COMP7630 – Web Intelligence and its Applications

Recommender Systems

Valentino Santucci

(valentino.santucci@unistrapg.it)

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - Collaborative Filtering
    - Memory-based CF
    - Model-based CF
- Evaluation Metrics

Information Overflow!!!

Information Overflow!!!

- *Products* to buy
- *Holidays* to spend
- *Movies* to watch
- *People* to follow
- *News articles* to read

- Psychological considerations of different stakeholders
  - Buyer: want to figure out what they need
  - Seller: want to make you buy what they desire

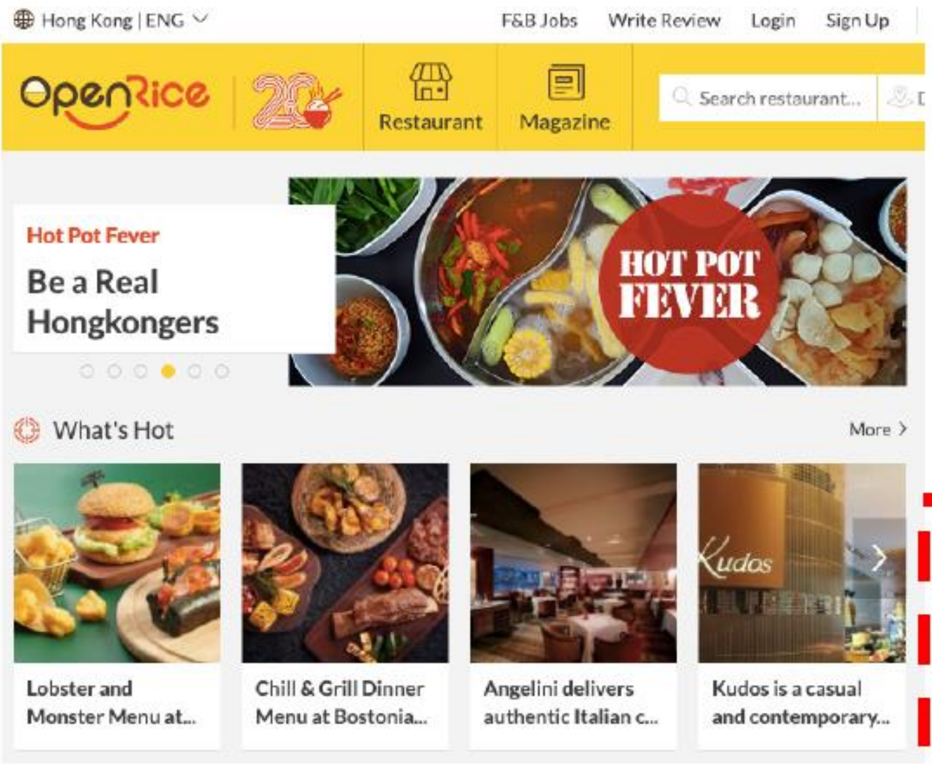
- Further complication: context/situation dependent …
  - where you are, purpose behind, individual/group, …

