Social Media Mining

Recommendation in Social Media
Difficulties of Making Decision

- Which digital camera should I buy?
- Where should I spend my holiday?
- Which movie should I rent?
- Whom should I follow?
- Where should I find interesting news article?
- Which movie is the best for our family?

- If interested, see two recent conference tutorials
  – SIGKDD2014, Recommendation in Social Media
  – RecSys2014, Personalized Location Recommendation
When Does This Problem Occur?

• There are many choices
• There are no obvious advantages among them
• We do not have enough resources to check all options (information overload)
• We do not have enough knowledge and experience to choose, or
  – I’m lazy, but don’t want to miss out on good stuff
  – Defensive decision making

Goal of Recommendation:
To come up with a short list of items that fits user’s interests
Common Solutions to the Problem

• Consulting friends
• Obtaining information from a trusted third party
• Hiring a team of experts
• Search the Internet
• Following the crowd (pick the item from top-n lists, e.g., best sellers on Amazon)

• **Can we automate all of the above?**
  – Using a recommender algorithm (a.k.a. a recommender system)
Recommender Systems - Examples

Book recommendation in Amazon

Product Recommendation in eBay

Video clip recommendation in YouTube

Restaurant Recommendation in Yelp

Social Media Mining
Main Idea behind Recommender Systems

Use historical data such as the user’s past preferences or similar users’ past preferences to predict future likes

• Users’ preferences are likely to remain stable, and change smoothly over time.
  – By watching the past users’ or groups’ preferences, we try to predict their future likes
  – Then we can recommend items of interest to them

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

$$f : U \times I \rightarrow \mathbb{R}$$
Recommendation vs. Search

• One way to get answers is using search engines
• Search engines find results that match the query provided by the user
• The results are generally provided as a list ordered with respect to the relevance of the item to the given query
• Consider the query “best 2014 movie to watch”
  – The same results for an 8 year old and an adult

Search engines’ results are not customized
Challenges (I)

• The Cold Start Problem
  – Many recommendation systems use historical data or information provided by the user to recommend items, products, etc. However, when individuals join sites, they still haven’t bought any product, or they have no history. This makes it hard to infer what they are going to like when they start on a site. The problem is referred to as the cold-start problem.

• Data Sparsity
  – Similar to the cold-start problem, data sparsity is when not enough historical or prior information is available. Unlike the cold start problem, this is in the system as a whole and is not specific to an individual.
Challenges (II)

• Attacks
  – Push Attack: pushing the ratings up by creating fake users
  – Nuke attack: DDoS attacks, stop the whole recommendation systems

• Privacy
  – Employing user’s private information to recommend to others.

• Explanation
  – Recommendation systems often recommend items without any explanation of why recommending them
Classical Recommendation Algorithms

- Content-based algorithms
- Collaborative filtering
Content-Based Methods

• Content-based recommendation systems are based on the fact that a user’s interest should match the description of the items that she should be recommended by the system.
• In other words, 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.

Finding 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

## Book Knowledge Base

<table>
<thead>
<tr>
<th>Title</th>
<th>Genre</th>
<th>Author</th>
<th>Type</th>
<th>Price</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>The Lace Reader</em></td>
<td>Fiction, Mystery</td>
<td>Brunonia Barry</td>
<td>Hardcover</td>
<td>49.90</td>
<td>American contemporary fiction, detective, historical</td>
</tr>
<tr>
<td><em>Into the Fire</em></td>
<td>Romance, Suspense</td>
<td>Suzanne Brockmann</td>
<td>Hardcover</td>
<td>45.90</td>
<td>American fiction, murder, neo-Nazism</td>
</tr>
</tbody>
</table>

## User Profile

<table>
<thead>
<tr>
<th>Title</th>
<th>Genre</th>
<th>Author</th>
<th>Type</th>
<th>Price</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>Fiction, Suspense</td>
<td>Brunonia Barry, Ken Follett</td>
<td>Paperback</td>
<td>25.65</td>
<td>detective, murder, New York</td>
</tr>
</tbody>
</table>
1. **Describe the items to be recommended**

