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. }

PD-GOV **Public Domain – Government**: Works that are produced by the U.S. Government. (USC 17 § 105)

PD-EXP **Public Domain – Expired**: Works that are no longer protected due to an expired copyright term.

PD-SELF **Public Domain – Self Dedicated**: Works that a copyright holder has dedicated to the public domain.

ZERO **Creative Commons – Zero Waiver**

BY **Creative Commons – Attribution License**

BY-SA **Creative Commons – Attribution Share Alike License**

BY-NC **Creative Commons – Attribution Noncommercial License**

BY-NC-SA **Creative Commons – Attribution Noncommercial Share Alike License**

GNU-FDL **GNU – Free Documentation License**

Make Your Own Assessment

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

PD-INEL **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 **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 5:
User-User Recommender

SI583: Recommender Systems

SCHOOL OF INFORMATION
UNIVERSITY OF MICHIGAN

Generating recommendations

- Core problem: predict how much a person “Joe” (is likely to) like an item “X”
- Then, can decide to recommend most likely successes, filter out items below a threshold, etc.

user / item	A	B	C		. . .				X
Joe	7	4		4	2		5		**?**
Sue	7	5	6	5			6		**8**
John	2		3		7				**2**

User-User recommenders: Intuition

- Assumption: If Joe and another user agreed on other items, they are more likely to agree on X
- Collaborative filtering approach:
  – For each user, find how similar that user is to Joe on other ratings
  – Find the pool of users "closest" to Joe in taste
  – Use the ratings of those users to come up with a prediction

User-user algorithm: Details to be formalized

- How is similarity measured?
  - how are ratings normalized?
- How is the pool of neighbors selected?
- How are different users' ratings *weighted* in the prediction for Joe?

CF Algorithms in the Literature

- Sometimes classified as *memory-based vs. model-based*

- *Model based:* statistically predict an unknown rating
  - Fit a statistical model, then estimate
  - E.g., SVD

- *Memory-based:* ad-hoc use of previous ratings
  - *No explicit class of models, although sometimes retrofit*
  - *E.g., user-user, item-item*

Measures of similarity

user \ item	A	B	C	D	. . .				
Joe	7	3	7	3					
Sue	6	4	6	4					
John	7	7	7	7					
Amy	9	2	3	2					
Bob	7	3							

Some possible similarity metrics

For all our metrics: focus on the ratings on items that both i and j have rated

- *Similarity(i,j)* = number of items on which *i* and *j* have exactly the same rating

Some possible similarity metrics

- *Similarity(i,j)* = number of items on which *i* and *j* have the same rating
  - intuitive objection: we would have similarity(Joe,John) > similarity(Joe, Sue)

Some possible similarity metrics

- *Similarity(i,j)* = number of items on which *i* and *j* have the same rating
  - intuitive objection: we would have similarity(Joe,John) > similarity(Joe, Sue)

- *Similarity(i,j)* =

  (i's rating vector).(j's rating vector)^T

Some possible similarity metrics

- *Similarity(i,j)* = number of items on which *i* and *j* have the same rating
  – intuitive objection: we would have similarity(Joe,John) > similarity(Joe, Sue)

- *Similarity(i,j)* =
  (i's rating vector).(j's rating vector)^T
  – intuitive objection: we would have similarity(Joe,John) > similarity(Joe, Sue)

Some possibilities..

- Normalize for mean rating:
  - Let μ_i = i's average rating
  - Let *i*'s normalized rating vector

    $\boldsymbol{x_i}$ = (rating on A - μ_i , rating on B - μ_i, ...)
  - *Define similarity(i,j)* = $\boldsymbol{x_i . x_j^T}$

Mean-normalized ratings

user \ item	A	B	C	D	. . .				
Joe	2	-2	2	-2					
Sue	1	-1	1	-1					
John	0	0	0	0					
Amy	5	-2	-1	-2					
Bob	2	-2							

Some possibilities..

- Normalize for mean rating:
  - Let μ_i = i's average rating
  - Let *i*'s normalized rating vector

    $\boldsymbol{x_i}$ = (rating on A - μ_i , rating on B - μ_i, ...)
  - *Define similarity(i,j) =* $\boldsymbol{x_i . x_j^T}$

- Objection:

  similarity(Joe, Amy) > similarity(Joe, John)

Normalizing for mean and standard deviation

- Normalize for mean rating:
  - Let μ_i = i's average rating
  - Let *i*'s normalized rating vector

    $\mathbf{x}_i$ = (rating on A - μ_i , rating on B - μ_i, ...)
- Then, normalize for standard deviation
  - $\mathbf{z}_i = (1/\sigma_i)\mathbf{x}_i$,
  - where $\sigma_i = ||xi|| = \sqrt{x_i(A)^2 + x_i(B)^2 + ../(\#items\ rated\ by\ i)}$
- Define

  $similarity(i,j) = \mathbf{z}_i \cdot \mathbf{z}_j^T$

Mean-std.dev normalized ratings (z-scores)

user \ item	*A*	*B*	*C*	*D*	. . .				
Joe	1	-1	1	-1					
Sue	1	-1	1	-1					
John	0	0	0	0					
Amy	1.7	-0.7	-0.3	-0.7					
Bob	1	-1							

