

Refining after search

Use user feedback

- or pseudo-feedback
 - Approximate feedback with first results
- or implicit feedback
 - e.g. clicks
- change ranking of current results
- search again with modified query
- change ranking for future searches

5

7

Explicit user feedback

- User must participate
- User marks (some) relevant results
 or
- · User changes order of results
 - Can be more nuanced than relevant or not
 - Can be less accurate than relevant or not
 - Example: User moves 10th item to first
 - $-\operatorname{says}$ 10th better than first 9
 - Does not say which, if any, of first 9 relevant

Implicit user feedback

- Click-throughs
 - Use as relevance judgment
 - Use as reranking:
 When click result, moves it ahead of all results
 - didn' t click that come before it
 - Problems?
- Better implicit feedback signals?

User feedback in classic vector model

- Run query
 - Query represented as vector of term weights
- User marks top p documents for relevance p = 10 to 20 "typical"
- Construct new weights for terms in query vector
 - Modifies query
- Rerun query
 - Could use just on initial results to re-rank *

Deriving new query for vector model

For collection C of n doc.s

 Let C_r denote set all relevant docs in collection,

Perfect knowledge Goal:

Vector **q**_{opt} =

 $1/|C_r| * (sum of all vectors d_j in C_r) - 1/(n-|C_r|) * (sum of all vectors d_k not in C_r)$

centroids

9

Deriving new query for vector model: Rocchio algorithm

Give query **q** and relevance judgments for a subset of retrieved docs

- · Let D_r denote set of docs judged relevant
- + Let D_{nr} denote set of docs judged not relevant

Modified query:

Vector $\mathbf{q}_{new} = \alpha \mathbf{q} + \beta/|D_r|^*$ (sum of all vectors \mathbf{d}_j in D_r) - $\gamma/(|D_{nr}|)^*$ (sum of all vectors \mathbf{d}_k in D_{nr})

For tunable weights α , β , γ

Remarks on new query

- α: importance original query
- β: importance effect of terms in relevant docs
- γ: importance effect of terms in docs not relevant
- Usually terms of docs not relevant are least important
 - Reasonable values α =1, β =.75, γ =.15
- · Reweighting terms leads to long queries
 - **Many** more non-zero elements in query vector $\boldsymbol{q}_{\text{new}}$
 - Can reweight only most important (frequent?) terms
- · Most useful to improve recall
- Users don't like: work + wait for new results 11

Simple example user feedback in vector model

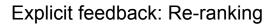
10

12

- **q** = (1,1,0,0)
- Relevant: **d1** = (1,0,1,1) **d2** = (1,1,1,1)
- Not relevant: **d3**=(0,1,1,0)
- α, β, γ **= 1**
- $\mathbf{q}_{\text{new}} = (1,1,0,0) + (1, 1/2, 1, 1) (0,1,1,0)$
 - = (2, 1/2, 0, 1)

Term weights change New term

Observe: Can get negative weights



- Can disambiguate within given results
 jaguar car versus jaguar animal
- Can modify rankings for future searches
- Algorithms usually based on machine learning

13

15

- Learn ranking function that best matches partial ranking(s) given
- · Simpler strategies:
 - use for repeat of same search
 - · user reorder or select best
 - Google experiment circa 2007

Behavior History

- Going beyond behavior on *same* query.
- Personal history versus Crowd history
 - Crowd history
 - · Primarily search history
 - Google's claim Bing copies
 - Personal history
 - · characterize behavior
 - characterize interests: topics
 - what use?

14

16

Personal History: sources

- Your searches
- Your social networks
 - Your content
- Other behavior browsing, mail?, ...

Collaborative history

- History of people "like" you
- How get?
 - For "free": social networks
 - friends, lists, ...
 - Deduce: Crowd history + personal history
 recommendations
- · How characterize?
 - Shared behaviors
 - Shared topics

Crowd versus friends

- · Content and properties can be
 - 1. Yours
 - 2. Your friends
 - 3. The crowd's
- 1 and 2 provide personalization

Social Network Sites and Obtaining Information

Social network sites

17

19

- · Catch-all term for
 - social networking sites
 - Facebook
 - microblogging sites
 - Twitter
 - blog sites (for some purposes)
- Now interested in social networking information
 - friending/following concept
 - not totality of Web
 - not Wikipedia encyclopedia pages
 - yes Wikipedia talk pages?

Ways we can use social networks to find information

18

- Search site
- Aggregate site information to get trends
- Use site content as meta-information for search
- Use site properties as meta-information for search

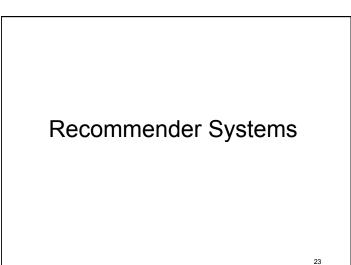
Use site content as metainformation for search

- disambiguate queries (Teeven et al 2011 suggested)
 - search Twitter with query
 - analyze content of matching tweets to identify most current, most popular meaning
- factor in ranking URLs (Dong et. al. 2010 studied)
 harvest URLs mentioned in tweets
 - associate a URL with tweeted text surrounding it
- other uses for tweet text?
- similar analyses of social networking sites such as Facebook?

Use site properties as metainformation for search

- interactions: friends, followers, likes, retweets, more?
- Uses
 - ranking by popularity of content
 - ranking by influence of author
- temporal relevance
 - ranking
 - discover URLs faster (Dong et. al. 2010)

22



Relationship to information retrieval?

