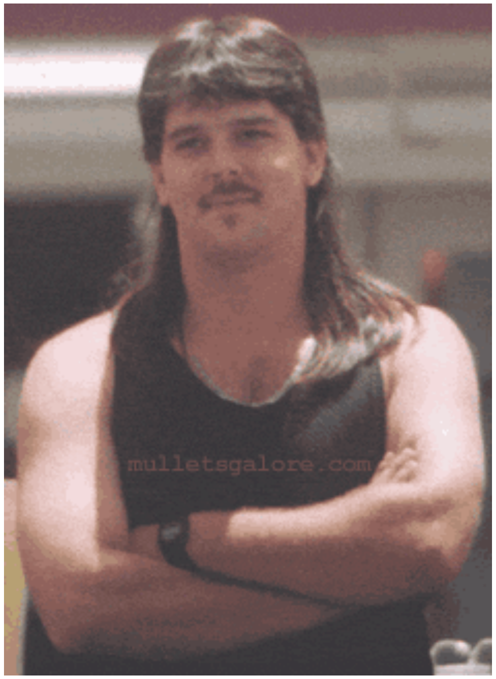
Welcome to

DS504/CS586: Big Data Analytics
Recommender System

Prof. Yanhua Li

Time: 6:00pm –8:50pm Thu.
Location: AK 232
Fall 2016

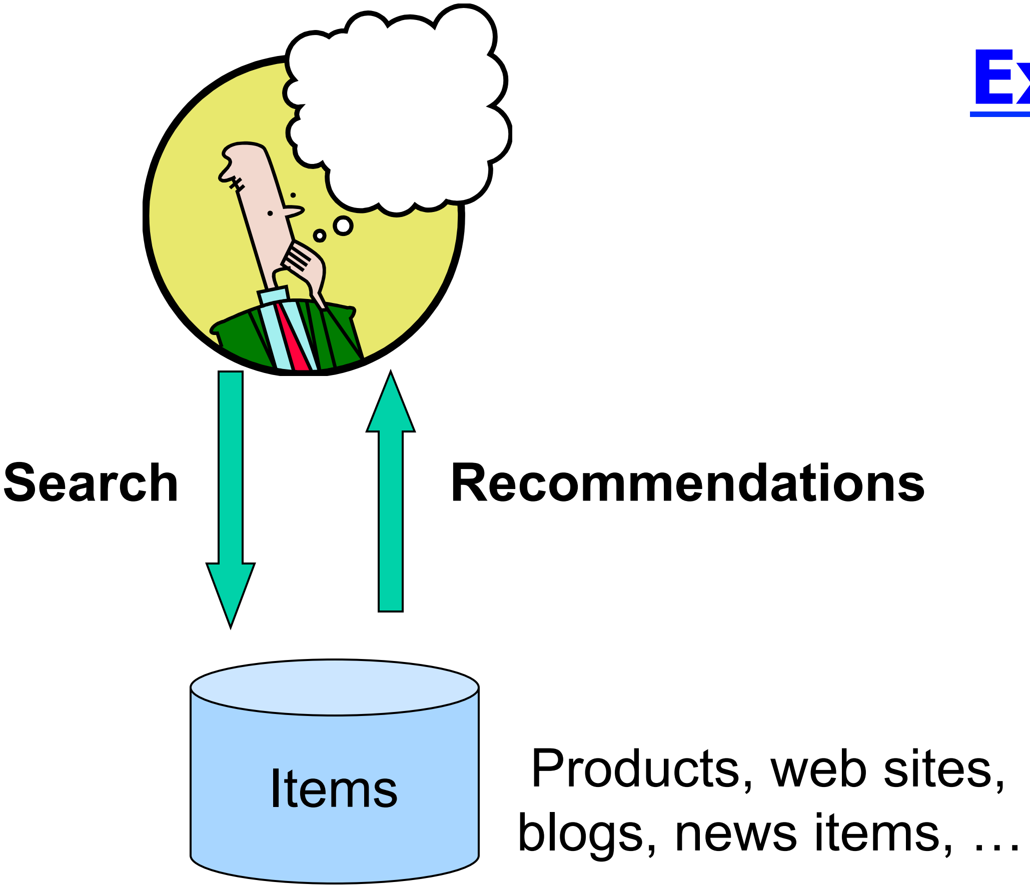
Example: Recommender Systems



- **Customer X**
  - Star War I
  - Star War II

- **Customer Y**
  - Does search on Star War I
  - Recommender system suggests Star War II from data collected about customer **X**

Recommendations



Examples:

amazon.com

PANDORA

StumbleUpon

del.icio.us

NETFLIX

movielens
helping you find the *right* movies

last.fm
the social music revolution

Google News

YouTube

XBOX LIVE

From Scarcity to Abundance

- **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, e.g., Amazon, Target online, eBay, etc.
- **More choices necessitates better filters**
  - Recommendation engines

Types of Recommendations

- **Editorial and hand curated**
  - List of favorites
  - Lists of "essential" items

- **Simple aggregates**
  - Top 10, Most Popular, Recent Uploads

- **Tailored to individual users**
  - Amazon, Netflix, …

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) Estimate 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 estimation 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
- **Implicit**
  - Learn ratings from user actions
    - E.g., purchase implies high rating
  - What about low ratings?

(2) Estimating 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

- **Approaches to recommender systems:**
    - **1)** Content-based
    - **2)** Collaborative filtering

Content-based Recommender Systems

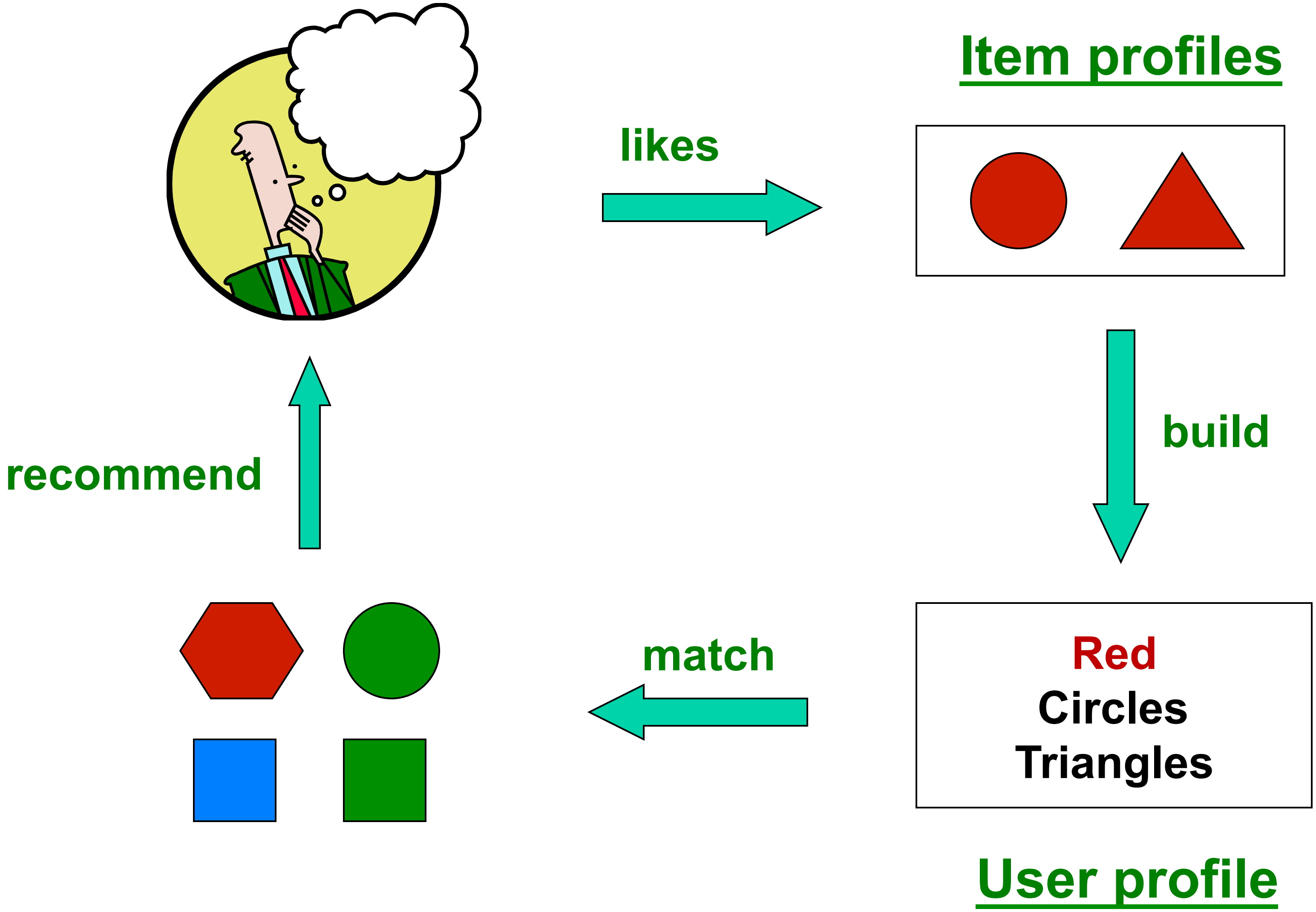
Content-based Recommendations

- **Main idea:** Recommend items to customer ***x*** similar to previous items rated highly by ***x***
  - ***Look at x's items vs all items***

Example:

- **Movie recommendations**
  - Recommend movies with same actor(s), director, genre, …
- **Websites, blogs, news**
  - Recommend other sites with "similar" content

Plan of Action



likes

Item profiles

build

User profile

Red
Circles
Triangles

match

recommend

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**

Sidenote: TF-IDF

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

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

Note: we normalize TF by the frequency of the most frequent term 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

$$w_j = (w_{1j}, \ldots, w_{ij}, \ldots, w_{kj})$$

User Profiles and Prediction

- User profile possibilities:
  - Weighted average of rated item profiles
  - Variations: weight by difference from average rating for item

$$w_x = \sum_{j=1\ldots N_x} w_j(r_{xj} - \overline{r}_x)$$

- Prediction heuristic:
  - Given user profile w_x and item profile w_j, estimate

$$r_{xj} = \cos(w_x, w_j) = w_x w_j / \| w_j \| \| w_x \|$$

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 item cold-start
- **+: 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