2. **Create a profile of the user that describes the types of items the user likes**

3. **Compare items with the user profile to determine what to recommend**

The profile is often created, and updated automatically in response to feedback on the desirability of items that are presented to the user.
Content-based Recommendation: Example

Edit Favorites

Mark the categories that interest you the most:

- Books

Your Books Favorites

Categories

- Biographies & Memoirs
- Nonfiction
- Business & Investing
- Computers & Internet
- More Categories

Add to Your Favorites

- Arts & Photography
- Children's Books
- Comics & Graphic Novels
- Cooking, Food & Wine
- Entertainment
- Health, Mind & Body
- History
- Home & Garden
- Religion & Spirituality

Recommended For You > Books

Recommended by Category
In Books

Your Favorites Edit Business & Investing
Computers & Internet
Biographies & Memoirs
Nonfiction
More Categories
Arts & Photography
Children's Books
Comics & Graphic Novels
Cooking, Food & Wine
Entertainment
Gay & Lesbian
Health, Mind & Body
History
Home & Garden
...
Content-Based Methods

• To formalize a content-based method, we first represent both user profiles and item descriptions by vectorizing them using a set of k keywords

• We can vectorize (e.g., using TF-IDF) both users and items and compute their similarity

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

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

• We can recommend the top most similar items to the user
Content-Based Recommendation 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.

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;
4: $s_{i,j} = sim(U_i, I_j)$;
5: Return top $r$ items with maximum similarity $s_{i,j}$.

- In content-based recommendation, we compute the topmost similar items to a user $j$ and then recommend these items in the order of similarity.
Collaborative Filtering

• Collaborative filtering is the process of selecting information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc.

• The main advantage of this method is that the recommender system does not 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.
Rating Matrix: An Example

Movies You’ve Rated

Based on your 745 movie ratings, this is the list of movies you’ve seen. As you discover movies on the website that you’ve seen, rate them and they will show up on this list. On this page, you may change the rating for any movie you’ve seen, and you may remove a movie from this list by clicking the ‘Clear Rating’ button.

<table>
<thead>
<tr>
<th>Title</th>
<th>MPAA</th>
<th>Genre</th>
<th>Star Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 Angry Men (1957)</td>
<td>UR</td>
<td>Classic</td>
<td>[5 Stars]</td>
</tr>
<tr>
<td>The 39 Steps (1935)</td>
<td>UR</td>
<td>Classic</td>
<td>[4 Stars]</td>
</tr>
<tr>
<td>The Andromeda Strain (1971)</td>
<td>G</td>
<td>Sci-Fi &amp; Fantasy</td>
<td>[3 Stars]</td>
</tr>
<tr>
<td>Apollo 13 (1996)</td>
<td>PG</td>
<td>Drama</td>
<td>[2 Stars]</td>
</tr>
<tr>
<td>The Battle of Algiers (1966)</td>
<td>PG</td>
<td>Drama</td>
<td>[1 Star]</td>
</tr>
<tr>
<td>Big Deal on Madonna Street (1969)</td>
<td>PG</td>
<td>Foreign</td>
<td>[5 Stars]</td>
</tr>
<tr>
<td>Being There (1979)</td>
<td>PG</td>
<td>Drama</td>
<td>[4 Stars]</td>
</tr>
<tr>
<td>Big Bad (1979)</td>
<td>PG</td>
<td>Foreign</td>
<td>[3 Stars]</td>
</tr>
</tbody>
</table>

Table 9.1: User-Item Matrix

<table>
<thead>
<tr>
<th></th>
<th>Lion King</th>
<th>Aladdin</th>
<th>Mulan</th>
<th>Anastasia</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Joe</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Jill</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Jane</td>
<td>3</td>
<td>?</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Jorge</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Social Media Mining
Rating Matrix

Users rate (rank) the items (purchased, watched)

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

- **Implicit ratings**: inferred from other user behavior
  - Play lists or music listened to, for a music Rec system
  - The amount of time users spent on a webpage
Collaborative Filtering

Types of Collaborative Filtering Algorithms:

• **Memory-based**: Recommendation is directly based on previous ratings in the stored matrix that describes user-item relations.

