Recommendation Systems

Stony Brook University
CSE545, Fall 2017

Recommendation Systems

- What other item will this **user** like? (based on previously liked items)
- How much will user like item X?

Recommendation Systems

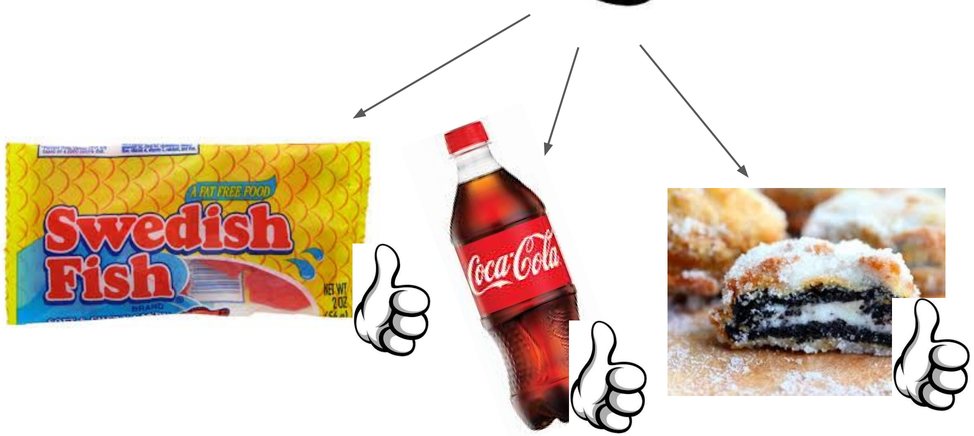
- What other item will this **user** like? (based on previously liked items)

- How much will user like item X?

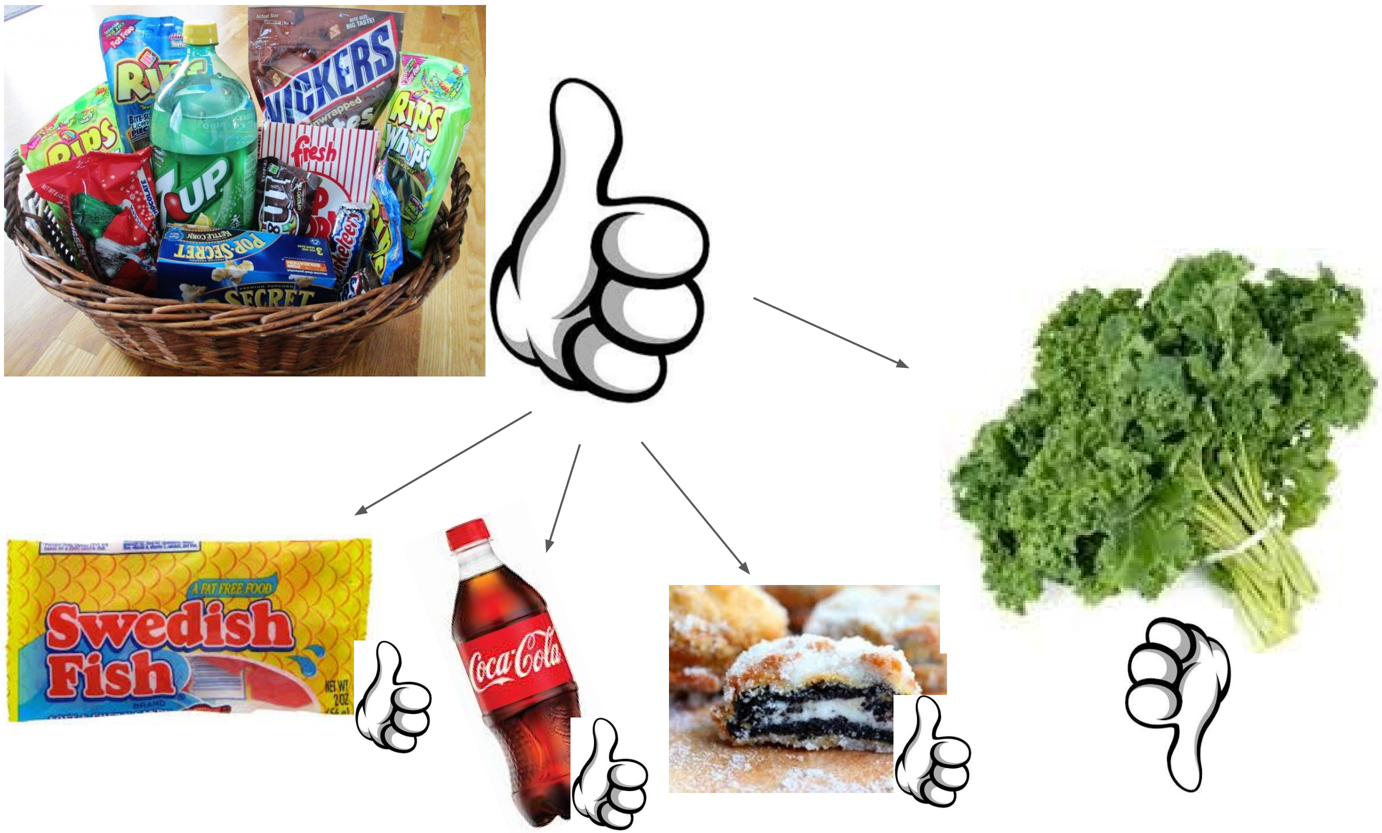
?

