

## CSCE 670 - Information Storage and Retrieval

# Lecture 11: Recommender Systems (Collaborative Filtering)

Yu Zhang

yuzhang@tamu.edu

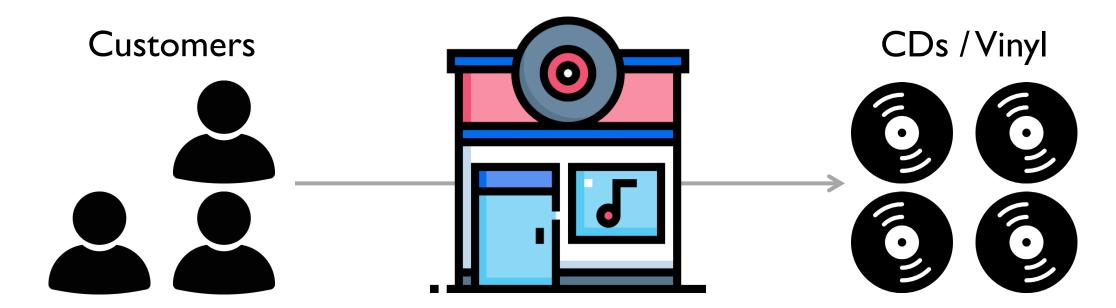
September 30, 2025

Course Website: https://yuzhang-teaching.github.io/CSCE670-F25.html

#### Recap: Phase 1

- Phase I: Search Engines
  - basics, Boolean and ranked retrieval, link analysis, evaluation, learning to rank (ML + ranking), ...
- Phase 2: Recommender Systems
  - basics, non-personalized recommendation, collaborative filtering, matrix factorization, implicit recommendation, ...
- Phase 3: From Foundations to Modern Methods
  - embedding learning, Transformer, "small" language models, ... (for search and recommendation)
- Phase 4: Large Language Models (!!)

#### Our Capabilities So Far

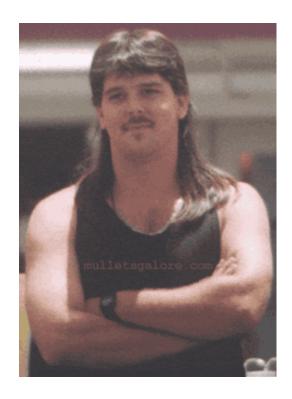


- Given a query, find relevant CDs
  - Exact matching (Boolean, phrase, proximity, wildcard)
  - Ranked retrieval (TF-IDF, BM25, learning to rank)
  - Link analysis (PageRank, HITS)

#### Phase 2

- Phase I: Search Engines
  - basics, Boolean and ranked retrieval, link analysis, evaluation, learning to rank (ML + ranking), ...
- Phase 2: Recommender Systems
  - basics, non-personalized recommendation, collaborative filtering, matrix factorization, implicit recommendation, ...
- Phase 3: From Foundations to Modern Methods
  - embedding learning, Transformer, "small" language models, ... (for search and recommendation)
- Phase 4: Large Language Models (!!)

## Example of Recommender Systems

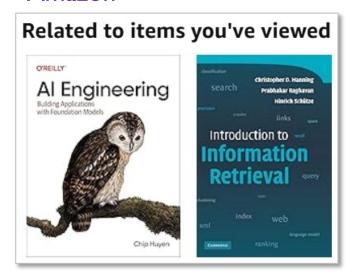


- Customer X
  - Buys Metallica CD
  - Buys Megadeth CD

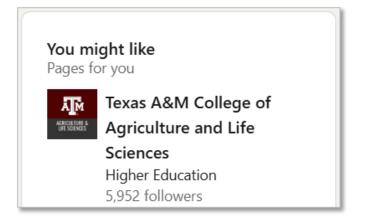


- Customer Y
  - Does search on Metallica
  - Recommender systems should suggest Megadeth from data collected about Customer X