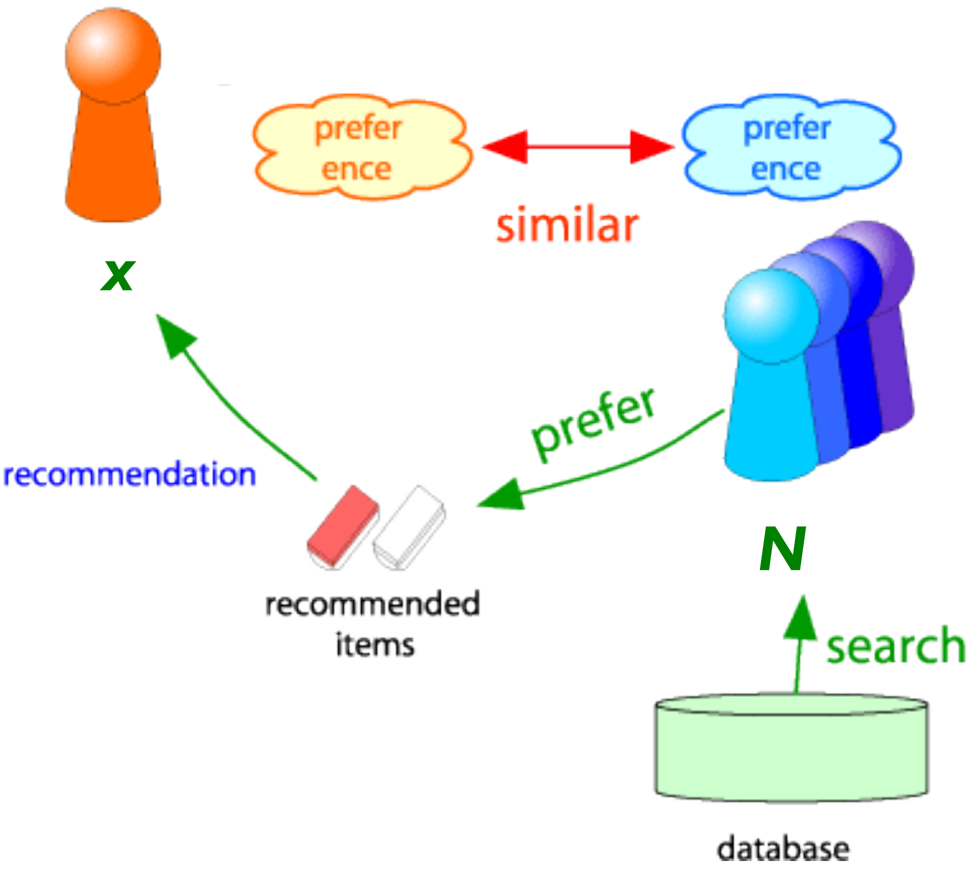
- **–: Finding the appropriate features is hard**
  - E.g., images, movies, music
- **–: Recommendations for new users**
  - **How to build a user profile?**
  - **User code-start problem**
- **–: Overspecialization**
  - Never recommends items outside user's content profile
  - People might have multiple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

Collaborative Filtering

Harnessing quality judgments of other users

Collaborative Filtering

- Consider user ***x***
- 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 = [*, _, _, *, ***]
r_y = [*, _, **, **, _]

- Let r_x be the vector of user x’s ratings
- Jaccard similarity measure

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

  - Problem: Ignore the value of the ratings:
- Cosine Similarity measure
  - $Sim(x,y)=\cos(r_x, r_y)=r_x r_y / ||r_x||\ ||r_y||$
  - Problem: Treading missing ratings as negatives
- Pearson correlation coefficient

- $Sim(x,y)=(r_x-r_{x,ave})(r_y-r_{y,ave})/||r_x-r_{x,ave}||\ ||r_y-r_{y,ave}||$

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

Similarity Metric

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- **Intuitively we want:**
    - **sim(*A*, *B*) > sim(*A*, *C*)**
- **Jaccard similarity:** 1/5 < 2/4
- **Cosine similarity:** 0.386 > 0.322
    - Considers missing ratings as "negative"
    - **Solution: subtract the (row) mean**

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

Notice cosine sim. is correlation when data is centered at 0

User-User Collaborative Filtering

- For user u, find other similar users
- Estimate rating for item ***i*** based on ratings from similar users

$$pred(u,i) = \frac{\sum_{n \subset neighbors(u)} sim(u,n) \cdot r_{ni}}{\sum_{n \subset neighbors(u)} sim(u,n)}$$

Sim(u,n)… similarity of user ***u*** and ***n***
r_{ui}…rating of user ***u*** on item ***i***
neighbor(u)… set of users similar to user ***u***

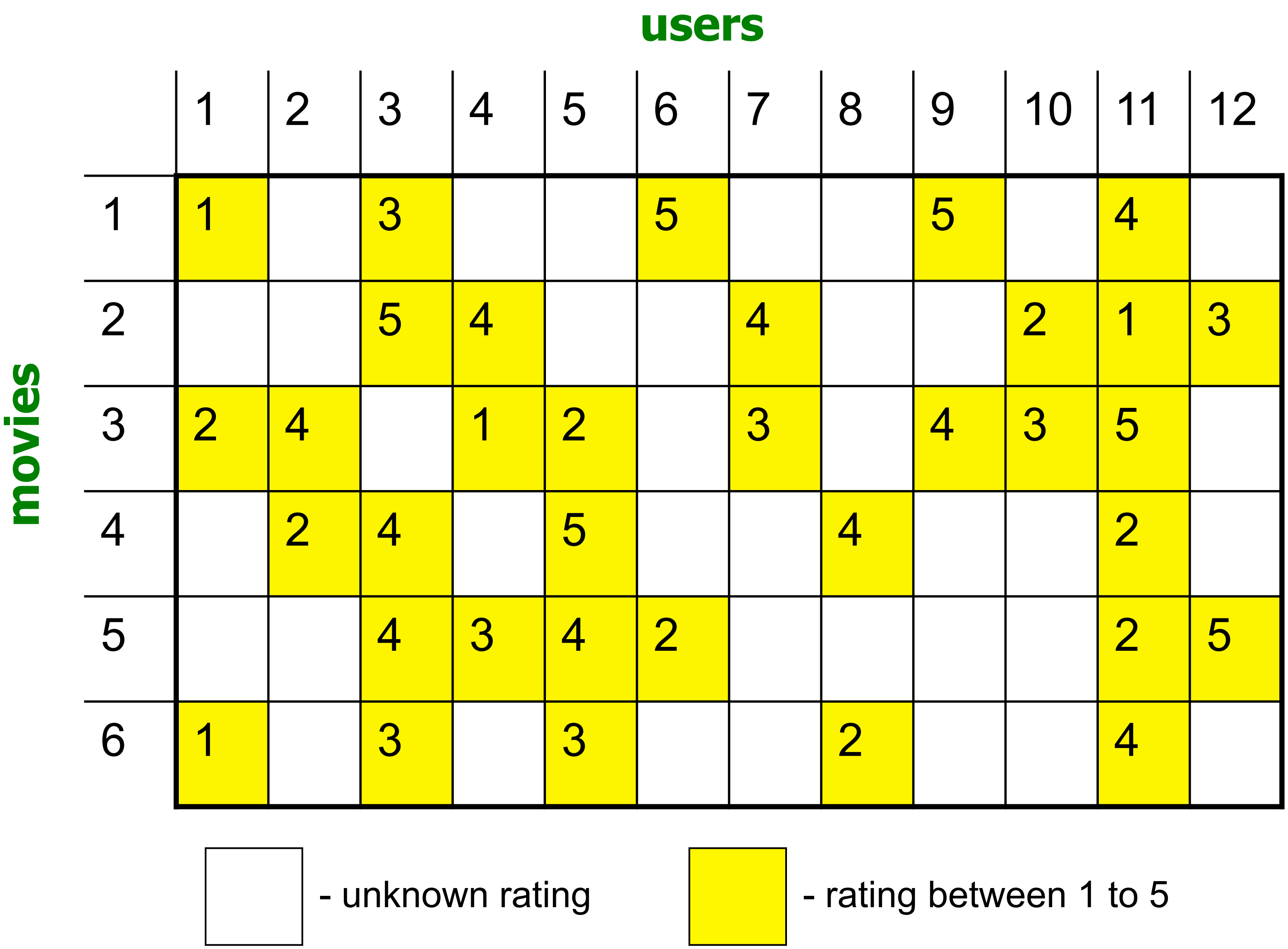
Item-Item Collaborative Filtering

- **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

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

s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** 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)