• **Model-based**: Alternatively, one can assume that an underlying model (hypothesis) governs the way users rate items. This model can be approximated and learned. The model is then used to recommend ratings. A model, for example, is that users rate low budget movies poorly.
The most important assumptions of collaborative filtering are:

- **User-based CF**
  Users with similar *previous* ratings for items are likely to rate future items similarly.

- **Item-based CF**
  Items that have received similar *previously* ratings from users are likely to receive similar ratings from future users (*item-based CF*)
Collaborative Filtering: Algorithm

1. Weigh all users/items with respect to their similarity with the current user/item
2. Select a subset of the users/items (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
• A and B are following C, and B is following D
  – A might be interested in following D
Measure Similarity between Users (or Items)

- Cosine Similarity

\[
sim(U_i, U_j) = \cos(U_i, U_j) = \frac{U_i \cdot U_j}{||U_i|| \ ||U_j||} = \frac{\sum_k r_{ik}r_{jk}}{\sqrt{\sum_k r_{ik}^2} \ \sqrt{\sum_k r_{jk}^2}}.\]

- Pearson Correlation Coefficient

\[
sim(U_i, U_j) = \frac{\sum_k (r_{ik} - \bar{r}_i)(r_{jk} - \bar{r}_j)}{\sqrt{\sum_k (r_{ik} - \bar{r}_i)^2} \ \sqrt{\sum (r_{jk} - \bar{r}_j)^2}}.\]
User-based Collaborative Filtering

• In user-based collaborative filtering, the system finds the most similar user (users) to the current user and uses their preferences for recommendation.

• The user-based approach is not as popular as the item-based approach.
  – For a system that handles a large user base, even the smallest change in the user data is likely to reset the entire group of similar users.
User-based CF

Updating the ratings:

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

- Predicted rating of user \( u \) for item \( i \)
- User \( u \)‘s mean rating
- User \( v \)‘s mean rating
- Observed rating of user \( v \) for item \( i \)
User-based CF, Example

<table>
<thead>
<tr>
<th></th>
<th>Lion King</th>
<th>Aladdin</th>
<th>Mulan</th>
<th>Anastasia</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Joe</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Jill</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Jane</td>
<td>3</td>
<td>?</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Jorge</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

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

\[
sim(Jane, John) = \frac{3 \times 3 + 1 \times 3 + 0 \times 3}{\sqrt{10} \sqrt{27}} = 0.73
\]

\[
sim(Jane, Joe) = \frac{3 \times 5 + 1 \times 0 + 0 \times 2}{\sqrt{10} \sqrt{29}} = 0.88
\]

\[
sim(Jane, Jill) = \frac{3 \times 1 + 1 \times 4 + 0 \times 2}{\sqrt{10} \sqrt{21}} = 0.48
\]

\[
sim(Jane, Jorge) = \frac{3 \times 2 + 1 \times 0 + 0 \times 1}{\sqrt{10} \sqrt{5}} = 0.84
\]
3- Calculate Jane’s rating for Aladdin, 
Assume that neighborhood size = 2

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

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

\[ = 1.33 + \frac{0.88(4 - 2.75) + 0.84(2 - 1.25)}{0.88 + 0.84} = 2.33 \]
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, Example

1- Calculate average ratings

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

\[
\bar{r}_{\text{Aladdin}} = \frac{0 + 4 + 2 + 2}{4} = 2.
\]

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

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

2- Calculate item-item similarity

\[
sim(\text{Aladdin}, \text{Lion King}) = \frac{0 \times 3 + 4 \times 5 + 2 \times 1 + 2 \times 2}{\sqrt{24} \times \sqrt{39}} = 0.84
\]

\[
sim(\text{Aladdin}, \text{Mulan}) = \frac{0 \times 3 + 4 \times 0 + 2 \times 4 + 2 \times 0}{\sqrt{24} \times \sqrt{25}} = 0.32
\]

\[
sim(\text{Aladdin}, \text{Anastasia}) = \frac{0 \times 3 + 4 \times 2 + 2 \times 2 + 2 \times 1}{\sqrt{24} \times \sqrt{18}} = 0.67
\]