#### Amazon



#### LinkedIn



#### Example of Recommender Systems

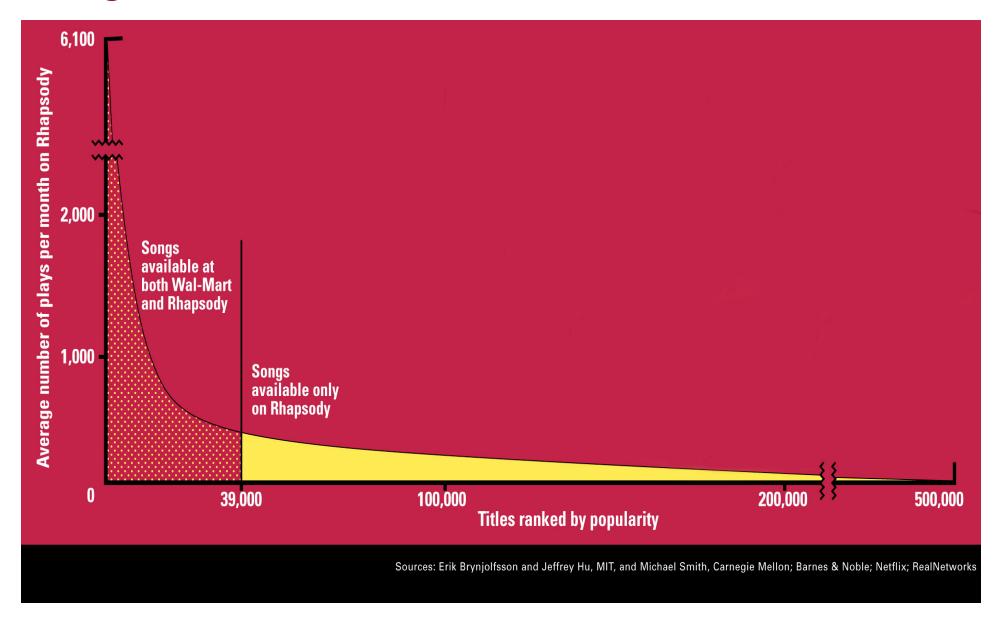




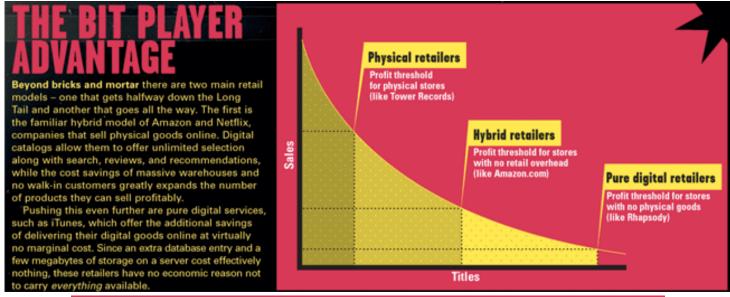
#### Why do we need recommendation?

- Shelf space is a scarce commodity for traditional retailers
  - Also:TV networks, movie theaters,...
- Web enables near-zero-cost dissemination of information about products
  - From scarcity to abundance
- More choice necessitates better filters
  - How Into Thin Air (published in 1997) made Touching the Void (published in 1988) a bestseller
  - Touching the Void did not become a bestseller until a similar bestseller book appears 9
    years later.
  - Amazon's recommendation engine

## The Long Tail



## The Long Tail





#### Types of Recommendations

- Editorial and hand-curated (not personalized)
  - "Store Manager's Pick"
  - Promoted items
- Simple aggregates (not personalized)
  - Most liked/clicked this month/week/day
  - Most recent
- Personalized approaches
  - Collaborative Filtering (today)
  - Matrix Factorization (Oct 2 and Oct 7)
  - Bayesian Personalized Ranking (Oct 9)

#### Formal Setup

- $\mathcal{X}$ : A set of users
- S:A set of items
- Utility function  $u: \mathcal{X} \times \mathcal{S} \to \mathcal{R}$ 
  - $\mathcal{R}$  = set of ratings, which is a totally ordered set
    - E.g., I-5 stars
    - E.g., real number in [0,1]

# Utility Matrix U

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

#### Key Problems

- Problem I: Gathering "known" ratings for matrix
  - How to collect the data in the utility matrix?
  - Evaluating the quality of an item solely based on its average rating?
- Problem 2: Extrapolate unknown ratings from the known ones
  - Mainly interested in high unknown ratings
  - We are not interested in knowing what you don't like but what you like
- Problem 3: Evaluating extrapolation methods
  - How to measure success/performance of recommendation methods?