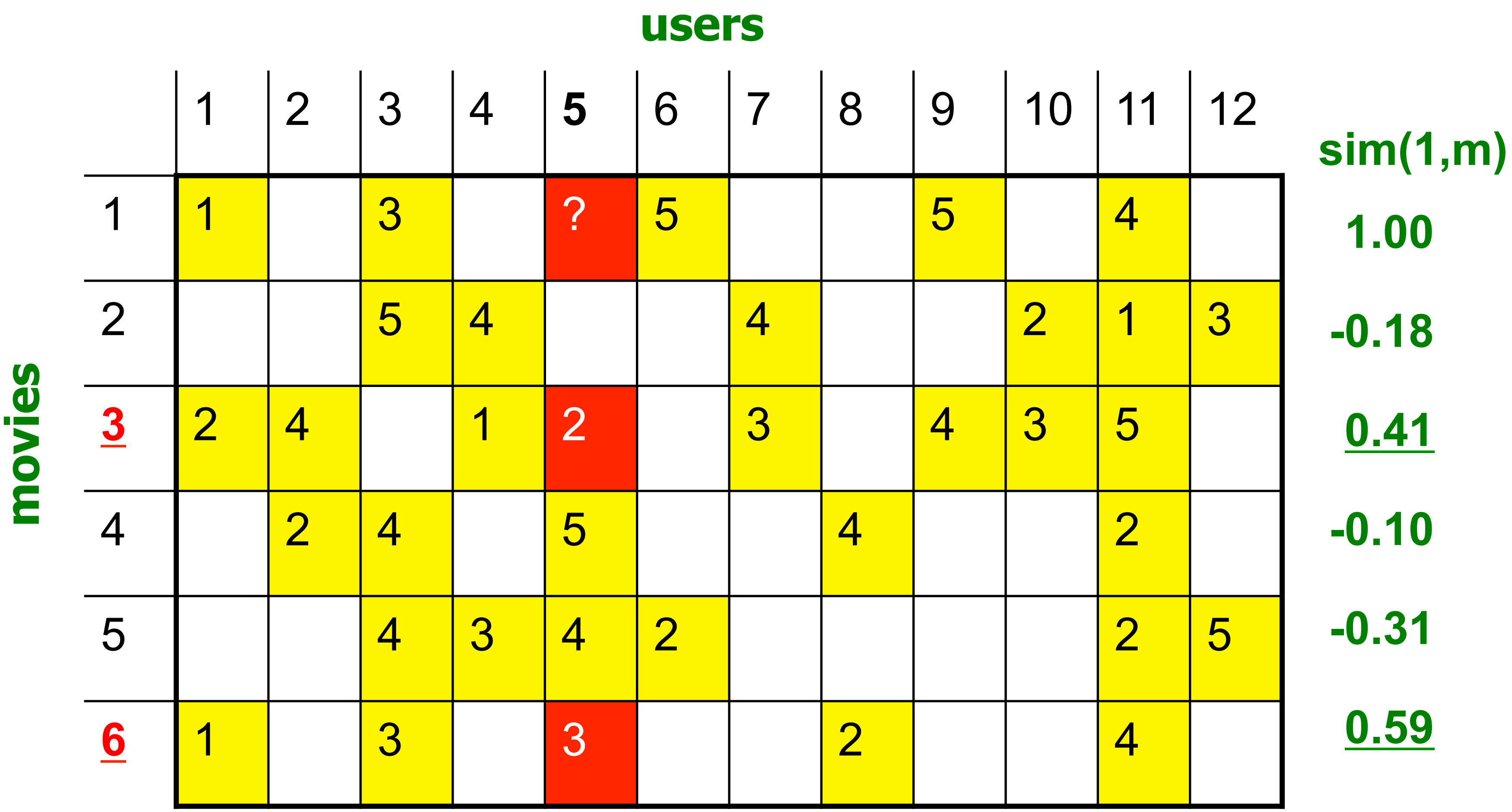
users

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

movies

	1	2	3	4	**5**	6	7	8	9	10	11	12	**sim(1,m)**
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	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**
6	1		3		3			2			4		**0.59**

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)

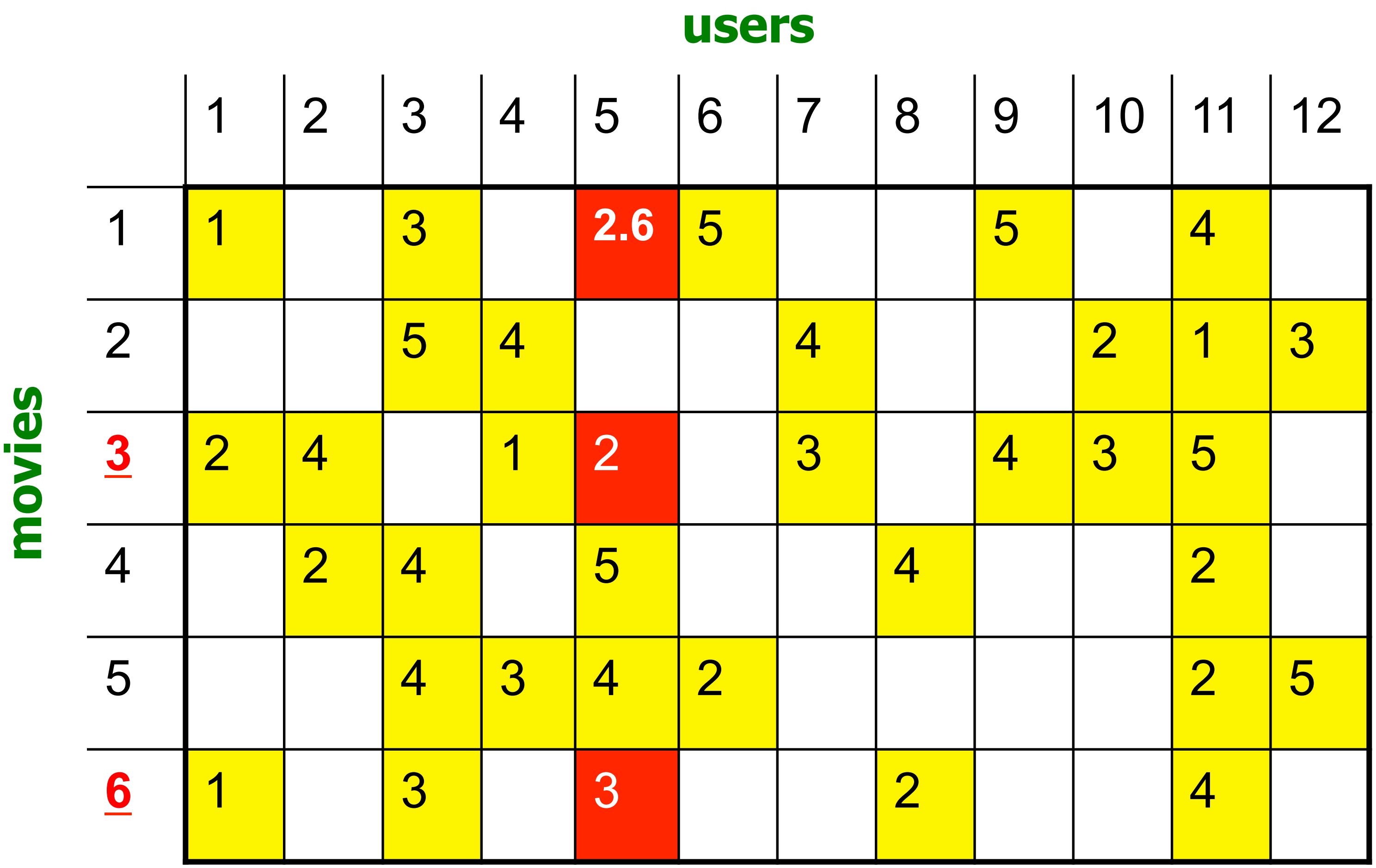
users

movies	1	2	3	4	5	6	7	8	9	10	11	12	**sim(1,m)**
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	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**
6	1		3		3			2			4		**0.59**

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
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	

Predict by taking weighted average:

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

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

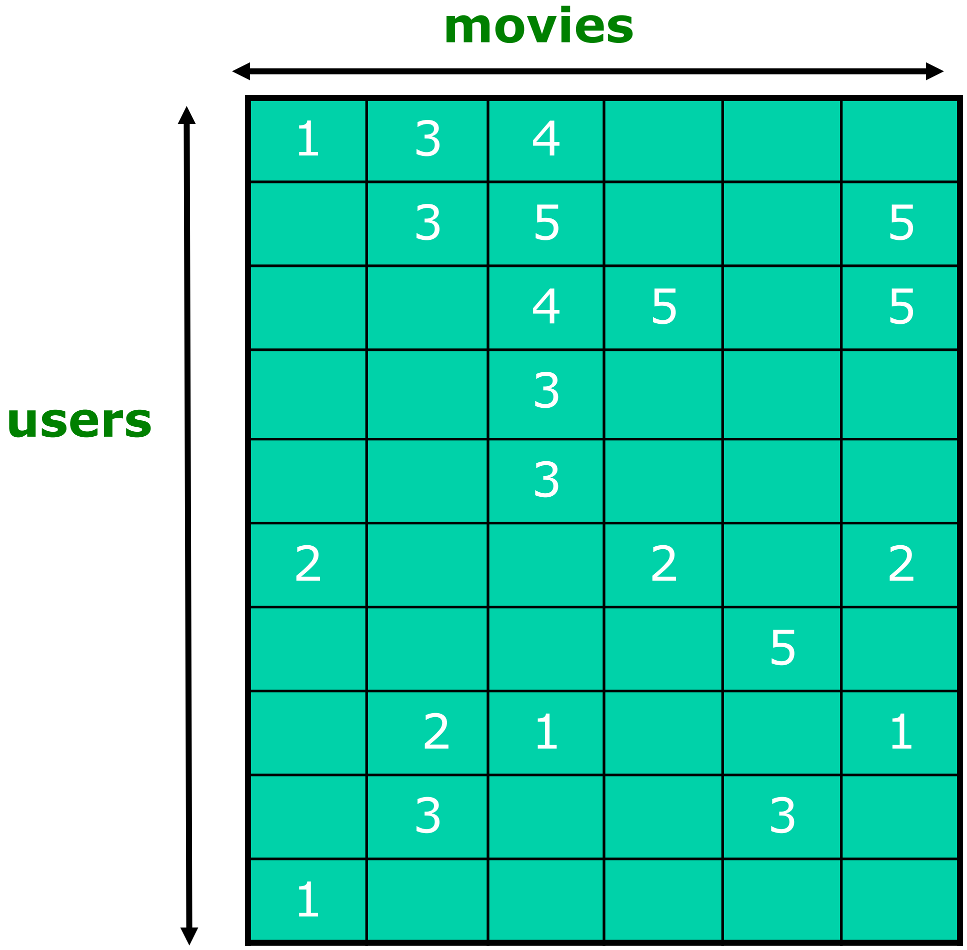
Pros/Cons of Collaborative Filtering

- **+ 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

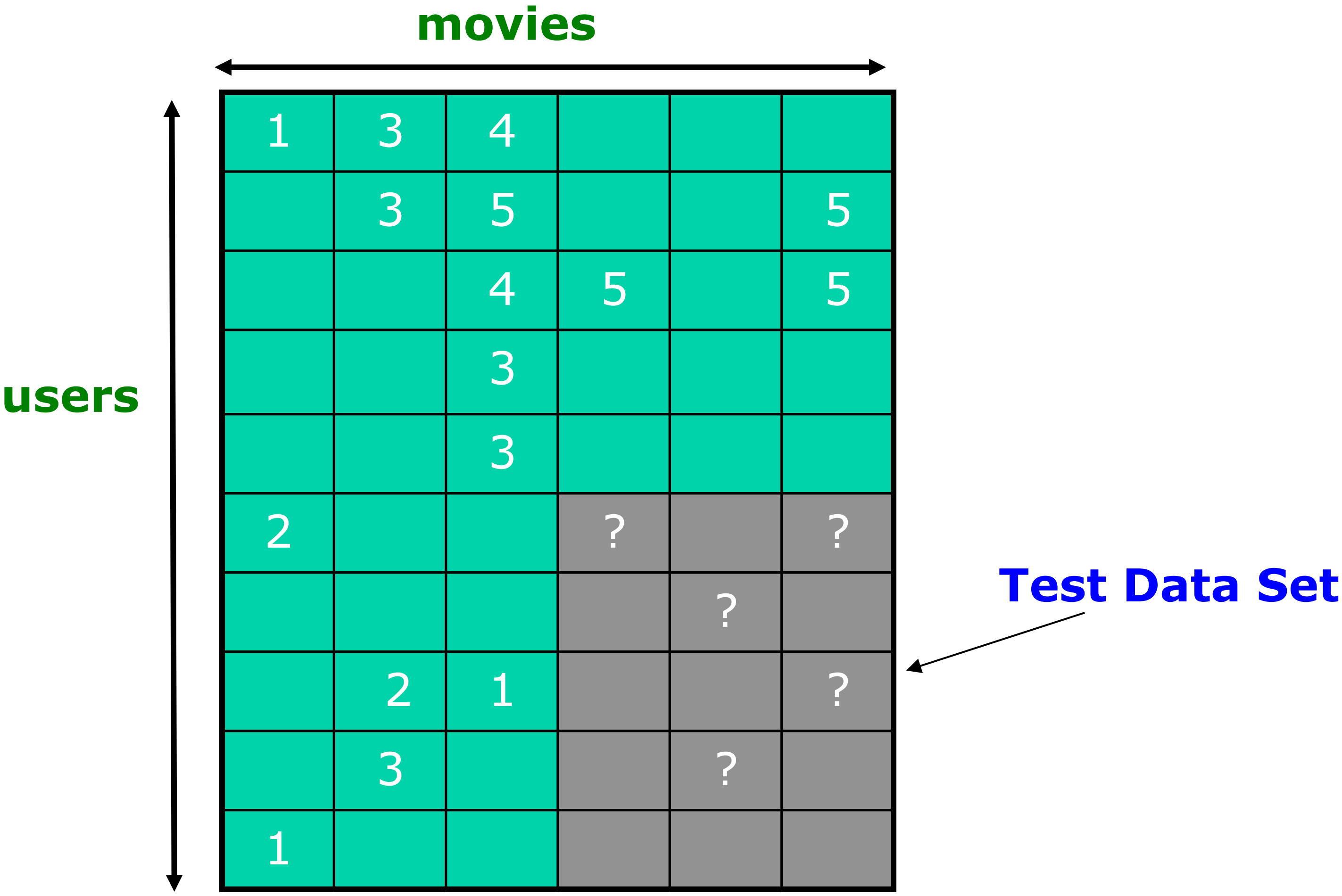
- **Implement two or more different recommenders and combine predictions**
  - Perhaps using a linear model
- **Add content-based methods to collaborative filtering**
  - Item profiles for new item problem
  - Demographics to deal with new user problem

Evaluation



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

Evaluation



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

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 ·|X|)**
  - X … set of customers
- **We already know how to do this!**
  - Near-neighbor search in high dimensions
  - Clustering
  - Dimensionality reduction

Location-based & Preference-Aware Recommendation Using Sparse Geo-Social Networking Data

Jie Bao Yu Zheng Mohamed F. Mokbel

Microsoft Research Asia
Beijing, China

Department of Computer Science &Engineering
University of Minnesota

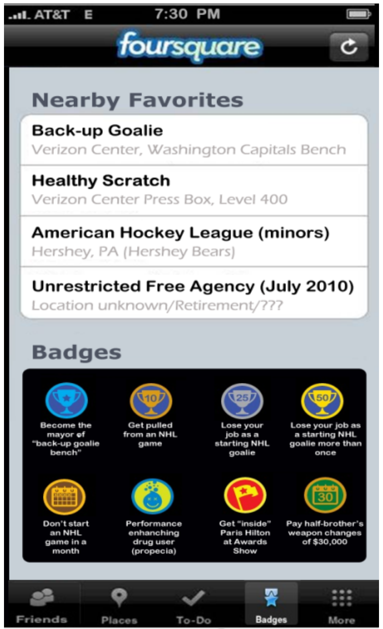
Background

- ❖ Location-based Social Networks

Loopt

Me
5 mins ago
Last check-in at:
Peet's Coffee & Tea

Foursquare



Nearby Favorites

Back-up Goalie
Verizon Center, Washington Capitals Bench

Healthy Scratch
Verizon Center Press Box, Level 400

American Hockey League (minors)
Hershey, PA (Hershey Bears)

Unrestricted Free Agency (July 2010)
Location unknown/Retirement/???

Badges

STAYCLASSY.NET

Facebook Places

Zhen Fang dawn over embarcadero
7 hours ago
9 comments 9 like this

Tony Tung
Facebook HQ
Tony checked in at Facebook HQ.
Tagged: **Erick Tseng**
10 minutes ago

Morin Oluwole > Dave Fetterman
Happy birthday Dave! Enjoy your day!
10 minutes ago

Jenny Wiens launched!

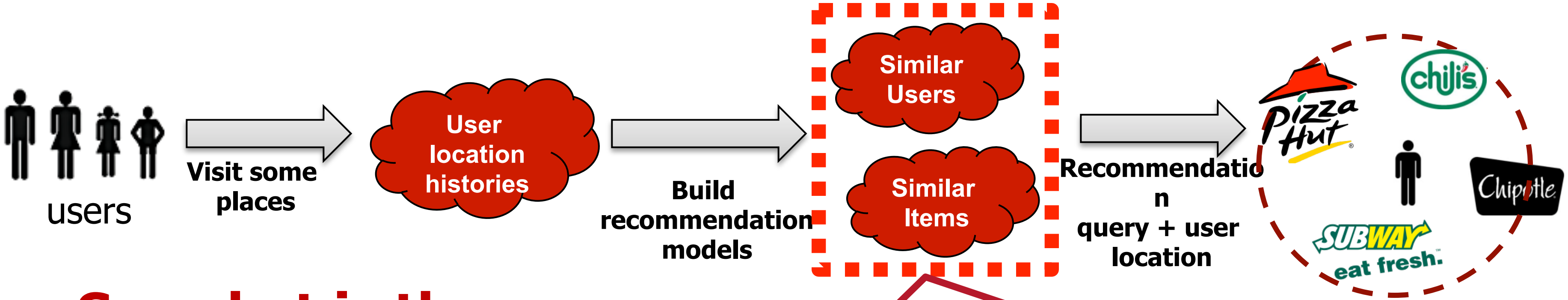
Dianping

- Users share photos, comments or check-ins at a location
- Expanded rapidly, e.g., Foursquare gets over 3 million check-ins every day

http://blog.foursquare.com/2011/04/20/an-incredible-global-4sqday/

Introduction

- Location Recommendations in LBSN
  - Recommend locations using a user’s location histories and community opinions
  - Location bridges gap between physical world & social networks
- Existing Solutions
  - Based on item/user collaborative filtering
  - Similar users gives the similar ratings to similar items



So, what is the PROBLEM here?