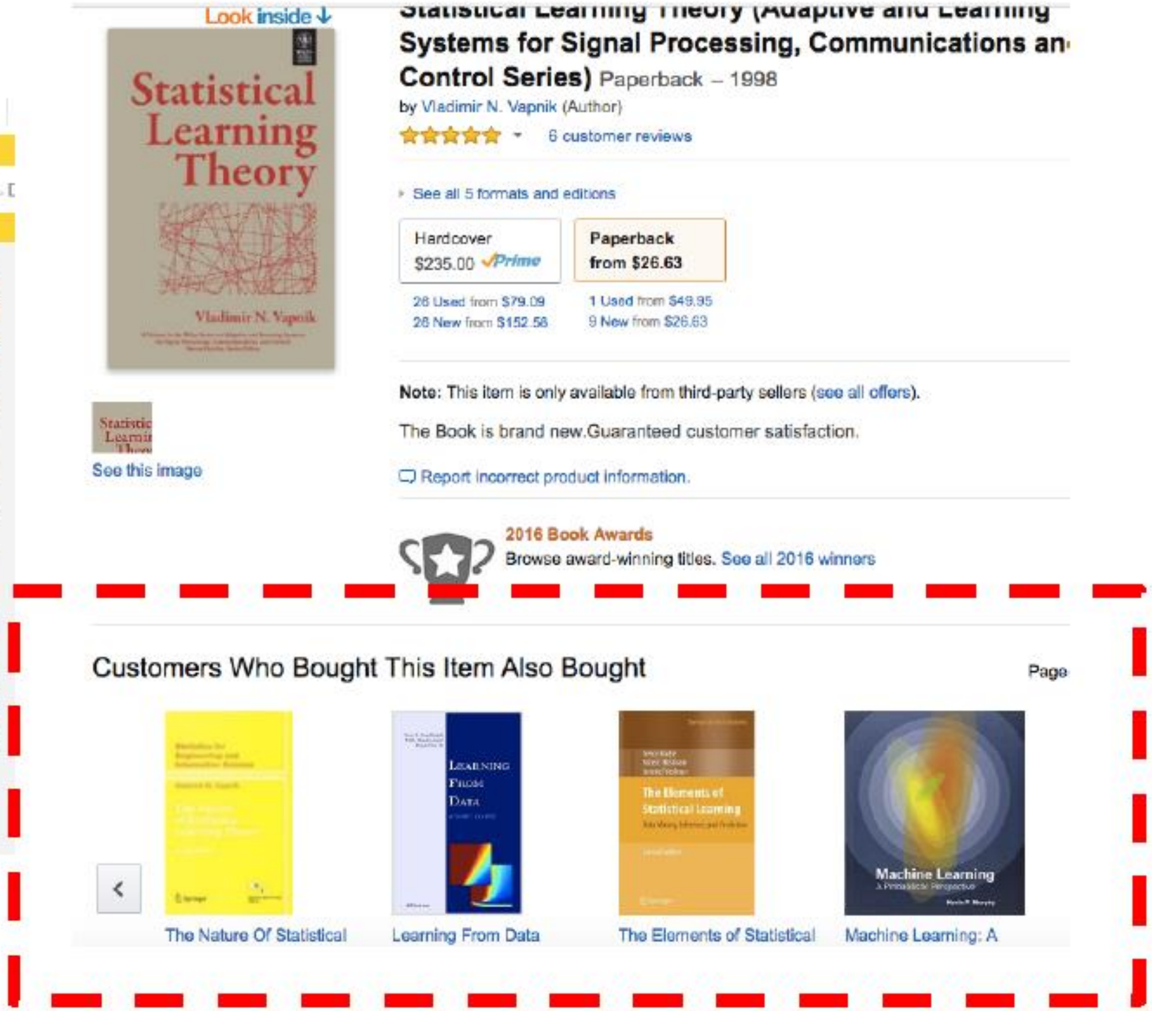
From Pull to Push

- Information Retrieval (Pull Information)
  - Query -> Matched Results -> Manual Filtering

- Recommender Systems (Push Information)
  - Potential Requirements -> Machine Filtering -> Recommendation

Recommender Systems





Problem Space of RS

RS: Assumptions behind

- Users' preferences remain stable for some time, and yet may change smoothly over time

- Steps:
  - Observe the past users' or groups' preferences,
  - Predict their future interests
  - Recommend specific items of interest

- Also called customization, personalization, targeted advertisement, …

Formalization of Recommendation Problem

- Formally, a recommender system takes a set of users U and a set of items I and learns a function f such that:

$$f : U \times I \to \mathbb{R}$$

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - Collaborative Filtering
    - Memory-based CF
    - Model-based CF
- Evaluation Metrics

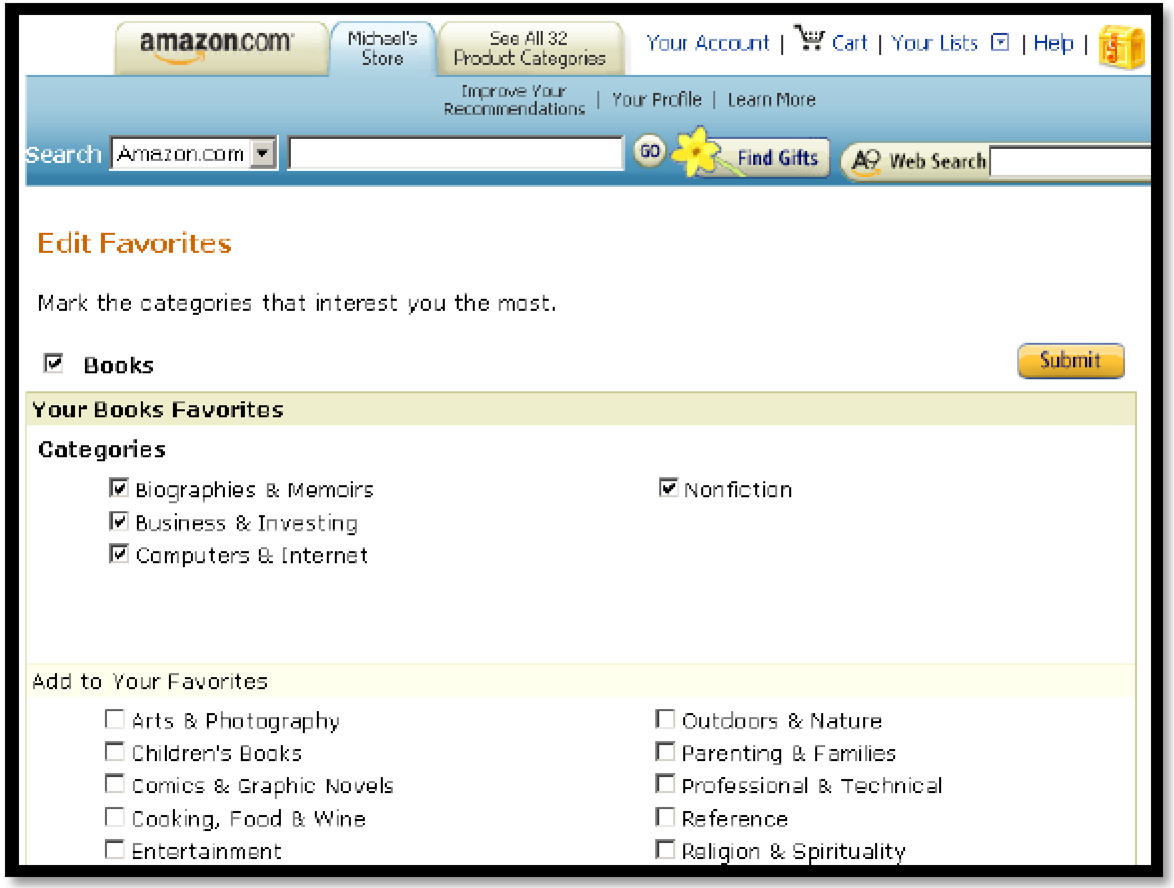
Content-based RS

- Compute content features for all the items based on their description (how?)
- Compute a user profile that characterizes the types of items the user likes (how?).
- Compare items with the user profile to determine what to recommend (how?).

The user interest profiles are often created explicitly via online questionnaires of different formats (your design).

Content-based RS

User Profile Acquisition



Items Recommended

Content-based RS – TFIDF again!

- We represent user profiles and item descriptions by vectorizing them using a set of k keywords
- Vectorize (e.g., using **TF-IDF**) both users and items and compute their similarity

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

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

- Recommend the top-most similar items to the user

Content-based RS algorithm

Algorithm 9.1 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.

return r recommended items.

$U_i = (u_1, u_2, \ldots, u_k)$ = user i's profile vector;

$\{I_j\}_{j=1}^{n} = \{(i_{j,1}, i_{j,2}, \ldots, i_{j,k})$ = item j's description vector$\}_{j=1}^{n}$;

$s_{i,j} = sim(U_i, I_j), 1 \leq j \leq n$;

Return top r items with maximum similarity $s_{i,j}$.

Limitations of Content-based algorithm

- Similar items could be described differently
  - Synonymy (“happy” / “joyful”, “love” / “passion”, …)
  - Polysemy (“bright students” / “bright light bulbs”, “feet of a body” / “5 feet long”, …)
- Content-based recommendation systems make recommendations to users based on their past behavior or preferences
- Tend to suggest items that are similar to what the user has already consumed, which can lead to a lack of variety and discovery
- Rely on static user profiles that do not update frequently. Thus, they may not be able to adapt to changes in the user's preferences over time.
- …
- What if user's profile is not available!?

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - <u>Collaborative Filtering</u>
    - Memory-based CF
    - Model-based CF
- Evaluation Metrics

Collaborative Filtering

- Automating the word-of-mouth process

"You can trust me on this, because I heard it from a friend of a friend of a Facebook friend."

Collaborative Filtering (CF)

- Filtering information using techniques involving collaborative consideration of multiple viewpoints, multiple data sources, etc.

- Based on **users' ratings** or purchase records (or related information) [secondary use of the data]

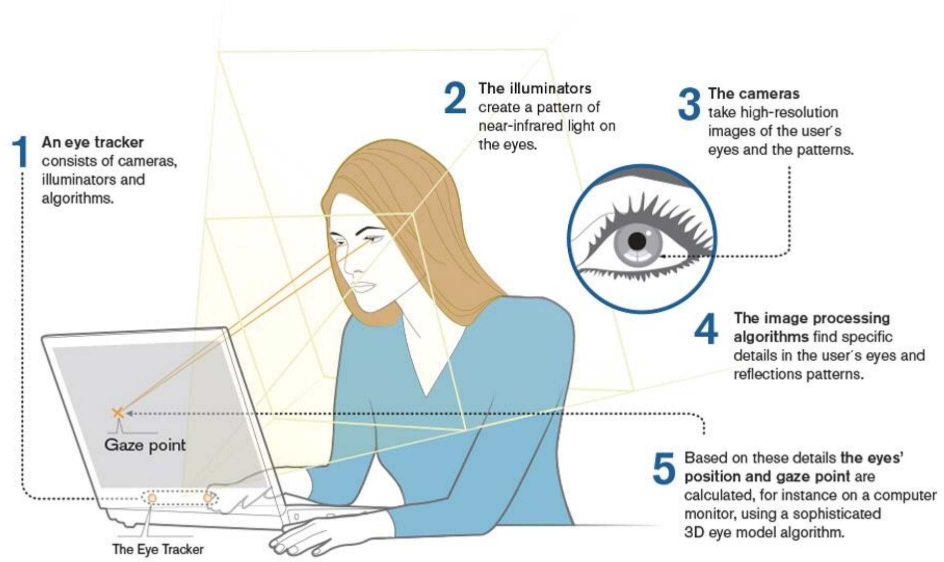
- Not required to have textual descriptions about the users or content of the items (though hybrid approaches are possible)

How to obtain users' ratings?

- Explicit ratings:
  - entered by a user directly
  - i.e., “Please rate this on a scale of 1-5”

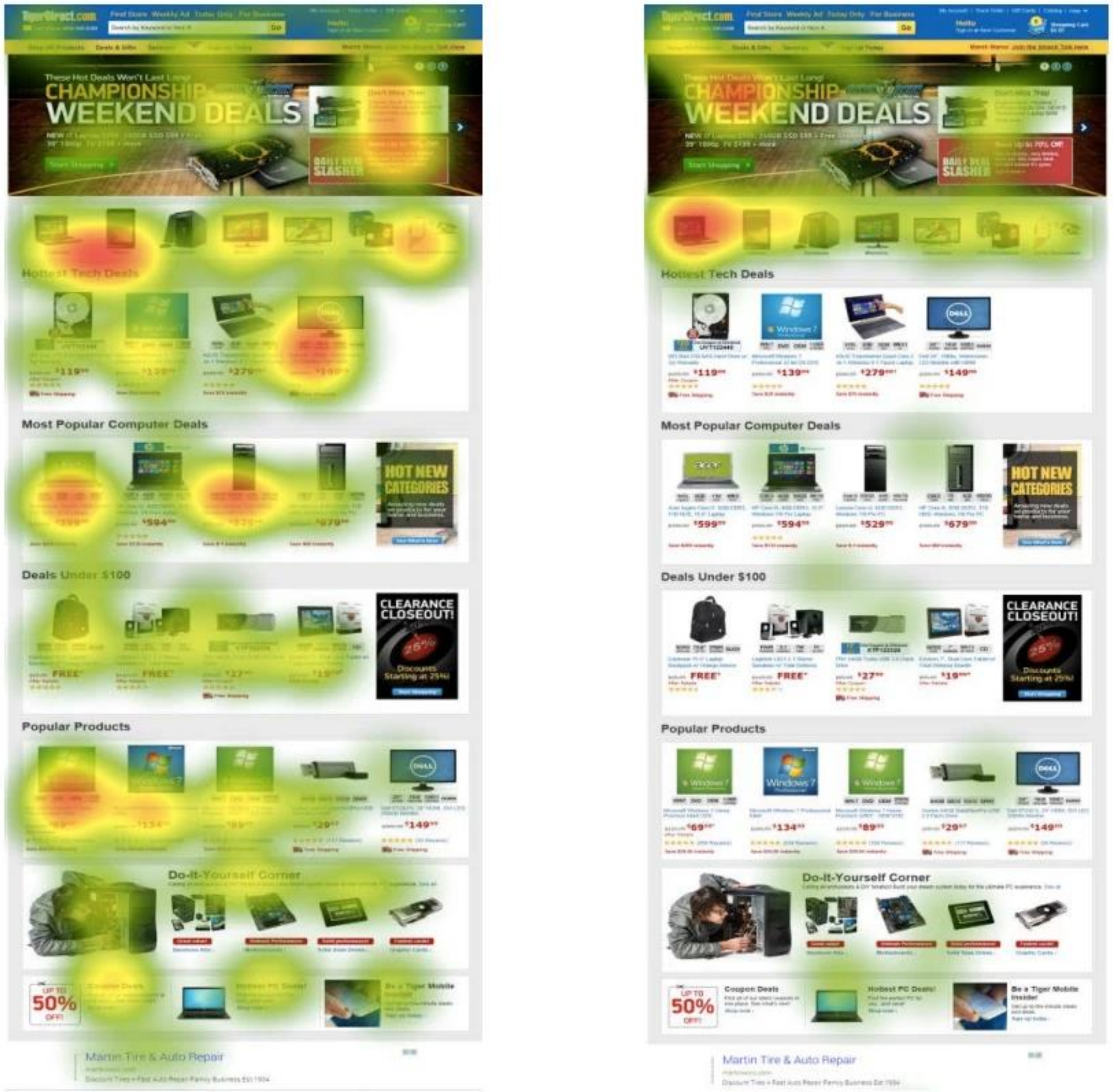
- Implicit ratings:
  - Inferred from (online) user behavior
  - E.g., songs played for the past few months
  - E.g., amount of time spent on different webpages (or even different parts of a web page!)

Eye Tracking



https://connect.tobii.com/s/article/How-do-Tobii-eye-trackers-work?language=en_US

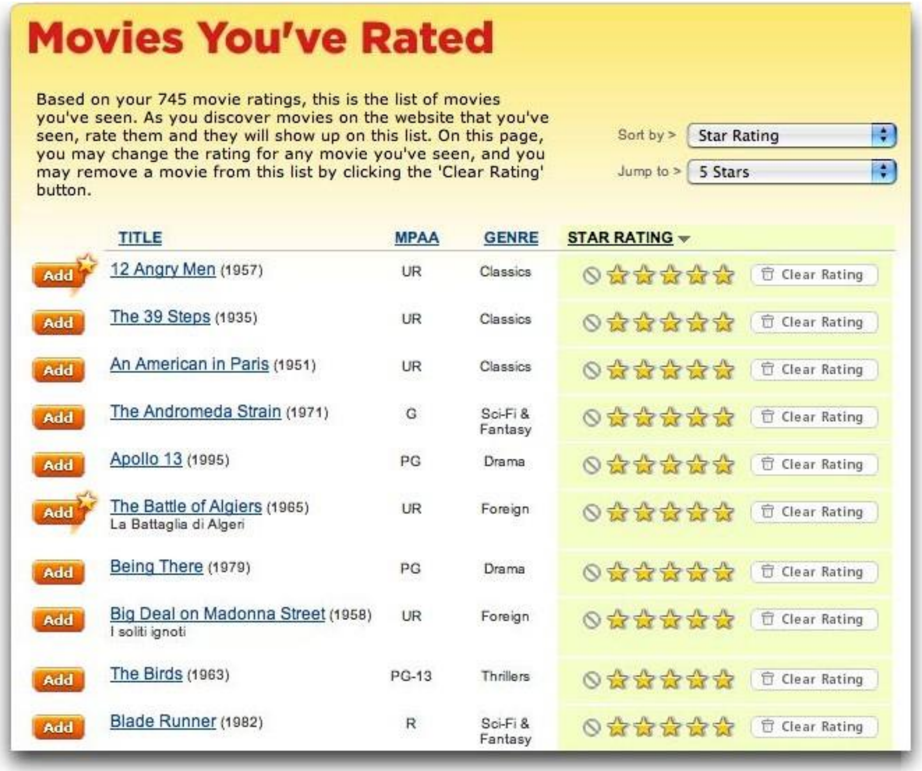
https://www.shopify.com/enterprise/ecommerce-eye-tracking



a. Baby Boomer

b. Generation Y

Rating Matrix



Value	Graphic representation	Textual representation
5	☆ ☆ ☆ ☆ ☆	Excellent
4	☆ ☆ ☆ ☆	Very good
3	☆ ☆ ☆	Good
2	☆ ☆	Fair
1	☆	Poor

Table 9.1: User-Item Matrix

	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

Rating Matrix

- Rating matrix contains several unknown entries… why?

Table 9.1: User-Item Matrix

	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

- In CF, one aims to predict the missing ratings and possibly recommend the item with the highest predicted rating to the user

Memory-based CF vs Model-based CF

- **Memory-based**: Recommendation is directly based on previous ratings in the stored matrix that describes user-item relations
- **Model-based**: Assumes that an underlying model (hypothesis) governs how users rate items.
  - This model can be approximated and learned.
  - The model is then used to recommend ratings.
  - **Example**: users rate low budget movies poorly

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - Collaborative Filtering
    - <u>Memory-based CF</u>
    - Model-based CF
- Evaluation Metrics

Memory-based CF

Two memory-based methods:

User-based CF

Users with similar **previous** ratings for items are likely to rate future items similarly

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

Item-based CF

Items that have received similar ratings **previously** from users are likely to receive similar ratings from future users

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

Rating Prediction in User-based CF

Observed rating of user v for item i

User v's mean rating

User u's mean rating

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in N(u)} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in N(u)} sim(u,v)}$$

Predicted rating of user u for item i

Neighborhood of user u

Similarity between Users

Cosine Similarity

$$sim(U_u, U_v) = cos(U_u, U_v) = \frac{U_u \cdot U_v}{||U_u||\ ||U_v||} = \frac{\sum_i r_{u,i} r_{v,i}}{\sqrt{\sum_i {r_{u,i}}^2}\sqrt{\sum_i {r_{v,i}}^2}}.$$

Pearson Correlation Coefficient

$$sim(U_u, U_v) = \frac{\sum_i (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_i (r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_i (r_{v,i} - \bar{r}_v)^2}}$$

User-based CF

1. Weigh all users with respect to their similarity with the current user
2. Select a subset of the users (neighbors) as recommenders
3. Predict the user rating for a specific item using neighbors’ ratings for the same item
4. Recommend items with the highest predicted ranks

User-based CF: an example (1/2)

[Cosine Similarity is used as similarity]

	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

Predict Jane’s rating for Aladdin

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: an example (2/2)

[Cosine Similarity is used as similarity]

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

$$
\begin{aligned}
r_{Jane,Aladdin} &= \bar{r}_{Jane} + \frac{sim(Jane, Joe)(r_{Joe,Aladdin} - \bar{r}_{Joe})}{sim(Jane, Joe) + sim(Jane, Jorge)} \\
&\quad + \frac{sim(Jane, Jorge)(r_{Jorge,Aladdin} - \bar{r}_{Jorge})}{sim(Jane, Joe) + sim(Jane, Jorge)} \\
&= 1.33 + \frac{0.88(4 - 2.75) + 0.84(2 - 1.25)}{0.88 + 0.84} = 2.33
\end{aligned}
$$

Item-based CF

- Calculate the similarity between items and then predict new items based on the past ratings for similar items

$$r_{u,i} = \bar{r}_i + \frac{\sum_{j \in N(i)} sim(i,j)(r_{u,j} - \bar{r}_j)}{\sum_{j \in N(i)} sim(i,j)}$$

Item i's mean rating

i and j are two items

Item-based CF: an example
[Cosine Similarity is used as similarity]

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)

$$\begin{aligned} r_{Jane,Aladdin} &= \bar{r}_{Aladdin} + \frac{sim(Aladdin, Lion\,King)(r_{Jane,Lion\,King} - \bar{r}_{Lion\,King})}{sim(Aladdin, Lion\,King) + sim(Aladdin, Anastasia)} \\ &\quad + \frac{sim(Aladdin, Anastasia)(r_{Jane,Anastasia} - \bar{r}_{Anastasia})}{sim(Aladdin, Lion\,King) + sim(Aladdin, Anastasia)} \\ &= 2 + \frac{0.84(3-2.8) + 0.67(0-1.6)}{0.84+0.67} = 1.40 \end{aligned}$$

User-based vs Item-based CF

- User-based collaborative filtering is more effective when new items are added to the system as it relies on user behavior and preferences rather than item characteristics. Item-based collaborative filtering requires historical data on items to establish relationships, and hence, it may not be effective when new items are added.

- For e-commerce, user-based CF sometimes is less preferred than the item-based CF
  - Consider that the item set changes less often than users
  - With a large number of users, even the smallest change in the user data is likely to reset the entire group of similar users

Recommendation + Social Network

- Instead of determining "neighbors" based on ratings …
- We can limit the set of individuals that can contribute to the ratings of a user to the set of friends of the user

- Let $S(i)$ be the set of the k most similar friends of an individual

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in S(u)} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in S(u)} sim(u,v)}$$

Example of User-based CF in a Social Network

Adjacency matrix of the social network

$$A = \begin{array}{c} \\ John \\ Joe \\ Jill \\ Jane \\ Jorge \end{array} \begin{bmatrix} John & Joe & Jill & Jane & Jorge \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

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

Neighborhood size k = 2

Average Ratings

$$\bar{r}_{John} = \frac{4+3+2+2}{4} = 2.75.$$
$$\bar{r}_{Joe} = \frac{5+2+1+5}{4} = 3.25.$$
$$\bar{r}_{Jill} = \frac{2+5+0}{3} = 2.33.$$
$$\bar{r}_{Jane} = \frac{1+3+4+3}{4} = 2.75.$$
$$\bar{r}_{Jorge} = \frac{3+1+1+2}{4} = 1.75.$$

User Similarity

$$sim(Jill, John) = \frac{2\times 4+5\times 3+0\times 2}{\sqrt{29}\sqrt{29}} = 0.79$$
$$sim(Jill, Joe) = \frac{2\times 5+5\times 2+0\times 5}{\sqrt{29}\sqrt{54}} = 0.50$$
$$sim(Jill, Jane) = \frac{2\times 1+5\times 3+0\times 3}{\sqrt{29}\sqrt{19}} = 0.72$$
$$sim(Jill, Jorge) = \frac{2\times 3+5\times 1+0\times 2}{\sqrt{29}\sqrt{14}} = 0.54$$

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

$$= 2.33 + \frac{0.72(4-2.75) + 0.54(1-1.75)}{0.72+0.54} = 2.72$$

Combine User-based and Item-based CF

- A simple extension is to form a convex combination of user-based and item-based CF
- Convex combination = Linear combination with non-negative weigths summing to 1
- $r_{u,i} = w^{user} r_{u,i}^{user} + w^{item} r_{u,i}^{item}$
  with $w^{user}, w^{item} \geq 0$ and $w^{user} + w^{item} = 1$

Limitations of CF

- The Cold Start Problem (New Customer)
  - When users first join, they still haven't bought any product, i.e., they have no purchase record.
  - CF cannot be applied.

- Data Sparsity
  - Sometimes historical or prior information is insufficient.
  - Unlike the cold start problem, this is the situation for the system as a whole and is not specific to new customers.

Cold Start Problem

- When users/items first join the system, they do not have any rating
- CF cannot be applied

- Solution: combine content-based and collaborative recommender systems
- A user is described by a vector of features, so it is possible to find the most similar user already in the system, so the new user can "clone" its ratings
- Since also items are described by a vector of features, it is possible to devise the same mechanism for items too

Sparsity Problem

- Sometimes historical or prior information is insufficient
- Lot of missing values
- Simple solutions:
  - Use default rating on the basis of some domain knowledge
  - Replace them with mean of users/items ratings

- But when sparsity is a huge issue, better model-based CF …

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - Collaborative Filtering
    - Memory-based CF
    - <u>Model-based CF</u>
- Evaluation Metrics

Model-based CF

- **In memory-based methods**
  - We predict the missing ratings based on similarities between users or items.

- **In model-based collaborative filtering**
  - We assume that an underlying model governs how users rate.

- We learn that model and use it to predict the missing ratings.
  - Among a variety of model-based techniques, we focus on a well-established model-based technique that is based on singular value decomposition (SVD).

Model-based CF

- Apply SVD (Singular Value Decomposition) to the rating matrix and take the best rank-k approximation X_k of the user-item matrix X.
- Note: both X_k and X have the same $m \times n$ shape, but X_k has rank k and it is dense (much less zeros than X)

SVD of the Sparse rating matrix

$$X = U\Sigma V^T$$

$$X_k = U_k \Sigma_k V_k{}^T$$

Dense transformation of the rating matrix

Users matrix: any row encodes a user in a "denoised" k-dimensional space

Items matrix: any column encodes an item in a "denoised" k-dimensional space

Model-based CF using SVD

- Missing entries before applying Truncated SVD?
    - They can be set to zero, then Truncated SVD will (implicitly) learn a semantically meaningful value for them
    - Or, alternatively, the internal optimization process of Truncated SVD can be adapted to consider only non-missing values
        - Recall: Truncated SVD actually minimizes $||X - X_k||_F$ where X_k has rank k

Example

Table 9.2: An User-Item Matrix

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

$$U = \begin{bmatrix} \boxed{\begin{matrix} -0.4151 & -0.4754 \\ -0.7437 & 0.5278 \\ -0.4110 & -0.6626 \\ -0.3251 & 0.2373 \end{matrix}} & \begin{matrix} -0.7679 & 0.1093 \\ 0.0169 & -0.4099 \\ 0.6207 & -0.0820 \\ 0.1572 & 0.9018 \end{matrix} \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 8.0265 & 0 & 0 \\ 0 & 4.3886 & 0 \\ 0 & 0 & 2.0777 \\ 0 & 0 & 0 \end{bmatrix}$$

$$V^T = \begin{bmatrix} -0.7506 & -0.5540 & -0.3600 \\ 0.2335 & 0.2872 & -0.9290 \\ -0.6181 & 0.7814 & 0.0863 \end{bmatrix}$$

Considering a rank 2 approximation (i.e., k = 2), we truncate all three matrices:

$$U_k = \begin{bmatrix} -0.4151 & -0.4754 \\ -0.7437 & 0.5278 \\ -0.4110 & -0.6626 \\ -0.3251 & 0.2373 \end{bmatrix}$$

$$\Sigma_k = \begin{bmatrix} 8.0265 & 0 \\ 0 & 4.3886 \end{bmatrix}$$

$$V_k^T = \begin{bmatrix} -0.7506 & -0.5540 & -0.3600 \\ 0.2335 & 0.2872 & -0.9290 \end{bmatrix}$$

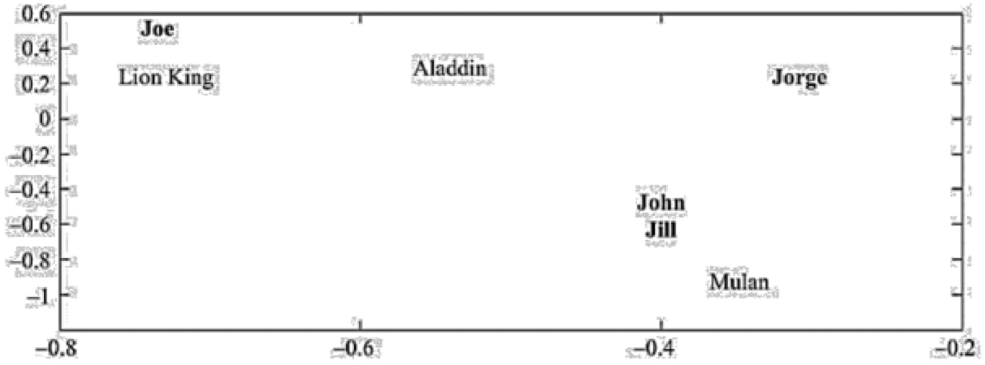


Figure 9.1: Users and Items in the 2-D Space.

Latent Space

- Users and items preferences are projected to a lower dimensional space
- The lower dimensional space is formed by latent/hidden features which captures relevant aspects of the preferences …
- … though they are difficult to interpret, but …

- … the new matrix is denser than before and semantically meaningful!

- Note: SVD and matrix factorization are not the only model-based CF approaches (like Non-Negative Matrix Factorization).

Additional uses for the Latent Space

- Rows of U_k, possibly multiplied by the singular values, are semantic representations of the users
- Rows of V_k (i.e., columns of V_k^T), possibly multiplied by the singular values, are semantic representations of the items

- Cosine similarity can be calculated on these semantic representations in order to gauge the semantic similarity of users or items pairs
- **Clustering algorithms can be executed on those representations in order to cluster together similar users or items**
  - Improve scalability and diversity: consider cluster of users instead of users for providing recommendations
- Also classification algorithms can trained and executed on those semantic representations

Outline

- Need of Recommender Systems
- Recommendation Algorithms
  - Content-based
  - Collaborative Filtering
    - Memory-based CF
    - Model-based CF
- <u>Evaluation Metrics</u>

Accuracy Metrics

- Ultimate goal: which RS approach is better for the recommendation problem at hand?
- Three types of metrics:
  - Rating Value Accuracy: Closeness between RS's predicted ratings to true ratings
  - Classification Accuracy: Ratio with correct vs. incorrect decisions about whether an item is good.
  - Ranking Accuracy: Closeness between RS's predicted ranking to true ranking
- What is required: we have the predictions of our systems, but we need true/actual data to compare with the predictions.
  - For example, true data may be acquired by questionnaires, surveys, eye-trackers, …

Rating Value 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

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

Measuring Error Rate (Example)

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

$$
\begin{aligned}
MAE &= \frac{|1-3|+|2-5|+|3-3|+|4-2|+|4-1|}{5} = 2 \\
NMAE &= \frac{MAE}{5-1} = 0.5 \\
RMSE &= \sqrt{\frac{(1-3)^2+(2-5)^2+(3-3)^2+(4-2)^2+(4-1)^2}{5}} \\
&= 2.28
\end{aligned}
$$

Classification Accuracy

Confusion Matrix

	Selected	Not Selected	Total
Relevant	N_{rs}	N_{rn}	N_r
Irrelevant	N_{is}	N_{in}	N_i
Total	N_s	N_n	N

Precision: a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved

$$P = \frac{N_{rs}}{N_s}$$

Recall: a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items

$$R = \frac{N_{rs}}{N_r}$$

F-measure

- Precision and Recall evaluates different aspects, but may be synthesized by averaging them using harmonic mean

- F-measure is the harmonic mean of precision and recall

$$F = \frac{2PR}{P + R}.$$

Precision and Recall (Example)

	Selected	*Not Selected*	*Total*
Relevant	*9*	*15*	*24*
Irrelevant	*3*	*13*	*16*
Total	*12*	*28*	*40*

$$
\begin{aligned}
P &= \frac{9}{12} = 0.75 \\
R &= \frac{9}{24} = 0.375 \\
F &= \frac{2 \times 0.75 \times 0.375}{0.75 + 0.375} = 0.5
\end{aligned}
$$

Ranking Accuracy

- Spearman's Rank Correlation
  - Pearson correlation coefficient between two rankings $\boldsymbol{x}$ & $\boldsymbol{y}$

$$\rho = 1 - \frac{6\sum_{i=1}^{n}(x_i - y_i)^2}{n^3 - n}$$

- Kendall's Tau (τ)
  - Compare recommended ranking and ground truth ranking
  - *Concordant*: An item pair with its ordering preserved in the recommended ranking. Otherwise, it is *disconcordant*.

$$\tau = \frac{c-d}{\binom{n}{2}} = \frac{c-d}{n(n-1)/2}$$

  - 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

$$\begin{aligned} (i_1, i_2) &: \textit{concordant} \\ (i_1, i_3) &: \textit{concordant} \\ (i_1, i_4) &: \textit{concordant} \\ (i_2, i_3) &: \textit{discordant} \\ (i_2, i_4) &: \textit{discordant} \\ (i_3, i_4) &: \textit{concordant} \end{aligned}$$

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

Ties in Kendall's-tau

- A pair $(x_i, x_j), (y_i, y_j)$ is said to be *tied* if $x_i = x_j$ or $y_i = y_j$
- A tied pair is neither concordant nor discordant
- So Kendall's-tau formula is unchanged if there are ties

References

- R. Zafarani, M. A. Abbasi, and H. Liu, Social Media Mining: An Introduction, Cambridge University Press, 2014 [Chapter 9]. URL: http://socialmediamining.info/