RECOMMENDATION ENGINE DEMYSTIFIED

NEIGHBORHOOD METHODS

COLLABORATIVE FILTERING

Alex Lin

Senior Architect

Intelligent Mining

Outline

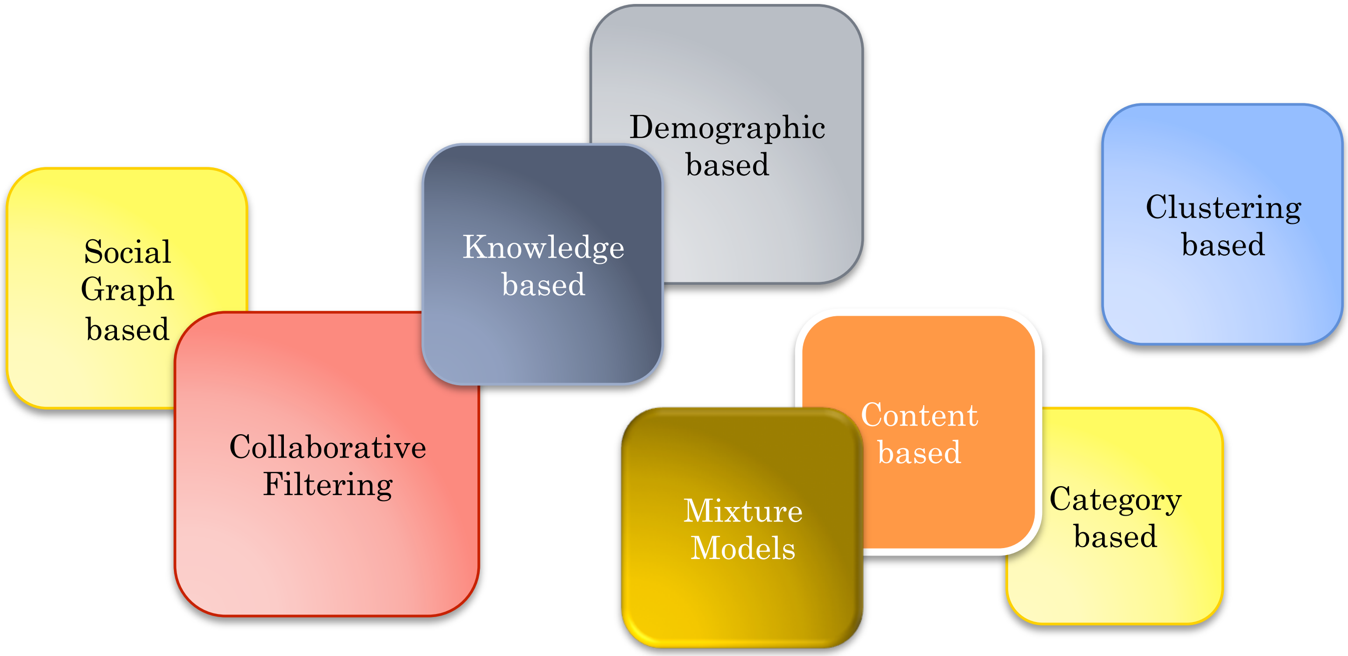
- Introduction
- User-oriented Collaborative Filtering
- Item-oriented Collaborative Filtering
- Challenges
- Best Practices

Recommendation Engine

What is a Recommendation Engine (RE)?