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

\[
\hat{r}_{\text{Jane, Aladdin}} = \bar{r}_{\text{Aladdin}} + \frac{\text{sim}(\text{Aladdin}, \text{Lion King})(\hat{r}_{\text{Jane, Lion King}} - \bar{r}_{\text{Lion King}})}{\text{sim}(\text{Aladdin}, \text{Lion King}) + \text{sim}(\text{Aladdin, Anastasia})} + \frac{\text{sim}(\text{Aladdin, Anastasia})(\hat{r}_{\text{Jane, Anastasia}} - \bar{r}_{\text{Anastasia}})}{\text{sim}(\text{Aladdin, Lion King}) + \text{sim}(\text{Aladdin, Anastasia})}
\]

\[
= 2 + \frac{0.84(3 - 2.8) + 0.67(0 - 1.6)}{0.84 + 0.67} = 1.40
\]
Model-Based Collaborative Filtering

• **In memory-based methods** (either item-based or user-based), we aim to predict the missing ratings based on similarities between users or items.

• **In model-based collaborative filtering**, we assume that an underlying model governs the way users rate.

• We aim to learn that model and then use that model 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).
Singular Value Decomposition (SVD)

• SVD is a linear algebra technique that, given a real matrix $X \in \mathbb{R}^{m \times n}$, $m \geq n$, and factorizes it into three matrices,

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

where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ are orthogonal matrices and $\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix.

• The product of these matrices is equivalent to the original matrix; therefore, no information is lost.
Low-rank Matrix Approximation

- A Low-rank matrix approximation of matrix $X$ is another matrix $C \in \mathbb{R}^{m \times n}$.
- $C$ approximates $X$, and $C$’s rank (the maximum number of linearly independent columns) is a fixed number $k << \min(m, n)$.
  \[
  \text{Rank}(C) = k
  \]
- The best low-rank matrix approximation is a matrix $C$ that minimizes $||X - C||_F$.
  \[
  ||X||_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} X_{ij}^2}
  \]
  - Low-rank approximations of matrices remove noise by assuming that the matrix is not generated at random and has an underlying structure.
  - SVD can help remove noise by computing a low-rank approximation of a matrix.
Low-rank Matrix Approximation with SVD

1. Create $\Sigma_k$ from $\Sigma$ by keeping only the first $k$ elements on the diagonal. This way, $\Sigma_k \in \mathbb{R}^{k \times k}$.

2. Keep only the first $k$ columns of $U$ and denote it as $U_k \in \mathbb{R}^{m \times k}$, and keep only the first $k$ rows of $V^T$ and denote it as $V_k^T \in \mathbb{R}^{k \times n}$.

3. Let $X_k = U_k \Sigma_k V_k^T$, $X_k \in \mathbb{R}^{m \times n}$.

$X_k$ is the best low-rank approximation of a matrix $X$

Theorem 9.1 (Eckart-Young-Mirsky Low-Rank Matrix Approximation). Let $X$ be a matrix and $C$ be the best low-rank approximation of $X$; if $\|X - C\|_F$ is minimized, and $\text{rank}(C) = k$, then $C = X_k$.
Model-based CF, Example

Table 9.2: An User-Item Matrix

<table>
<thead>
<tr>
<th></th>
<th>Lion King</th>
<th>Aladdin</th>
<th>Mulan</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Joe</td>
<td>5</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Jill</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Jorge</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

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

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

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

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

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

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

\[
V^T_k = \begin{bmatrix}
-0.7506 & -0.5540 & -0.3600 \\
0.2335 & 0.2872 & -0.9290
\end{bmatrix}
\]

Figure 9.1: Users and Items in the 2-D Space.
Recommendation to a Group
Recommendation to Groups

• Find content of interest to all members of a group of socially acquainted individuals

• Examples:
  – A movie for friends to watch together
  – A travel destination for a family to spend a holiday break
  – A good restaurant for colleagues to have a working lunch
  – A music to be played in a public area
Tasks of a Group Recommender System

