Refining searches

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

Refining after search
- use user feedback
  - approximate feedback with first results
    - pseudo-feedback
      - example: “yahoo assist”
  - change ranking of current results
  - search again with modified query

Explicit user feedback
- user must participate
  - user marks (some) relevant results
  - user changes order of results
    - pros and cons?

Explicit user feedback
- user must participate
  - user marks (some) relevant results
  - 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

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
Deriving new query for vector model

For collection \( C \) of \( n \) docs

- Let \( C_r \) denote set all relevant docs in collection,

Perfect knowledge Goal:

Vector \( q_{\text{opt}} = \frac{1}{|C_r|} \) * (sum of all vectors \( d_j \) in \( C_r \)) - \( \frac{1}{n - |C_r|} \) * (sum of all vectors \( d_k \) not in \( C_r \))

Remarks on new query

- \( \alpha \): importance original query
- \( \beta \): importance effect of terms in relevant docs
- \( \gamma \): importance effect of terms in docs not relevant

- Usually terms of docs not relevant are least important
  - Reasonable values \( \alpha = 1, \beta = .75, \gamma = .15 \)
- Reweighting terms leads to long queries
  - Many more non-zero elements in query vector \( q_{\text{new}} \)
- Can reweight only most important (frequent?) terms
- Most useful to improve recall
- Users don’t like: work + wait for new results

Simple example user feedback in vector model

- \( q = (1, 1, 0, 0) \)
- Relevant: \( d_1 = (1, 0, 1, 1) \)
  \( d_2 = (1, 1, 1, 1) \)
- Not relevant: \( d_3 = (0, 1, 1, 0) \)
  \( \alpha, \beta, \gamma = 1 \)

- \( q_{\text{new}} = q + \beta/(|D_r|) * (\text{sum of all vectors } d_j \text{ in } D_r) - \gamma/(|D_{nr}|) * (\text{sum of all vectors } d_k \text{ in } D_{nr}) \)

Term weights change

Observe: Can get negative weights

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 orderings
- Inversion = \# inversions

Kendall’s Tau (order1, order2)

\[ = \frac{(A- I)/(A+ I)}{1 - \frac{1}{(\frac{1}{2})(n-1)}} \]

since \( A+I = 1/2(n)(n-1) \)
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?

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?

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?

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

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 threshold for recommendation

<table>
<thead>
<tr>
<th>1st person</th>
<th>romance</th>
<th>mystery</th>
<th>sci-fi</th>
</tr>
</thead>
<tbody>
<tr>
<td>book 1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>book 2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>new book A</td>
<td>1</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>new book B</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

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?
$p \cdot A = 0.5$
$p \cdot B = 0$

<table>
<thead>
<tr>
<th>1st per</th>
<th>rom</th>
<th>mys</th>
<th>sci-fi</th>
</tr>
</thead>
<tbody>
<tr>
<td>user pref</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>book 1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>book 2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>new A</td>
<td>1</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>new B</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Content-based: issues

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

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

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 characteristics
  - hybrid content/collaborative

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)

Memory-Based: Preliminaries

- Notation
  - \( r(u,i) = \) rating of \( i \)th item by user \( u \)
  - \( I_u = \) set of items rated by user \( u \)
  - \( I_{uv} = \) set of items rated by both users \( u \) and \( v \)
  - \( U_{ij} = \) set of users that rated items \( i \) and \( j \)
- Adjust scales for user differences
  - Use average rating by user:
    - \( r_u^{avg} = (1/|I_u|) \times \sum_{i \in I_u} r(u,i) \)
  - Adjusted ratings: \( r_{adj}(u,i) = r(u,i) - r_u^{avg} \)

One Memory-Based method: User Similarities

- similarity between users \( u \) and \( v \)
  - Pearson correlation coefficient
    \[
    sim(u,v) = \frac{\sum_{i \in I_u} (r_{adj}(u,i) \times r_{adj}(v,i))}{(\sum_{i \in I_u} (r_{adj}(u,i))^2 \times \sum_{i \in I_v} (r_{adj}(v,i))^2)^{1/2}}
    \]
One Memory-Based Method: 
Item Similarities

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

\[ \text{sim}(i,j) = \frac{\sum_{u \in U_{ij}} (r_{adj}(u,i) \cdot r_{adj}(u,j))}{\left( \sum_{u \in U_i} (r_{adj}(u,i))^2 \cdot \sum_{u \in U_j} (r_{adj}(u,j))^2 \right)^{1/2}} \]

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

For item i not rated by user u

\[ r_{\text{pred}}(u,i) = r_u + \frac{\sum_{v \in S} \text{sim}(u,v) \cdot r_{adj}(v,i)}{\sum_{v \in S} |\text{sim}(u,v)|} \]

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

Collaborative filtering example

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>user 2</td>
<td>x</td>
<td>5</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>user 3</td>
<td>3</td>
<td>1</td>
<td>x</td>
<td>2</td>
</tr>
<tr>
<td>user 4</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>user 2</td>
<td>x</td>
<td>-1</td>
<td>-2</td>
<td>1</td>
</tr>
<tr>
<td>user 3</td>
<td>1</td>
<td>-1</td>
<td>x</td>
<td>0</td>
</tr>
<tr>
<td>user 4</td>
<td>2</td>
<td>-2</td>
<td>0</td>
<td>?</td>
</tr>
</tbody>
</table>

Collaborative filtering example

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

- predict \( r(u4, \text{book}4) = 2 + \frac{(-2) \cdot .894 + 1 \cdot (-.447) + 0 \cdot 1}{.894 + .447 + 1} \)
  \[ = 2 \cdot .955 \approx 1 \]

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
Recommendation techniques and search

Content-based ⇔ query refinement with user feedback

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

Recommendation techniques 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