Recommendation Systems

- What other item will this **user** like? (based on previously liked items)
- How much will user like item X?

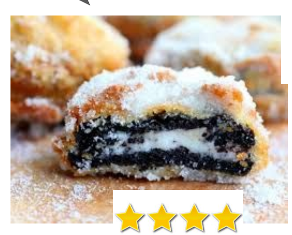


Recommendation Systems



Recommendation Systems

Past User Ratings

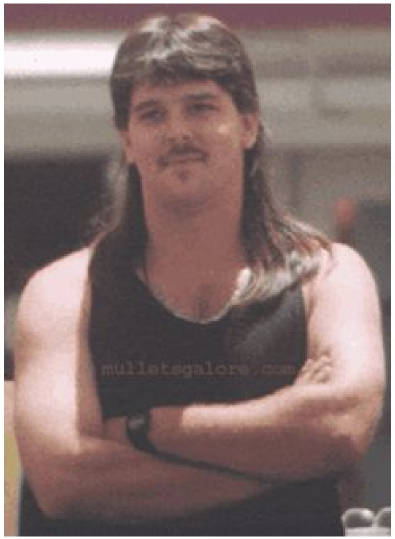


Recommendation Systems

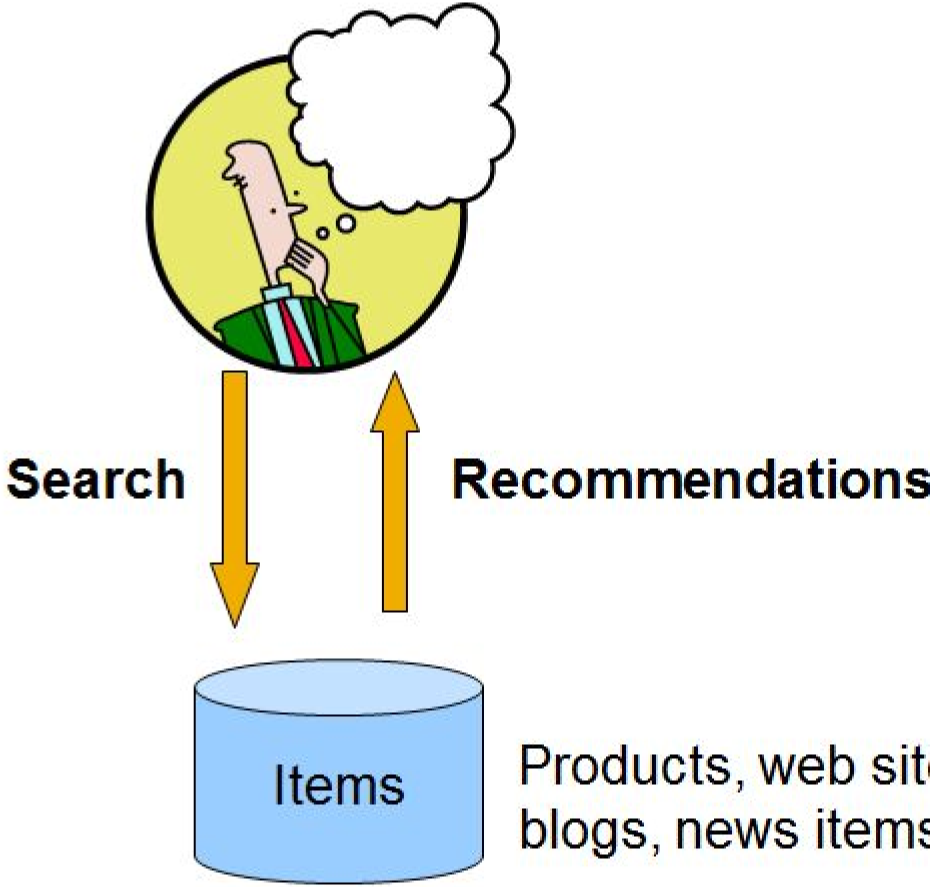
Why Big Data?

