci-Fi

[next >](#)

To get a new set of movies click the [next>](#) link.

movielens
helping you find the *right* movies

Welcome **ricci@itc.it** (Log Out)
You've rated **16** movies.
You're the 26th visitor in the past hour.

★★★★ = Must See
★★★★ = Will Enjoy
★★★★ = It's OK
★★★★ = Fairly Bad
★★★★ = Awful

[Home](#) | [Forums](#) | [Manage Buddies](#) | [Your Account](#) | [Help](#)

Shortcuts [Search](#)

- Top Picks For You
- Your Ratings
- Your Wishlist
- Newest Additions
- Rate Random Movies
- Most Often Rated
- Your Tags
- Suggest Title
- About Your Ratings
- New Drama
 - V for Vendetta (2006)
 - Inside Man (2006)
 - Match Point (2005)
- New DVDs
 - Howl's Moving Castle...
 - Three... Extremes (2...
 - Jarhead (2005)
- New Movies
 - Why We Fight (2005)
 - V for Vendetta (2006)
 - Inside Man (2006)

[How to create shortcuts](#)
[Publish your shortcuts](#)

News and Updates [\(archives\)](#)

23 Mar 2006: We may have lost some ratings between 3pm and 7pm central standard time today (Thursday)! Sorry about that! Our database became angry, but we have tamed it. Please **check your ratings** if you rated during that time!

20 Feb 2006: We have added a "show / hide tags" setting on the [account page](#). [More about tags...](#)

7 Feb 2006: We are thinking of redesigning the main page, and would like to [hear your thoughts](#) as we decide how to move forward.

20 Dec 2005: Try out [Movielens QuickPick](#), our new gift recommender. [More...](#)

New Forum Messages

Message Subject	Author	Date
Re: Front page redesign thoughts	robert_kraut	2006-03-27
Re: What's the last thing you watched and what did you rate it?	vargus	2006-03-27
Re: Upcoming Films You Are Most Looking Forward To	Vigilans	2006-03-27
* More threads		

Recently Applied Tags [\(tag your movies\)](#) [\(more about tags\)](#)

My DVDs (124), revolutionary (1), Futuristmovies.com (136), 1984 (1), guy fawkes (1), john hurt (1), dry funny (1), west virginia (1), canoga park (1), creepy (1).

movielens
helping you find the *right* movies

Welcome **ricci@itc.it** (Log Out)
You've rated **16** movies.
You're the 26th visitor in the past hour.

★★★★ = Must See
★★★★ = Will Enjoy
★★★★ = It's OK
★★★★ = Fairly Bad
★★★★ = Awful

[Home](#) | [Forums](#) | [Manage Buddies](#) | [Your Account](#) | [Help](#)

Shortcuts [Search](#)

Search Titles

Use selected buddies

Combined Search

All Genres: All Dates:

Domain: All movies

Tag:

Use selected buddies

Advanced Search

Select Buddies

Test Buddy

What are buddies?

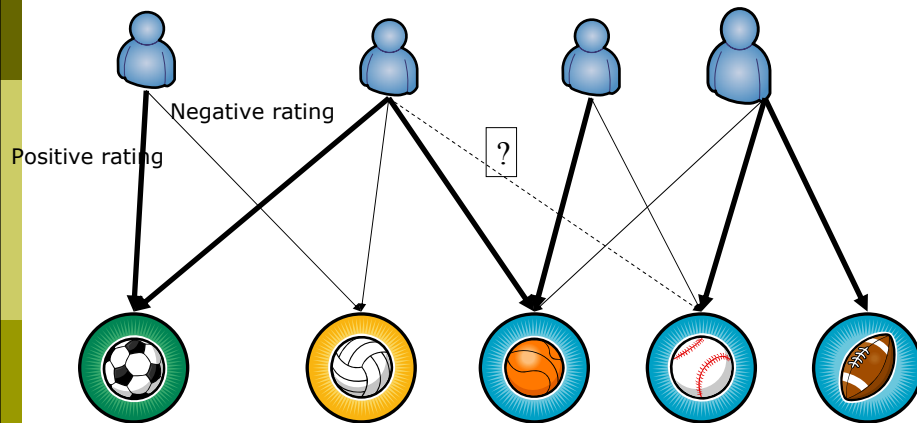
You've searched for **all titles**.
Found **8220** movies, sorted by **Prediction**
Genres: **All** | Exclude Genres: **None**
Dates: **All** | Domain: **All** | Format: **All** | Languages: **All**
[Show Printer-Friendly Page](#) | [Download Results](#) | [Suggest a Title](#)