## Problem 1: Gathering Ratings

- Explicit
  - Ask people to rate items
  - Doesn't work well in practice people can't be bothered
- Implicit
  - Learn ratings from user actions
  - E.g., purchase implies high rating
  - What about low ratings?

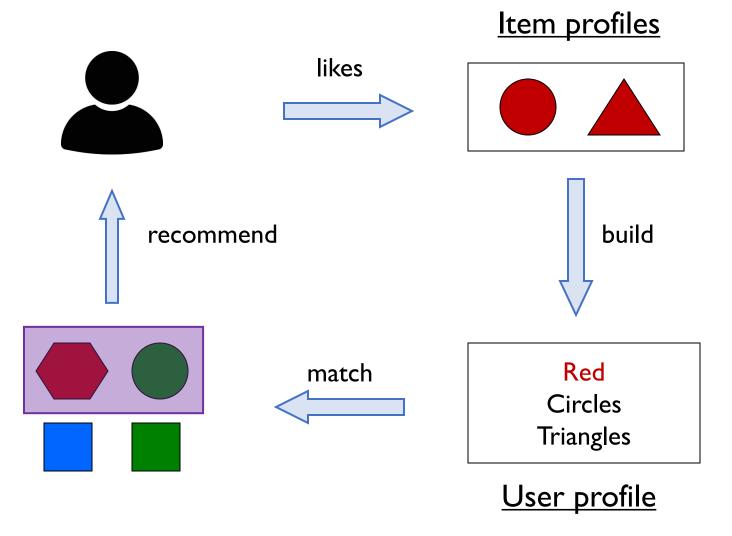
## Problem 2: Extrapolating Utilities

- Key problem: Utility matrix *U* is sparse
  - Most people have not rated most items
  - Cold start:
    - New items have no ratings
    - New users have no history
- Solutions to be introduced today:
  - Content-Based Approach
  - Collaborative Filtering

# Content-Based Approach (Calculating User-Item Similarity)

#### Content-Based Recommender Systems

- Idea: Recommend items to user x similar to previous items rated highly by x
- Example:
  - Movie recommendations:
     Recommend movies with same actor(s), director, genre, ...
  - Websites, blogs, news:
     Recommend other sites
     with "similar" content



#### **Profiles**

- Item Profile *i* 
  - A set (vector) of features
  - Movies: author, title, actor, director, ...
  - Text: Set of "important" words in document (e.g., based on TF-IDF)
- User Profile x
  - Weighted average of rated item profiles
- Prediction
  - Given x and i, estimate  $u(x, i) = \cos(x, i) = \frac{x^T i}{\|x\| \cdot \|i\|}$

#### Example

- Rating scale:  $\mathcal{R} = \{1,0,-1\}$
- User x has rated 3 songs
  - Song I: "sunny day", rating: 1
  - Song 2: "cloudy day", rating: 0
  - Song 3: "rainy day", rating: -1
- Let's simplify the model and use Boolean representation (rather than TF-IDF)
- Item profiles

	sunny	cloudy	rainy	day
Song I	I	0	0	1
Song 2	0	I	0	I
Song 3	0	0	I	I

#### Example

- User x has rated 3 songs
  - Song I: "sunny day", rating: 1
  - Song 2: "cloudy day", rating: 0
  - Song 3: "rainy day", rating: -1
- Item profiles

	sunny	cloudy	rainy	day
Song I	I	0	0	Ī
Song 2	0	I	0	I
Song 3	0	0	I	I

• User profile: User  $x = 1 \times \text{Song I} + 0 \times \text{Song 2} + (-1) \times \text{Song 3}$ 

	sunny	cloudy	rainy	day
User x	I	0	-1	0

#### Example

• User profile: User  $x = 1 \times \text{Song I} + 0 \times \text{Song 2} + (-1) \times \text{Song 3}$ 

	sunny	cloudy	rainy	day
User x	I	0	-1	0

- New Song: "sunny cloudy"
- Will User x like it?
- Profile of New Song

	sunny	cloudy	rainy	day
User x	I	I	0	0

• 
$$u(User x, New Song) = cos \begin{pmatrix} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{pmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2} \times \sqrt{2}} = \frac{1}{2}$$

