

## 6-1: Introduction to Item-Item Collaborative Filtering

Introduction to Recommender Systems

### Motivation

- User-User CF was great, except ...
- Issues of Sparsity
  - With large item sets, small numbers of ratings, too often there are points where no recommendation can be made (for a user, for an item to a set of users, etc.)
  - Many solutions proposed here, including “filterbots”, item-item, and dimensionality reduction

## Learning Objectives

- To understand the motivation, history, and intuition behind item-item CF algorithms
- To gain a basic understanding of the algorithm idea, preparing you to master the details later this module
- To understand some of the practical strengths and weaknesses of the algorithm

Introduction to Recommender Systems

### Motivation (2)

- Computational performance
  - With millions of users (or more), computing all-pairs correlations is expensive
  - Even incremental approaches were expensive
  - And user profiles could change quickly – needed to compute in real time to keep users happy

# The Item-Item Insight

- Item-Item similarity is fairly stable ...
  - This is dependent on having many more users than items
    - Average item has many more ratings than an average user
    - Intuitively, items don't generally change rapidly – at least not in ratings space (special case for time-bound items)
- Item similarity is a route to computing a prediction of a user's item preference

Introduction to Recommender Systems

## Item-Item Top-N

- Item-Item similarity model can be used to compute top-N directly:
  - Simplify model by limiting items to small “neighborhoods” of  $k$  most-similar items (e.g., 20)
  - For a profile set of items, compute/merge/sort the  $k$ -most similar items for each profile item
    - Straightforward matrix operation from Deshpande and Karypis

Introduction to Recommender Systems

# A little more detail ...

- Two step process:
  - Compute similarity between pairs of items
    - Correlation between rating vectors
      - co-rated cases only (only useful for multi-level ratings)
    - Cosine of item rating vectors
      - can be used with multi-level or unary ratings
    - Adjusted cosine (normalize each user's ratings)
      - to adjust for differences in rating scales
    - Some use conditional probability (unary)
  - Predict user-item rating
    - Weighted sum of rated “item-neighbors”
    - Linear regression to estimate rating

Introduction to Recommender Systems

## Benefits of Item-Item

- It actually works quite well
  - Good MAE performance on prediction; good rank performance on top-N
- Efficient implementation
  - At least in cases where  $|U| \gg |I|$
  - Benefits of precomputability
- Broad applicability and flexibility
  - As easy to apply to a shopping cart as to a user profile

Introduction to Recommender Systems

# Core Assumptions/Limitations

- Item-item relationships need to be stable ...
  - Mostly just a corollary of stable user preferences
  - Could have special cases that are difficult (e.g., calendars, short-lived books, etc.)
  - Many of these issues are general temporal issues
- Main limitation/complaint: lower serendipity
  - This is a user/researcher complaint, not fully studied; intuition is clear

Introduction to Recommender Systems

# Moving Forward

- Next Lectures
  - Breaking down the core item-item algorithm
  - Looking at the special cases of unary/binary ratings
  - Programming item-item (for programmers)

Introduction to Recommender Systems

## 6-1: Introduction to Item-Item Collaborative Filtering

	1	2	3	4	5	
ann		5	3	1		5 1
bob	4	5	3			1 5
chen	2	1	3	5	2 2	1 4
doris		2	3			
eddie	5	1	2	4		5 3
						2 3
						1 2