- Acquiring preferences
- Generating recommendations
- Explaining recommendations
- Helping group members to achieve consensus
Aggregation Strategies

- **Maximizing Average Satisfaction**
  - Average everyone’s ratings and choose the max

- **Least Misery**
  - This approach tries to minimize the dissatisfaction among group’s members (Max of the mins of all)

- **Most Pleasure**
  - The maximum of individuals’ maximum ratings is taken as group’s rating
### Recommendation to Group, an Example

**Table 9.3: User-Item Matrix**

<table>
<thead>
<tr>
<th></th>
<th>Soda</th>
<th>Water</th>
<th>Tea</th>
<th>Coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Joe</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Jill</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Jorge</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Juan</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

### Average Satisfaction

- \( R_{Soda} = \frac{1 + 2 + 3}{3} = 2 \)
- \( R_{Water} = \frac{3 + 2 + 3}{3} = 2.66 \)
- \( R_{Tea} = \frac{1 + 4 + 4}{3} = 3 \)
- \( R_{Coffee} = \frac{1 + 2 + 5}{3} = 2.66 \)

### Least Misery

- \( R_{Soda} = \min\{1, 2, 3\} = 1 \)
- \( R_{Water} = \min\{3, 2, 3\} = 2 \)
- \( R_{Tea} = \min\{1, 4, 4\} = 1 \)
- \( R_{Coffee} = \min\{1, 2, 5\} = 1 \)

### Most Pleasure

- \( R_{Soda} = \max\{1, 2, 3\} = 3 \)
- \( R_{Water} = \max\{3, 2, 3\} = 3 \)
- \( R_{Tea} = \max\{1, 4, 4\} = 4 \)
- \( R_{Coffee} = \max\{1, 2, 5\} = 5 \)
Recommendation Using Social Context

- Recommendation using social context alone
- Extending classical methods with social context
- Recommendation constrained by social context
Information Available in Social Context

• In social media, in addition to ratings of products, there is additional information available, such as the friendship network among individuals.

• This information can be used to improve recommendations, based on the assumption that an individual’s friends have an impact on the ratings ascribed to the individual.

• This impact can be due to homophily, influence, or confounding
Recommendation Using Social Context Alone

- Consider a network of friendships for which no user-item rating matrix is provided.

- In this network, we can still recommend users from the network to other users for friendship.

- This is an example of friend recommendation in social networks.
Extending Classical Methods

- Social information can be used in addition to a user-item rating matrix to improve recommendation.
- Addition of social information can be performed by assuming that users that are connected (i.e., friends) have similar tastes in rating items.

\[ R = U^T V \quad R \in \mathbb{R}^{n \times m}, U \in \mathbb{R}^{k \times n}, V \in \mathbb{R}^{k \times m} \]

\[
\min_{U,V} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij}(R_{ij} - U_i^T V_j)^2
\]

\[
\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_1}{2} \|U\|_F^2 + \frac{\lambda_2}{2} \|V\|_F^2
\]
The taste for user i is close to that of all his friends j ∈ \( F(i) \)

\[
\sum_{i=1}^{n} \sum_{j \in F(i)} \text{sim}(i, j) \|U_i - U_j\|_F^2
\]

sim(i, j) denotes the similarity between user i and j (e.g., cosine similarity or Pearson correlation between their ratings) and \( F(i) \) denotes the friends of i

\[
\min_{U,V} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij}(R_{ij} - U_i^T V_j)^2 + \beta \sum_{i=1}^{n} \sum_{j \in F(i)} \text{sim}(i, j) \|U_i - U_j\|_F^2
\]

\[
+ \frac{\lambda_1}{2} \|U\|_F^2 + \frac{\lambda_2}{2} \|V\|_F^2
\]
Recommendation Constrained by Social Context

• In classical recommendation, to estimate ratings of an item, we determine similar users or items. In other words, any user similar to the individual can contribute to the predicted ratings for the individual.