- Data with many potential features (and sometimes observations)
- An application of techniques for finding similar items
  - locality sensitive hashing
  - dimensionality reduction

Recommendation System: Example



- **Customer X**
  - Buys Metallica CD
  - Buys Megadeth CD
- **Customer Y**
  - Does search on Metallica
  - Recommender system suggests Megadeth from data collected about customer **X**



Products, web sites, blogs, news items, …

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

Enabled by Web Shopping

- Does Wal-Mart have everything you need?

Enabled by Web Shopping

- Does Wal-Mart have everything you need?

(thelongtail.com)

Enabled by Web Shopping

- Does Wal-Mart have everything you need?

- A lot of products are only of interest to a small population (i.e. "long-tail products").
- However, most people buy many products that are from the long-tail.

- Web shopping enables more choices
  - Harder to search
  - Recommendation engines to the rescue

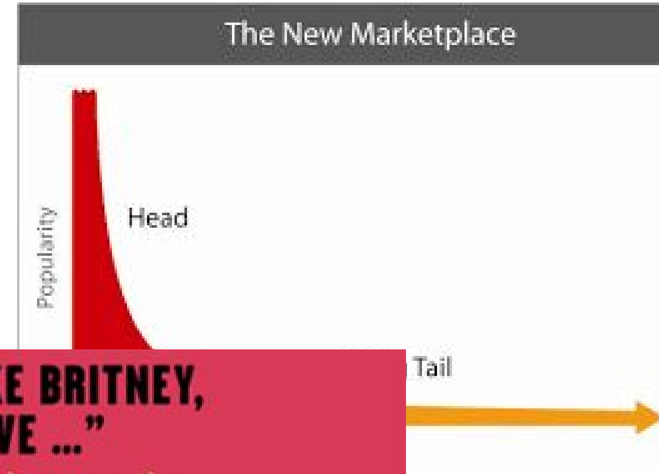
(thelongtail.com)

Enabled by Web Shopping

- Does Wal-Mart have everything you need?

- A lot of products are only of interest to a small population (i.e. “long-tail products”).
- However, most people buy many products that are fro

- Web shopp
  - Harder t
  - Recomm



A Model for Recommendation Systems

Given: *users*, *items, utility matrix*

A Model for Recommendation Systems

Given: *users*, *items*, *utility matrix*

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4	5	3		3
B	5			4	2
C			5	2	

A Model for Recommendation Systems

Given: *users*, *items*, *utility matrix*

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4	5	3		3
B	5			4	2
C	**?**	**?**	5	2	**?**

Recommendation Systems

Problems to tackle:

1. Gathering ratings

2. Extrapolate unknown ratings
   a. Explicit: based on user ratings and reviews
   (problem: only a few users engage in such tasks)
   b. Implicit: Learn from actions (e.g. purchases, clicks)
   (problem: hard to learn low ratings)

3. Evaluation

Recommendation Systems

Common Approaches

1. Content-based
2. Collaborative
3. Latent Factor

Problems to tackle:

1. Gathering ratings
2. Extrapolate unknown ratings
   a. Explicit: based on user ratings and reviews (problem: only a few users engage in such tasks)
   b. Implicit: Learn from actions (e.g. purchases, clicks) (problem: hard to learn low ratings)
3. Evaluation

Recommendation Systems

Common Approaches

1. Content-based
2. Collaborative
3. **Latent Factor**

Problems to tackle:

1. Gathering ratings
2. Extrapolate unknown ratings
   a. Explicit: based on user ratings and reviews (problem: only a few users engage in such tasks)
   b. Implicit: Learn from actions (e.g. purchases, clicks) (problem: hard to learn low ratings)
3. Evaluation

Concept, In Matrix Form:

columns:
p features

f1, f2, f3, f4, … fp

o1
o2
o3
…

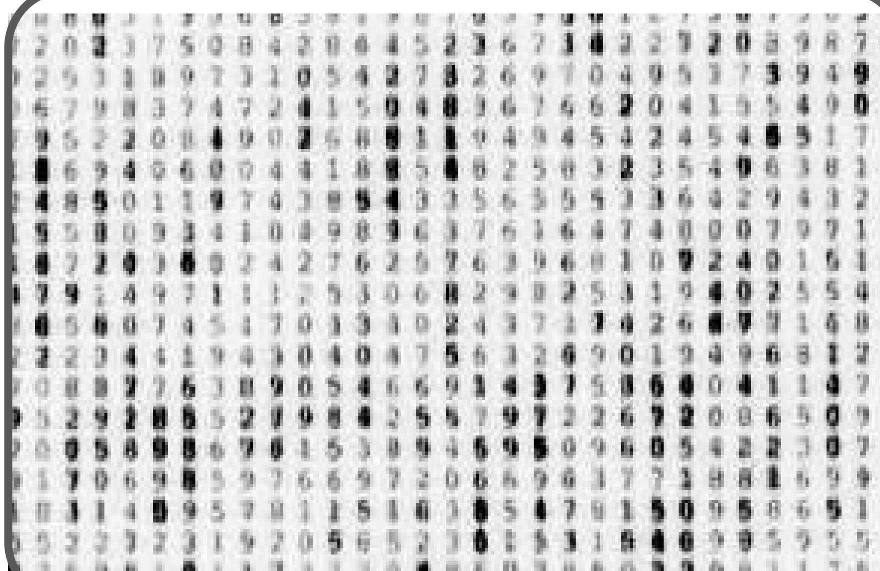
rows:
N observations

oN

Concept, In Matrix Form:

f1, f2, f3, f4, … fp

o1
o2
o3
…



oN

Concept, In Matrix Form:

Dimensionality reduction

Try to best represent but with on p’ columns.

f1, f2, f3, f4, … fp

o1
o2
o3
…
oN

c1, c2, c3, c4, … cp’