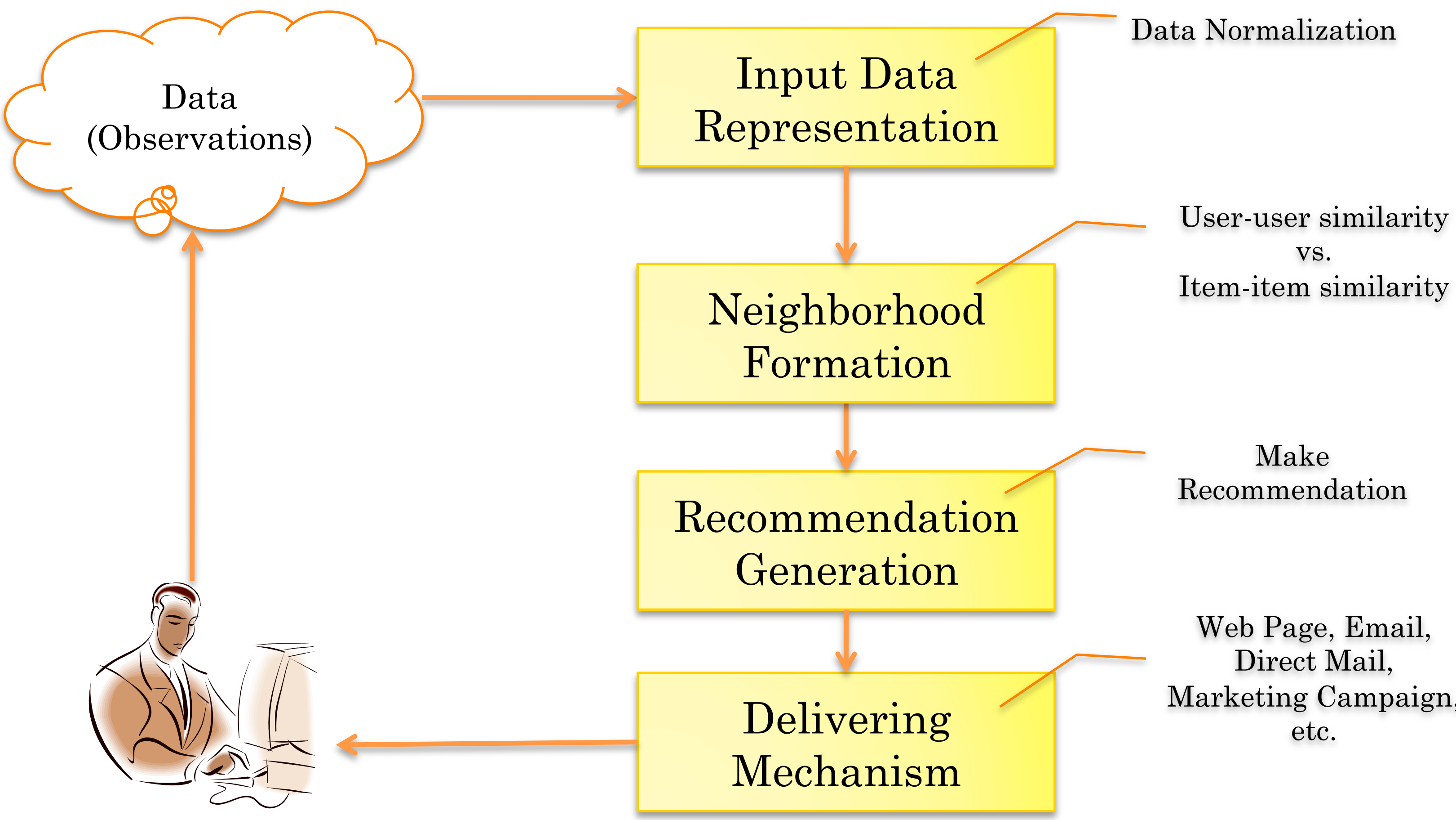
- RE takes "observation" data and uses machine learning / statistical algorithms to predict outcomes or levels of interest.
- "Recommender systems form a specific type of information filtering (IF) technique that attempts to present information items (movies, music, books, news, images, web pages, etc.) that are likely of interest to the user." – Wikipedia

Recommendation Engine



- This presentation will focus on Neighborhood-based Collaborative Filtering
  - User-oriented method
  - Item-oriented method

Neighborhood-based Collaborative Filtering



Outline

- Introduction
- User-oriented Collaborative Filtering
- Item-oriented Collaborative Filtering
- Challenges
- Best Practices

User-oriented CF

- Input Data Representation: Users / Items matrix.

users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	1		1			1				1		
2							1	1	1			
3	1	1		1				1	1			
4		1			1			1	1			
⋮			1				1					
m												

- Cell value “1” means user purchased the item.
- Data Normalization is not shown on this slide.

User-oriented CF

- Neighborhood Formation:
  - Find the k most like-minded users in the system.

users

items

	1	2	3	4	5	6	7	**8**	9	10	…	n
1	1	1	1			1				1		
2							1	1	1			
3	1	1		1				1	1			1
4		1			1			1	1			
⋮			1				1					
m				1					1			

User-oriented CF

- Neighborhood Formation:
  - Find the k most like-minded users in the system.

users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	1	1	1			1				1		
2							1	1	1			
3	1	1		1				1	1			1
4		1			1			1	1			
⋮			1				1					
m				1					1			

- Identify U_9 and U_2 are similar to U_8

User-oriented CF

- Recommendation Generation:

users

items

	1	2	3	4	5	6	7	8	9	10	…	n
1	1	1	1			1				1		
2							1	1	1			
3	1	1		1				1	1			1
4		1			1			1	1			
⋮			1				1					
m				1					1			

- Identify I_1 and I_9 are not yet purchased by U_8

User-oriented CF

- Recommendation Generation:

users

items	1	**2**	3	4	5	6	7	**8**	**9**	10	…	n
1	1	1	1			1		**0.7**		1		
2							1	1	1			
3	1	1		1				1	1			1
4		1			1			1	1			
⋮			1				1					
m				1				**0.9**	1			

