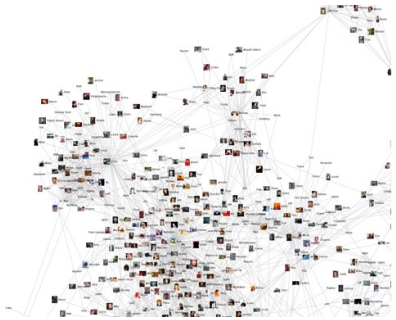
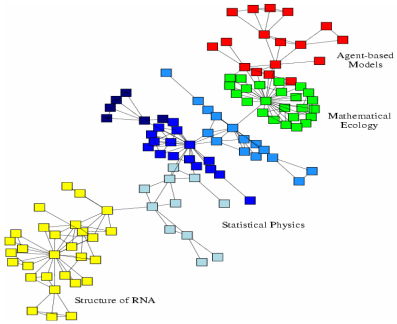
Personalised Data Analytics

Personalization with Recommender Systems

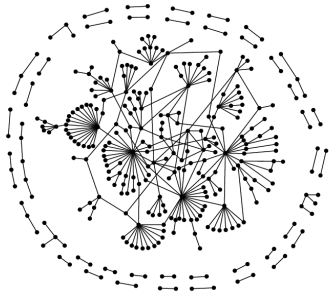
Last Week Recap:
Many Data are Networks



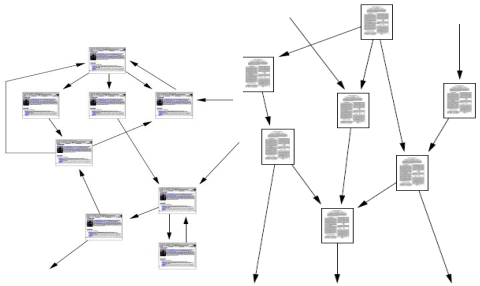
Social networks



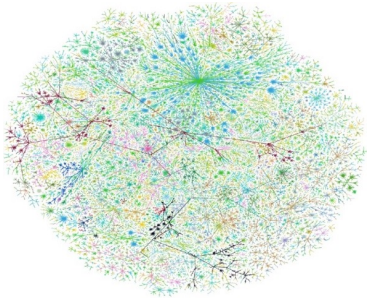
Economic networks



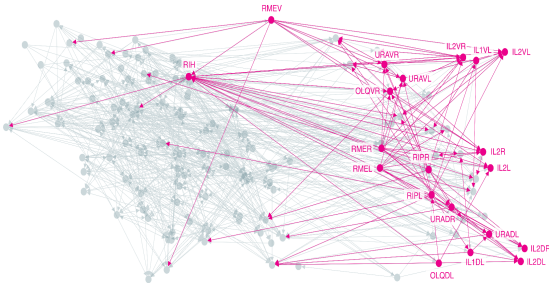
Biomedical networks



Information networks:
Web & citations



Internet



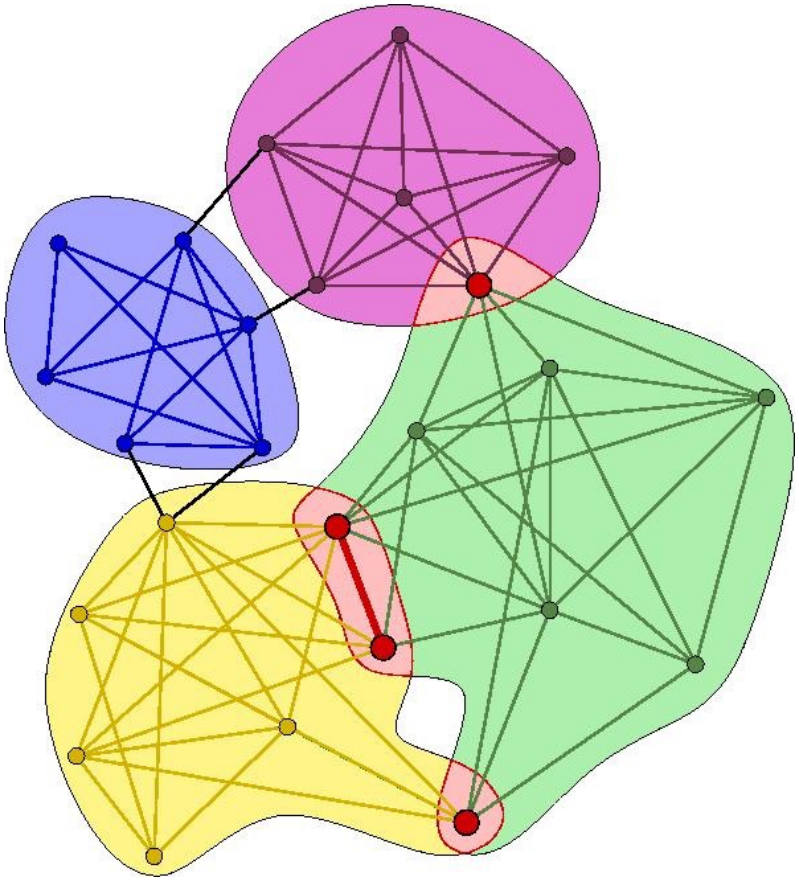
Networks of neurons

Last Week Recap: Centrality

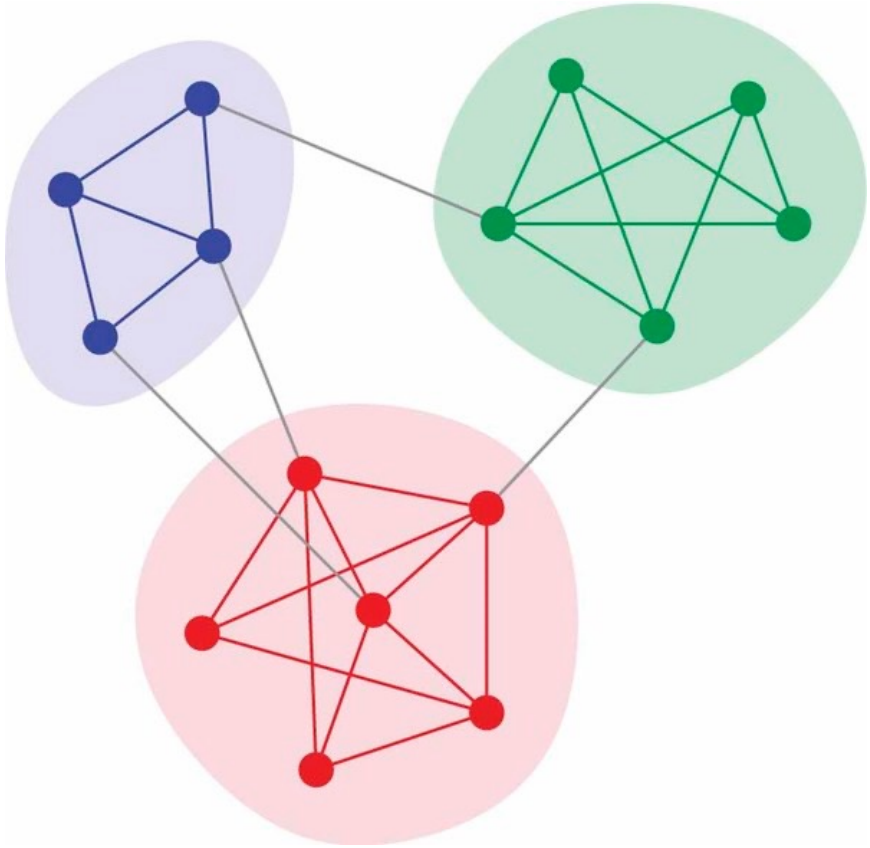
- What is centrality?
  - Centrality defines how important an actor is within a network