• We can limit the set of individuals that can contribute to the ratings of a user to the set of friends of the user.
  – $S(i)$ is the set of $k$ most similar friends of an individual

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

\[ A = \begin{bmatrix}
  \text{John} & \text{Joe} & \text{Jill} & \text{Jane} & \text{Jorge} \\
  \text{John} & 0 & 1 & 0 & 0 & 1 \\
  \text{Joe} & 1 & 0 & 1 & 0 & 0 \\
  \text{Jill} & 0 & 1 & 0 & 1 & 1 \\
  \text{Jane} & 0 & 0 & 1 & 0 & 0 \\
  \text{Jorge} & 1 & 0 & 1 & 0 & 0 
\end{bmatrix} \]

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

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

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

\[
= 2.33 + \frac{0.72(4 - 2.75) + 0.54(1 - 1.75)}{0.72 + 0.54} = 2.72
\]
Evaluation of Recommender Systems
Evaluating Recommender Systems is difficult

- Different algorithms may be better or worse on different datasets (applications)
  - Many algorithms are designed specifically for datasets where there are many more users than items or vice versa. Similar differences exist for rating density, rating scale, and other properties of datasets

- The goals to perform evaluation may differ
  - Early evaluation work focused specifically on the "accuracy" of algorithms in "predicting" withheld ratings.
  - Other properties different from accuracy also have important effect on user satisfaction and performance

- There is a significant challenge in deciding what combination of measures should be used in comparative evaluation
Evaluating Recommender Systems

• A myriad of algorithms are proposed, but
  – Which one is the best in a given application domain?
  – What are the success factors of different algorithms?
  – Comparative analysis based on an optimality criterion?

• Main questions are:
  – Is a RS efficient with respect to specific criteria like accuracy, user satisfaction, response time, serendipity, online conversion, ....
  – Do customers like/buy recommended items?
  – Do customers buy items they otherwise would have not?
  – Are they satisfied with a recommendation after purchase?
How Do We Evaluate Recommenders

• Application outcomes
  – Add-on sales
  – Click-through rates
  – The number of products purchased (and not returned!)

• Research measures
  – User satisfaction

• Metrics
  – To anticipate the above beforehand (offline)
Accuracy Metrics

- Predictive accuracy
  - Predictive accuracy metrics measure *how close* the recommender system’s predicted ratings are to the true user ratings

- Classification accuracy
  - Classification metrics measure the ratio with which a recommender system makes correct vs. incorrect decisions about whether an item is good.
  - Classification metrics are thus appropriate for tasks such as *Find Good Items* when users have binary preferences.

- Rank accuracy
### Predictive accuracy - Metrics measure error rate

- **Mean Absolute Error (MAE)** measures the average absolute deviation between a predicted rating \( (p) \) and the user’s true rating \( (r) \)
  
  \[
  NMAE = \frac{MAE}{(r_{\text{max}} - r_{\text{min}})}
  \]

- **Root Mean Square Error (RMSE)** is 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

Consider the following table with both the predicted ratings and true ratings of five items.

<table>
<thead>
<tr>
<th>Item</th>
<th>Predicted Rating</th>
<th>True Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

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

\[
NMAE = \frac{MAE}{5 - 1} = 0.5
\]

\[
RMSE = \sqrt{\frac{(1 - 3)^2 + (2 - 5)^2 + (3 - 3)^2 + (4 - 2)^2 + (4 - 1)^2}{5}} = 2.28
\]
Relevance: Precision and Recall

- **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

\[ R = \frac{N_{rs}}{N_r} \]

<table>
<thead>
<tr>
<th></th>
<th>Selected</th>
<th>Not Selected</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant</td>
<td>( N_{rs} )</td>
<td>( N_m )</td>
<td>( N_r )</td>
</tr>
<tr>
<td>Irrelevant</td>
<td>( N_{is} )</td>
<td>( N_{in} )</td>
<td>( N_i )</td>
</tr>
<tr>
<td>Total</td>
<td>( N_s )</td>
<td>( N_n )</td>
<td>( N )</td>
</tr>
</tbody>
</table>
### Evaluating Relevancy, Example