- Predict by taking weighted sum

User-oriented CF
Practical Implementation

- Compute and store all user-user similarities.
  - Cosine similarity: $sim(u,v) = \cos(\vec{u},\vec{v}) = \dfrac{\vec{u} \bullet \vec{v}}{\left\|\vec{u}\right\|_2 * \left\|\vec{v}\right\|_2}$
- Find N items that will be most likely purchased by user **u**.
  - Find **k** most similar users to **u**, save to U_{sim}
  - Get all items purchased by U_{sim}, save to $I_{candidate}$
  - Remove unavailable items in $I_{candidate}$
  - Get all items purchased by **u**, save to $I_{purchased}$
  - Take $I_{candidate} - I_{purchased} = I_{recmd}$
  - Re-order items in I_{recmd} based on sum of user-user similarity

$$pred(u,i) = \frac{\sum_{v \in k-similarUser(u)} userSim(u,v) * r_{vi}}{\sum_{v \in k-similarUser(u)} userSim(u,v)}$$

Outline

- Introduction
- User-oriented Collaborative Filtering
- Item-oriented Collaborative Filtering
- Challenges
- Best Practices

Item-oriented CF

- Input Data Representation: Users / Items matrix.

users

items

	1	2	3	4	5	6	7	8	9	10	…	n
1	1		1			1				1		
2							1	1	1			
3	1	1		1				1	1			
4		1			1			1	1			
⋮			1				1					
m												

- Cell value "1" means user purchased the item.
- Data Normalization is not shown on this slide.

Item-oriented CF

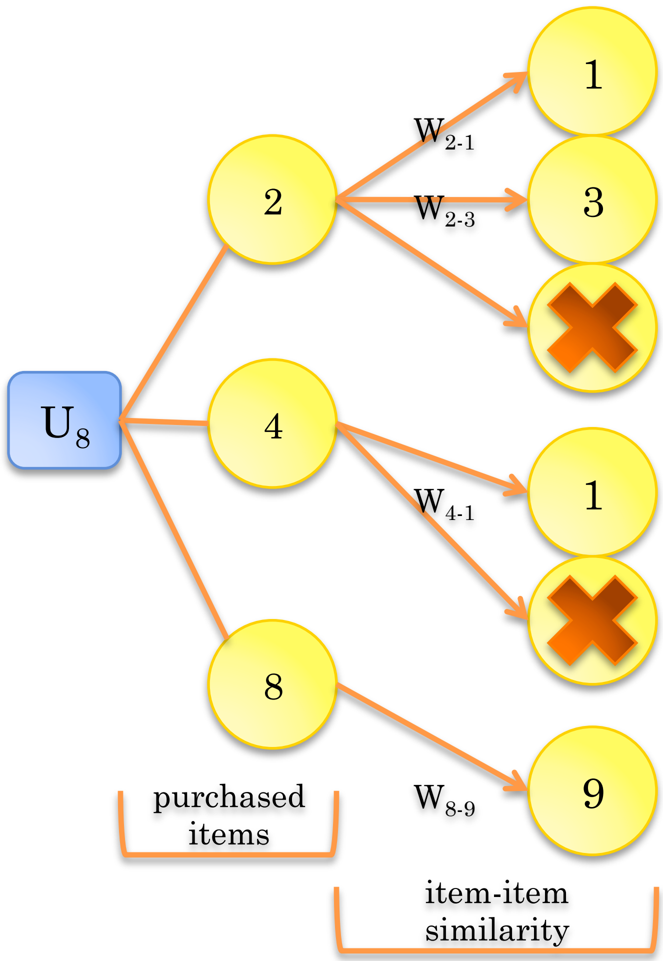
- Neighborhood Formation:
  - Find the k items that have the most similar user vectors

users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	1		1			1				1		
2		1					1	1	1			
3	1			1				1	1			
4		1			1		1	1	1			
⋮			1				1					
m				1					1			

Item-oriented CF – cont.

- Recommendation Generation



Predict by taking weighted sum

TopN Recmd. for U_8 : {1,9,3}

Item-oriented CF
Practical Implementation

- Compute and store all item-item similarities.
  - Cosine similarity: $sim(a,b) = \cos(\vec{a},\vec{b}) = \dfrac{\vec{a} \bullet \vec{b}}{\|\vec{a}\|_2 * \|\vec{b}\|_2}$

- Find N items that will be most likely purchased by user **u**.
  - Get all items purchased by **u**, save to $I_{purchased}$
  - For each item in $I_{purchased}$, find **k** most similar items save them to $I_{candidate}$
  - Remove unavailable items in $I_{candidate}$
  - Get all items purchased by **u**, save to $I_{purchased}$
  - Take $I_{candidate} - I_{purchased} = I_{recmd}$
  - Re-order items in I_{recmd} based on

$$pred(u,i) = \frac{\sum_{j \in purchasedItems(u)} itemSim(i,j) * r_{uj}}{\sum_{j \in purchasedItems(u)} itemSim(i,j)}$$