- Why centrality? a measure of influence
  - The act or power of producing an effect without apparent exertion of force or direct exercise of command

Last Week Recap: Community



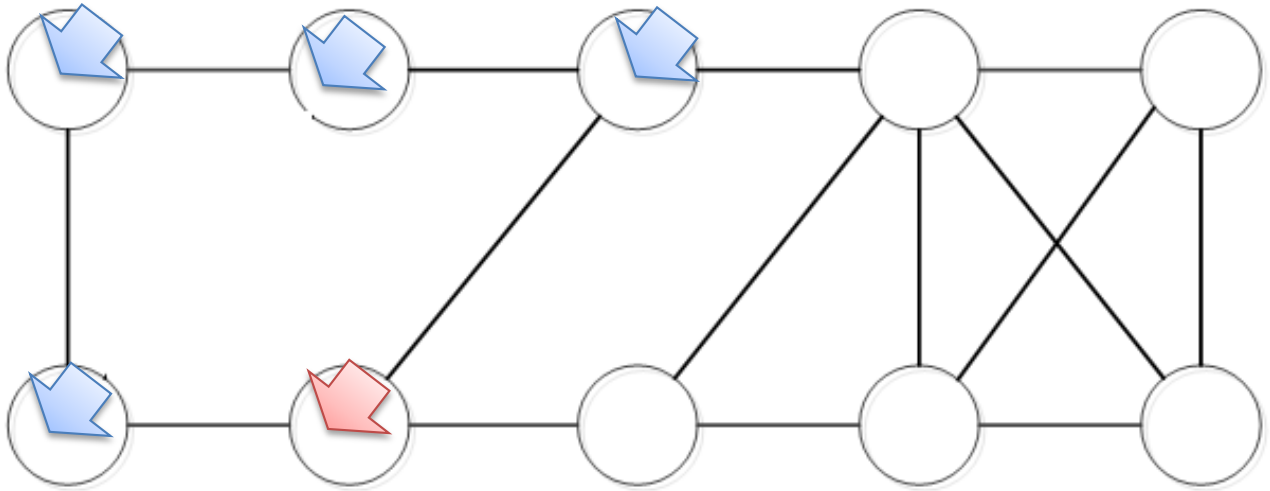
Overlapping Communities



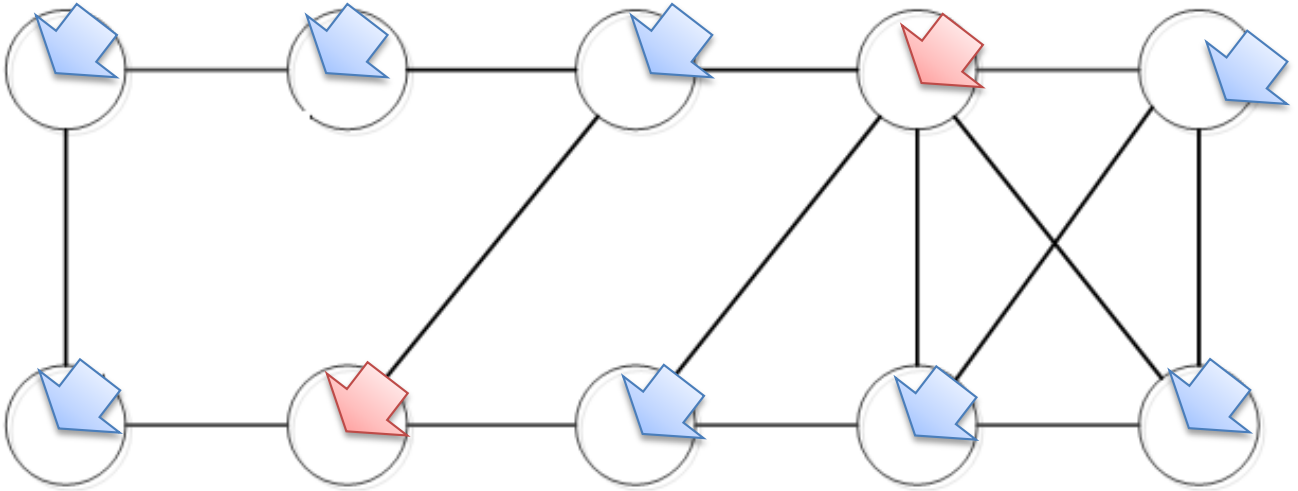
Disjoint Communities

Last Week Recap: Information Diffusion

k=1 seed



k=2 seeds



Course structure

W1. Data Processing with Python

W2. Data Exploration with Python

W3. Data Modeling with Pytyhon

W4. Data Analytics for Timeseries

Holiday

W5-6-7. Data Analytics for Texts

W8. Data Analytics for Images

W9. Data Analytics for Graphs

W10-11. **Data Analytics for Other Data**

W12. Revision

Personalized Data Analytics

I. Why Personalization?

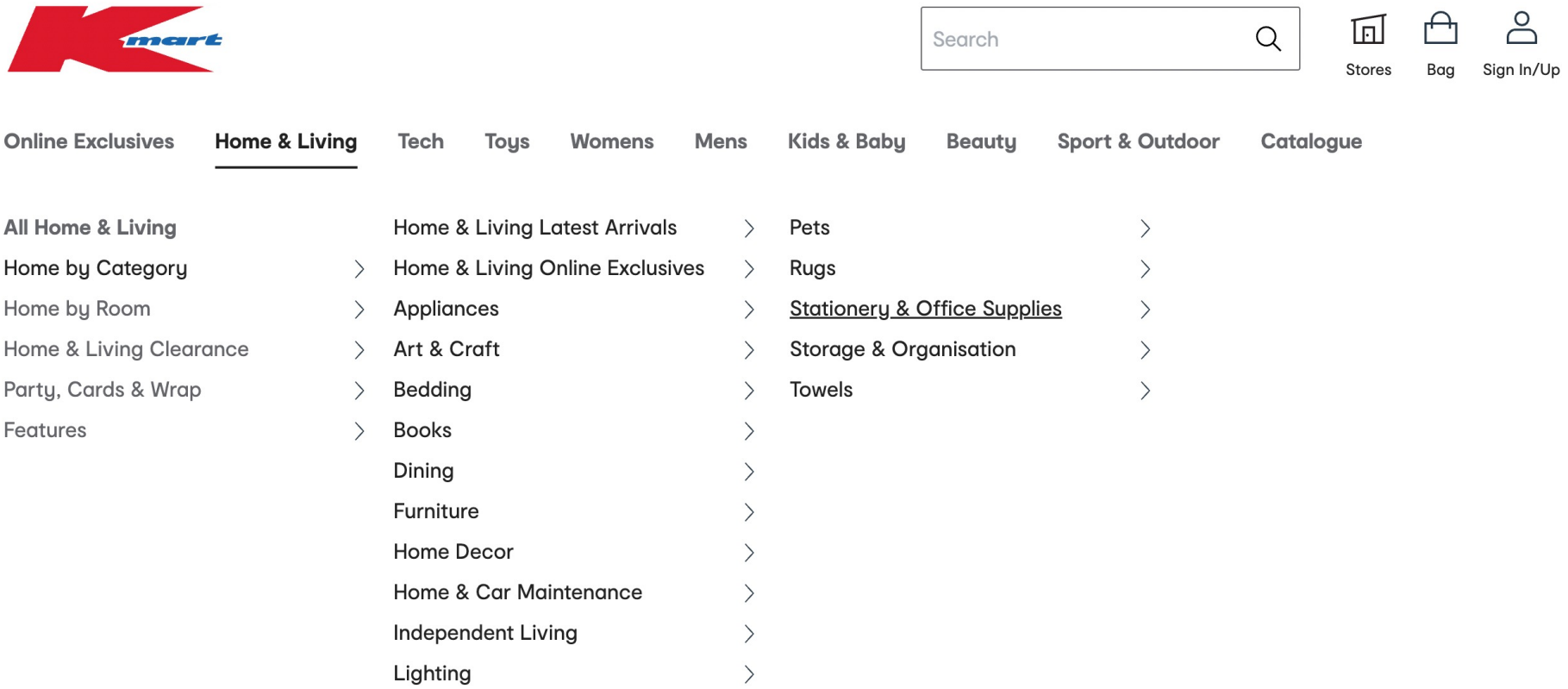
II. Recommender Systems

1. Content-based Recommendation
2. Rating-based Recommendation (Collaborative Filtering)