Normalizing for mean and standard deviation

- Normalize for mean rating:
  - Let μ_i = i's average rating
  - Let *i*'s normalized rating vector

    $\boldsymbol{x_i}$ = (rating on A - μ_i , rating on B - μ_i, ...)
- Then, normalize for standard deviation
  - $\boldsymbol{z_i} = (1/\sigma_i)\boldsymbol{x_i}$,
  - where $\sigma_i = ||xi|| = \sqrt{x_i(A)^2 + x_i(B)^2 + .. / (\#items)}$
- (Modified for different numbers of ratings):

  similarity(i,j) = $\boldsymbol{z_i \cdot z_j^T / (\#items)}$ ***[Pearson correlation coefficient]***

Pearson correlation coefficient

- Intuitively:similarity measure that
  - adjusts for different average rating for different users
  - adjusts for different swing magnitudes for different users
  - adjusts for different numbers of common ratings
- Also has a good statistical justification
  - arises naturally in a statistical model..

Correlation: Statistical justification

Statistical model:

- Item *w* drawn randomly from some space
- Each user's rating is a random variable:
  - *i*'s rating can be represented by $r_i(w)$
- *Goal: Estimate* r_{Joe}*(item) from observing* r_{Sue}*(item),* r_{John}*(item), etc..*
- If r_j is *independent* of r_i, r_j is useless for estimating r_i
- The more correlated r_j is with r_i, the more useful it is (independence => correlation = 0)
- Correlation can be estimated from common ratings

Linear Algebra Representation

- **R**: [n$\times$ m] matrix representing n users' ratings on m items
- ***X***: [n$\times$ m] matrix representing ratings normalized by user means
- **Z**: [n$\times$ m] matrix representing *z-scores* (normalized ratings)

Mathematical representation

- **R**: [n× m] matrix representing n users' ratings on m items
- ***X***: [n× m] matrix representing ratings normalized by user means
- **Z**: [n× m] matrix representing *z-scores* (normalized ratings)

If matrices are complete:

- $C=XX^T$ is an [n× n] matrix of covariances
  - C_{ij} /(#items i&j rated) estimates covariance of r_i, r_j
- $P=ZZ^T$ is an [n× n] matrix of correlations
  - P_{ij} /(#items i&j rated) estimates correlation of r_i, r_j

Other similarity measures

- Any distance measure between vectors can be used to define a similarity
- e.g., "cosine similarity"
  - treat rating vectors as lines in space, similarity based on how small the angle between *i* and *j* is
- How do you decide which one is best?

Other similarity measures

- Any distance measure between vectors can be used to define a similarity
- e.g., "cosine similarity"
  - treat rating vectors as lines in space, similarity based on how small the angle between *i* and *j* is
- How do you decide which one is best?
  - intuitively judge what normalizations are important
  - try them out empirically on your data!

User-user algorithm: Details to be formalized

- How is similarity measured?
  - how are ratings normalized?
- How is the pool of neighbors selected?
- How are different users' ratings *weighted* in the prediction for Joe?

Choosing a pool of neighbors

- Common approach: *k-nearest neighbors*
  - Pick up to k users who have rated *X*, in order of decreasing similarity to *X*
  - *parameter k is typically about 20-50*
- *Alternative: Thresholding*
  - Pick all users with correlation coefficients greater than t who have rated *X*
  - *threshold t >0 is recommended*

Weighting users

- Users' ratings on X are weighted according to computed similarities
- Prediction for Joe is weighted average
  - w_{ij} = Pearson correlation similarity(i,j)
  - predicted $z_{Joe}(x) = \sum_{i \text{ in pool}} w_{i,Joe}\, z_i(x)$
- Denormalize to compute predicted rating
  - predicted $r_{Joe(x)} = \mu_{Joe} + z_{Joe}(x)\sigma_{Joe}$

Example: Predict Joe's rating for X

user \ item	*A*	*B*	*C*	*D*	*. . .*				*X*
Joe	7	3	7	3					
Sue	6	4	6	4					4
John	7	7	7	7					
Amy	9	2	3	2					6
Bob	7	3							6

Example: z-scores

user \ item	A	B	C	D	. . .				X
Joe	1	-1	1	-1					
Sue	1	-1	1	-1					-1
John	0	0	0	0					
Amy	1.7	-0.7	-0.3	-0.7					0.8
Bob	1	-1							0.6

Example: weights and predictions

- similarity (Amy,Joe) = 0.95
- similarity (Sue,Joe) = 1
- similarity (Bob,Joe) = 1

- *predicted* $z_{Joe}(x) = -0.36$
- *predicted rating = 5 -2*0.36 = 4.22*

Recommendations [Herlocker et al, Information and Retrieval, 2002]

Table 8. A tabulation of recommendations based on the results presented in this chapter.

	Recommended	Not recommended
Similarity weighting (Section 5.1)	Pearson correlation	Spearman, entropy, vector similarity, mean-squared difference
Significance weighting (Section 5.2)	Yes	
Selecting neighbors (Section 6)	Set max number of neighbors (potentially in the range of 20–60 nbors)	Weight thresholding
Rating normalization (Section 7.1)	Deviation-from-mean or z-score	No normalization
Weighting neighbor contributions (Section 7.2)	Yes	

 Herlocker et al, Information and Retrieval, 2002