Outline

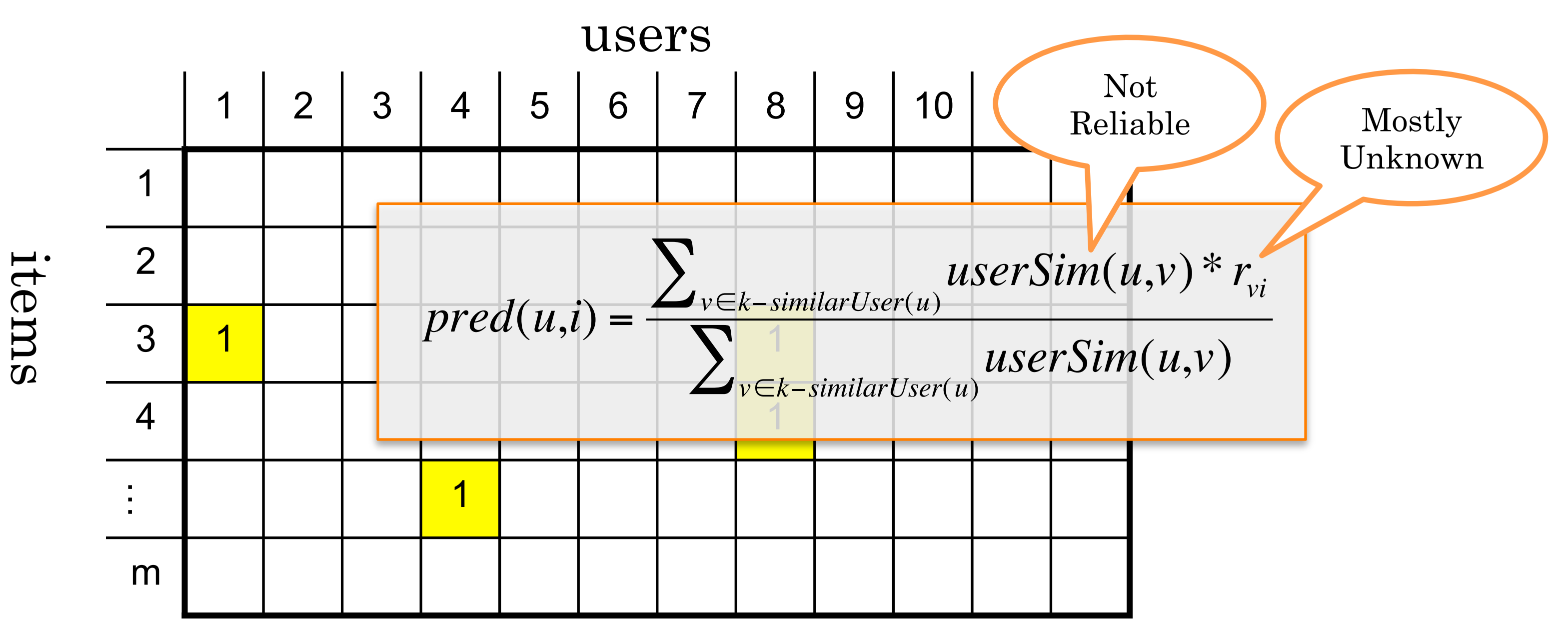
- Introduction
- User-oriented Collaborative Filtering
- Item-oriented Collaborative Filtering
- Challenges
- Best Practices

The Challenges you will face

- Data Sparsity Issue
- Cold Start Problem
- Curse of Dimensionality
- Scalability

Data Sparsity Issue

- Missing values in the Users / Items matrix.



$$pred(u,i) = \frac{\sum_{v \in k-similarUser(u)} userSim(u,v) * r_{vi}}{\sum_{v \in k-similarUser(u)} userSim(u,v)}$$

- Netflix Prize data set: 98.82% of cells are blank
- Typical e-commerce txn data set can be 10-100 time more sparse than Netflix Prize data set !!

Cold Start Problem

- It occurs when new item or new user is added to the data matrix.