3. Hybrid Recommendation
4. Clustering-based Recommendation

III. Performance measures

I. PERSONALIZATION

No Personalization

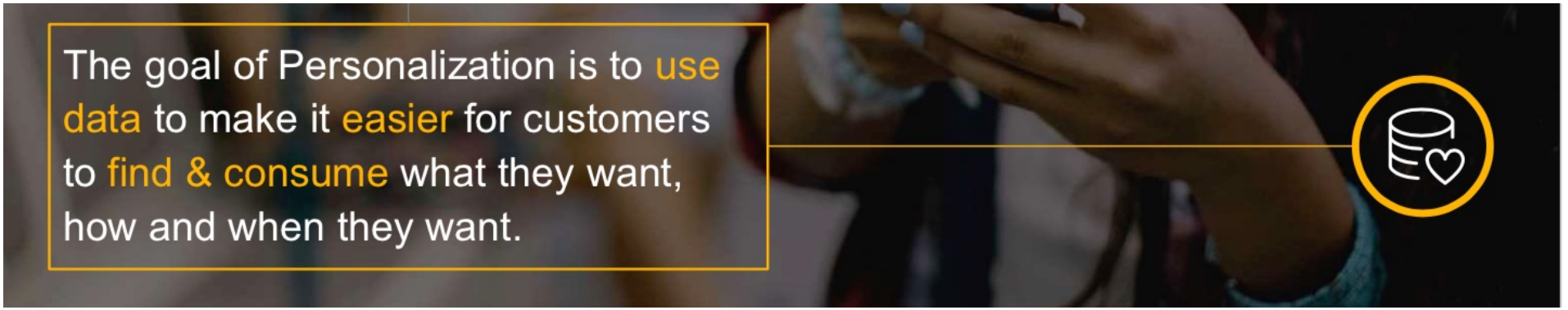


- All users see the same list of products
- Need to browse a lot to find what they want

Why Personalization?

Why Personalization?

What is Personalization?



3 dimensions of Personalization

Personalization Algorithms

- **Recognize:** classification, clustering, community analysis etc.
- **Relevance:** information retrieval, search engine, information diffusion etc.
- **Recommend:** Recommender Systems

II. RECOMMENDER SYSTEMS

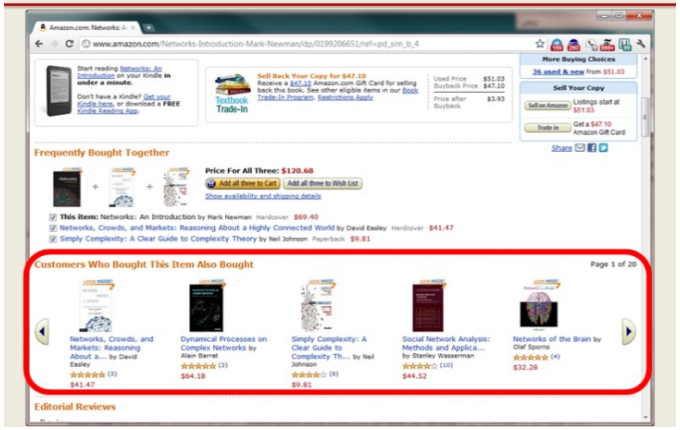
What is Recommendation?

- Which mobile phone should I buy?
- Where should I visit for my business trip?
- Whom should I follow on Twitter?
- Where should I invest my money?

- Which tour is the best for **<u>our class</u>**?