Tags Related to Your Search: [In Netflix queue \(178\)](#), [Futuristmovies.com \(134\)](#), [My DVDs \(123\)](#), [Oscar \(Best Cinematography\) \(90\)](#), [Oscar \(Best Picture\) \(85\)](#), [\(about tags\)](#)

Page 1 of 548 | Go to page: [1...](#)[109...](#)[218...](#)[327...](#)[436...](#)[545...](#)[last](#) page 2 >

(hide) Predictions for you	Your Ratings	Movie Information	Wish List
★★★★★	Not seen	Cat Returns, The (Neko no ongaeshi) (2002) DVD info imdb Adventure, Animation, Children, Fantasy - Japanese [add tag] Popular tags: anime , cats , In Netflix queue	<input type="checkbox"/>
★★★★★	Not seen	Immigrant, The (1917) DVD VHS info imdb add tag Comedy - Silent	<input type="checkbox"/>
★★★★★	Not seen	Experiment, The (Das Experiment) (2001) DVD VHS info imdb add tag Drama, Thriller - German	<input type="checkbox"/>
★★★★★	Not seen	Thesis (Tesis) (1996) DVD info imdb add tag Drama, Horror, Thriller - Spanish	<input type="checkbox"/>
★★★★★	Not seen	Howl's Moving Castle (Hauru no ugoku shiro) (2004) DVD info imdb Adventure, Animation, Children, Fantasy, Romance - Japanese [add tag] Popular tags: 06 Oscar Nominated Best Movie - Animation , In Netflix queue	<input type="checkbox"/>
★★★★★	Not seen	Why We Fight (2005) info imdb Documentary [add tag] Popular tags: Military , In Netflix queue , controversial	<input type="checkbox"/>

Collaborative Filtering



The CF Ingredients

- List of **m Users** and a list of **n Items**
- Each user has a **list of items** he/she expressed their **opinion** about (can be a null set)
- **Explicit opinion** - a rating score (numerical scale)
- Sometime the rating is **implicitly** - purchase records
- **Active user** for whom the CF prediction task is performed
- A **metric** for measuring **similarity between users**
- A method for selecting a **subset of neighbors** for prediction
- A method for **predicting a rating** for items not currently rated by the active user.

Collaborative-Based Filtering

- The collaborative based filtering recommendation techniques proceeds in these steps:
 1. For a **target/active** user (the user to whom a recommendation has to be produced) the set of his ratings is identified
 2. The **users more similar** to the target/active user (according to a similarity function) are identified (neighbor formation)
 3. The **products bought by these similar users** are identified
 4. For **each one of these products a prediction** - of the rating that would be given by the target user to the product - is generated
 5. Based on this predicted rating a set of top **N products are recommended.**

A Simplified Model of Recommendation

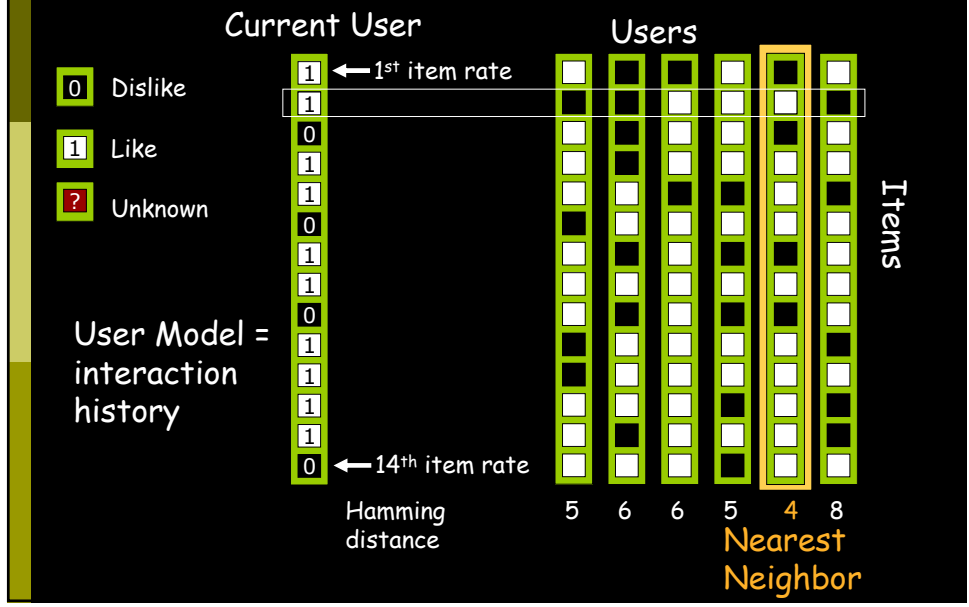
1. Two types of entities: Users and Items
2. A background knowledge:
 - A set of ratings: a map $R: \text{Users} \times \text{Items} \rightarrow [0,1] \cup \{?\}$
 - A set of ~~"features" of the Users and/or Items~~
3. A **method** for eliminating all or part of the '?' values for some (user, item) pairs – substituting '?' with the true values:

$$R(u, i) = \text{Average} \{R(su, i)\}_{su \text{ is similar to } u}$$

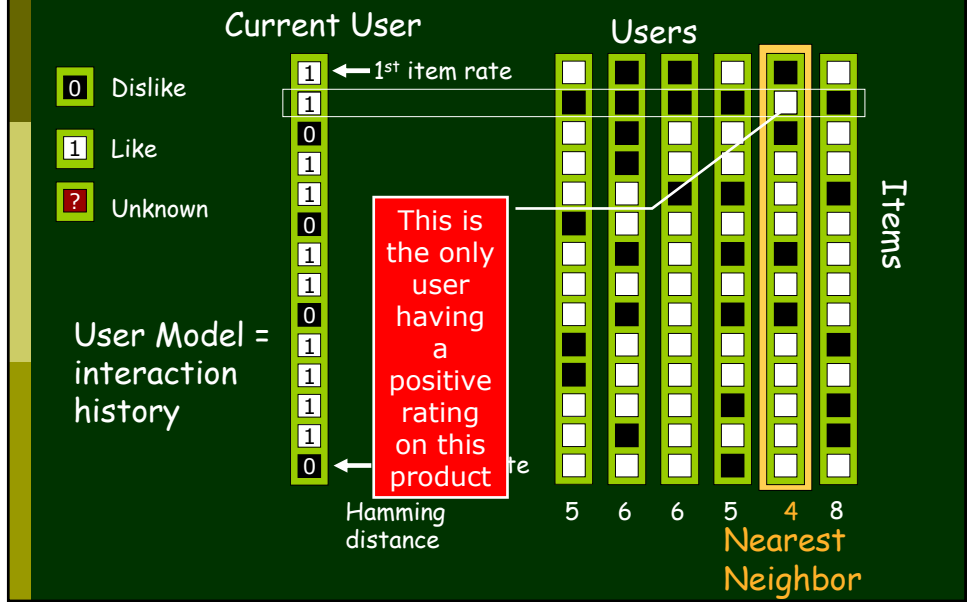
4. A method for selecting the items to recommend
 - Recommend to u the item $i^* = \arg \max_{i \in \text{Items}} \{R(u, i)\}$

[Adomavicius et al., 2005]

Nearest Neighbor Collab.-Based Filtering



1-Nearest Neighbor can be easily wrong



Matrix of ratings

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
a																									
b																									
c																									
d																									
e																									
f																									
g																									
h																									
i																									
j																									
k																									
l																									
m																									
n																									
o																									
p																									
q																									
r																									
s																									
t																									
u																									
v																									
w																									
x																									
y																									

Collaborative-Based Filtering

- A collection of user $u_i, i=1, \dots, n$ and a collection of products $p_j, j=1, \dots, m$
- A $n \times m$ matrix of ratings v_{ij} , with $v_{ij} = ?$ if user i did not rate product j
- Prediction for user i and product j is computed as

$$v_{ij}^* = v_i + K \sum_{v_{kj} \neq ?} u_{ik} (v_{kj} - v_k)$$

- Where, v_i is the average rating of user i , K is a normalization factor such that the sum of u_{ik} is 1, and

$$u_{ik} = \frac{\sum_j (v_{ij} - v_i)(v_{kj} - v_k)}{\sqrt{\sum_j (v_{ij} - v_i)^2 \sum_j (v_{kj} - v_k)^2}}$$

Similarity of users i and k

- Where the sum (and averages) is over j s.t. v_{ij} and v_{kj} are not "?".

Example

	p_j	
u_5	4	$v_5 = 4$
u_i	?	$v_i = 3.2$
u_8	3	$v_8 = 3.5$
u_9	5	$v_9 = 3$

Users' similarities: $u_{i5} = 0.5$, $u_{i8} = 0.5$, $u_{i9} = 0.8$

$$v_{ij}^* = v_i + K \sum_{v_{kj} \neq ?} u_{ik} (v_{kj} - v_k)$$

$$\begin{aligned} v_{ij}^* &= 3.2 + 1/(0.5+0.5+0.8) * [0.5 (4 -4) + 0.5 (3 - 3.5) + 0.8 (5 -3)] \\ &= 3.2 + 1/1.8 * [0 - 0.25 + 1.6] = 3.2 + 0.75 = 3.95 \end{aligned}$$