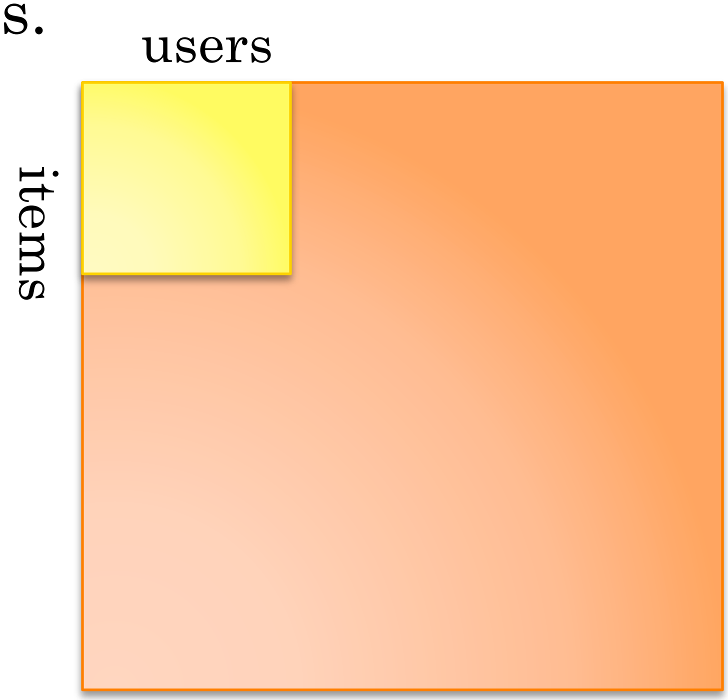
users

items	1	2	3	4	5	6	7	8	9	10	…	n
1	1		1			1				1		
2							1	1	1			
3	1	1		1				1	1			1
4		1			1			1	1			
⋮			1				1					
m												

- RE does not have enough knowledge about this new user or this new item yet.
- Content-based REs can be incorporated to alleviate cold start problem.

Curse of Dimensionality

- Adding more features (items or users) can increase the noise, and hence the error.
- There aren't enough observations to get good estimates.



Scalability

- User neighborhood formation: $O(n^2)$ for n users
- Item neighborhood formation: $O(m^2)$ for m items
- When m (# of items) << n (# of users), item-based CF will be more efficient than user-based CF
- Ability to update neighborhood incrementally

Outline

- Introduction
- User-oriented Collaborative Filtering
- Item-oriented Collaborative Filtering
- Challenges
- Best Practices

Best Practices

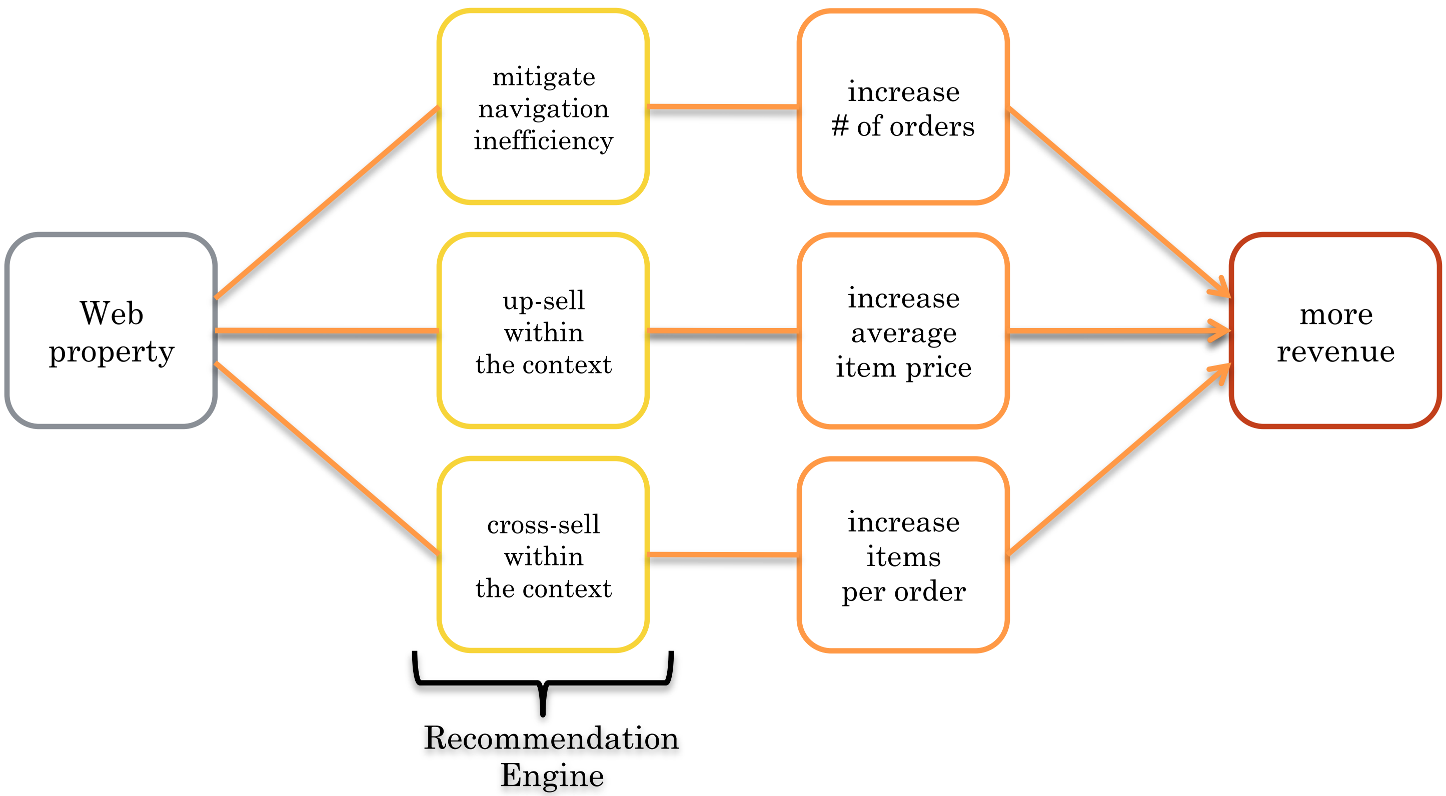
- Understand the data thoroughly
- Define business objectives and conversion metrics judiciously
- Understand context and user intent
- Apply adaptive reinforcement learning
- Optimize RE using cost-based methods
- Be aware of data-shift issue
- Optimize marketing messages delivered with recommendation results