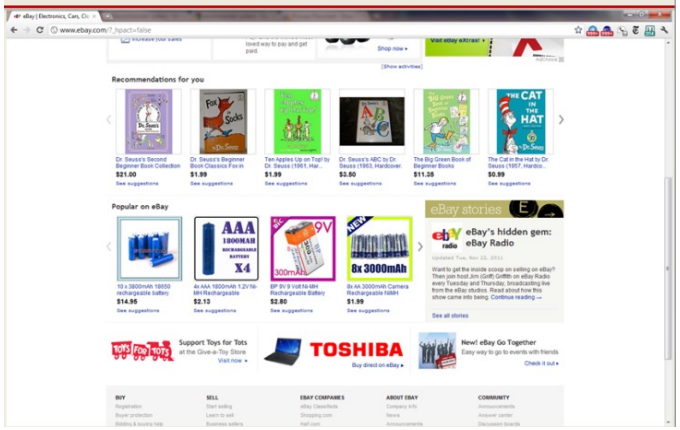
Recommender Systems - Applications

Book recommendation in Amazon

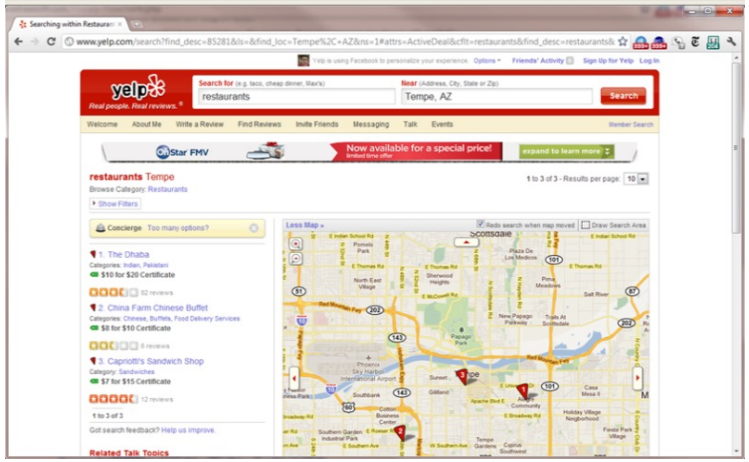


Video clip recommendation in YouTube

Product Recommendation in ebay



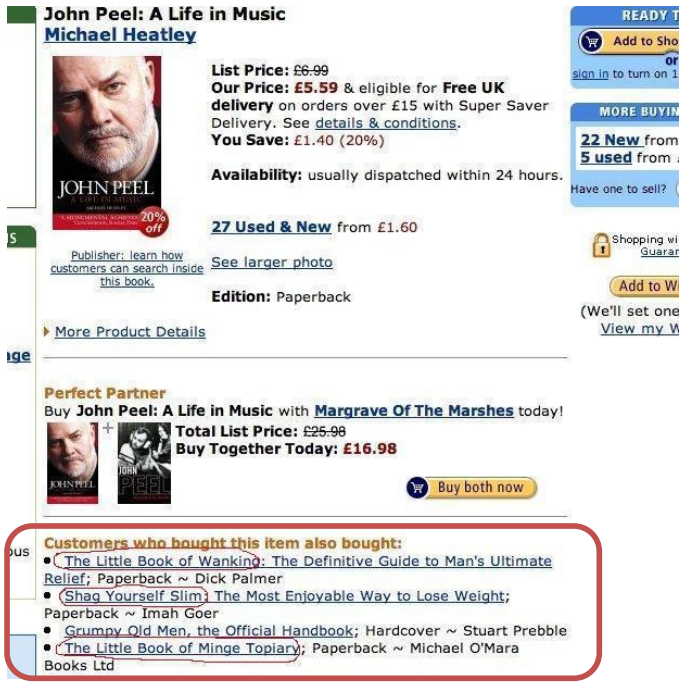
Restaurant Recommendation in Yelp



recommendation = personalized prediction

The value of recommendations

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



Recommender Systems - Problem

- Estimate a utility function that automatically predicts how a user will like an item
  - Formally, a recommender system takes a set of users U and a set of items I and learns a function f such that: $f: U \times I \rightarrow R$
  - For each user $u \in U$, we want to choose an item $i \in I$ that maximize f.
- Based on:
  - **Past behavior**
  - **Relations to other users**
  - **Item similarity**
  - **Context**
  - …

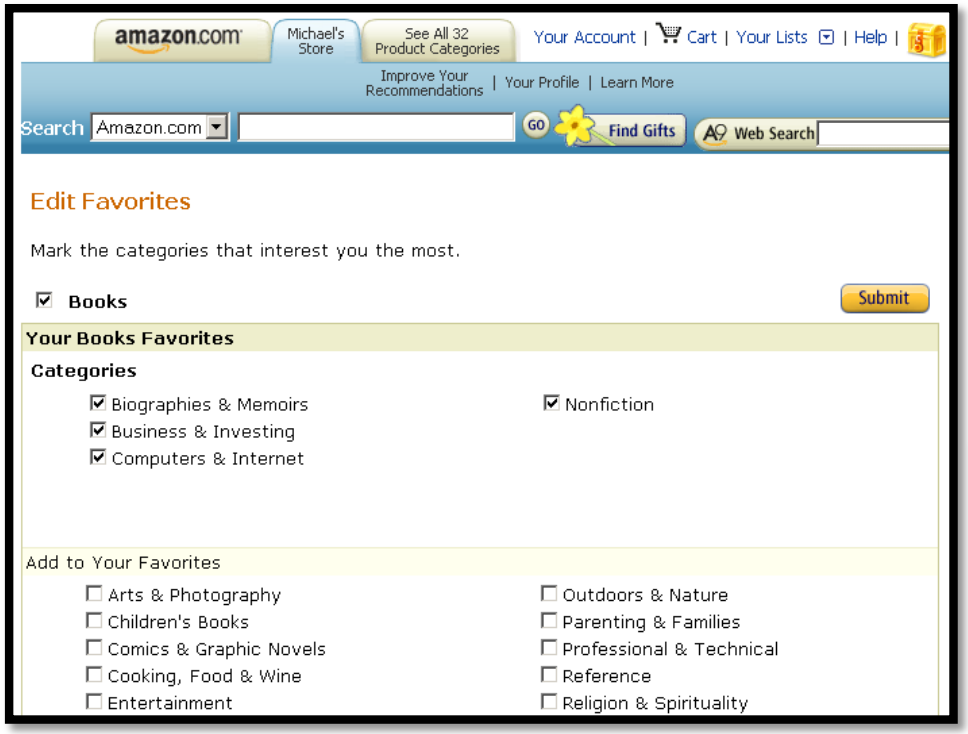
Types of Recommender Systems

1. Content-based recommendation:
   - Recommend **based on similarity** between user features and item features
2. Rating-based recommendation (Collaborative Filtering)
   - Recommend **based on rating** matrix
3. Clustering-based recommendation
   - Recommend **based on clusters of rating** matrix

1. Content-based recommendation

- Assumption: a **user's interest** should match the description of the items that the user should be recommended by the system.
  - The more similar the item's description to that of the user's interest, the more likely that the user finds the item's recommendation interesting.
- Goal: **find the similarity** between the user and all of the existing items is the core of this type of recommender systems

Content-based Recommendation: An Example



User profile

Items recommended

Content-based Recommendation: Information Retrieval Approach

- We represent user profiles and item descriptions by vectorizing them using a set of k keywords (e.g., using **TF-IDF**)

$$I_j = (i_{j,1}, i_{j,2}, \ldots, i_{j,k})$$
$$U_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,k})$$

- Compute their (cosine) similarity

$$sim(U_i, I_j) = cos(U_i, I_j) = \frac{\sum_{l=1}^{k} u_{i,l} i_{j,l}}{\sqrt{\sum_{l=1}^{k} {u_{i,l}}^2} \sqrt{\sum_{l=1}^{k} {i_{j,l}}^2}}$$

- Recommend the top most similar items to the user

Algorithm Content-based recommendation

Require: User i's Profile Information, Item descriptions for items $j \in \{1, 2, \ldots, n\}$, k keywords, r number of recommendations.

