

机器学习与数据挖掘

Machine Learning & Data Mining

权小军 教授

中山大学数据科学与计算机学院

quanxj3@mail.sysu.edu.cn

浅谈搜索与推荐



浅谈搜索与推荐



今日头条
你关心的 才是头条



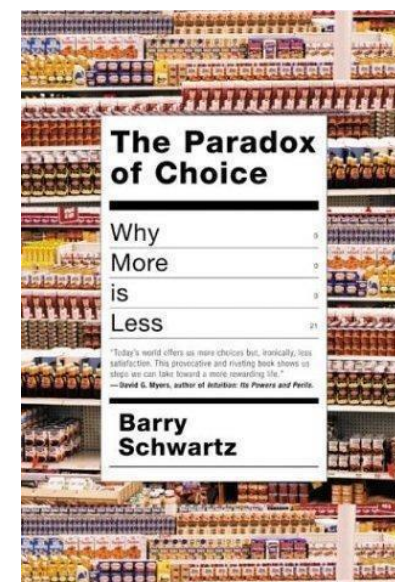
JD.COM 京东

Lecture 18

Recommender Systems I

1. What is a Recommender System?

Information overload



Information overload occurs when you have too much information about a task or issue to effectively understand it, or make a decision on how to solve it.

Information overload

Here are some more highlights regarding the information produced online (Bonardi, 2018):

- 2.5 quintillion bytes of data are produced every day;
- It would take an average user 200,000 years to read all information produced online;
- Over 103 million spam emails are sent every day;
- 456,000 tweets are sent daily.

Search Engine and Recommender System

The value of recommendations

- Netflix: 2/3 of the movies watched are recommended
- Google News: recommendations generate 38% more clickthrough
- Amazon: 35% sales from recommendations
- Choicestream: 28% of the people would buy more music if they found what they liked.

[Japan Halts US Beef Imports After Banned Meat Found \(update\)](#)

Bloomberg - 1 hour ago
Jan. 20 (Bloomberg) -- Japan stopped imports of beef from the US after inspectors found banned cattle parts in a shipment, disrupting trade that resumed last month following a two-year halt because of mad-cow disease. ...
[Japan halts US beef imports due to fears of mad cow](#) San Diego Union Tribune
[US to probe beef shipment to Japan](#) San Jose Mercury News
[Boston Globe](#) - [Guardian Unlimited](#) - [MarketWatch](#) - [CNN](#) - [all 1,045 related »](#)

[Recommended for gprice@gmail.com »](#)



[Serena in denial over her terminal decline](#)

Guardian Unlimited - 7 hours ago - It was in Australia eight years ago that the Williams sisters were seen competing at the same grand slam for the first ...
[International Herald Tribune](#) - [TennisReporters.net](#) - [Forbes](#) - [all 319 related »](#)

[2 dozen hurt in Tel Aviv bombing](#)

San Francisco Chronicle - 20 hours ago - Jerusalem -- At least two dozen Israelis were wounded Thursday when a suicide bomber detonated explosives he was ...
[Los Angeles Times](#) - [Detroit Free Press](#) - [San Jose Mercury News](#) - [all 836 related »](#)

[US plans to shift diplomats to developing countries](#)

Boston Globe - Jan 19, 2006 - By Farah Stockman, Globe Staff | January 19, 2006. WASHINGTON -- Secretary of State Condoleezza Rice announced ...
[International Herald Tribune](#) - [Sydney Morning Herald](#) - [Financial Times](#) - [all 70 related »](#)

[Phone Cancer Link Downplayed](#)

Red Herring - all 170 related »

['American Idol' Gets a Little Mean](#)

Ceres Courier - all 575 related »

[Deadline to kill US journalist passes with no news](#)

Khaleej Times - all 2,858 related »

[From here, Oscar race goes inside Hollywood](#)

Reuters - all 114 related »

[NASA starry-eyed at comet's samples](#)

Houston Chronicle - all 219 related »

[REGION: Annan urges Iran to resume talks with EU](#)

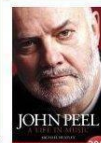
Daily Times - all 1,382 related »

[In The News](#)

Osama bin Laden - [Midnight Hour](#)
Albert Brooks - [Mustang Sally](#)
Tel Aviv - [Afr Sahara](#)
Jet Airways - [Mehmet Ali Agca](#)
Wilson Pickett - [Jill Carroll](#)

John Peel: A Life in Music

Michael Heatley



List Price: £6.99
Our Price: **£5.59** & eligible for **Free UK delivery** on orders over £15 with Super Saver Delivery. See [details & conditions](#).
You Save: £1.40 (20%)

Availability: usually dispatched within 24 hours.