<table>
<thead>
<tr>
<th></th>
<th>Selected</th>
<th>Not Selected</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant</td>
<td>9</td>
<td>15</td>
<td>24</td>
</tr>
<tr>
<td>Irrelevant</td>
<td>3</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>Total</td>
<td>12</td>
<td>28</td>
<td>40</td>
</tr>
</tbody>
</table>

\[
P = \frac{9}{12} = 0.75
\]

\[
R = \frac{9}{24} = 0.375
\]

\[
F = \frac{2 \times 0.75 \times 0.375}{0.75 + 0.375} = 0.5
\]
Evaluating Ranking of Recommendation

- **Spearman’s Rank Correlation**
  \[ \rho = 1 - \frac{6 \sum_{i=1}^{n} (x_i - y_i)^2}{n^3 - n} \]

- **Kendall’s \( \tau \)**
  - It checks the concordant the items of the recommended ranking list against the ground truth ranking list
    - If the two orders are consistent, it is concordant
    - For top 4 items in ranking list, there are 4*3/2=6 pairs
  \[ \tau = \frac{c-d}{\binom{n}{2}} \]
  - where \( c \) is the number of concordants and \( d \) of discordants
Ranking, 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:

<table>
<thead>
<tr>
<th></th>
<th>Predicted Rank</th>
<th>True Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i_1 )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( i_2 )</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>( i_3 )</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>( i_4 )</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

Pair of items and their status {c}oncordant/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

\[
\tau = \frac{4 - 2}{6} = 0.33
\]
Extra Slides
Beyond Accuracy, Relevance, and Rank

• Coverage
  – Measure of the domain of items in the system over which the system can form predictions or make recommendations

• Novelty and Serendipity
  – Helping users to find a surprisingly interesting item he might not have otherwise discovered

• Confidence
  – How sure is the RS that its recommendation is accurate?

• Diversity
• Risk
• Robustness
• Privacy
• Adaptivity
• Scalability
Content Recommendation in Social Media
Video Recommendation

• Related videos defined as videos that a user is likely to watch after having watched a video

• Approaches to video recommendation:
  – Association rule mining
  – co-visitation
Video Recommendation: Association Rule Mining

System calculates the probability of watching $v_j$ after the user watched $v_i$ and recommends top-N of highly ranked videos

$$Rank(v_i, v_j) = P(v_j|v_i)$$
Video Recommendation: Co-visitation

Co-visitation score is a number that shows in a given time period, how often two videos co-watched within sessions

\[
Co - visiting(v_i, v_j) = \frac{c_{ij}}{f(v_i, v_j)}
\]

c_{ij} is the number of co-watches for videos v_i and v_j and f (v_i, v_j) is a normalization factor regarding the popularity of two videos, e.g., e the product of the videos’ popularity (the number of views).
Tag Recommendation

- Tag recommendation is the process of recommending appropriate tags to be applied by the user per specific item annotation.

- Approaches:
  - Recommending the most popular tags
  - Collaborative Filtering
  - Content-based Tag Recommendation
  - Graph-based
Tag Recommendation: Popularity-based and CF Approaches

**Recommend the most popular tags**
- Popular tags already assigned for the target item
- Frequent tags previously used by the user, and
- Tags co-occurred with already assigned tags.

**Collaborative filtering**
- It can use item-based or user-based approaches.
- Or it may use a hybrid approach by recommending tags given by similar users to similar items.
Tag Recommendation: Content- and Graph-based Approaches

Content-based Tag Recommendation
– This can be done by recommending keywords from the item’s associated text or tags that have the highest co-occurrence with important keywords.

Graph-based Approaches
– The FolkRank algorithm is an example. Its idea is: a resource which is tagged with tags by important users should be important.
News Recommendation

• A recommended news should be of interest to the user if it is recent or fresh, diverse, and not very similar (the same) to the other news the user recently read.

• Regular recommender systems might not be used for news as recency is one of the most important factors for a piece of news.
Blog Recommendation

• The blog recommendation is the task of finding relevant blogs in response to a query

• Blog relevance ranking differs from classical document retrieval and ranking in several ways:
  – How to deal with blog posts
  – How to come up with reliable queries as user queries represent current interests in the topic
Blog Recommendation: An Algorithm

• A solution for the first problem is using *two different* document models
  – Large document model: entire blog as a whole
  – Small document model: each blog post as a document.

• The second problem can be solved by *query expansion*
  – Query Expansion (QE) is the process of reformulating a seed query to improve retrieval performance in information retrieval operations).
    • Using wikipedia is an often used method for query expansion: the query is sent to Wikipedia, and the retrieved wiki articles are now *new queries* and will be used for blog searching.

• After solving these two problems of blogs, we can use existing recommendation systems to get the blog recommendation.
Social Media Content Recommendation: Tag Based

- People use tags to summarize, remember, and organize information.
- Tags are a powerful tool for social navigation, helping people to share and discover new information contributed by other community members.
- Tags promote social navigation by their vocabulary, or the set of tags used by members of the community.
- Instead of imposing controlled vocabulary or categories, tagging systems’ vocabulary emerges organically from the tags chosen or created by individual members.
- A tag-based recommendation system uses tags to recommend items.
Tag Based Recommendation

• Algorithms combining tags with recommenders provide both the automation inherent in recommenders, and the flexibility and conceptual comprehensibility inherent in the tagging system.
Hybrid Approaches to Recommendation
Pipeline hybridization

- This approach uses more than one recommender system and puts the recommender systems in a line.
- The result of one recommender is the input for another one.
- The earlier recommender can make a model of the input and pass it to the next recommender system or can generate a list of recommendations to be used by the next recommender system.
Parallelized hybridization

• We use more than one recommender algorithm.
• The hybridization process gets the result of recommenders, combines them and generates the final recommendation.
• Different methods can be used to combine the results of other recommenders such as:
  – Mixed (a union of results from all recommender systems),
  – Weighted (a weighted combination of the results),
  – Switching (use results from specific recommender systems for specific tasks),
  – Majority voting.
Recommendation
Explanation

The digital camera X is the best for you because ...
Confidence

• Why would someone distrust a recommendation?
  – Can I trust the provider?
  – How does this work, anyway?
  – Does the system know enough about me?
  – Does the system know enough about the item it is recommending?
  – How sure is it?
The Confidence Challenge

• Why should users believe recommendations?
• When should users believe them?
• Approaches
  – Confidence indicators
  – Explain the recommendations
    • Reveal data and processes
    • Corroborating data, track records
  – Offer opportunities to correct mistaken data
Objectives of explanations

• Transparency
  – Provide information such that the user can comprehend the reasoning used to generate a specific recommendation

• Validity
  – Explanations can be generated in order to allow a user to check the validity of a recommendation

• Trustworthiness
  – Explanations aiming to build trust in recommendations reduce the uncertainty about the quality of a recommendation

• Comprehensibility
  – Explanations targeting comprehension support the user by relating her known concepts to the concepts employed by the recommender

• Education
  – Deep knowledge about the domain helps the customer rethink her preferences and evaluate the pros and cons of different solutions
Objectives of explanations

• Persuasiveness
  – In this sense persuasive explanations for recommendations aim to change the user's buying behavior

• Effectiveness
  – The support a user receives for making high-quality decisions

• Efficiency
  – A system's ability to support users in order to reduce the decision-making effort e.g. time

• Satisfaction
  – Explanations can attempt to improve the overall satisfaction stemming from the use of a recommender system.

• Relevance
  – Additional information may be required in conversational recommenders. Explanations can be provided to justify why additional information is needed from the user
Examples

- Similarity between items
- Similarity between users
- Tags
  - Tag relevance (for items)
  - Tag preference (of users)
Explanation types

• Nearest neighbor explanation
  – Customers who bought item X also bought items Y,Z
  – Item Y is recommended because you rated related item X

• Content based explanation
  – This story deals with topics X,Y which belong to your topic of interest

• Social-network based explanation
  – People leverage their social network to reach information and make use of trust relationships to filter information.
    • Your friend X wrote that blog
    • 50% of your friends liked this item (while only 5% disliked it)