1: **return** r recommended items.
2: $U_i = (u_1, u_2, \ldots, u_k)$ = user i's profile vector;
3: $\{I_j\}_{j=1}^{n} = \{(i_{j,1}, i_{j,2}, \ldots, i_{j,k})$ = item j's description vector$\}_{j=1}^{n}$;
4: $s_{i,j} = sim(U_i, I_j), 1 \leq j \leq n$;
5: Return top r items with maximum similarity $s_{i,j}$.

Content-based RecSys: Summary

(+) Not depend on other users
(+) Lower calculation cost
(+) Can recommend niche items that very few other users are interested in

(-) Hard to collect description of item, need hand-engineered features
(-) Limited ability to expand on the users' existing interests.

2. Rating-based Recommendation

- AKA Collaborative filtering: recommend items by only **users' past behavior**
- Advantage: we don't need to have additional information about the users or content of the items
  - Users' **rating** or **purchase history** is the only information that is needed to work
- Input: Rating matrix
  - Users rate (rank) items (purchased, watched)
  - Explicit ratings: entered by a user directly
  - Implicit ratings: inferred from other user behavior
    - e.g. the amount of time users spent on a webpage
    - e.g. the number of times users listen to a song

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?

2.1. Nearest-Neighbour Approach

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

	I1	I2	I3	I4
U1	1	2	4	4
U2	1	2	4	?
U3	2	5	2	2
U4	5	2	3	3

User-based

- Find similar users to me and recommend what they liked
- Why?: users with similar **previous** ratings for items are likely to rate future items similarly

Item-based

- Find similar items to those that I have previously liked
- Why?: Items that have received similar ratings **previously** from users are likely to receive similar ratings from future users

Nearest-Neighbour Approach (Cont'd)

User-based

1. Weigh all users with respect to their similarity with the current user
2. Select a subset of the users (e.g. top-k neighbors) as recommenders
3. Predict the rating of the user for specific items using neighbors' ratings for the same (or similar) users
4. Recommend items with the highest predicted rank

Item-based

1. Weigh all items with respect to their similarity with the current item
2. Select a subset of the items (e.g. top-k neighbors) as recommenders
3. Predict the rating of the user for specific items using neighbors' ratings for the same (or similar) items
4. Recommend items with the highest predicted rank

User-based vs. Item-based CF

- Finds the **most similar users** to the current user
- Cosine Similarity:

$$sim(u_1, u_2) = \frac{u_1 . u_2}{||u_1||.||u_2||}$$

$$= \frac{\sum_i r_{1,i} r_{2,i}}{\sqrt{\sum_i r_{1,i}^2} \sqrt{\sum_i r_{2,i}^2}}$$

- Pearson Similarity (Correlation Coefficient):

$$sim(u_1, u_2)$$

$$= \frac{\sum_i (r_{1,i} - \overline{r_{u_1}})(r_{2,i} - \overline{r_{u_2}})}{\sqrt{\sum_i (r_{1,i} - \overline{r_{u_1}})^2} \sqrt{\sum_i (r_{2,i} - \overline{r_{u_2}})^2}}$$

- Finds the **most similar items** to the current item
- Cosine Similarity:

$$sim(i_1, i_2) = \frac{i_1 . i_2}{||i_1||.||i_2||}$$

$$= \frac{\sum_u r_{u,1} r_{u,2}}{\sqrt{\sum_u r_{u,1}^2} \sqrt{\sum_u r_{u,2}^2}}$$

- Pearson Similarity (Correlation Coefficient):

$$sim(i_1, i_2)$$

$$= \frac{\sum_u (r_{u,1} - \overline{r_{i_1}})(r_{u,2} - \overline{r_{i_2}})}{\sqrt{\sum_u (r_{u,1} - \overline{r_{i_1}})^2} \sqrt{\sum_u (r_{u,2} - \overline{r_{i_2}})^2}}$$

User-based vs. Item-based CF

❖ Update the ratings:

$$r_{u,i} = \overline{r_u} + \frac{\sum_{v \in N(u)} sim(u,v)(r_{v,i} - \overline{r_v})}{\sum_{v \in N(u)} sim(u,v)}$$

$r_{u,i}$: predicted rating of user u for item i

$\overline{r_u}$: user u's mean rating

$\overline{r_v}$: user v's mean rating

$r_{v,i}$: observed rating of user v for item i

❖ Update the ratings:

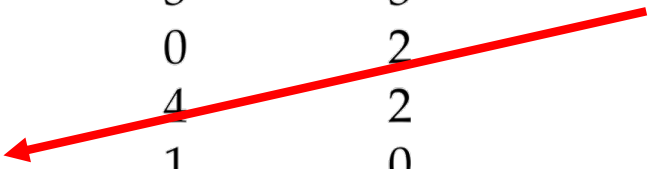
$$r_{u,i} = \overline{r_i} + \frac{\sum_{j \in N(i)} sim(i,j)(r_{u,j} - \overline{r_j})}{\sum_{j \in N(i)} sim(i,j)}$$

$\overline{r_i}$: item i's mean rating

$\overline{r_j}$: item j's mean rating

User-based CF, Example

	Lion King	Aladdin	Mulan	Anastasia
John	3	0	3	3
Joe	5	4	0	2
Jill	1	2	4	2
Jane	3	?	1	0
Jorge	2	2	0	1