27 Used & New from £1.60

[Publisher: learn how customers can search inside this book.](#)

[See larger photo](#)

Edition: Paperback

[More Product Details](#)

Perfect Partner

Buy **John Peel: A Life in Music** with **Margrave Of The Marshes** today!

Total List Price: £25.98
Buy Together Today: **£16.98**

[Buy both now](#)

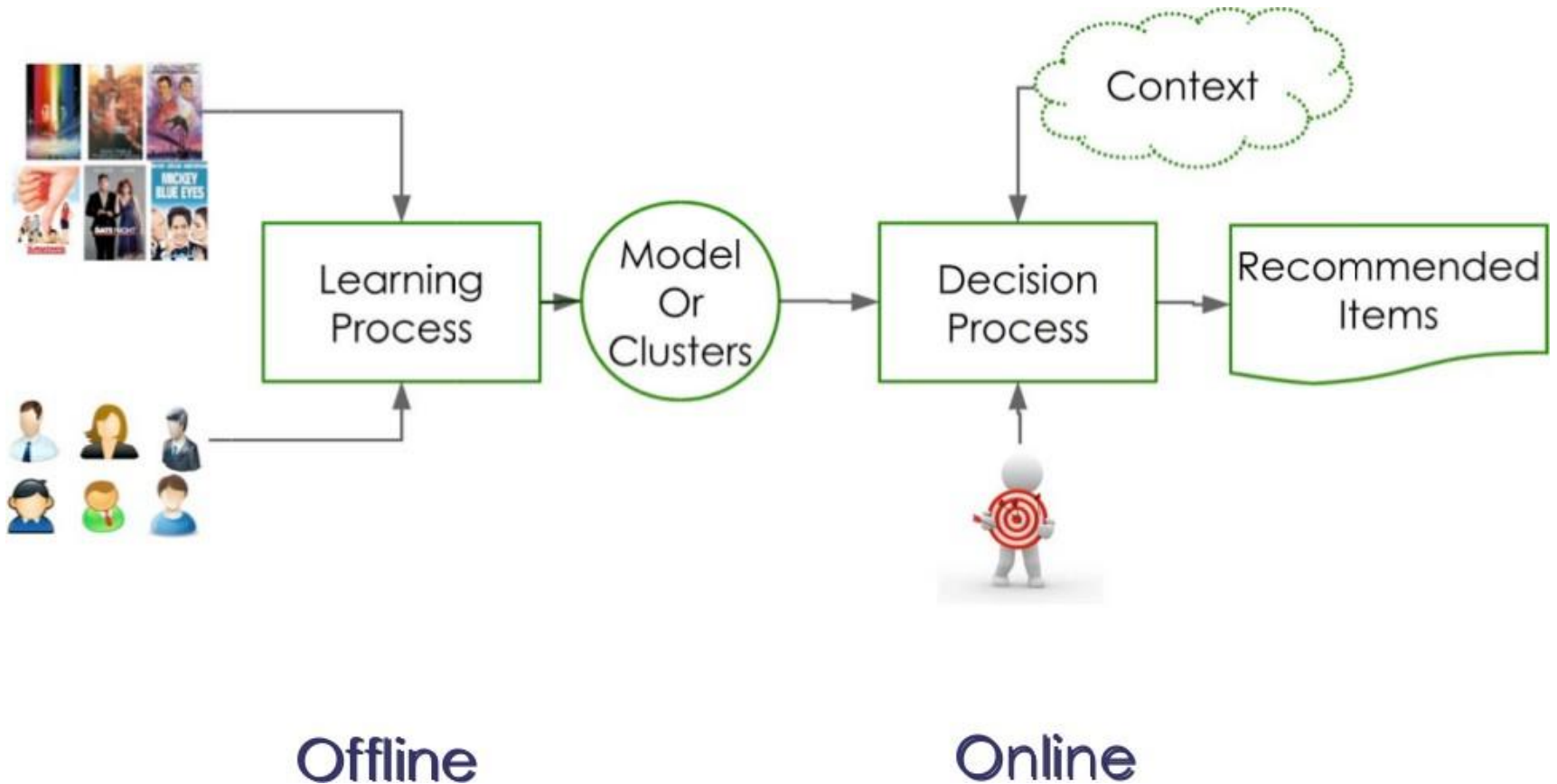
Customers who bought this item also bought:

- [The Little Book of Wanking: The Definitive Guide to Man's Ultimate Relief](#); Paperback ~ Dick Palmer
- [Shag Yourself Slim: The Most Enjoyable Way to Lose Weight](#); Paperback ~ Imrah Goer
- [Gummy Old Men, the Official Handbook](#); Hardcover ~ Stuart Prebble
- [The Little Book of Minge Topiary](#); Paperback ~ Michael O'Mara

The “Recommender problem”

- Estimate a utility function that automatically predicts how a user will like an item.
- Based on:
 - Past behavior
 - Relations to other users
 - Item similarity
 - Context
 - ...

Two-step process

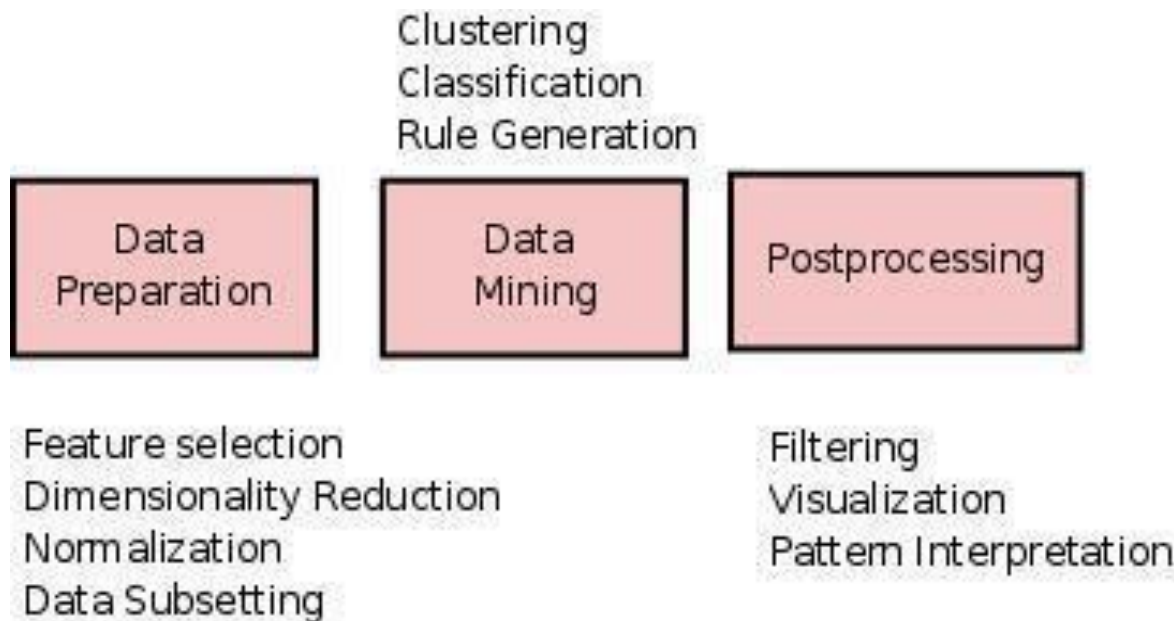


Approaches to Recommendation

- ❑ **Collaborative Filtering:** Recommend items based only on the users past behavior
 - **User-based:** Find similar users to me and recommend what they liked
 - **Item-based:** Find similar items to those that I have previously liked
- ❑ **Content-based:** Recommend based on item features
- ❑ **Personalized Learning to Rank:** Treat recommendation as a ranking problem
- ❑ **Demographic:** Recommend based on user features
- ❑ **Social recommendations** (trust-based)
- ❑ **Hybrid:** Combine any of the above