Understand the data thoroughly

- What data are available?
- E-commerce data set typically contains:
  - Clickstream
  - Shopping cart / Saved Items / Wish list / Shared Item
  - Order / Return
  - User profile
  - User ratings
- How are these data points being collected?
- Is there pre-existing bias in the data? or leakage?
- Is the data related to what we want to predict?

Define business objectives and conversion metrics judiciously

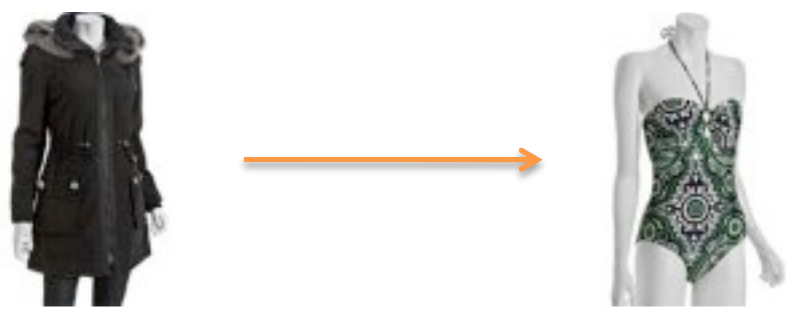
- Defining correct conversion metrics can be a competitive advantage.



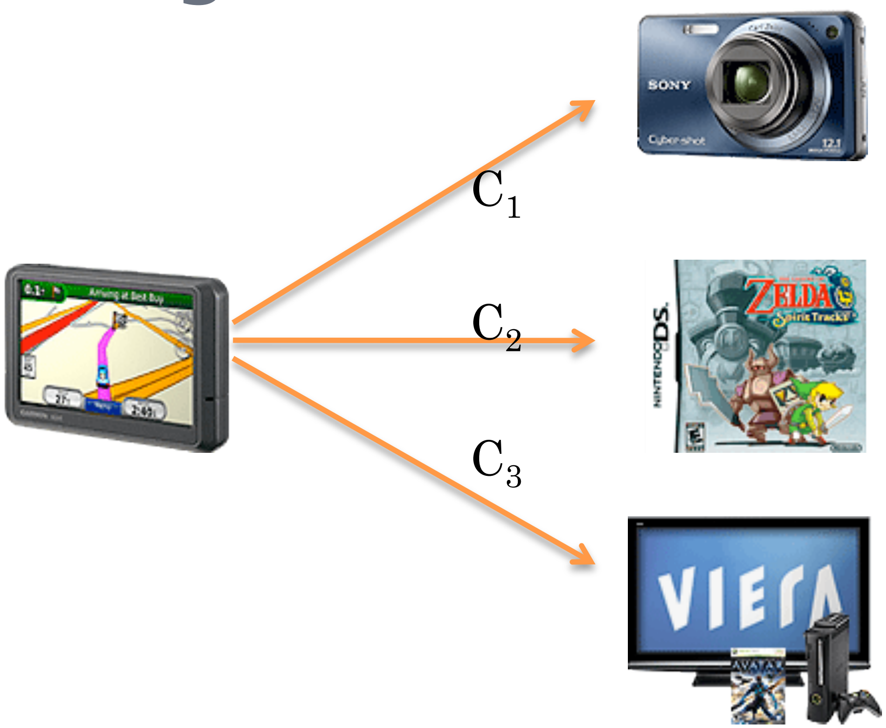
Understand context and user intent

- Context should be considered when RE making recommendation.