1- Calculate average ratings

$$\bar{r}_{John} = \frac{3+3+0+3}{4} = 2.25$$

$$\bar{r}_{Joe} = \frac{5+4+0+2}{4} = 2.75$$

$$\bar{r}_{Jill} = \frac{1+2+4+2}{4} = 2.25$$

$$\bar{r}_{Jane} = \frac{3+1+0}{3} = 1.33$$

$$\bar{r}_{Jorge} = \frac{2+2+0+1}{4} = 1.25$$

2- Calculate user-user similarity

$$sim(Jane, John) = \frac{3\times3+1\times3+0\times3}{\sqrt{10}\sqrt{27}} = 0.73$$

$$sim(Jane, Joe) = \frac{3\times5+1\times0+0\times2}{\sqrt{10}\sqrt{29}} = 0.88$$

$$sim(Jane, Jill) = \frac{3\times1+1\times4+0\times2}{\sqrt{10}\sqrt{21}} = 0.48$$

$$sim(Jane, Jorge) = \frac{3\times2+1\times0+0\times1}{\sqrt{10}\sqrt{5}} = 0.84$$

User-based CF, Example (cont'd)

3- Calculate Jane's rating for Aladdin, Assume that neighborhood size = 2

$$r_{Jane,Aladdin} = \bar{r}_{Jane} + \frac{sim(Jane,Joe)(r_{Joe,Aladdin} - \bar{r}_{Joe})}{sim(Jane,Joe) + sim(Jane,Jorge)}$$

$$+ \frac{sim(Jane,Jorge)(r_{Jorge,Aladdin} - \bar{r}_{Jorge})}{sim(Jane,Joe) + sim(Jane,Jorge)}$$

$$= 1.33 + \frac{0.88(4 - 2.75) + 0.84(2 - 1.25)}{0.88 + 0.84} = 2.33$$

Item-based CF, Example

1- Calculate average ratings

$$\bar{r}_{Lion\ King} = \frac{3+5+1+3+2}{5} = 2.8$$

$$\bar{r}_{Aladdin} = \frac{0+4+2+2}{4} = 2.$$

$$\bar{r}_{Mulan} = \frac{3+0+4+1+0}{5} = 1.6$$

$$\bar{r}_{Anastasia} = \frac{3+2+2+0+1}{5} = 1.6$$

2- Calculate item-item similarity

$$sim(Aladdin, Lion\ King) = \frac{0\times3+4\times5+2\times1+2\times2}{\sqrt{24}\sqrt{39}} = 0.84$$

$$sim(Aladdin, Mulan) = \frac{0\times3+4\times0+2\times4+2\times0}{\sqrt{24}\sqrt{25}} = 0.32$$

$$sim(Aladdin, Anastasia) = \frac{0\times3+4\times2+2\times2+2\times1}{\sqrt{24}\sqrt{18}} = 0.67$$

3- Calculate Jane's rating for Aladdin, Assume that neighborhood size = 2

$$r_{Jane,Aladdin} = \bar{r}_{Aladdin} + \frac{sim(Aladdin, Lion\ King)(r_{Jane,Lion\ King} - \bar{r}_{Lion\ King})}{sim(Aladdin, Lion\ King) + sim(Aladdin, Anastasia)} + \frac{sim(Aladdin, Anastasia)(r_{Jane,Anastasia} - \bar{r}_{Anastasia})}{sim(Aladdin, Lion\ King) + sim(Aladdin, Anastasia)}$$

$$= 2 + \frac{0.84(3-2.8) + 0.67(0-1.6)}{0.84+0.67} = 1.40$$

2.2. Rating-based Recommendation: Embedding/Latent Approach (OPTIONAL)

- Assume each item has a feature vector of d dimensions
  - But don't know what these features are
- Similarly, each user also has a preference vector of d dimensions
  → Each user has different preferences on these features

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0	?	?
Cute puppies of love	?	4	0	?	?	?
Nonstop car chases	0	0	5	4	?	?
Swords vs. karate	0	0	5	?	?	?

2.2.1. Dual Optimization Method

Minimizing $x^{(1)}, \ldots, x^{(n_m)}$ and $\theta^{(1)}, \ldots, \theta^{(n_u)}$ simultaneously

$$J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)}) =$$

$$\frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^{(i)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

Want: $\min_{\substack{x^{(1)}, \ldots, x^{(n_m)} \\ \theta^{(1)}, \ldots, \theta^{(n_u)}}} J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$

Dual Optimization Method (cont'd)

Iterative Algorithm:

1. Initialize $x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)}$ to small random values.
2. Minimize $J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$ using gradient descent (or an advanced optimization algorithm). E.g. for every $j = 1, \ldots, n_u, i = 1, \ldots, n_m$:

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

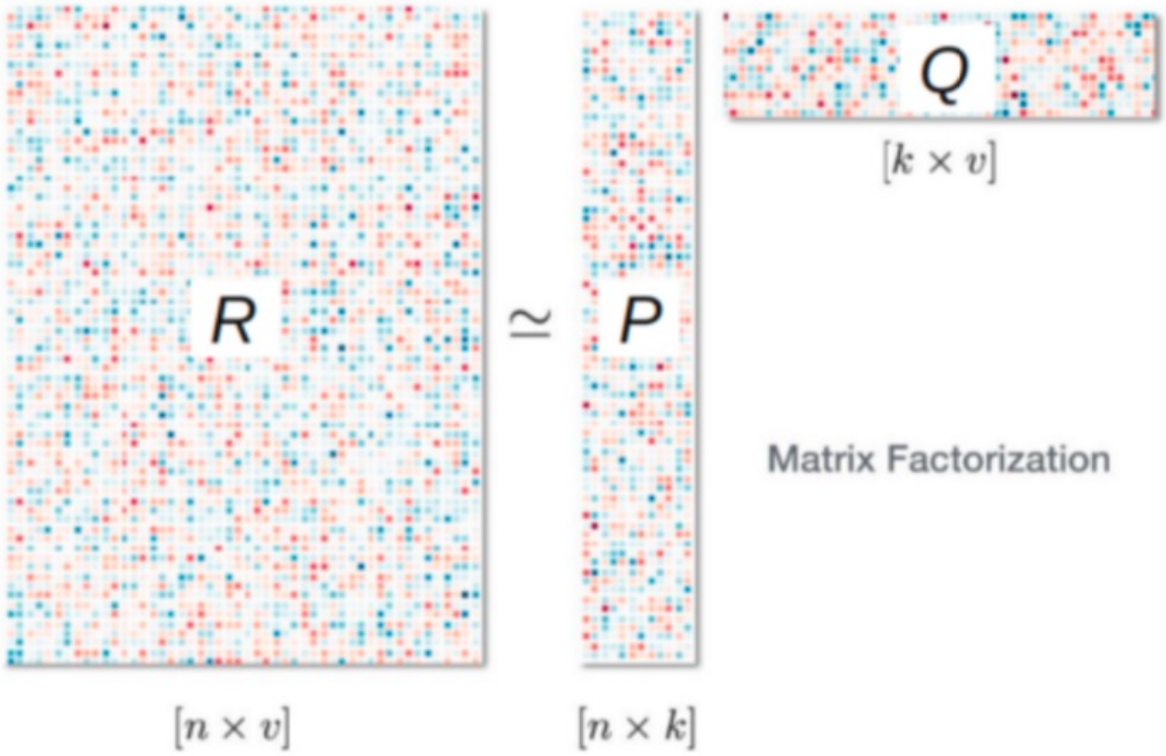
$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