#### Content-Based Recommender Systems: Pros

- + No need for data on other users
  - No cold-start or sparsity problems
- + Able to recommend to users with unique tastes
- + Able to recommend new & unpopular items
  - No first-rater problem
- + Able to provide explanations
  - Can provide explanations of recommended items by listing content-features that caused an item to be recommended

#### Content-Based Recommender Systems: Cons

- - Finding the appropriate features is hard
  - E.g., images, movies, music
- - Recommendations for new users
  - How to build a user profile?
- - Overspecialization
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - Unable to exploit quality judgments of other users

# Collaborative Filtering (Harnessing Quality Judgments of Other Users)

## Collaborative Filtering

- Consider user x
- Step I: Find a set  $\mathcal{N}$  of other users whose ratings are "similar" to x's ratings
- Step 2: Estimate x's ratings based on ratings of users in  $\mathcal{N}$



#### Step 1: Finding Similar Users

- How to define User-User similarity?
- Example:

	ltem l	Item 2	Item 3	Item 4	Item 5
User x	*			*	***
User y	*		**	**	

• (Bad) Solution I: Jaccard Similarity (between two sets)

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

$$x: \{1, 4, 5\},$$
  $y: \{1, 3, 4\},$   $J(x, y) = \frac{1}{4}$ 

Problem: Ignores the value of the rating

## Step 1: Finding Similar Users

- How to define User-User similarity?
- Example:

	ltem l	Item 2	Item 3	Item 4	Item 5
User x	*			*	***
User y	*		**	**	

- (Bad) Solution 2: Cosine Similarity (between two vectors)
  - $x: [1, 0, 0, 1, 3]^T$
  - $y: [1, 0, 2, 2, 0]^T$
  - Problem: Treats missing ratings as "negative"

## Step 1: Finding Similar Users

- How to define User-User similarity?
- Example:

	ltem l	Item 2	Item 3	Item 4	Item 5
User x	*			*	***
User y	*		**	**	

- Solution 3: Pearson Correlation Coefficient
  - Consider  $S_{xy}$  = items rated by both users x and y

$$sim(x,y) = \frac{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)(U_{yi} - \overline{U}_y)}{\sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)^2} \sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{yi} - \overline{U}_y)^2}}$$

 $\overline{U}_x$ ,  $\overline{U}_y$ : average rating given by x and y

#### How to understand the Pearson Correlation Coefficient?

$$sim(x,y) = \frac{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)(U_{yi} - \overline{U}_y)}{\sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)^2} \sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{yi} - \overline{U}_y)^2}}$$

Original Table

	ltem l	Item 2	Item 3	Item 4	Item 5
User x	*			*	***
User y	*		**	**	

• Step I: Subtract the (row) mean

	ltem l	Item 2	Item 3	Item 4	Item 5
User x	1-5/3=-2/3			1-5/3=-2/3	3-5/3=4/3
User y	1-5/3=-2/3		2-5/3=1/3	2-5/3=1/3	

#### How to understand the Pearson Correlation Coefficient?

$$sim(x,y) = \frac{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)(U_{yi} - \overline{U}_y)}{\sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{xi} - \overline{U}_x)^2} \sqrt{\sum_{i \in \mathcal{S}_{xy}} (U_{yi} - \overline{U}_y)^2}}$$

Original Table

	Item I	Item 2	Item 3	Item 4	Item 5
User x	*			*	***
User y	*		**	**	

 Step 2: Only keep the column rated by both x and y

	ltem l	Item 4
User x	1-5/3=-2/3	1-5/3=-2/3
User y	1-5/3=-2/3	2-5/3=1/3

- Step 3: Calculate the Cosine Similarity
  - Pearson is equivalent to Cosine after some data normalization steps!