Recommendation as data mining

The core of the Recommendation Engine can be assimilated to a general **data mining** problem:



Formal Model

- X = set of **Customers**
- S = set of **Items**
- **Utility function** $u: X \times S \rightarrow R$
 - R = set of ratings
 - R is a totally ordered set
 - e.g., **0-5** stars, real number in **[0,1]**

Utility Matrix

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

Key Problems

- **(1) Gathering “known” ratings for matrix**
 - How to collect the data in the utility matrix
- **(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
- **(3) Evaluating extrapolation methods**
 - How to measure success/performance of recommendation methods

(1) Gathering Ratings

■ Explicit

- Ask people to rate items
- Doesn't work well in practice – people can't be bothered
- Crowdsourcing: Pay people to label items

■ Implicit

- Learn ratings from user actions
 - E.g., purchase implies high rating
- What about low ratings?

(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
- **Three approaches to recommender systems:**
 - 1) Content-based
 - 2) Collaborative
 - 3) Latent factor based

} **Today!**

Content-based Recommender Systems

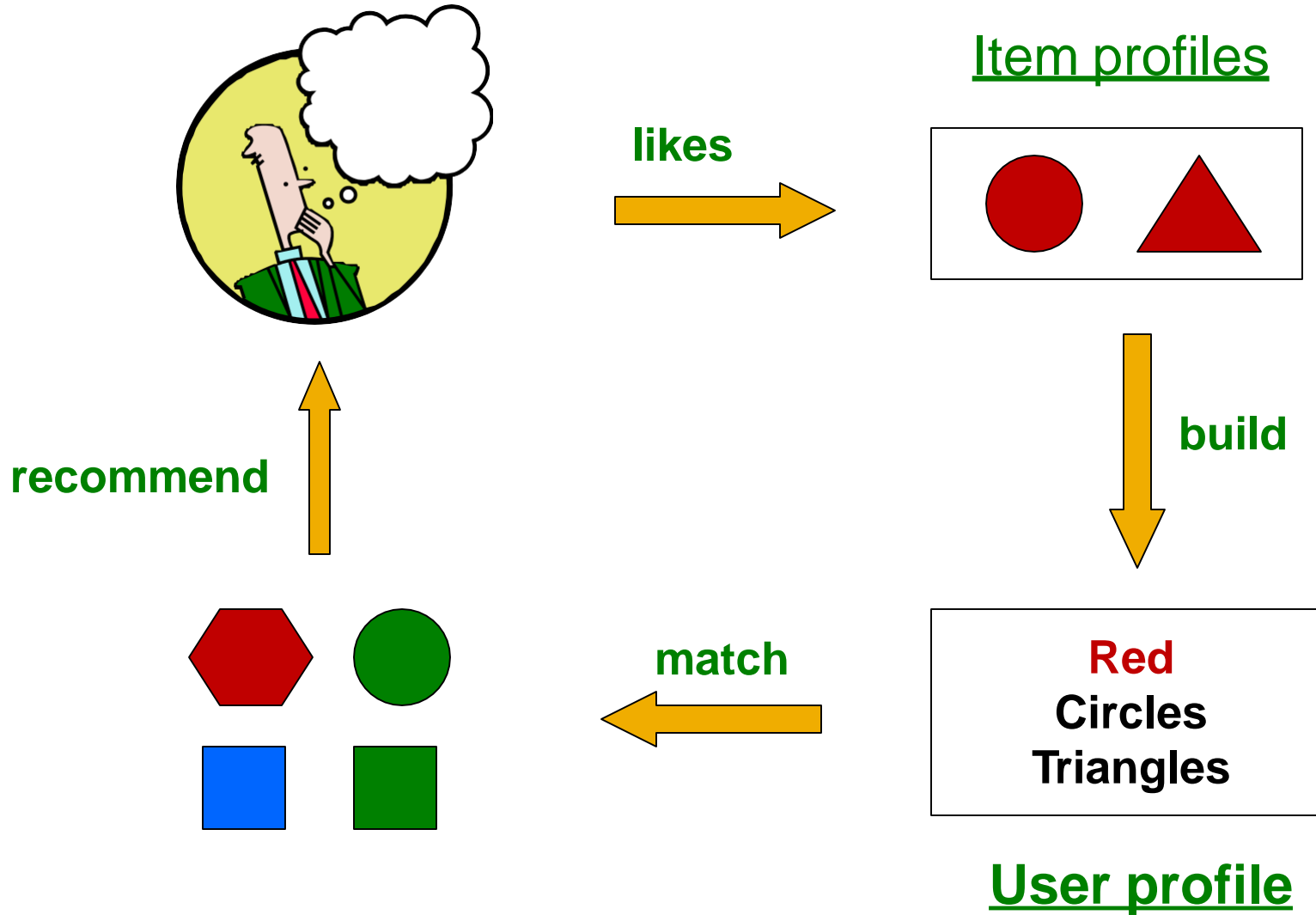
Content-based Recommendations

- **Main idea:** Recommend items to customer 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

Plan of Action



Item Profiles

- For each item, create an **item profile**
- **Profile is a set (vector) of features**
 - **Movies:** author, title, actor, director,...
 - **Text:** Set of “important” words in document
- **How to pick important features?**
 - Usual heuristic from text mining is **TF-IDF**
(Term frequency * Inverse Doc Frequency)
 - **Term ... Feature**
 - **Document ... Item**

TF-IDF

f_{ij} = frequency of term (feature) i in doc (item) j

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$

Note: we normalize TF to discount for “longer” documents

n_i = number of docs that mention term i

N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Doc profile = set of words with highest **TF-IDF** scores, together with their scores

User Profiles and Prediction

■ User profile possibilities:

- Weighted average of rated item profiles
- **Variation:** weight by difference from average rating for item
- ...

■ Prediction heuristic:

- Given user profile \mathbf{x} and item profile \mathbf{i} , estimate

$$u(\mathbf{x}, \mathbf{i}) = \cos(\mathbf{x}, \mathbf{i}) = \frac{\mathbf{x} \cdot \mathbf{i}}{||\mathbf{x}|| \cdot ||\mathbf{i}||}$$

Pros: Content-based Approach

- +: No need for data on other users
- +: 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

