Author(s): Rahul Sami, 2009

License: Unless otherwise noted, this material is made available under the terms of the Creative Commons Attribution Noncommercial Share Alike 3.0 License: http://creativecommons.org/licenses/by-nc-sa/3.0/

We have reviewed this material in accordance with U.S. Copyright Law and have tried to maximize your ability to use, share, and adapt it. The citation key on the following slide provides information about how you may share and adapt this material.

Copyright holders of content included in this material should contact open.michigan@umich.edu with any questions, corrections, or clarification regarding the use of content.

For more information about how to cite these materials visit http://open.umich.edu/education/about/terms-of-use.
Citation Key
for more information see: http://open.umich.edu/wiki/CitationPolicy

Use + Share + Adapt
{ Content the copyright holder, author, or law permits you to use, share and adapt. }

Public Domain – Government: Works that are produced by the U.S. Government. (USC 17 § 105)
Public Domain – Expired: Works that are no longer protected due to an expired copyright term.
Public Domain – Self Dedicated: Works that a copyright holder has dedicated to the public domain.
Creative Commons – Zero Waiver
Creative Commons – Attribution License
Creative Commons – Attribution Share Alike License
Creative Commons – Attribution Noncommercial License
Creative Commons – Attribution Noncommercial Share Alike License
GNU – Free Documentation License

Make Your Own Assessment
{ Content Open.Michigan believes can be used, shared, and adapted because it is ineligible for copyright. }

Public Domain – Ineligible: Works that are ineligible for copyright protection in the U.S. (USC 17 § 102(b)) *laws in your jurisdiction may differ

{ Content Open.Michigan has used under a Fair Use determination. }

Fair Use: Use of works that is determined to be Fair consistent with the U.S. Copyright Act. (USC 17 § 107) *laws in your jurisdiction may differ

Our determination DOES NOT mean that all uses of this 3rd-party content are Fair Uses and we DO NOT guarantee that your use of the content is Fair.
To use this content you should do your own independent analysis to determine whether or not your use will be Fair.
Lecture 6: Applications; Implementation

SI583: Recommender Systems
Taxonomy of E-Commerce Applications [Schafer, Konstan, Riedl]

- Characterize systems based on
  - Functional Inputs & Outputs
    • note: navigational inputs
  - Recommendation Method
    • user-user, item-item, PageRank, etc.
  - Other design issues (esp., personalization)
A Taxonomy for Recommender Applications

Recommendation Method
- Raw retrieval
- Manually selected
- Statistical summarization
- Attribute-based
- Item-to-Item correlation
- User-to-User correlation

Targeted Customer Inputs
- Implicit Navigation
- Explicit Navigation
- Keyword/Item
- Attribute
- Ratings
- Purchase History

Community Inputs
- Item Attribute
- External Item
- Popularity
- Purchase History
- Ratings
- Text Comments

Outputs
- Suggestion
- Prediction
- Ratings
- Reviews

E-store Engine

Degree of Personalization
- Non-personalized
- Ephemeral
- Persistent

Delivery
- Push
- Pull
- Passive
Degrees of Personalization

- Unpersonalized

- Ephemeral personalization
  - e.g., based on shopping cart alone
  - user profile is not long-lived

- Persistent personalization

- What factors would influence your choice?
Software Architecture

- Don’t try to do the entire recommendation process *online* (i.e., *in real time*)

- Goal: precompute as much as possible, and do as little as necessary when you have to generate a recommendation
Software Architecture

- Don’t try to do the entire recommendation process online (i.e., in real time)

- Goal: precompute as much as possible, and do as little as necessary when you have to generate a recommendation
  - Tradeoff: precomputed values may be “stale”
To recommend items to Joe:

- Normalize all ratings by user means, standard deviations
- Compute similarity (Pearson correlation coefficient) between Joe and each other user
- Compare a set of nearest neighbors based on similarity scores
- Compute the weighted average of other users’ z-scores on each item X

Either:
- denormalize and report predicted value
- or, sort and report ranked list of items
User-User algorithm: Precompute what?

To recommend items to Joe:

- Normalize all ratings by user means, standard deviations
- Compute similarity (Pearson correlation coefficient) between Joe and each other user
- Compare a set of nearest neighbors based on similarity scores
- Compute the weighted average of other users’ z-scores on each item X
- Either:
  - denormalize and report predicted value
  - or, sort and report ranked list of items
Rationale

- Similarity between users is more likely to be stable over time => it should not matter too much if you use slightly old value

- Neighborhoods decided using only similarity info=> no additional damage if they are also pre-computed

- Recent items may have many new ratings => pre-computing these would lose a lot of information
Software modules

- UI
- Reco. generation
- Indexed DB
- Similarities/model weights
- Pearson Comp.
- sort, normalize

Diagram:
- visit site
- reco. items
Recap: Term papers

- A short paper that is a mock “consultant’s report” which
  - identifies a potential application for a recommender system
  - explores the design space of a recommender system for that domain
  - suggests a design
  - points out strengths and weaknesses/pitfalls

- Due by Feb 20th (before winter break)
Case Study: Recommending email messages from a list

- Domain: email list for an online community
- How a recommender might help: guide users to interesting messages