o1
o2
o3
…
oN

Dimensionality Reduction - PCA - Example

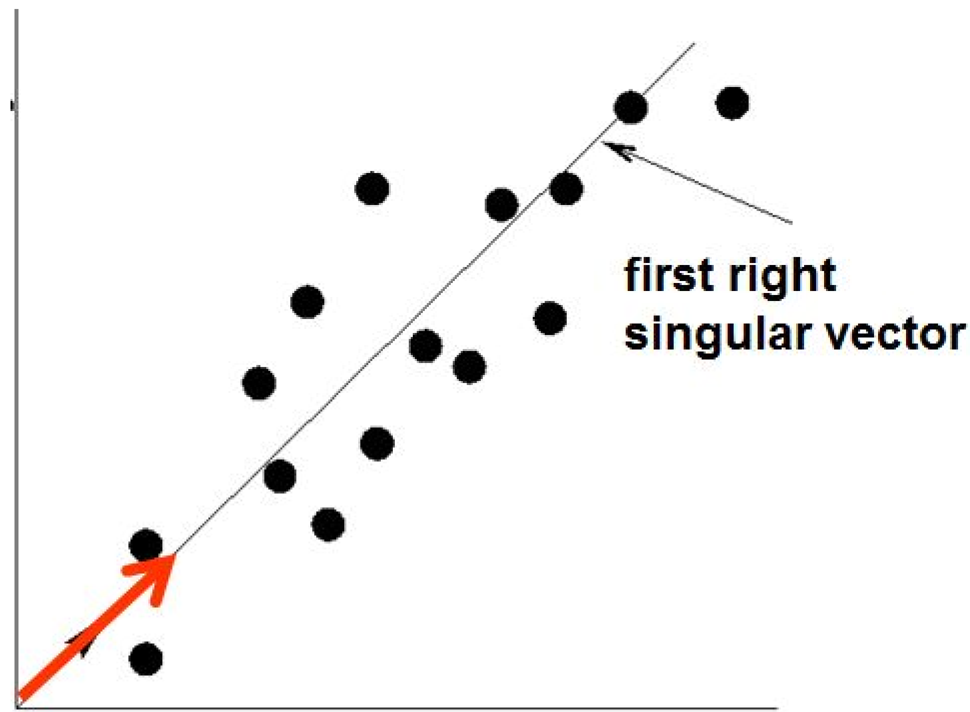
$$X_{[n \times p]} = U_{[n \times r]} \, D_{[r \times r]} \, V_{[p \times r]}^{T}$$

$$\begin{array}{cc}
 & \begin{matrix}\text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie}\end{matrix} \\
\begin{matrix}\\ \text{SciFi} \\ \\ \\ \\ \text{Romnce} \\ \\ \end{matrix} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
3 & 3 & 3 & 0 & 0 \\
4 & 4 & 4 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 2 & 0 & 4 & 4 \\
0 & 0 & 0 & 5 & 5 \\
0 & 1 & 0 & 2 & 2
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.13 & 0.02 & -0.01 \\
0.41 & 0.07 & -0.03 \\
0.55 & 0.09 & -0.04 \\
0.68 & 0.11 & -0.05 \\
0.15 & -0.59 & 0.65 \\
0.07 & -0.73 & -0.67 \\
0.07 & -0.29 & 0.32
\end{bmatrix}
\times
\begin{bmatrix}
12.4 & 0 & 0 \\
0 & 9.5 & 0 \\
0 & 0 & 1.3
\end{bmatrix}
\times
\begin{bmatrix}
0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\
0.40 & -0.80 & 0.40 & 0.09 & 0.09
\end{bmatrix}$$

Users to movies matrix

Dimensionality Reduction - PCA - Example

$$X_{[n x p]} = U_{[n x r]} D_{[r x r]} V_{[p x r]}^T$$



Dimensionality Reduction - PCA

Linear approximates of data in r dimensions.

Found via *Singular Value Decomposition:*

$$X_{[nxp]} = U_{[nxr]} D_{[rxr]} V_{[pxr]}^T$$

X: original matrix,
U: “left singular vectors”,
D: “singular values” (diagonal),
V: “right singular vectors”

Projection (dimensionality reduced space) in 3 dimensions:

$$(U_{[nx3]} D_{[3x3]} V_{[px3]}^T)$$

To reduce features in new dataset:

$$X_{new} V = X_{new\_small}$$

Recommendation Systems

Problems to tackle:

1. Gathering ratings

2. Extrapolate unknown ratings
    a. Explicit: based on user ratings and reviews (problem: only a few users engage in such tasks)
    b. Implicit: Learn from actions (e.g. purchases, clicks) (problem: hard to learn low ratings)