based on the model of *co-rating* and *co-visit*

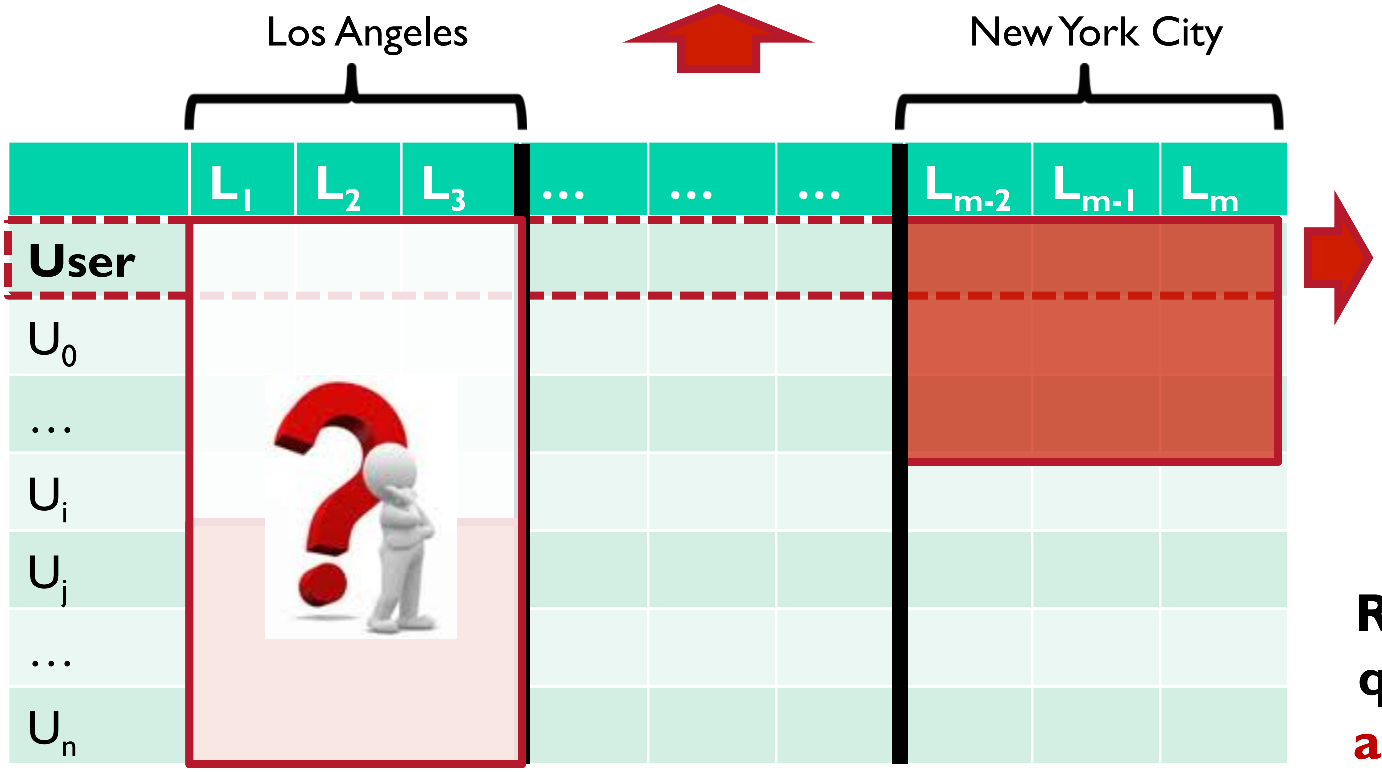
Why?

Mao Ye, Peifeng Yin, Wang-Chien Lee: “Location recommendation for location-based social networks.” GIS2010
stin J. Levandoski, Mohamed Sarwat, Ahmed Eldawy, and Mohamed F. Mokbel: “LARS: A Location-Aware Recommender System.” ICDE2

Motivation (1/2)

- User-item rating/visiting matrix

***Millions** of locations around the world*



	L_1	L_2	L_3	…	…	…	L_{m-2}	L_{m-1}	L_m
User									
U_0									
…									
U_i									
U_j									
…									
U_n									

A user visit **~100 locations**

User location histories are locally clustered

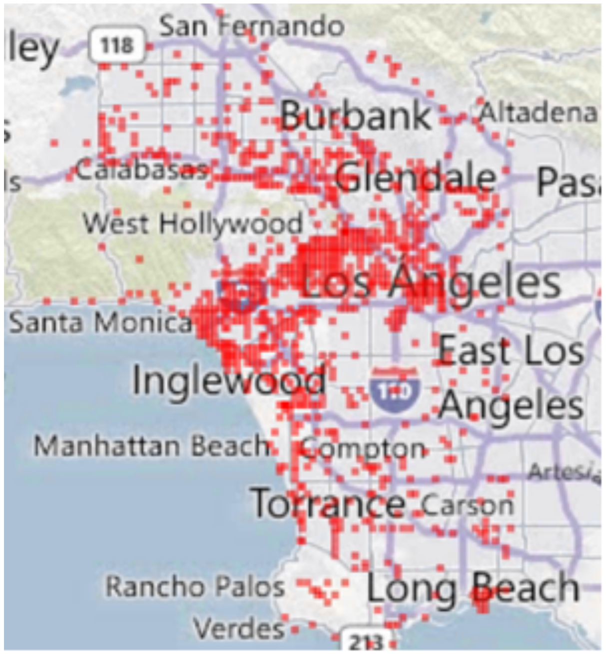
Recommendation queries target an area (very specific subset)

Noulas, S. Scellato, C Mascolo and M Pontil "An Empirical Study of Geographic User Activity Patterns in Foursquare " (ICWSM 2011)

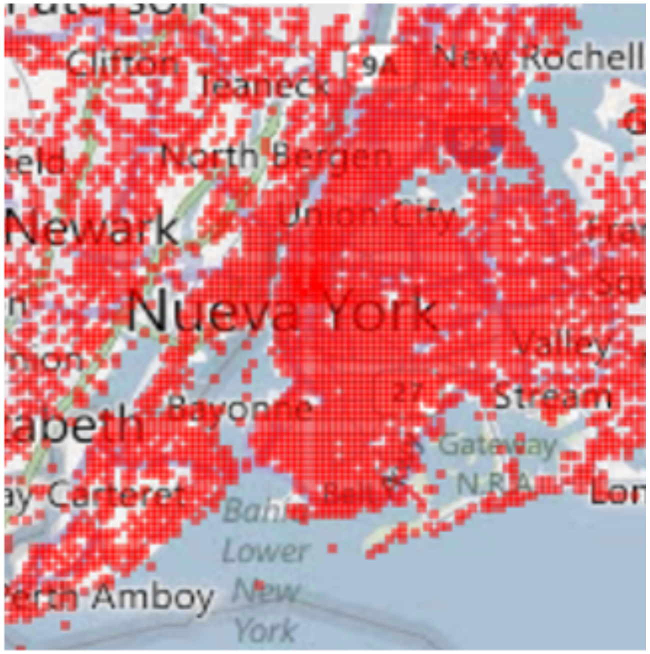
Motivation (2/2)

- User's activities are very limited in distant locations
  - May **NOT** get any recommendations in some areas
  - Things can get worse in **NEW Areas** *(small cities and abroad)*

(Where you need recommendations the most)



(a) New York users in Los Angels

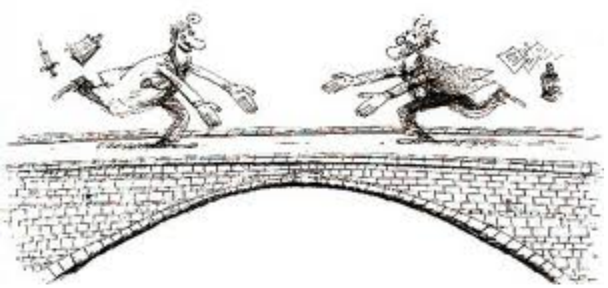


(b) New York users in New York City.

Key Components in Location Recommendation

1. User position & locations around

2. User Personal Interests/Preferences



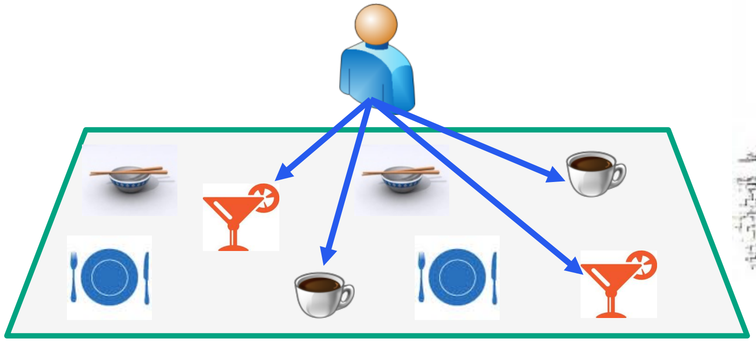
Recommender System

3. Social/Community Opinions

Our Main Ideas

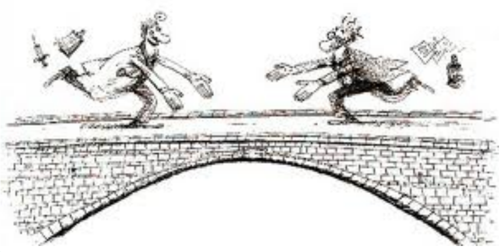
User Personal Interests/Preferences

Main idea #1:
Identify user preference using **semantic information** from the location history



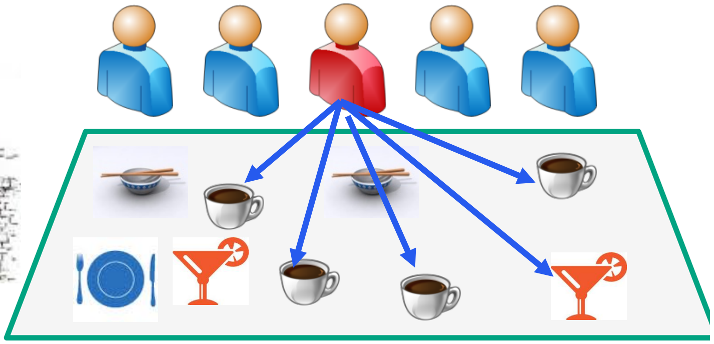
User position & locations around

Main idea #3:
Use **local experts** & **user preferences** for recommendation



Social/Community Opinions

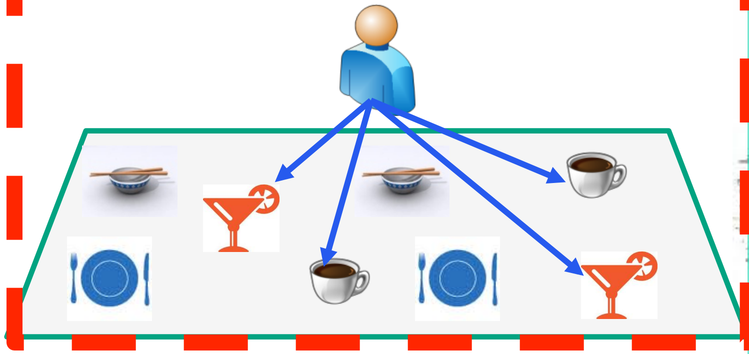
Main idea #2:
Discover **local experts** for different categories in a specific area



Offline Modeling User preferences discovery

User Personal Interests/Preferences

Main idea #1:
Identify user preference using **semantic information** from the location history