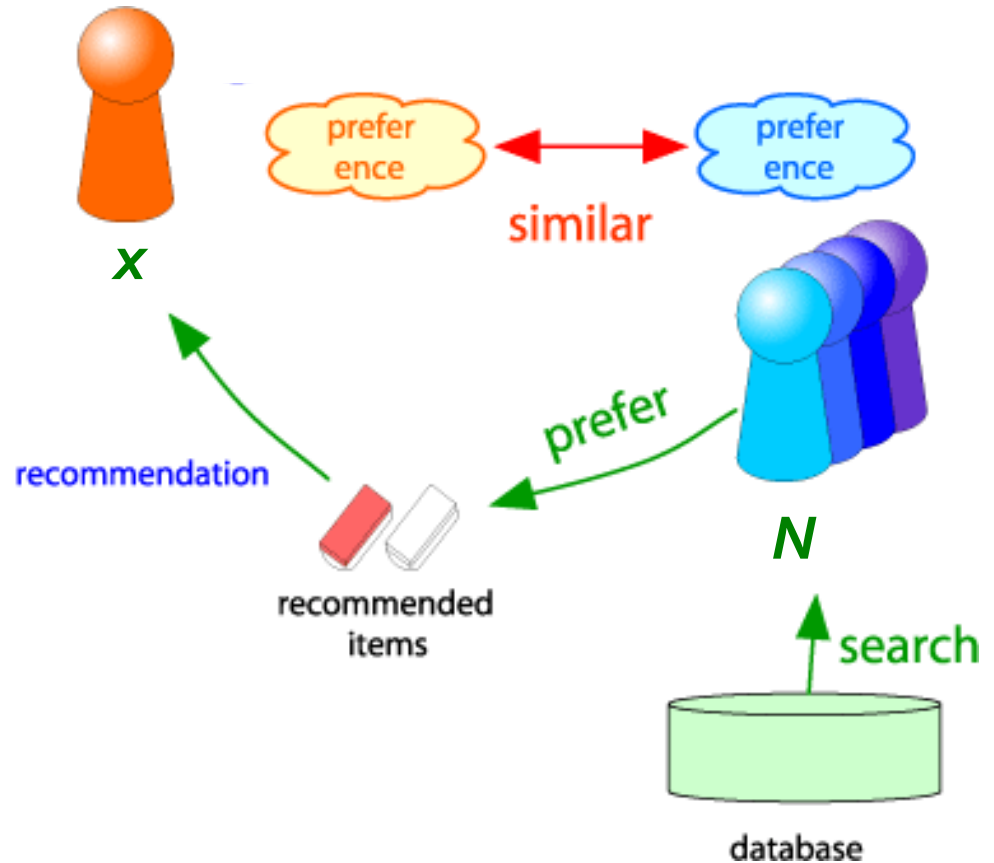
Cons: Content-based Approach

- —: 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

Collaborative Filtering

- Consider user x
- Find set N of other users whose ratings are “similar” to x ’s ratings
- Estimate x ’s ratings based on ratings of users in N



Finding “Similar” Users

$$r_x = \begin{bmatrix} * & _ & _ & * & *** \end{bmatrix}$$
$$r_y = \begin{bmatrix} * & _ & ** & ** & _ \end{bmatrix}$$

- Let r_x be the vector of user x 's ratings

- **Jaccard similarity measure**

- **Problem:** Ignores the value of the rating

r_x, r_y as sets:

$$r_x = \{1, 4, 5\}$$

$$r_y = \{1, 3, 4\}$$

- **Cosine similarity measure**

- $\text{sim}(x, y) = \arccos(r_x, r_y) = \frac{r_x r_y}{\|r_x\| \|r_y\|}$

r_x, r_y as points:

$$r_x = \{1, 0, 0, 1, 3\}$$

$$r_y = \{1, 0, 2, 2, 0\}$$

- **Problem:** Treats missing ratings as “negative”

- **Pearson correlation coefficient**

- S_{xy} = items rated by both users x and y

$$\text{sim}(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)(r_{ys} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \bar{r}_y)^2}}$$

Rating Predictions

From similarity metric to recommendations:

- Let \mathbf{r}_x be the vector of user \mathbf{x} 's ratings
- Let \mathbf{N} be the set of k users most similar to \mathbf{x} who have rated item i
- **Prediction for item s of user x :**

- $$r_{xi} = \frac{1}{k} \sum_{y \in \mathbf{N}} r_{yi}$$

- $$r_{xi} = \frac{\sum_{y \in \mathbf{N}} s_{xy} \cdot r_{yi}}{\sum_{y \in \mathbf{N}} s_{xy}}$$

Shorthand:

$$s_{xy} = \text{sim}(\mathbf{x}, \mathbf{y})$$

- Other options?

- **Many other tricks possible...**