3. Evaluation

Common Approaches

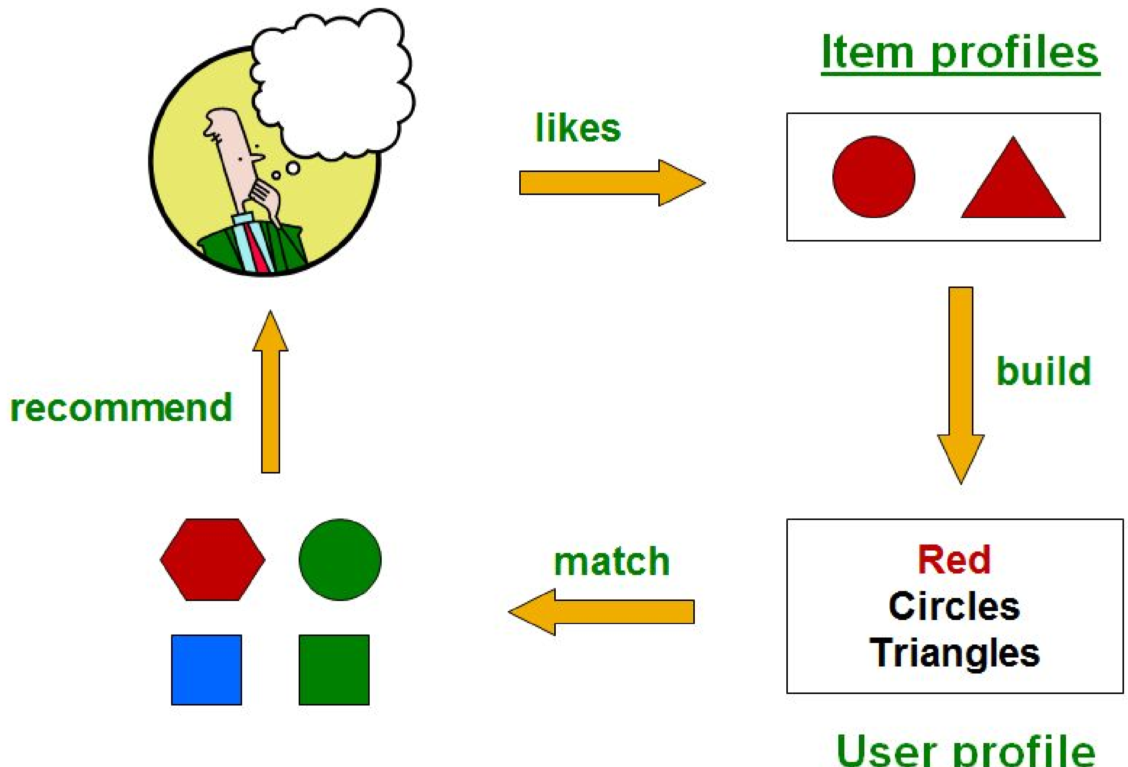
1. Content-based
2. **Collaborative**
3. Latent Factor

Content-based Rec Systems

Based on similarity of items to past items that they have rated.

Content-based Rec Systems

Based on similarity of items to past items that they have rated.



J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Content-based Rec Systems

Based on similarity of items to past items that they have rated.

1. `Build profiles of items (set of features);` examples:

    shows: producer, actors, theme, review

    people: friends, posts

    pick words with tf-idf

Content-based Rec Systems

Based on similarity of items to past items that they have rated.

1. `Build profiles of items (set of features);` examples:
   shows: producer, actors, theme, review
   people: friends, posts
   pick words with tf-idf
2. `Construct user profile from item profiles;` approach:
   average all item profiles
   variation: weight by difference from their average

Content-based Rec Systems

Based on similarity of items to past items that they have rated.

1. `Build profiles of items (set of features);` examples:

    shows: producer, actors, theme, review

    people: friends, posts

    pick words with tf-idf

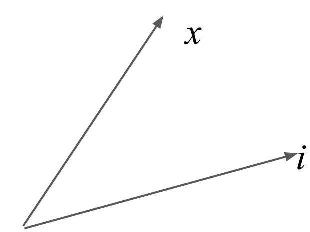
2. `Construct user profile from item profiles;` approach:

    average all item profiles of items they've purchased

    variation: weight by difference from their average ratings

3. `Predict ratings for new items;` approach:

$$utility(user, i) = \cos(x, i) = \frac{x \cdot i}{||x|| \cdot ||i||}$$



Why Content Based?

- Only need users history
- Captures unique tastes
- Can recommend new items
- Can provide explanations

Why Content Based?

- Only need users history
- Captures unique tastes
- Can recommend new items
- Can provide explanations

- Need good features
- New users don’t have history
- Doesn’t venture “outside the box” (Overspecialized)

Why Content Based?