Month: December
Temperature: 45°F



Apply adaptive reinforcement learning

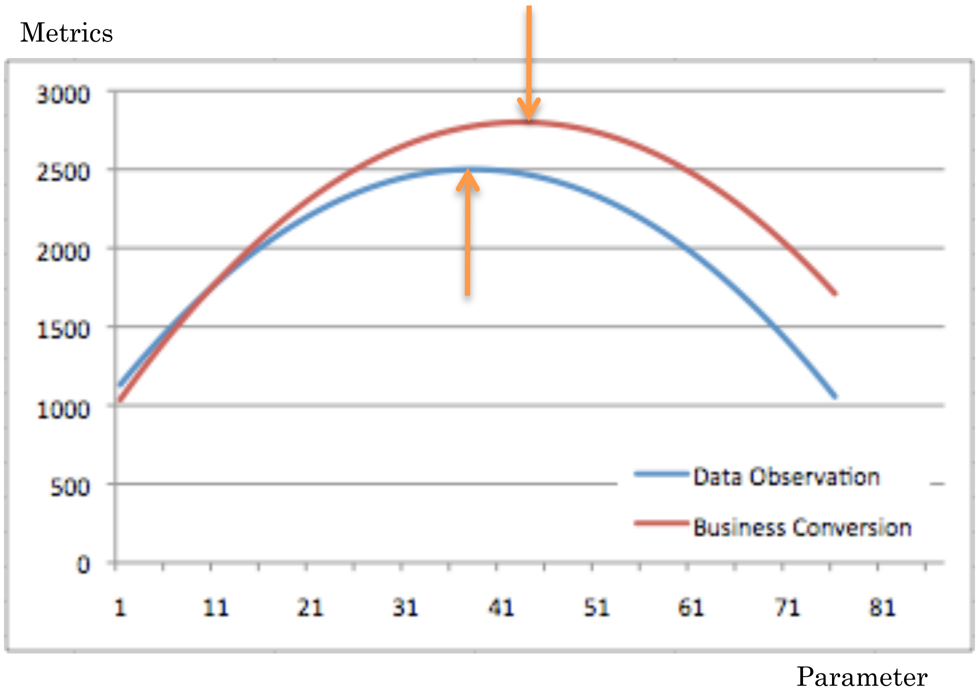


Incorporating clickstream adaptive reinforcement

$$pred(u,i) = \frac{\sum_{j \in purchasedItems(u)} itemSim(i,j) * r_{uj}}{\sum_{j \in purchasedItems(u)} itemSim(i,j)} + W_i C_i$$

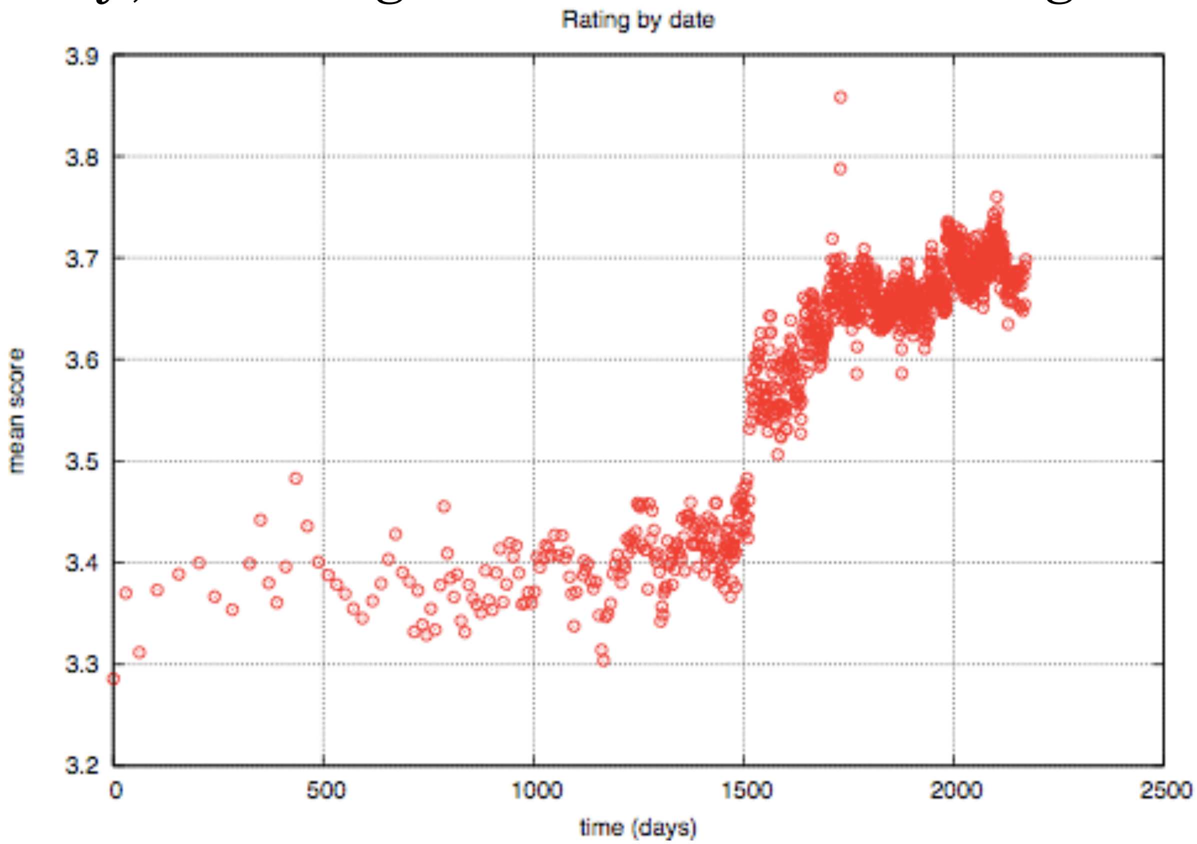
Optimize RE using cost-based models

- Cost-based engine optimization



Be aware of the data-shift issue

- Data collection UI changes will influence data significantly, creating artificial data shifting