#### Step 2: Rating Prediction

- Let  $\mathcal N$  be the set of k users that are most similar to x who have rated item i
- Prediction for item *i* of user *x*:
  - Simple average:

$$U_{xi} = \frac{1}{k} \sum_{y \in \mathcal{N}} U_{yi}$$

Weighted by User-User similarity:

$$U_{xi} = \frac{\sum_{y \in \mathcal{N}} sim(x, y) \cdot U_{yi}}{\sum_{y \in \mathcal{N}} sim(x, y)}$$

- Many other tricks possible...
- So far, User-User Collaborative Filtering
  - Using User-User similarity to predict User-Item similarity

#### How about Item-Item Collaborative Filtering?

- Using Item-Item similarity to predict User-Item similarity
- Step I: For item i, find other similar items
- Step 2: Estimate rating for item i based on ratings for similar items
- Can use same similarity metrics and prediction functions as in user-user model
  - E.g., Weighted by Item-Item similarity:

$$U_{xi} = \frac{\sum_{j \in \mathcal{N}} sim(i, j) \cdot U_{xj}}{\sum_{j \in \mathcal{N}} sim(i, j)}$$

sim(i,j): Pearson Correlation Coefficient between item i and item j  $\mathcal{N}$ : the set of items rated by x that are similar to i

#### users

		1	2	3	4	5	6	7	8	9	10	11	12
CDs	1	1		3			5			5		4	
	2			5	4			4			2	1	3
	3	2	4		1	2		3		4	3	5	
	4		2	4		5			4			2	
	5			4	3	4	2					2	5
	6	1		3		3			2			4	
			- u	ınkno	wn ra	ting		- rating between 1 to 5					

#### users

		1	2	3	4	5	6	7	8	9	10	11	12
CDs	1	1		3		?	5			5		4	
	2			5	4			4			2	1	3
	3	2	4		1	2		3		4	3	5	
	4		2	4		5			4			2	
	5			4	3	4	2					2	5
	6	1		3		3			2			4	



- estimate rating of CD I by user 5

#### users

		1	2	3	4	5	6	7	8	9	10	11	12	$sim(1,\cdot)$
	1	1		3		?	5			5		4		1.00
	2			5	4			4			2	1	3	-0.18
CDs	<u>3</u>	2	4		1	2		3		4	3	5		0.41
	4		2	4		5			4			2		-0.10
	5			4	3	4	2					2	5	-0.31
	<u>6</u>	1		3		3			2			4		0.59

Neighbor selection: Identify movies that are similar to CD 1, rated by user 5

#### users

		1	2	3	4	5	6	7	8	9	10	11	12	$sim(1,\cdot)$
	1	1		3		2.6	5			5		4		1.00
	2			5	4			4			2	1	3	-0.18
CDs	<u>3</u>	2	4		1	2		3		4	3	5		0.41
	4		2	4		5			4			2		-0.10
	5			4	3	4	2					2	5	-0.31
	<u>6</u>	1		3		3			2			4		0.59

Predict by taking weighted average:

$$U_{51} = (0.41 \times 2 + 0.59 \times 3) / (0.41 + 0.59) = 2.6$$

## Collaborative Filtering: Common Practice

• So far,

$$U_{xi} = \frac{\sum_{j \in \mathcal{N}} sim(i, j) \cdot U_{xj}}{\sum_{j \in \mathcal{N}} sim(i, j)}$$

• In practice,

$$U_{xi} = b_{xi} + \frac{\sum_{j \in \mathcal{N}} sim(i,j) \cdot (U_{xj} - b_{xj})}{\sum_{j \in \mathcal{N}} sim(i,j)}$$

- $b_{xi}$ : baseline estimate for  $U_{xi}$   $(b_{xi} = \mu + b_x + b_i)$
- $\mu$ : overall mean CD rating
- $b_x$ : rating deviation of user x, which is the (avg. rating given by user x)  $\mu$
- $b_i$ : rating deviation of item i, which is the (avg. rating given to item i)  $\mu$