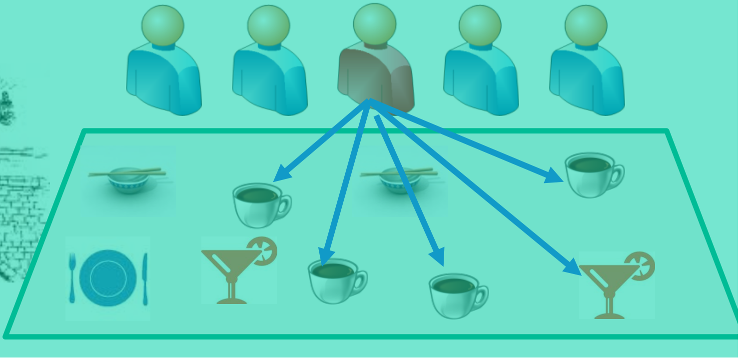


User position & locations around

Main idea #3:
Use **local experts & user preferences** for recommendation

Social/Community Opinions

Main idea #2:
Discover **local experts** for different categories in a specific area

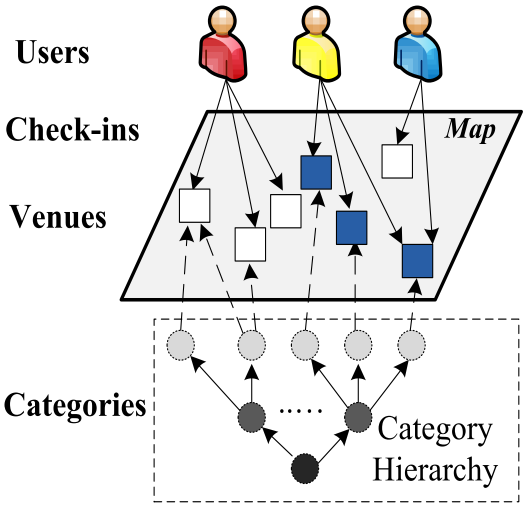


User preference discovery (1/2) Our Solution

- A natural way to express a user's preference
  - E.g., Jie likes shopping, football…..

1. User preferences is not that spatial-aware
2. User preferences is more semantic

- Can we extract such preferences from user locations? **YES!**



(a) Overview of a location-based social network

Category Name	Number of sub-categories
Arts & Entertainment	17
College & University	23
Food	78
Great Outdoors	28
Home, Work, Other	15
Nightlife Spot	20
Shop	45
Travel Spot	14

(b) Detailed location category hierarchy in FourSquare

Millions of locations

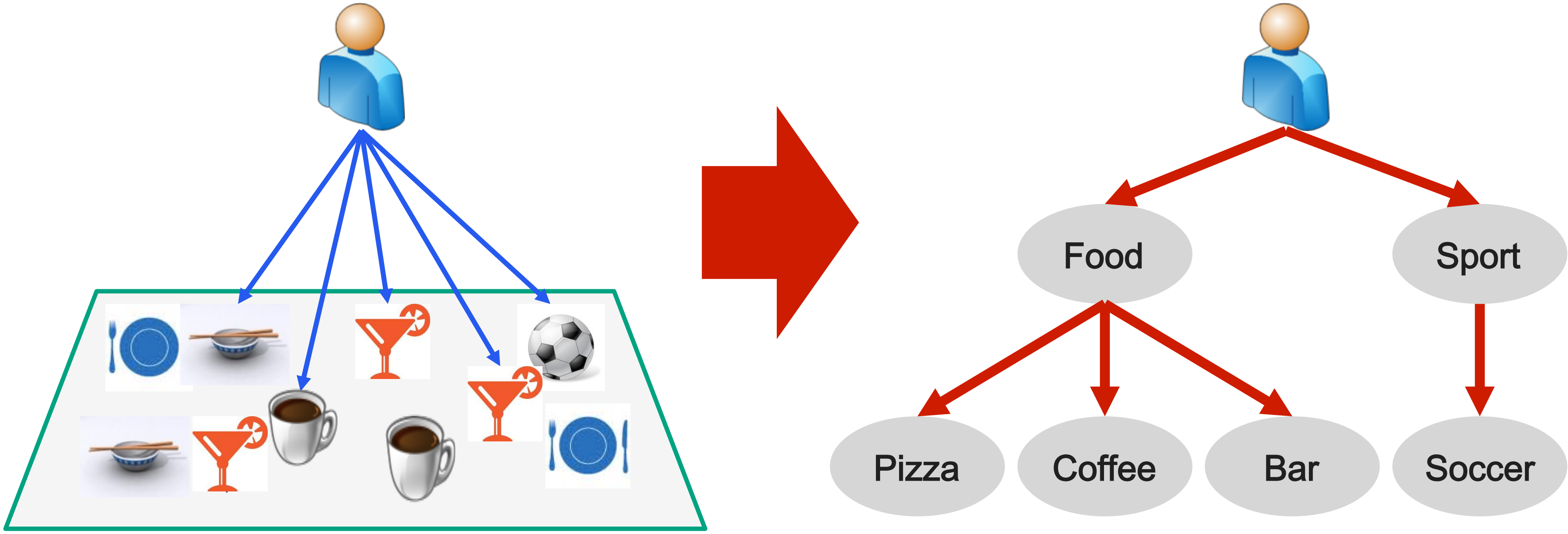
Hundreds of categories

AND

NOT limited only to the residence areas

User preference discovery (2/2)
Weighted Category Hierarchy

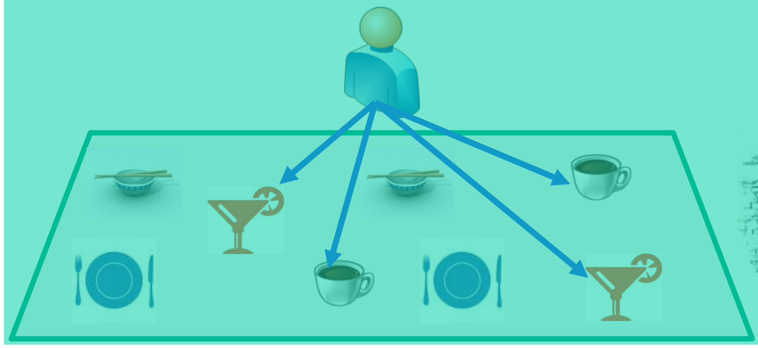
- User preferences discovery
  - Location history
  - Semantic information
  - User preference hierarchy
    - Use TF-IDF approach to minimize the bias



Offline Modeling (2/2) Social Knowledge Learning

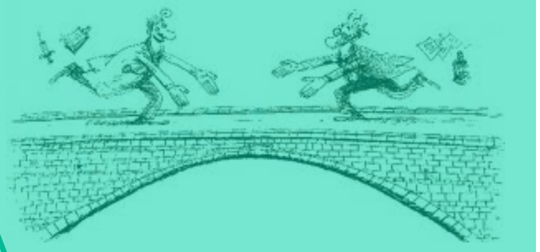
User Personal Interests/Preferences

Main idea #1:
Identify user preference using **semantic information** from the location history



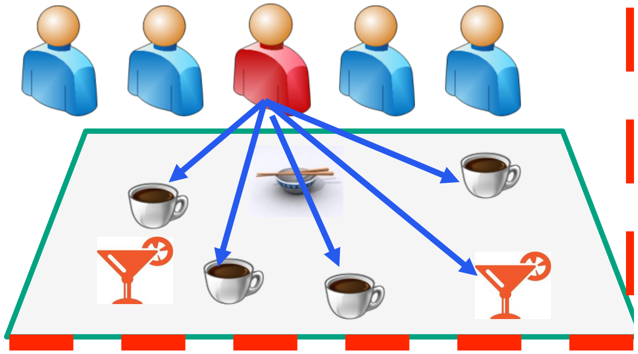
User position & locations around

Main idea #3:
Use **local experts & user preferences** for recommendation



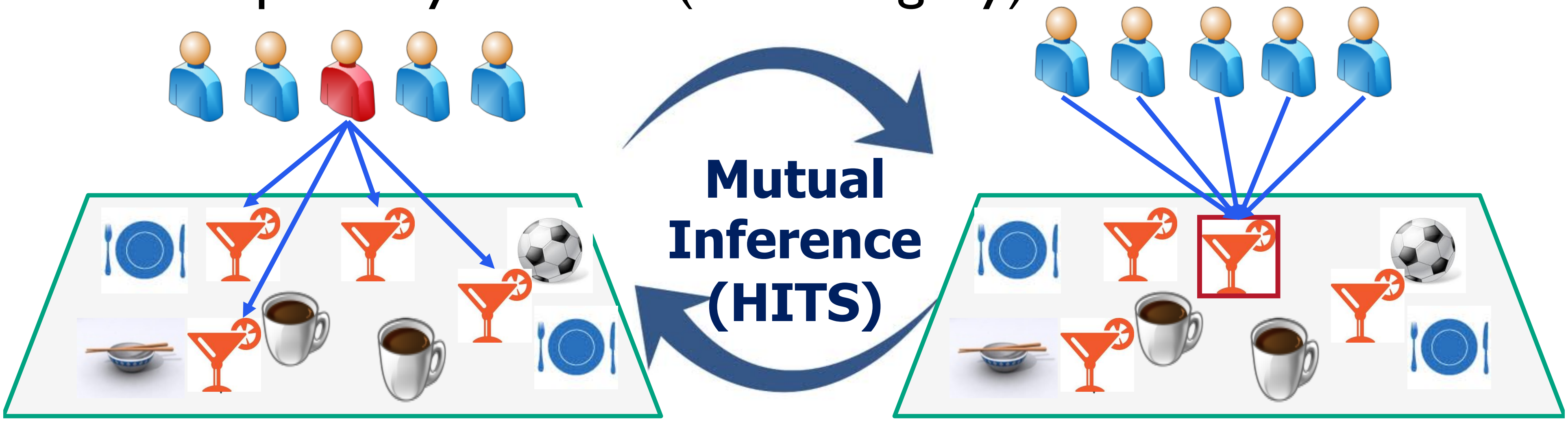
Social/Community Opinions

Main idea #2:
Discover **local experts** for different categories in a specific area



