Where are we?

- Refinement/Personalization of results
- Study techniques of

  Recommender systems

  - Content filtering
    - Applying content filtering to search
  - Collaborative filtering
    - Nearest neighbor methods
      - Applying nearest neighbor method to search
    - ➢Matrix factorization methods

1

Matrix factorization motivation

- Discover/use latent factors
  - attributes, topics, features
- Factor matrices to uncover latent factors
- Don't know what latent factors represent
  - can conjecture

2

Matrix factorization for Collaborative Filtering

- Give ratings matrix R: M users X N items
  - R has holes- R_{ij} with no value
- Want to fill in holes => predict ratings
- Idea: decompose R:

  $$R=P*Q^T$$

  - P is M X f; Q is N X f
  - f dimensions are latent factors
    - no interpretation but can add one
  - must choose f

3

How does decomposition help?

- estimate P and Q, leaving no holes
- get estimate of R as $R_f = PQ^T$
  - R_f has holes of R filled in
- Several methods for estimation, e.g.
  - Gradient descent
  - Stochastic gradient descent
    - Koren et al. *Matrix Factorization Techniques for Recommender Systems*, IEEE Computer, Aug 2009
  - Least squares based calculations
    - Bell et al *Modeling Relat'ships at Multiple Scales to Improve Accuracy of Large Recom. Sys.*, KDD Aug 2007.

4

Optimization

- Minimize least squares error:

err(P,Q) is defined as

$$\sum_{(u,i) \text{ in } K}(R_{(u,i)} - (PQ^T)_{(u,i)})^2$$

for K the set of (u,i) for which $R_{(u,i)}$ has a value

5

Simple Step: Gradient Descent

- Minimize for one element change:
  - choose one element of P or one element of Q to vary, say $P_{(r,s)}$

    $(PQ^T)_{(r,j)} = (\sum_{k,\, k\neq s} P_{(r,k)} * Q_{(j,k)}) + x * Q_{(j,s)}$
  - err(P,Q) becomes equation with one unknown
    - look at only terms involving x
    - get sum over j for which $R_{(r,j)}$ has a value of:

$$(R_{(r,j)} - (PQ^T)_{(r,j)})^2 = (R_{(r,j)} - (\sum_{k,\, k\neq s} P_{(r,k)} * Q_{(j,k)}) - x * Q_{(j,s)})^2$$

  - take derivative wrt x, set to 0, solve

6

Update step

Solution:

$$x = \frac{\sum_{j \text{ in } K} Q_{(j,s)} (R_{(r,j)} - \sum_{k \neq s} P_{(r,k)} * Q_{(j,k)})}{\sum_j Q^2_{(j,s)}}$$

for K the set of (r,j) for which $R_{(r,j)}$ has a value

- Similar equation if set element of Q to unknown y
- Iterate through elements of P, Q, repeatedly
- Find local minimum
  - improvement threshhold
- Need initial values P, Q

7

Collaborative Filtering: Global effects

Effects over many or all of ratings

✓ different users have different rating scales
- metadata (attributes) for items and/or users
  hybrid content/collaborative
- date of rating
- trend of user's ratings over time
- trend of item's ratings over time

Reference: Scalable Collaborative Filtering w/ Jointly Derived Neighborhood Interpolation Weights, Bell and Koren, *IEEE Intern. Conf. Data Mining* (part of winning Netflix contest team)

8

Summary

- Looked at several techniques to modify search
- explicit user feedback
- user behavior: history
  - user history
  - crowd history
  - collaborative history: "people like you"
- role of social networks
  - general analysis
  - relationships
- models of recommender systems

9

Final thought

All techniques we've seen behavior or topic oriented

What about links? What about PageRank?

10

Refining PageRank

$$\boldsymbol{pr} = (\alpha/n, \alpha/n, \ldots \alpha/n)^T + (1-\alpha)\, L^T \boldsymbol{pr}$$

- let $\boldsymbol{v} = (1/n, 1/n, \ldots 1/n)$
- rewrite $\boldsymbol{pr} = (\alpha)\boldsymbol{v}^T + (1-\alpha)\, L^T \boldsymbol{pr}$
- Refinement choices
  - change ***v***
  - change ***L***

11

"Topic Sensitive" PageRank

Haveliwala

- Use pre-defined topics
  - Open Directory Project
    - "the largest, most comprehensive human-edited directory of the Web."
    - 16 top-level topics
- Each page has PageRank for each topic
- Calculate similarity of query to each topic
  - Use linear combination of topic PageRanks based on similarity values query to topic

12

Personalized PageRank

Kamvar et. al.

- Random leaps are biased by personal interests – change ***v***
- Combined with use of block structure to make more efficient:
  - Divide Web graph into blocks (clusters)
    - Use high-level domains (e.g. princeton.edu)
  - Calc. local PageRank within each block
  - Collapse each block into 1 node – new graph
    - Weighted edges between nodes
  - Calc. PageRank with biased leaps for block structure
  - Weight local PageRanks with block PageRank
    - Use to initialize power calculation

13