#### Item-Item vs. User-User

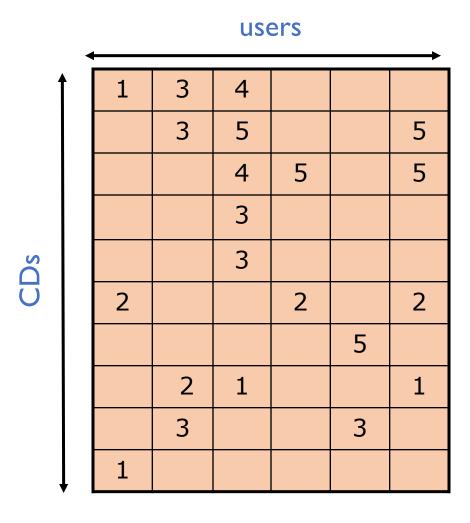
<ul> <li>In practice, it has been observed that item-item often works better than user-user!</li> </ul>		Avatar	LOTR	Matrix	Pirates
<ul> <li>Why? Items are simpler, users have multiple tastes</li> </ul>	Alice	1		0.8	
<ul> <li>It is impossible for a piece of music to be both 60's rock and 1700's baroque.</li> </ul>	Bob		0.5		0.3
<ul> <li>There are individuals who like both 60's rock and 1700's</li> </ul>	Carol	0.9		1	0.8
baroque, and who buy examples of both types of music.	David			1	0.4

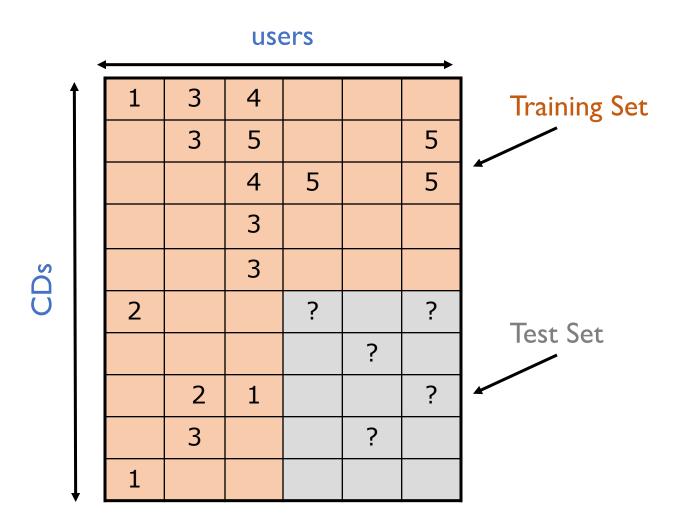
#### Collaborative Filtering: Pros and Cons

- + Works for any kind of item
  - No feature selection (e.g., text information) needed
- - Cold start
  - Need enough users in the system to find a match
- - Sparsity
  - The user/ratings matrix is sparse
  - Hard to find users that have rated the same items
- - First rater
  - Cannot recommend an item that has not been previously rated
  - New items & esoteric items
- - Popularity bias
  - Cannot recommend items to someone with unique taste
  - Tends to recommend popular items

#### Hybrid Methods

- Implement two or more different recommenders (e.g., content-based and collaborative filtering) and combine predictions
  - Perhaps using a linear model
  - "Learning to Recommend"
- Add content-based approaches to collaborative filtering
  - Building item profiles to deal with the new item problem
  - Building demographics to deal with the new user problem





- Compare predictions with *known* ratings. (There are many predictions whose ground truth is *unknown*.)
  - Root-mean-square error (RMSE)
    - $\sqrt{\frac{1}{M}\sum_{x,i}(U_{xi}-U_{xi}^*)^2}$  where  $U_{xi}$  is predicted, and  $U_{xi}^*$  is the true rating of x on i; M is the number of testing samples
  - Precision@10
    - Among the top 10 items with known ratings, how many are relevant (e.g., ≥4 stars)

10

- nDCG@10
- Recall@10
  - Among the top 10 items with known ratings, how many are relevant (e.g., ≥4 stars)
    Among ALL items with known ratings, how many are relevant (e.g., ≥4 stars)

#### Final Comments

- Problem with RMSE: In practice, our main interest is in accurately predicting high ratings.
  - It is far more important to predict whether you would give 5 stars or 4 stars to a CD you might like than to predict whether you would give 2 stars or 1 star to one you dislike.
  - However, RMSE may penalize methods that perform well on high ratings but poorly on others.
- Tip: Leverage ALL the data
  - Don't try to reduce data size in an effort to make fancy algorithms work
  - Simple methods on large data do best
  - More data beats better algorithms: <a href="http://anand.typepad.com/datawocky/2008/03/more-data-usual.html">http://anand.typepad.com/datawocky/2008/03/more-data-usual.html</a>



#### Thank You!

Course Website: https://yuzhang-teaching.github.io/CSCE670-F25.html