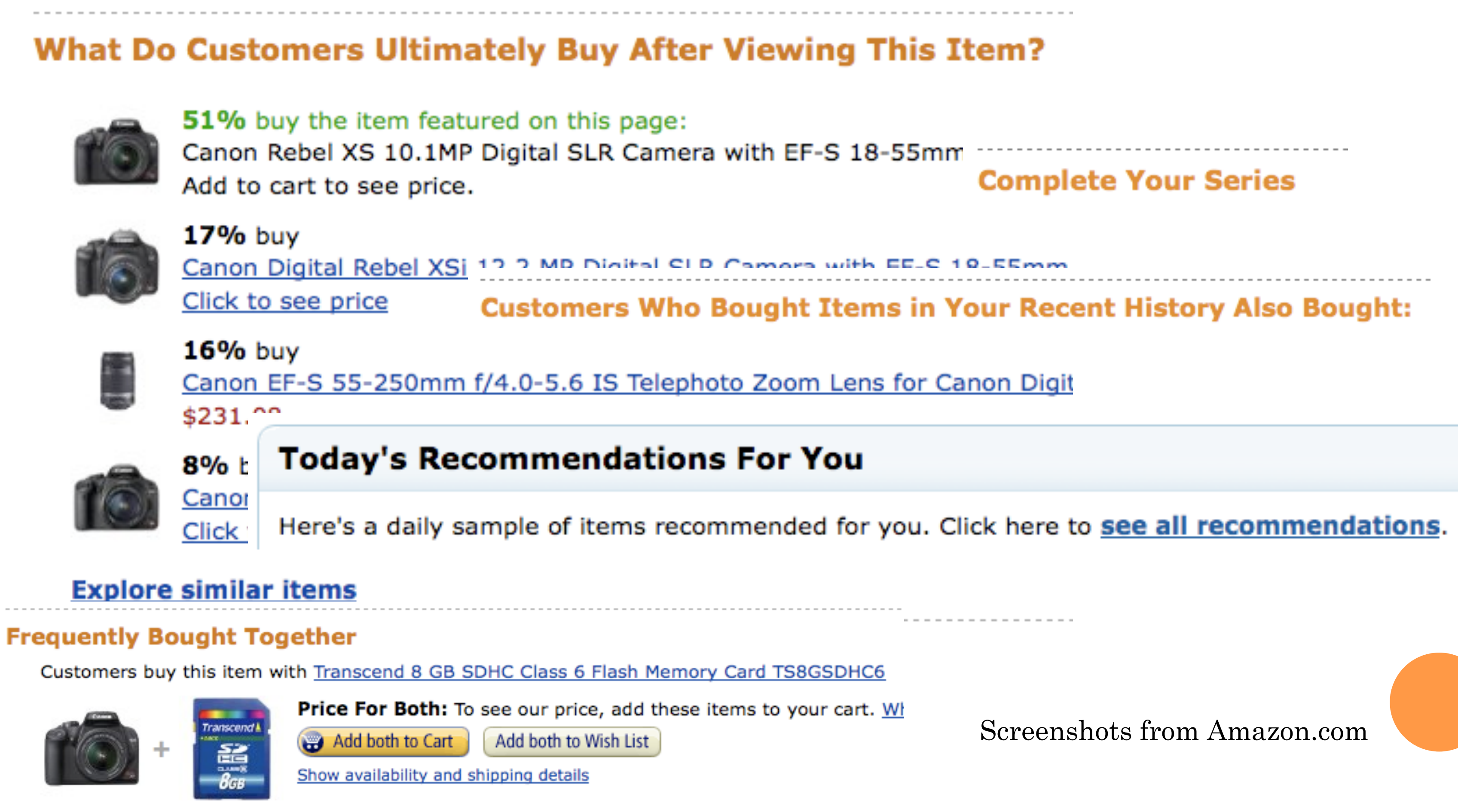


Netflix prize data set

Y. Koren, "Collaborative Filtering with Temporal Dynamics," Proc. 15th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 09), ACM Press, 2009, pp. 447-455.12

Optimize marketing message delivered with recommendation result

- It's How You Say It



Screenshots from Amazon.com

Recommendation Engine Demystified

Alex Lin
Intelligent Mining
Email: alin@intelligentmining.com
Twitter: DKALab