3. For a user with parameters θ and a movie with (learned) features x, predict a star rating of $\theta^T x$

2.2.2. Matrix factorization method

- Find two smaller matrices P and Q such that the ratings matrix equals to P · Q (or approximates it)

Item (embedding/latent) matrix



User (embedding/latent) matrix

How to compute? SVD, VD++, Deep-Learning, … (sorry, too many to cover)

Collaborative Filtering: Summary

(+) Don't need domain knowledge because the embeddings are automatically learned

(+) The model can help users discover new interests

(-) Cold-start problem: new item or new user?
(-) Sparsity problem: too few ratings to infer anything
(-) Hard to include side features

3. Hybrid Recommendation: Rating + Content (OPTIONAL)

d

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

$x^{(i)}$

- For each user j, learn a parameter vector $\theta^{(j)} \in R^d$. Predict that user j will rate movie i with $\theta^{(j)} \odot x^{(i)}$ stars

Regression Approach: Notations

$r(i,j) = 1$ if user j has rated movie i (0 otherwise)
$y^{(i,j)} =$ rating by user j on movie i (if defined)

$\theta^{(j)}$ = parameter vector for user j → User preference for each movie feature
$x^{(i)}$ = feature vector for movie i
For user j, movie i, predicted rating: $(\theta^{(j)})^T(x^{(i)})$

$m^{(j)}$ = no. of movies rated by user j
To learn $\theta^{(j)}$:

Regularized Mean Squared Error

Optimization objective:

To learn $\theta^{(j)}$ (parameter for user j):

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

To learn $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$:

$$\min_{\theta^{(1)},\ldots,\theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

Model Training

Optimization algorithm:

$$\min_{\theta^{(1)},\ldots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

Gradient descent update:

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} \quad (\text{for } k = 0)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} + \lambda\theta_k^{(j)}\right) \quad (\text{for } k \neq 0)$$

4. Clustering-based Recommendation

- **Limitations** of rating-based recommendation:
  - Hard to **scale** with large data
  - Bad with **sparse** rating matrix
  - Bad with **diversity** of users and items

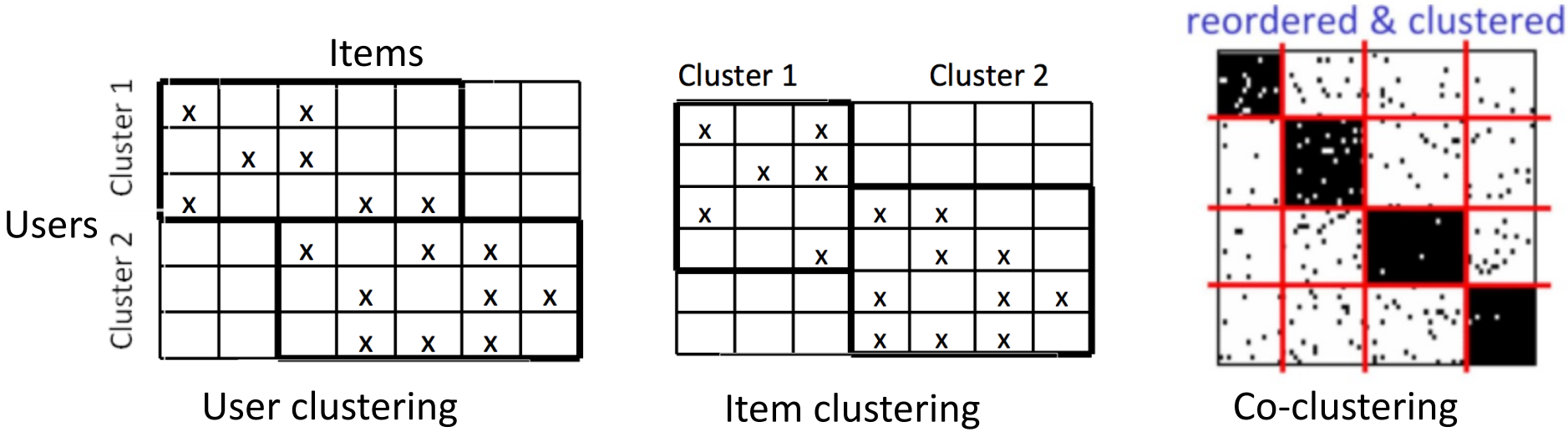
Items

Users

X		X				
	X	X				
X			X	X		
		X		X	X	
			X		X	X
			X	X	X	

Clustering-based Recommendation

- Solution: **clustering** the data before-hand
  - Clustering based on ratings: k-means [*], etc.
  - **One-dimension clustering**: user clustering, item clustering
  - **Co-clustering**
- Then, perform rating-based recommendation on each cluster



[*] Al Mamunur Rashid, Shyong K. Lam, George Karypis, and John Riedl. "ClustKNN: a highly scalable hybrid model-& memory-based CF algorithm." *Proceeding of WebKDD* 2006 (2006).

Clustering-based Recommendation – How does it work?

- Any customer that shall be classified as a member of **CLUSTER** will received recommendations based on preferences of the group:
  - Book 2 will be highly recommended to Customer F
  - Book 6 will also be recommended to some extent
- **Pros:**
  - Overcome the sparse data problem
  - Capture latent similarities between users and items
- **Cons:**
  - Recommendations (per cluster) maybe less relevant than collaborative filtering (per individual)

	Book1	Book2	Book3	Book4	Book5	Book6
Customer A	X			X		
Customer B		X	X		X	
Customer C		X	X			
Customer D		X				X
Customer E	X				X	
Customer F			X		X	

III. PERFORMANCE MEASURES

RecSys - Performance Measures

- Qualitative measures:
  - **User Satisfaction** (e.g. questionnaire)
- Quantitative measures:
  - If ground truth is not available:
    - Add-on sales
    - Click-through rates
    - Number of products purchased
  - If ground truth is available:
    - **Predictive accuracy**:
      - The ratio of predicted ratings being the true user ratings?
    - Rank accuracy

Predictive accuracy

- **Mean Absolute Error (*MAE*).** The average absolute deviation between a predicted rating (p) and the user's true rating (r)
  - $NMAE = MAE/(r_{max} - r_{min})$

$$MAE = \frac{\sum_{ij} |\hat{r}_{ij} - r_{ij}|}{n}$$

- **Root Mean Square Error (*RMSE*).** Similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n} \sum_{i,j} (\hat{r}_{ij} - r_{ij})^2}$$