 Ranked results of query can be considered recommendations based on constraints (query) placed by user



- · Look at classic model and techniques
 - Items
 - Users
 - Recommend Items to Users
- · Recommend new items based on:
 - similarity to items user liked in past: individual history "Content Filtering"
 - Liked by other users similar to this user: collaborative history
 - "Collaborative Filtering"
 - Liked by other users: crowd history
 easier case

25

Recommender System attributes

- · Need explicit or implicit ratings by user
 - Purchase is 0/1 rating
 - Movie tickets
 - Books
- · Have focused category
 - examples: music, courses, restaurants
 - hard to cross categories with content-based
 - easier to cross categories with collaborative-based
 - users share tastes across categories?

26

Content Filtering

- · Items must have characteristics
- · user values item
 - \Rightarrow values characteristics of item
- model each item as vector of weights of characteristics
 - much like vector-based IR
- user can give explicit preferences for certain characteristics

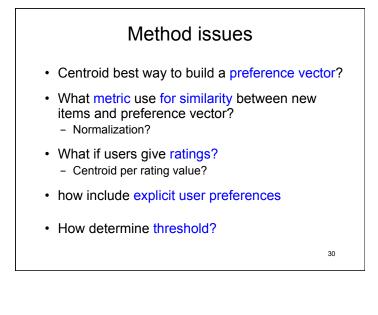
27

Buy/no buy prediction method: similarity with centroid

- Average vectors of items user bought

 user's centroid
- · Find similarity of new items to user's centroid
- · Decide threshold for "buy" recommendation

 user bought Average boo Score new b dot prodution decide threst 	book 1 and oks bought = oooks uct gives: sc	: (0, 1, 0.5, core(A) = 0.	5; score (B)= 1
	1 st person	romance	mystery	sci-fi
book 1	1 st person 0	romance	mystery 1	sci-fi 0
book 1 book 2	1 st person 0 0	romance 1 1	mystery 1 0	sci-fi 0 0
	0	romance 1 1 .5	1	0



Example with explicit user preferences How use scores of books bought? Try: preference vector p where component k = user pref for characteristic k if $\neq 0$ avg. comp. k of books bought when user pref =0 0 pref for user = "don' t care" 1st per rom mys sci-fi **p**=(0, 1, 0.5, -5) user pref 0 1 0 -5 New scores? book 1 0 1 1 0 **p**•A = 0.5 book 2 0 1 0 0 **p•**B = 0 .5 0 new A 1 0 0 0 .2 new B 1 31

Other methods: machine learning

- Major alternatives based on classifiers
 - Training set: items bought and not bought
 - Train classifier many algorithms
 - Classify new item as buy/no buy
- Observations
 - Uses books not bought. Problems?
 - Multiple rating values
 - Can use multiple classes

Limitations of Content Filtering

- Can only recommend items similar to those user rated highly
- New users
 - Insufficient number of rated items
- Only consider features explicitly associated with items
 - Do not include attributes of user

33

Applying content filtering methods to search

- Characterize documents (info. objects)
 - topic analysis?
 - other properties, e.g.:
 - Domain of source
 - Date of publication/update
- · Characterize individuals
 - deduce from properties of objects interact with
 - user provided preferences

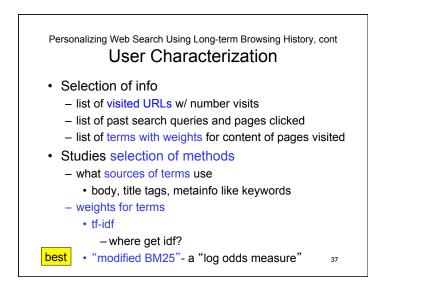
34

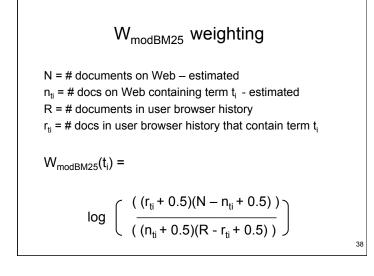
36

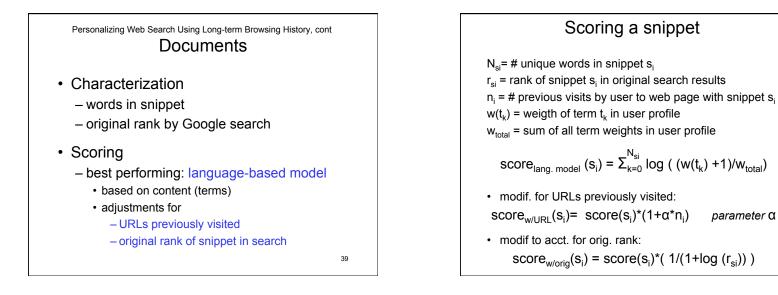
<section-header><list-item><list-item><list-item><list-item><list-item><list-item><list-item>

Example study: Personalizing Web Search Using Long-term Browsing History (in *WSDM11*)

- Goal: rerank
 top 50 results from Google query
- Query is initial filter to get results from Google
- · Strategy:
 - score snippets from search result against user profile
 - rerank based on snippet score







Personalizing Web Search Using Long-term Browsing History $\label{eq:search} Evaluation$

- "offline" evaluation:
 - relevance judgments by volunteers
 - used to select best of algorithmic variations
- online evaluation of best variations:
 - add-on to Browser by volunteers
 - interleave original results (no personalization) with results reranked by snippet score

41

– record clicks by user – which list from

Personalizing Web Search Using Long-term Browsing History $\label{eq:Results} Results$

- Offline: normalized DCG, avg. of 72 queries
 - Google's ranking w/out personalization: 0.502
 - best-performing of variations for reranking: 0.573
- Online
 - 8% queries: # clicks from original and reranked same
 - of rest: 60.5% queries: more clicks from reranked
 39.5% queries: more clicks from original

Observation

 Reranking can be done completely in browser if enough space for data for user profile
 42