Item-Item Collaborative Filtering

- So far: **User-user collaborative filtering**
- **Another view: Item-item**
 - For item i , find other similar items
 - Estimate rating for item i based on ratings for similar items
 - Can use same similarity metrics and prediction functions as in user-user model

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij} ... similarity of items i and j
 r_{xj} ... rating of user u on item j
 $N(i;x)$... set items rated by x similar to i

Item-Item CF ($|N|=2$)

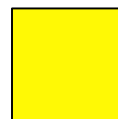
users

movies

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



- unknown rating



- rating between 1 to 5

Item-Item CF ($|N|=2$)

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12
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 movie **1** by user **5**

Item-Item CF ($|N|=2$)

		users													
		1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)	
movies	1	1		3		?	5			5		4		1.00	
	2			5	4			4			2	1	3	-0.18	
	<u>3</u>	2	4		1	2		3		4	3	5		<u>0.41</u>	
	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		<u>0.59</u>	

Neighbor selection:
Identify movies similar to
movie 1, rated by user 5

Here we use Pearson correlation as similarity:

- 1) Subtract mean rating m_i from each movie i
 $m_1 = (1+3+5+5+4)/5 = 3.6$
row 1: [-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]
- 2) Compute cosine similarities between rows

Item-Item CF ($|N|=2$)

		users												sim(1,m)
		1	2	3	4	5	6	7	8	9	10	11	12	
movies	1	1		3		?	5			5		4		1.00
	2			5	4			4			2	1	3	-0.18
	<u>3</u>	2	4		1	2		3		4	3	5		<u>0.41</u>
	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		<u>0.59</u>

Compute similarity weights:

$$s_{1,3}=0.41, s_{1,6}=0.59$$

Item-Item CF ($|N|=2$)

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3		2.6	5			5		4	
2			5	4			4			2	1	3
<u>3</u>	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
<u>6</u>	1		3		3			2			4	

Predict by taking weighted average:

$$r_{1.5} = (0.41 \cdot 2 + 0.59 \cdot 3) / (0.41 + 0.59) = 2.6$$

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

CF: Common Practice

Before:
$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

- Define **similarity** s_{ij} of items i and j
- Select k nearest neighbors $N(i; x)$
 - Items most similar to i , that were rated by x
- Estimate rating r_{xi} as the weighted average:

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

baseline estimate for r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

- μ = overall mean movie rating
- b_x = rating deviation of user x
= (avg. rating of user x) $- \mu$
- b_i = rating deviation of movie i

Item-Item vs. User-User

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.8	
Bob		0.5		0.3
Carol	0.9		1	0.8
David			1	0.4

- In practice, it has been observed that item-item often works better than user-user
- **Why?** Items are simpler, users have multiple tastes

Pros/Cons of Collaborative Filtering

- **+ Works for any kind of item**
 - No feature selection 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 and combine predictions**
 - Perhaps using a linear model
- **Add content-based methods to collaborative filtering**
 - Item profiles for new item problem
 - Demographics to deal with new user problem

Remarks & Practical Tips

Evaluation

- Error metrics
- Complexity / Speed

Evaluation

Diagram illustrating a user-movie rating matrix. The matrix is labeled "users" (vertical axis) and "movies" (horizontal axis).

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			2		2
				5	
	2	1			1
	3			3	
1					

Evaluation

movies

users

1	3	4			
	3	5			5
		4	5		5
		3			
		3			
2			?		?
				?	
	2	1			?
	3			?	
1					

Test Data Set

Evaluating Predictions

- **Compare predictions with known ratings**
 - **Root-mean-square error (RMSE)**
 - $\sqrt{\sum_{xi} (a_{xi} - \hat{a}_{xi})^2}$ where a_{xi} is predicted, \hat{a}_{xi} is the true rating of \mathbf{x} on i
 - **Precision at top 10:**
 - % of those in top 10
 - **Rank Correlation:**
 - Spearman's *correlation* between system's and user's complete rankings
- **Another approach: 0/1 model**
 - **Coverage:**
 - Number of items/users for which system can make predictions
 - **Precision:**
 - Accuracy of predictions
 - **Receiver operating characteristic (ROC)**
 - Tradeoff curve between false positives and false negatives

Problems with Error Measures

- **Narrow focus on accuracy sometimes misses the point**
 - Prediction Diversity
 - Prediction Context
 - Order of predictions
- **In practice, we care only to predict high ratings:**
 - RMSE might penalize a method that does well for high ratings and badly for others

Collaborative Filtering: Complexity

- Expensive step is finding k most similar customers: $O(|X|)$
- **Too expensive to do at runtime**
 - Could pre-compute
- Naïve pre-computation takes time $O(k \cdot |X|)$
 - X ... set of customers
- **We already know how to do this!**
 - Near-neighbor search in high dimensions (LSH)
 - Clustering
 - Dimensionality reduction

Tip: Add Data

- **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

- **Add more data**

- e.g., add IMDB data on genres

- **More data beats better algorithms**

<http://anand.typepad.com/datawocky/2008/03/more-data-usual.html>

The Netflix Prize

■ Training data

- 100 million ratings, 480,000 users, 17,770 movies
- 6 years of data: 2000-2005

■ Test data

- Last few ratings of each user (2.8 million)
- Evaluation criterion: root mean squared error (RMSE)
- Netflix Cinematch RMSE: 0.9514

■ Competition

- 2700+ teams
- \$1 million prize for 10% improvement on Cinematch

In-class question

How to address the ranking/recommendation cheat problem on JD/Tmall?

Acknowledgement

- CS246: Mining Massive Datasets, Stanford University

Thank you!

权小军 中山大学数据科学与计算机学院