Evaluation Example

Item	*Predicted Rating*	*True Rating*
1	*1*	*3*
2	*2*	*5*
3	*3*	*3*
4	*4*	*2*
5	*4*	*1*

$$MAE = \frac{|1-3| + |2-5| + |3-3| + |4-2| + |4-1|}{5} = 2$$

$$NMAE = \frac{MAE}{5-1} = 0.5$$

$$RMSE = \sqrt{\frac{(1-3)^2 + (2-5)^2 + (3-3)^2 + (4-2)^2 + (4-1)^2}{5}}$$

$$= 2.28$$

Rank Accuracy

- **Kendall's τ**
  - **Compares concordant** the items of the recommended ranking list **against the ground truth** ranking list
    - If the two orders are consistent, it is concordant
    - E.g., for top 4 items in ranking list, there are 4×3/2=6 pairs

$$\tau = \frac{c-d}{\binom{n}{2}}$$

    - c is the number of concordants
    - d is the number of disconcordants

Ranking Accuracy: Example

- Consider a set of four items $I = \{i_1, i_2, i_3, i_4\}$ for which the predicted and true rankings are as follows

	Predicted Rank	*True Rank*
i_1	*1*	*1*
i_2	*2*	*4*
i_3	*3*	*2*
i_4	*4*	*3*

Pair of items and their status {**concordant**/**discordant**} are

(i_1, i_2) : *concordant*

(i_1, i_3) : *concordant*

(i_1, i_4) : *concordant*

(i_2, i_3) : *discordant*

(i_2, i_4) : *discordant*

(i_3, i_4) : *concordant*

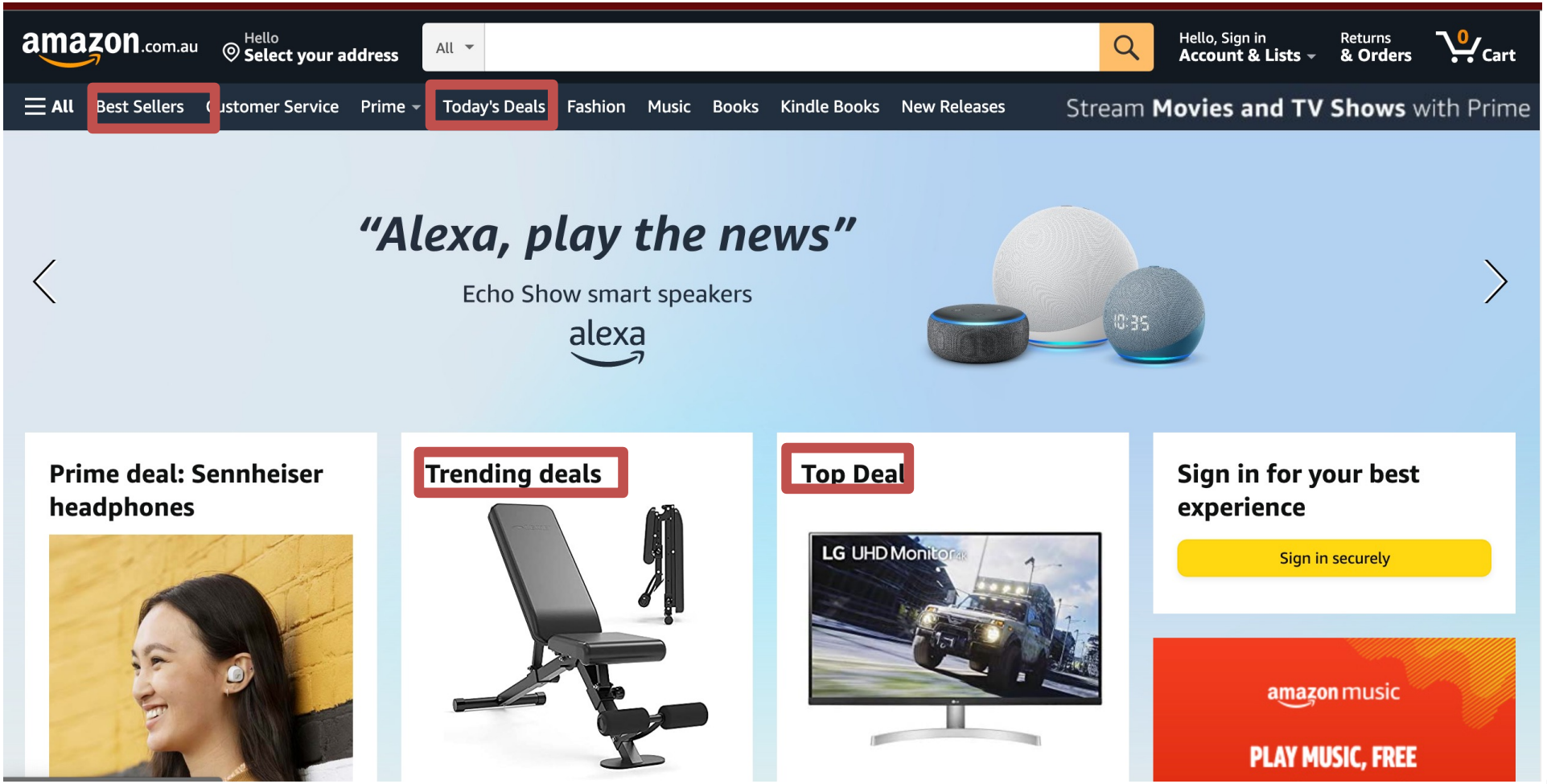
$$\tau = \frac{4-2}{6} = 0.33$$

RecSys – Further Challenges

- Cold-Start Problem
  - Recommender systems use **historical data** or information provided by the user to recommend items, products, etc.
  - When user join sites, they still haven't bought any product, or they have no history.
  - It is hard to infer what they are going to like when they start on a site.
- Data Sparsity
  - When historical or prior information is **insufficient**.
  - Unlike the cold start problem, this is in the system as a whole and is not specific to an individual.
- Attacks:
  - **Push Attack**: pushing ratings up by making fake users
  - **Nuke attack**: DDoS attacks, stop the whole recommendation systems
- Explanation
  - Recommender systems often recommend items with **no explanation** on why these items are recommended

Cold-Start Solutions

- Use Popular Products
- Use Cookies …



Summary

I. Why Personalization?

II. Recommender Systems

1. Content-based Recommendation
2. Rating-based Recommendation (Collaborative Filtering)
3. Hybrid Recommendation
4. Clustering-based Recommendation

III. Performance measures

References

[1] https://en.wikipedia.org/wiki/Recommender_system
[2] https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems)