Refining searches

1

Refine initially: query

- Commonly, query expansion
  - add synonyms
    - Improve recall
    - Hurt precision?
    - Sometimes done automatically
  - Modify based on prior searches
    - Not automatic
    - All prior searches
      vs
    - *your* prior searches
    - Example: Yahoo
      - Google does too

2

Refining after search

- Use user feedback
  or
- Approximate feedback with first results
  - Pseudo-feedback
  - Example: "Yahoo assist"

- change ranking of current results
  or
- search again with modified query

3

Explicit user feedback

- User must participate

- User marks (some) relevant results
  or
- User changes order of results
  - Pros and cons?

4

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
      - says 10th better than first 9
      - Does not say which, if any, of first 9 relevant

5

User feedback in classic vector model

- User marks top p documents for relevance
  p = 10 to 20 "typical"
- Construct new weights for terms in query vector
  - Modifies query
  - Could use just on initial results to re-rank

6

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 $\mathbf{q}_{opt}$ =
$1/|C_r|$ * (sum of all vectors d_j in C_r) -
$1/(n- |C_r|)$ * (sum of all vectors d_k not in C_r)
centroids

7

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

8

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 $\mathbf{q}_{new}$
  - Can reweight only most important (frequent?) terms
- Most useful to improve recall
- Users don't like: work + wait for new results

9

Simple example user feedback in vector model

- **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
- $\mathbf{q}_{new}$ = (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

10

Re-ranking

- Example - status?
  - Google experiment: only affects repeat of same search
  - learned in class is now SearchWiki feature for Google accounts
- Algorithms usually based on machine learning
  - Learn ranking function that best matches partial ranking given

11

Comparing rankings

Kendall Tau measure compares two orderings of the same set of n objects:

- Let A = # pairs whose orders agree in the two orderings
- Let I = # pairs whose orders disagree in the two ordering = # inversions

Kendall's Tau (order1, order2)
= (A- I)/(A+ I) Ordinal correlation

$= 1 - \frac{I}{(¼(n)(n-1))}$ since A+I = 1/2(n)(n-1)

12

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?

13

Single user feedback vs group

- Compare Recommender Systems
  - Items
  - Users
  - Recommend Items to Users
- Recommend new items based on similarity to items that:
  - User liked in past: Content-based
  - Liked by other users similar to this user: Collaborative Filtering
    - just "liked by other users" - easier case
- Documents matching search = items?

14

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?

15

Content-based recommendation

- Items must have characteristics
- user values item

  ⇒ 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

16

Content-based example

- user bought book 1 and book 2
  - what if actually rated?
- Average books bought = (0, 1, 0.5, 0)
- Score new books
  - dot product gives: score(A) = 0.5; score (B)= 1
- decide threshhold for recommendation

	1st person	romance	mystery	sci-fi
book 1	0	1	1	0
book 2	0	1	0	0
new book A	1	.5	0	0
new book B	0	1	0	.2

17

Example with explicit user preferences

How use scores of books bought?

Try: preference vector p where component k =

user pref for characteristic k if ≠ 0

avg. comp. k of books bought when user pref =0

0 pref for user = "don't care"

p=(0, 1, 0.5, -5)

New scores?

p•A = 0.5

p•B = 0

	1st per	rom	mys	sci-fi
user pref	**0**	**1**	**0**	**-5**
book 1	0	1	1	0
book 2	0	1	0	0
new A	1	.5	0	0
new B	0	1	0	.2

18

Content-based: issues

- Vector-based one alternative
- Major alternatives based on machine-learning
- For vector based
  - how build a preference vector
    - how combined vectors for items rated by user
      - our example only 0/1 rating
    - how include explicit user preferences
  - what metric use for similarity between new items and preference vector
  - normalization
  - threshold?

19

Limitations of Content-based

- 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

20

Collaborative Filtering

- Recommend new items liked by other users similar to this user
- need items already rated by user ***and other users***
- don't need characteristics of items
  - each rating by individual user becomes characteristic
- Can combine with item characterisitics
  - hybrid content/collaborative

21

Method types

(see Adomavicius and Tuzhilin paper)

- Memory-Based
  - Similar to vector model
  - Use (user × item) matrix
  - Use similarity function
  - Prediction based on previously rated items
- Model-Based
  - Machine-learning methods
  - Model of probabilities of (users × items)

22

Memory-Based: Preliminaries

- Notation
  - $r(u,i)$ = rating of i^{th} item by user u
  - I_u = set of items rated by user u
  - $I_{u,v}$ = set of items rated by both users u and v
  - $U_{i,j}$ = set of users that rated items i and j
- Adjust scales for user differences
  - Use average rating by user u:

    $$r_u^{avg} = (1/|I_u|) * \sum_{i \text{ in } I_u} r(u,i)$$
  - Adjusted ratings: $r_{adj}(u,i) = r(u,i) - r_u^{avg}$

23

One Memory-Based method: User Similarities

- similarity between users u and v
  - Pearson correlation coefficient

$$\text{sim}(u,v) = \frac{\sum_{i \text{ in } I_{u,v}} (r_{adj}(u,i) * r_{adj}(v,i))}{\left(\sum_{i \text{ in } I_{u,v}} (r_{adj}(u,i))^2 * \sum_{i \text{ in } I_{u,v}} (r_{adj}(v,i))^2 \right)^{1/2}}$$

24

One Memory-Based Method: Item Similarities

- similarity between items i and j
  - vector of ratings of users in $U_{i,j}$
  - cosine measure using adjusted ratings

$$sim(i,j) = \frac{\sum_{u \text{ in } U_{i,j}} (r_{adj}(u,i) * r_{adj}(u,j))}{\left(\sum_{u \text{ in } U_{i,j}} (r_{adj}(u,i))^2 \sum_{u \text{ in } U_{i,j}} (r_{adj}(u,j))^2 \right)^{1/2}}$$

25

Predicting User's rating of new item: User-based

For item i not rated by user u

$$r^{pred}(u,i) = r_u^{avg} + \frac{\sum_{v \text{ in } S} (sim(u,v) * r_{adj}(v,i))}{\sum_{v \text{ in } S} |sim(u,v)|}$$

S can be all users or just users *most similar* to u

26

Predicting User's rating of new item: Item-based

For item i not rated by user u

$$r^{item\text{-}pred}(u,i) = \frac{\sum_{j \text{ in } T} (sim(i,j) * r(u,j))}{\sum_{j \text{ in } T} |sim(i,j)|}$$

T can be all items or just items *most similar* to i

➢ Prediction uses only u's ratings, but similarity uses other users' ratings

27

Collaborative filtering example

user ratings

	book 1	book 2	book 3	book 4
user 1	5	1	2	0
user 2	x	5	2	5
user 3	3	1	x	2
user 4	4	0	2	**?**

adj. user ratings

	book 1	book 2	book 3	book 4
user 1	3	-1	0	-2
user 2	x	1	-2	1
user 3	1	-1	x	0
user 4	2	-2	0	**?**

Collaborative filtering example

- sim(u1,u4) = $(6+2)/(10*8)^{1/2}$ = .894
- sim(u2,u4) = $(-2)/(5*4)^{1/2}$ = -.447
- sim(u3,u4) = $(2+2)/(2*8)^{1/2}$ = 1

- predict r(u4, book4) = $2 + \frac{(-2)*.894 + 1*(-.447) + 0*1}{.894 + .447 + 1}$

  $= 2 - .955 \approx 1$

29

Limitations

- May not have enough ratings for new users
- New items may not be rated by enough users
- Need "critical mass" of users
  - All similarities based on user ratings

30

Recommendation techiques and search

Content-based ⇔ query refinement with user feedback

item	⇔	document
characteristic	⇔	term
user preferences	⇔	initial query
user rating of previous items	⇔	relevance ratings for initial results

31

Recommendation techiques and search: Collaborative filtering

- analogy with product recommendation?

  users behavior on same search - i.e. same query
  - item ⇔ search result
  - rating ⇔ clicked/not clicked on
  - predict whether user will click on based on behavior of similar users
  - user similarity based on what have both clicked on for ***this*** search
- more general predictions of best results based on notions of user similarity
  - hybrid content and collaboration

32