Offline Modeling (2/2) Social Knowledge Learning

- Why local experts
  - High quality
  - Less number **(Efficiency)**
- How to discover "local experts"
  - Local knowledge (in an area)
  - Speciality (in a category)



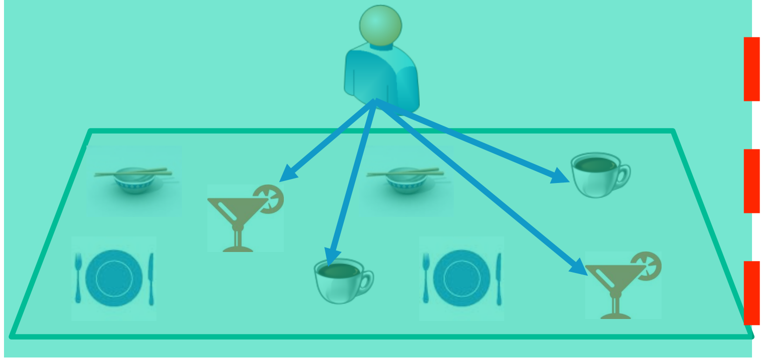
User hub nodes

Location authority nodes

Online Recommendation

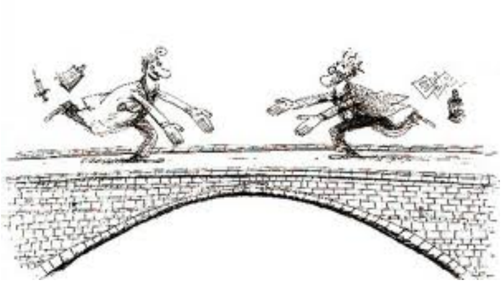
User Personal Interests/Preferences

Main idea #1:
Identify user preference using **semantic information** from the location history



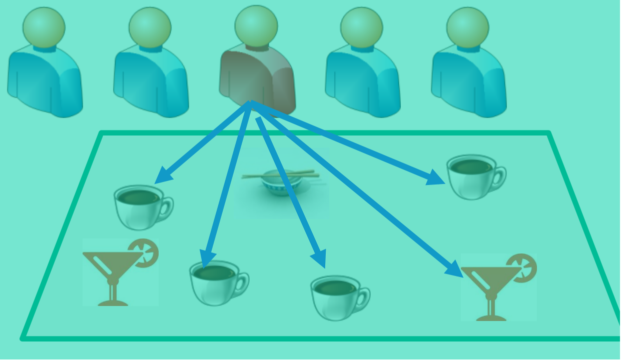
User position & locations around

Main idea #3:
Use **local experts** & **user preferences** for recommendation



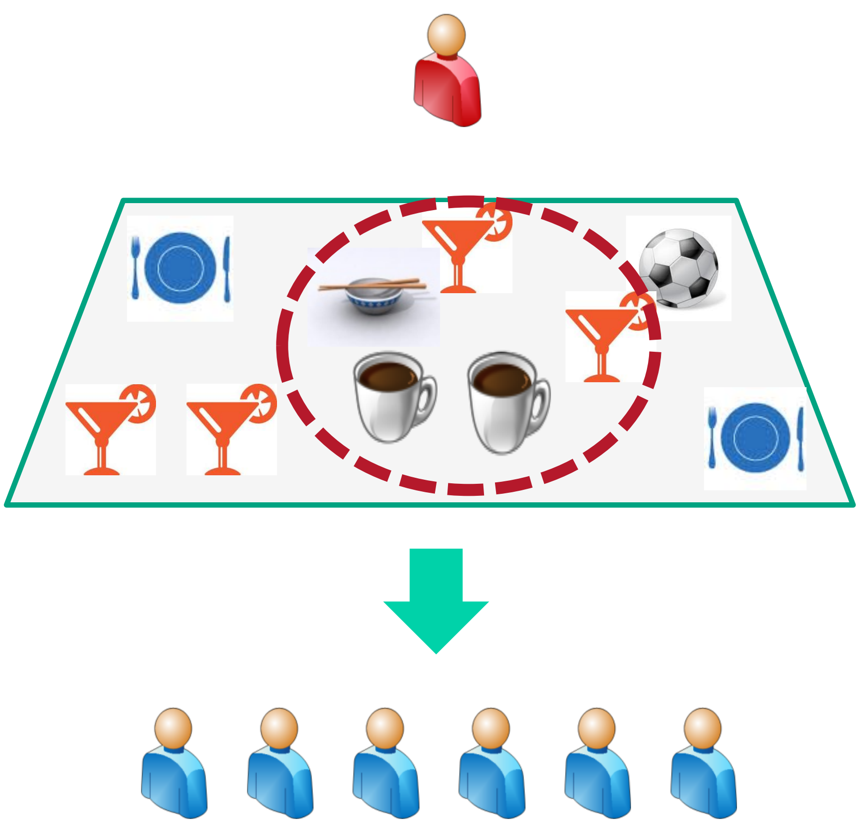
Social/Community Opinions

Main idea #2:
Discover **local experts** for different categories in a specific area

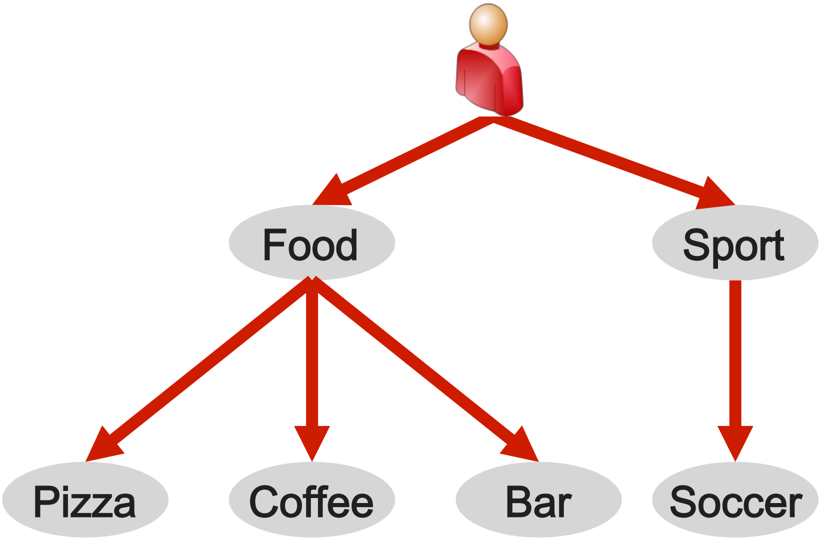


Online Recommendations (1/2)Candidate Selection

- Select the candidate locations and local experts



Candidate Local Experts



Food

Sport

Pizza

Coffee

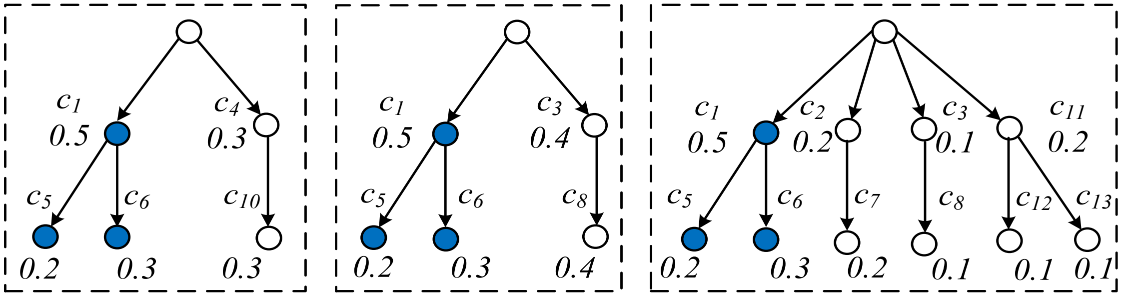
Bar

Soccer

More local experts are selected for the more preferred category

Online Recommendations (2/2)
Location Rating Inference

- Similarity Computing
  - Overlaps: Different weights for different levels
  - Diversity of user preferences
    - Based on entropy theory



(a) WCH of u_1 (b) WCH of u_2 (c) WCH of u_3

- Infer the ratings for the candidate locations

$$Sim(u,u') = \sum_{l=1}^{|l|} \beta \times \frac{LevelSim(u,u',l)}{1+|H(u,l)-H(u',l)|}$$

$$R_u(v) = \sum_{u'\in\mathcal{E}\&v\in\mathcal{V}} Sim(u,u') \times v(u',v),$$

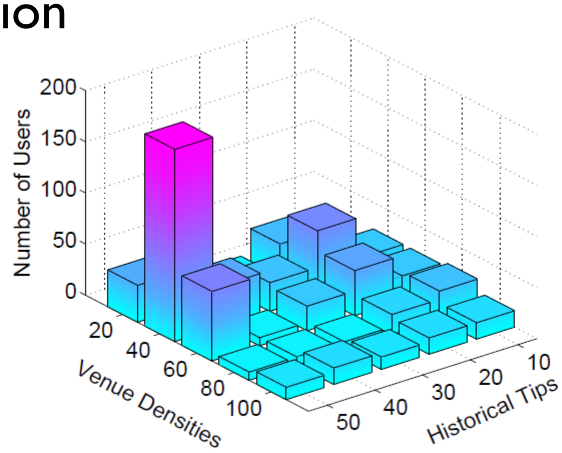
H(u, l) is user u's entropy at level l

Experiments Data Set

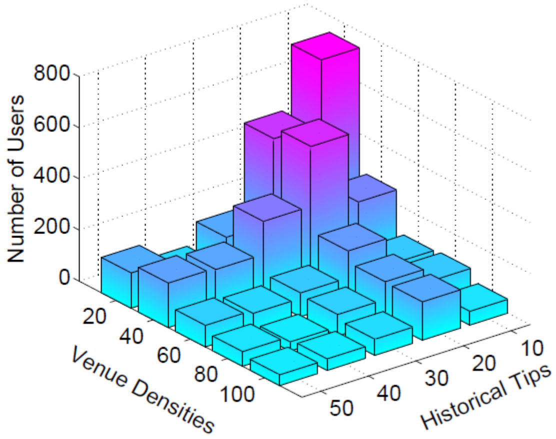
- Data Sets
  - 49,062 users and 221,128 tips in New York City (NYC)
  - 31,544 users and 104,478 tips in Los Angels (LA).
- Statistics