- Only need users history
- Captures unique tastes
- Can recommend new items
- Can provide explanations

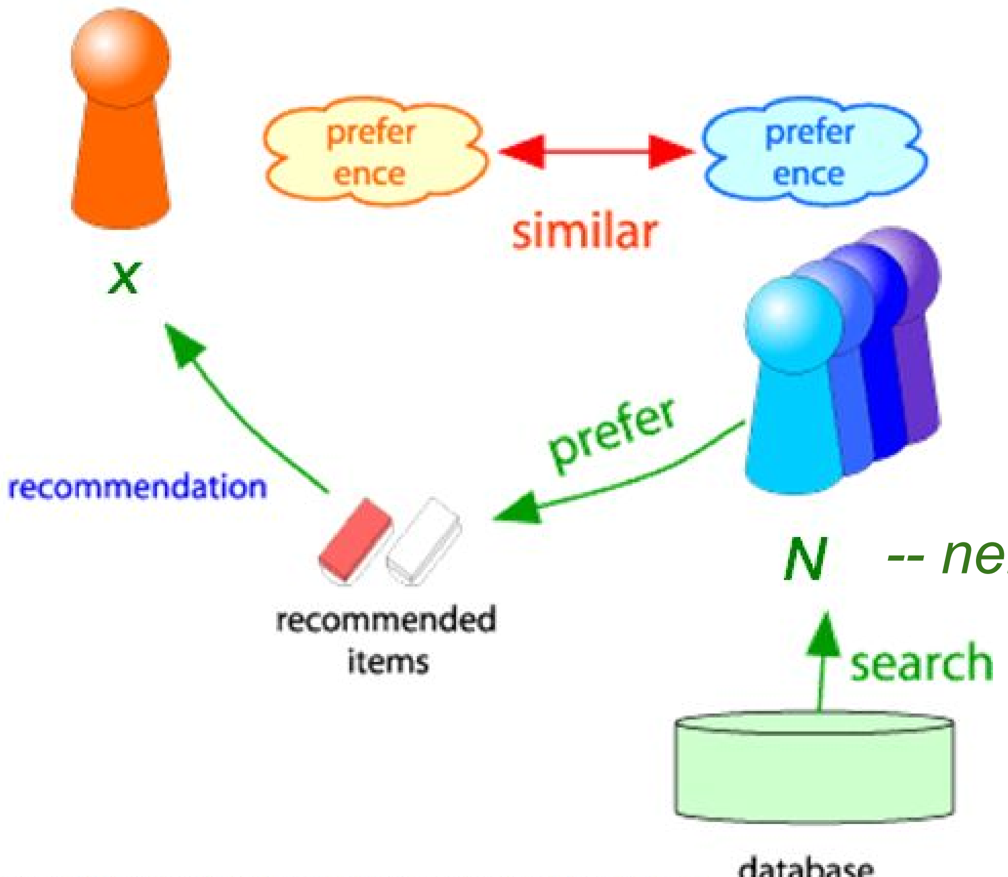
- Need good features
- New users don’t have history
- Doesn’t venture “outside the box” (Overspecialized)

(not exploiting other users judgments)

Collaborative Filtering Rec Systems

(not exploiting other users judgments)

Collaborative Filtering Rec Systems



-- neighborhood

J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4	5	2		3
B	5			4	2
C			5	2	

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4	5	2		3
B	5			4	2
C			5	2	

General Idea:

1) Find similar users = “neighborhood”

2) Infer rating based on how similar users rated

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4	5	2		3
B	5			4	2
C			5	2	

```
Given user, x, item, i
1.  Find neighborhood, N # set of k users most similar to
                              x who have also rated i
```

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4 => 0.5	5 => 1.5	2 => -1.5	=> 0	3 => -0.5
B	5			4	2
C			5	2	

Given: user, *x*; item, *i*; utility matrix, *u*

1. Find neighborhood, *N* # set of *k* users most similar to *x* who have also rated *i*

   Two Challenges: (1) user bias, (2) missing values

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4 => 0.5	5 => 1.5	2 => -1.5	=> 0	3 => -0.5
B	5			4	2
C			5	2	

Given: user, *x*; item, *i*; utility matrix, *u*

1. Find neighborhood, *N* # set of *k* users most similar to *x* who have also rated *i*

   Two Challenges: (1) user bias, (2) missing values

   Solution: subtract user's mean, add zeros for missing

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4 => 0.5	5 => 1.5	2 => -1.5	=> 0	3 => -0.5
B	5			4	2
C			5	2	

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k users most similar to
                              x who have also rated i
    -- sim(x, other) = cosine_sim(u[x], u[other])
    -- threshold to top k (e.g. k = 30)
```

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4 => 0.5	5 => 1.5	2 => -1.5	=> 0	3 => -0.5
B	5			4	2
C			5	2	

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k users most similar to
                                   x who have also rated i
    -- sim(x, other) = cosine_sim(u[x], u[other])
    -- threshold to top k (e.g. k = 30)
2. Predict utility (rating) of i based on N
```

Collaborative Filtering Rec Systems

user	Game of Thrones	Fargo	Ballers	Silicon Valley	Walking Dead
A	4 => 0.5	5 => 1.5	2 => -1.5	=> 0	3 => -0.5
B	5			4	2
C			5	2	

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k users most similar to
                                   x who have also rated i
   -- sim(x, other) = cosine_sim(u[x], u[other])
   -- threshold to top k (e.g. k = 30)
2. Predict utility (rating) of i based on N
   -- average, weighted by sim
```

$$utility(x,i) = \frac{\sum_{y \in N} Sim(x,y) \cdot utility(y,i)}{\sum_{y \in N} Sim(x,y)}$$

Collaborative Filtering Rec Systems

"User-User collaborative filtering"

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k users most similar to
                                    x who have also rated i
    -- sim(x, other) = cosine_sim(u[x], u[other])
    -- threshold to top k (e.g. k = 30)
2. Predict utility (rating) of i based on N
    -- average, weighted by sim
```

$$utility(x,i) = \frac{\sum_{y \in N} Sim(x,y) \cdot utility(y,i)}{\sum_{y \in N} Sim(x,y)}$$

Collaborative Filtering Rec Systems

"User-User collaborative filtering"

Item-Item:

Flip rows/columns of utility matrix and use same methods.
(i.e. estimate rating of item i, by finding similar items, j)

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k users most similar to
                                  x who have also rated i
    -- sim(x, other) = cosine_sim(u[x], u[other])
    -- threshold to top k (e.g. k = 30)
2. Predict utility (rating) of i based on N
    -- average, weighted by sim
```

$$utility(x,i) = \frac{\sum_{y \in N} Sim(x,y) \cdot utility(y,i)}{\sum_{y \in N} Sim(x,y)}$$

Collaborative Filtering Rec Systems

"User-User collaborative filtering"

Item-Item:

Flip rows/columns of utility matrix and use same methods.
(i.e. estimate rating of item i, by finding similar items, j)

```
Given: user, x; item, i; utility matrix, u
0. Update u: mean center, missing to 0
1.  Find neighborhood, N # set of k items most similar to
                                    i also rated by x
    -- sim(i, other) = cosine_sim(u[i], u[other])
    -- threshold to top k (e.g. k = 30)
2. Predict utility (rating) by x based on N
    -- average, weighted by sim
```

$$\text{utility}(x,i) = \frac{\sum_{j \in N} Sim(i,j) \cdot \text{utility}(x,j)}{\sum_{j \in N} Sim(i,j)}$$

Item-Item: Example

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: Example

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: Example

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**

Same as cosine sim when substracting the mean

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

$m_1 = (1+3+5+5+4)/5 = \mathbf{3.6}$

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: Example

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
<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: Example

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
<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>

***utility*(1, 5) = (0.41*2 + 0.59*3) / (0.41+0.59)**

$$\text{utility}(x,i) = \frac{\sum_{j \in N} Sim(i,j) \cdot \text{utility}(x,j)}{\sum_{j \in N} Sim(i,j)}$$

Item-Item v User-User

Item-item often works better than user-user. Why?

Users tend to be more different from each other than items are from other items.

(e.g. user A likes jazz + rock, user B likes classical + rock,
but user-A may still have same rock preferences as B;
Users span genres but items usually do not)