Home City	Querying City	Total Users	Tips in City	Tips /User	Footprint (miles)	All Tips
NJ	LA	228	2,553	11.20	5.31	9,836
NJ	NYC	2,886	72,170	25.01	3.93	106,870

- Visualization



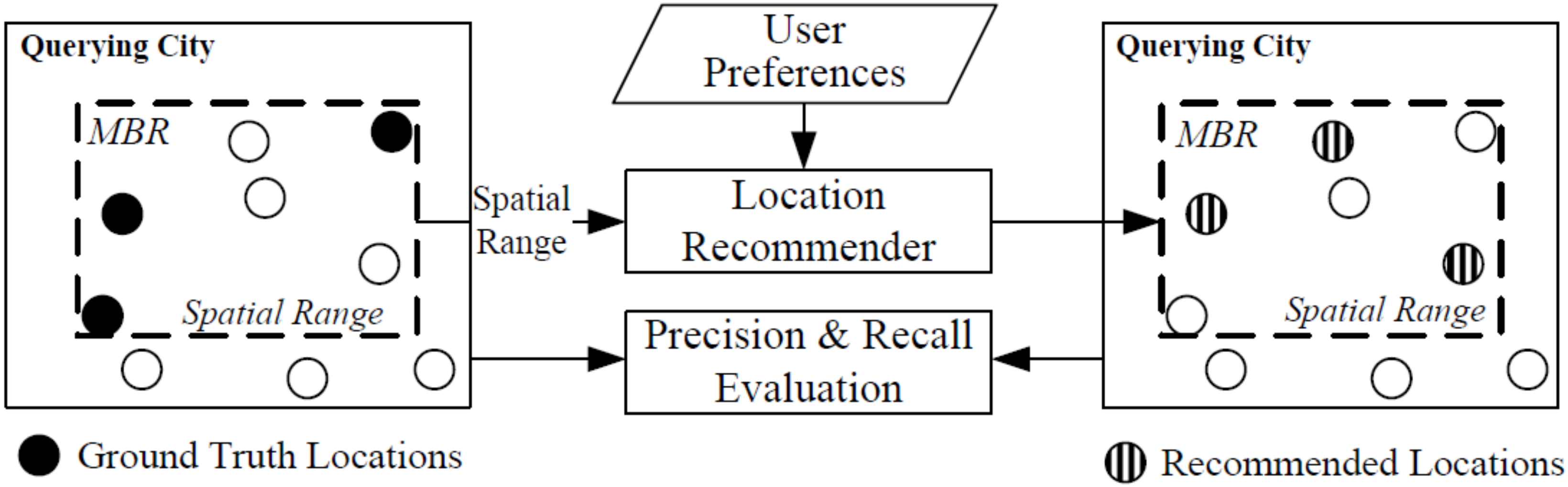
(a) New Jersey Users in LA.



(b) New Jersey Users in NYC.

Evaluation Framework

- Evaluation Method



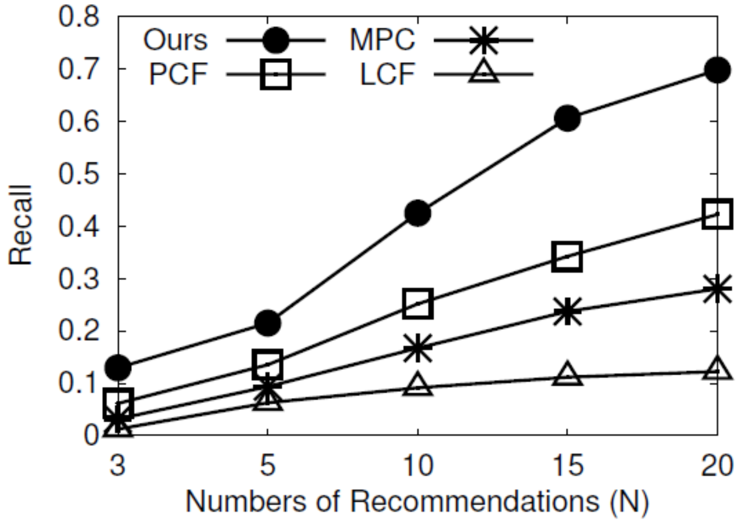
- Evaluation Metrics

$$precision = \frac{number\ of\ recovered\ ground\ truths}{total\ number\ of\ recommendations}$$

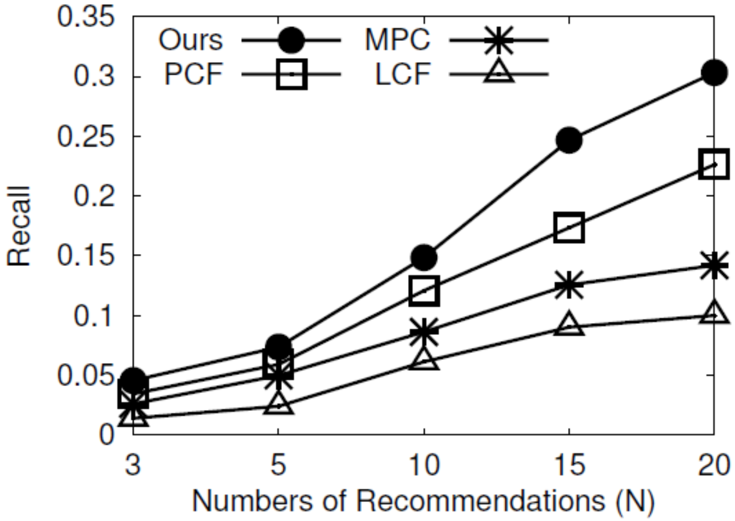
$$recall = \frac{number\ of\ recovered\ ground\ truths}{total\ number\ of\ ground\ truths}$$

Experimental Results

Method	**Social Opinion**	**Category of Location**	**Preference Hierarchy**	**Candidate Selection**
MPC		√	√	√
LCF	√			
PCF	√	√		
Ours w/o CS	√	√	√	
Ours	√	√	√	√



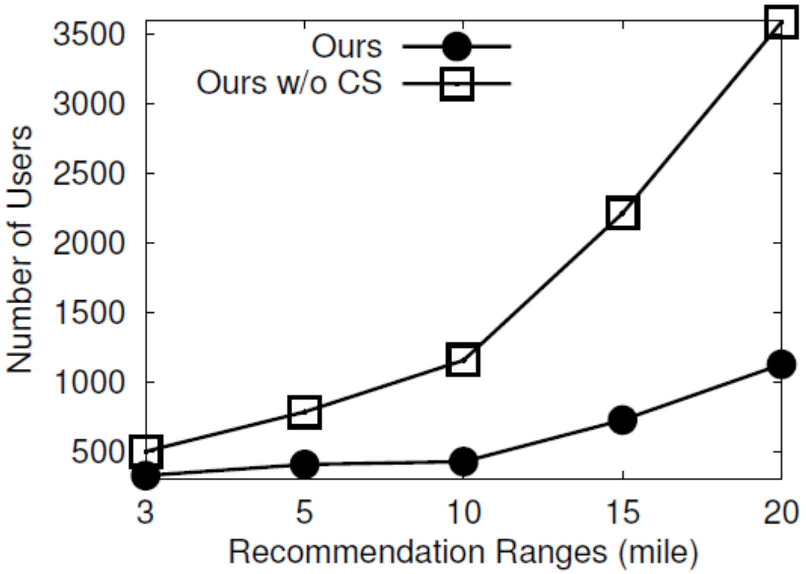
(a) New Jersey Users in LA.



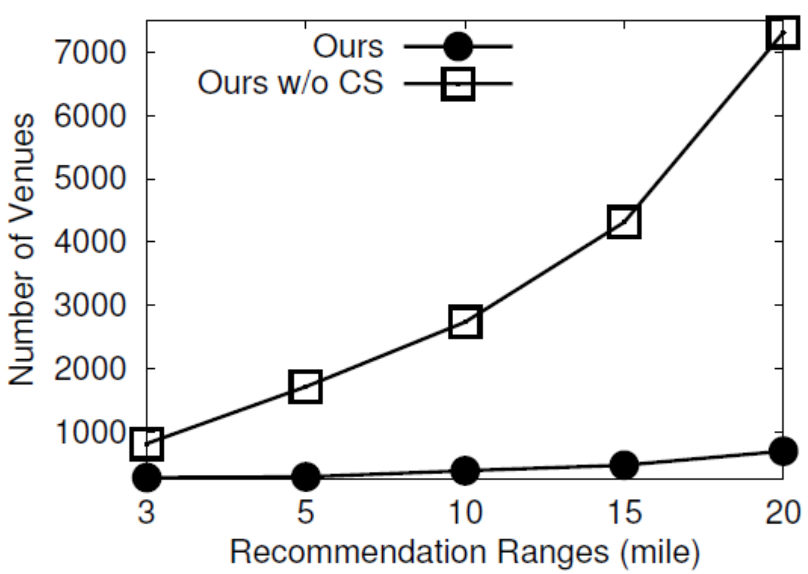
(b) New Jersey Users in NYC.

Experimental Results

- Efficiency



(a) Selected Users in LA.



(b) Candidate Venues in LA.

Conclusion

- Location Recommendations
  - Data sparsity is a big challenge in recommendation systems
  - Location-awareness amplify the data sparsity challenge

- Our Solution
  - Take advantage of **category information** to overcome the sparsity
  - Using the knowledge from the **local experts**
  - Dynamically select the local experts for recommendation based on user location

Before:

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

CF: Common Practice

- 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**)* **–** μ
- b_i = rating deviation of movie ***i***