Part 11: Collaborative Filtering

Francesco Ricci
Content

- An example of a Collaborative Filtering system: MovieLens
- The collaborative filtering method
  - Similarity of users
  - Methods for building the rating prediction
- The role of a recommender system
  - Service Provider
  - End user
- Evaluating a recommender system
  - Mean absolute error
  - Precision/Recall
  - Normalized discounted cumulative gain
It’s all about You

Yes, you.
You control the Information Age.
Welcome to your world.
User generated content

- **User-generated content** has been the key to success for many of today’s leading Web 2.0 companies, such as Amazon, eBay and Youtube.

- The community **adds value** to these sites, which, in many cases, are almost entirely built on user-generated content.

- User-generated **content types**:  
  - articles, reviews ([tripadvisor](https://www.tripadvisor.com))  
  - home videos ([youtube](https://www.youtube.com))  
  - photos ([flickr](https://www.flickr.com))  
  - items evaluations/ratings (all!)  
  - information that is gathered from the users’ **actions online** (e.g. in Amazon recommender system).
Recommender Systems

- In everyday life we rely on recommendations from other people either by word of mouth, recommendation letters, movie and book reviews printed in newspapers ...

- In a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients
  - Aggregation of recommendations
  - Match the recommendations with those searching for recommendations.

[Resnick and Varian, 1997]
Social Filtering
The Collaborative Filtering Idea

- Trying to **predict** the opinion the user will have on the different items and be able to recommend the “best” items to each user.
- It is based on: **the user’s previous likings** and the opinions of other like minded users.
- CF is a typical **Internet application** – it must be supported by a networking infrastructure:
  - At least many users and one server
  - But also a distributed model with many servers
- There is no stand alone CF application.
Welcome to the new MovieLens!

Existing MovieLens users: We'd like to welcome you back to MovieLens, and let you know we have a new MovieLens FAQ you might want to read. We hope you like what you will see!

Take me to MovieLens!

New MovieLens users: Thank you for joining MovieLens! In order to generate personalized movie recommendations, we need to know a little about what movies you have already seen. MovieLens will now display several lists of movies. If you have seen any of the listed movies, please rate them using the rating scale shown below.

Ratings are on a scale of 1 to 5:

⭐⭐⭐⭐⭐ = Must See
⭐⭐⭐⭐ = Will Enjoy
⭐⭐⭐⭐ = It's OK
⭐⭐⭐⭐ = Fairly Bad
⭐⭐⭐ = Awful

Remember: the more movies you rate, the more accurate MovieLens' predictions will be.

To rate a movie, just click on the pulldown next to the title of a movie you have seen. Blue stars will appear to indicate that your rating has been received.

⭐⭐⭐⭐⭐ 1.5 stars  
DVD, VHS, info | imdb
Comedy

This image shows that the movie ‘Dude, Where's My Car?’ was rated 1.5 stars.

I'm ready to start rating!
So far you have rated 0 movies. MovieLens needs at least 15 ratings from you to generate predictions for you. Please rate as many movies as you can from the list below.

<table>
<thead>
<tr>
<th>Your Rating</th>
<th>Movie Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>★★★</td>
<td>Austin Powers: International Man of Mystery (1997) Action, Adventure, Comedy</td>
</tr>
<tr>
<td>★★★★★</td>
<td>Contact (1997) Drama, Sci-Fi</td>
</tr>
<tr>
<td>???</td>
<td>Eraser (1996) Action, Drama, Thriller</td>
</tr>
<tr>
<td>???</td>
<td>Maverick (1994) Action, Comedy, Western</td>
</tr>
<tr>
<td>★★★★</td>
<td>Philadelphia (1993) Drama</td>
</tr>
<tr>
<td>★★★</td>
<td>Piano, The (1993) Drama, Romance</td>
</tr>
<tr>
<td>???</td>
<td>Toy Story 2 (1999) Adventure, Animation, Children, Comedy, Fantasy</td>
</tr>
</tbody>
</table>

To get a new set of movies click the **next >** link.
Congratulations!

MovieLens can now generate personalized movie recommendations for you.

Start Using MovieLens

Remember, you can always keep rating movies you have seen. The more movies you rate, the better your predictions will be. We’d also like to tell you about some other features of MovieLens you might be interested in:

- **Getting recommendations.** MovieLens has shortcuts like Top Picks For You that provide you with quick access to common searches. You can use the Search tab to perform more advanced searches that filter by genre, date, and more, and save your favorite searches as personal shortcuts.

- **Your Wishlist.** Here you can keep track of movies you haven’t yet seen. You can even print this list out and take it with you to your video store.

- **Movie buddies.** It can be a pain trying to decide what movie a group of people should see. Let MovieLens choose the right movie for you! You can add MovieLens users to be your buddies and be able to generate group movie recommendations.

We will keep adding more great features as time goes on, so look for them!

Start Using MovieLens
Welcome fricci@unibz.it (Log Out)
You've rated 70 movies.
You're the 18th visitor in the past hour.

Your Movies
- Your Ratings
- About Your Ratings
- Your Wishlist
- Your Tags

Rate and Find Movies
- Top Picks For You
- Newest Additions
- Most Often Rated
- Rate Random Movies
- Browse Movies by Tags

Your Account
- Your Profile (edit)
- Preferences
- Manage Buddies
- Manage RSS Feeds

Help MovieLens
- Volunteer Center
- Vote for Titles

New Movies
★★★★★ Lincoln Lawyer, The (2011)
★★★★★ Source Code (2011)
★★★★★ Limitless (2011)
★★★★★ Evangelion: 2.0 You Can (Not) Advance (Evangelion shin gekijôban: Ha) (2009)
★★★★★ Rango (2011)
★★★★★ Paul (2011)
★★★★★ Certified Copy (Copie conforme) (2010)
★★★★★ Ip Man 2 (2010)
★★★★★ Zeitgeist: Moving Forward (2011)
★★★★★ Adjustment Bureau, The (2011)

New DVDs
★★★★★ Black Swan (2010)
★★★★★ Social Network, The (2010)
★★★★★ Fighter, The (2010)
★★★★★ Fish Tank (2009)
★★★★★ Tillman Story, The (2010)
★★★★★ Inside Job (2010)
★★★★★ Letters to Father Jacob (Postia pappi Jaakobile) (2009)
★★★★★ Animal Kingdom (2010)
★★★★★ Made in Dagenham (2010)

0 new movies have been added since you last visited. See the newest additions.

Movie Tuner New!

Want a movie like Pulp Fiction but less "violent"?
Or a movie like Mission: Impossible but more "realistic"?

Movie Tuner lets you "tune" your movie selection along 1500 unique dimensions.

Look for Movie Tuner on the Movie Details and Search Results pages. More info
<table>
<thead>
<tr>
<th>Prediction</th>
<th>Rating</th>
<th>Movie Information</th>
<th>Wish List</th>
</tr>
</thead>
<tbody>
<tr>
<td>✭✭✭✭✭</td>
<td>Not seen</td>
<td>Work of Director Michel Gondry, The (2003) DVD **info</td>
<td>imdb</td>
</tr>
<tr>
<td>Comedy, Documentary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[add tag] Popular tags: quirky</td>
<td>notable soundtrack</td>
<td>surreal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✭✭✭✭✭</td>
<td>Not seen</td>
<td>Hearts and Minds (1974) DVD **info</td>
<td>imdb</td>
</tr>
<tr>
<td>Documentary, War</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[add tag] Popular tags: Vietnam War</td>
<td>racism</td>
<td>military</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✭✭✭✭✭</td>
<td>Not seen</td>
<td>Mad Love (1935) DVD **info</td>
<td>imdb</td>
</tr>
<tr>
<td>Horror, Romance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[add tag] Popular tags: DVD</td>
<td>not available from Netflix</td>
<td>DVD-RAM</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✭✭✭✭✭</td>
<td>Not seen</td>
<td>Double Indemnity (1944) DVD VHS **info</td>
<td>imdb</td>
</tr>
<tr>
<td>Crime, Drama, Film-Noir</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Welcome fricci@unibz.it (Log Out)
You've rated 70 movies.
You're the 18th visitor in the past hour.

Rate and Find Movies
- Top Picks For You
- Newest Additions
- Most Often Rated
- Rate Random Movies
- Browse Movies by Tags

Your Movies
- Your Ratings
- About Your Ratings
- Your Wishlist
- Your Tags

Your Account
- Your Profile (edit)
- Preferences
- Manage Buddies
- Manage RSS Feeds

Help MovieLens
- Volunteer Center
- Vote for Titles


Your Prediction: ★★★★★
Rate This Movie: Not seen Wish List: 

Movie Information (edit info) (flag)
Starring: Michel Gondry, Björk, Beck, David Grohl, David Cross, Jack White, Meg White, Cibo Matto
Directed By: Michel Gondry, Lance Bangs, Olivier Gondry
Genres: Comedy, Documentary
Languages: English French
Average rating: ★★★★★ (4.11 stars)
Your Prediction: ★★★★★ (5.0 stars)
Rated by: 71 users
Links: IMDb, Rotten Tomatoes

Movie Tags (more about tags)
Add and edit tags here or update all of your tags

Community Tags (?)
Tags represent how MovieLens users feel about this movie

01/11 02/11 03/11 bjork creative David Cross

The Work of Director Michel Gondry
The tireless creativity of director Michel Gondry is on vivid display in this collection of 27 music videos and other whimsical oddities. Released the year before Gondry's feature breakthrough Eternal Sunshine of the Spotless Mind, the compilation includes Kylie Minogue's "Come into My World," Bjork's "Human Behavior," Massive Attack's "Protection" and the White Stripes' Lego-centric stunner "Fell in Love with a Girl."

Report Wrong Movie
Delivered by Netflix (add to queue)
Collaborative Filtering
The CF Ingredients

- List of **m Users** and a list of **n Items**
- Each user has a **list of items** he/she expressed their **opinion** about (can be a null set)
- **Explicit opinion** - a rating score (numerical scale)
  - Sometime the rating is **implicitly** – purchase records
- **Active user** for whom the CF prediction task is performed
- A **metric** for measuring **similarity between users**
- A method for selecting a **subset of neighbors** for prediction
- A method for **predicting a rating** for items not currently rated by the active user.
Collaborative-Based Filtering

- The collaborative based filtering recommendation techniques proceeds in these steps:
  1. For a **target/active user** (the user to whom a recommendation has to be produced) the set of his ratings is identified
  2. The **users more similar** to the target/active user (according to a similarity function) are identified (neighbor formation)
  3. The **products evaluated by these similar users** are identified
  4. For each one of these products a **prediction** - of the rating that would be given by the target user to the product - is generated
  5. Based on this predicted ratings the set of **top N** products are recommended.
A Simplified Model of Recommendation

1. Two types of entities: **Users and Items**
2. A **background knowledge**:
   - A set of ratings: a map \( R : \text{Users} \times \text{Items} \rightarrow [0,1] \cup \{?\} \)
   - A set of “features” of the Users and/or Items
3. A **method** for eliminating all or part of the ‘?’ values for some (user, item) pairs – substituting ‘?’ with the true values

   \[
   R(u, i) = \text{Average} \ \{R(su, i)\} \ \text{su similar to u}
   \]

4. A method for selecting the items to recommend
   - Recommend to u the item \( i^* = \arg \max_{i \in \text{Items}} \{R(u,i)\} \)

   [Adomavicius et al., 2005]
Nearest Neighbor Collaborative-Based Filtering

User Model = interaction history

Dislike
Like
Unknown

Hamming distance

1st item rate

14th item rate

Nearest Neighbor
1-Nearest Neighbor can be easily wrong

Dislike
Like
Unknown

User Model = interaction history

This is the only user having a positive rating on this product

Hamming distance

1st item rate

5 6 6 5 4 8 Nearest Neighbor
### Movie rating data

#### Training data

<table>
<thead>
<tr>
<th>user</th>
<th>movie</th>
<th>date</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21</td>
<td>5/7/02</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>213</td>
<td>8/2/04</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>345</td>
<td>3/6/01</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>123</td>
<td>5/1/05</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>768</td>
<td>7/15/02</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>76</td>
<td>1/22/01</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>45</td>
<td>8/3/00</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>568</td>
<td>9/10/05</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>342</td>
<td>3/5/03</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>234</td>
<td>12/28/00</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>76</td>
<td>8/11/02</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>56</td>
<td>6/15/03</td>
<td>4</td>
</tr>
</tbody>
</table>

#### Test data

<table>
<thead>
<tr>
<th>user</th>
<th>movie</th>
<th>date</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>62</td>
<td>1/6/05</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>96</td>
<td>9/13/04</td>
<td>?</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>8/18/05</td>
<td>?</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>11/22/05</td>
<td>?</td>
</tr>
<tr>
<td>3</td>
<td>47</td>
<td>6/13/02</td>
<td>?</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>8/12/01</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>41</td>
<td>9/1/00</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>8/27/05</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>93</td>
<td>4/4/05</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>74</td>
<td>7/16/03</td>
<td>?</td>
</tr>
<tr>
<td>6</td>
<td>69</td>
<td>2/14/04</td>
<td>?</td>
</tr>
<tr>
<td>6</td>
<td>83</td>
<td>10/3/03</td>
<td>?</td>
</tr>
</tbody>
</table>
# Matrix of ratings

|     | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y |
| 1   | 1 | 4 | 5 | 4 | 3 | 2 | 4 | 2 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 2   | 4 | 3 | 5 | 1 | 3 | 4 | 2 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 3   | 5 | 4 | 4 | 3 | 5 | 4 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 4   | 3 | 5 | 4 | 1 | 5 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 5   | 2 | 4 | 2 | 5 | 1 | 4 | 5 | 4 | 2 | 4 | 5 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |
| 6   | 1 | 4 | 3 | 5 | 4 | 2 | 5 | 4 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 7   | 1 | 3 | 5 | 4 | 4 | 5 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 8   | 4 | 1 | 3 | 5 | 4 | 1 | 5 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 9   | 1 | 4 | 5 | 4 | 5 | 1 | 5 | 4 | 3 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |
| 10  | 4 | 4 | 5 | 4 | 5 | 4 | 2 | 5 | 5 | 5 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 11  | 4 | 5 | 4 | 5 | 4 | 5 | 2 | 5 | 5 | 5 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 12  | 1 | 4 | 3 | 4 | 5 | 1 | 5 | 4 | 2 | 4 | 4 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |
| 13  | 2 | 1 | 4 | 3 | 5 | 4 | 2 | 5 | 4 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |
| 14  | 4 | 1 | 4 | 2 | 2 | 5 | 4 | 5 | 4 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |
| 15  | 2 | 4 | 4 | 5 | 1 | 4 | 2 | 4 | 4 | 5 | 5 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |
| 16  | 1 | 4 | 3 | 4 | 5 | 1 | 5 | 4 | 4 | 3 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 17  | 2 | 1 | 4 | 3 | 5 | 4 | 2 | 4 | 5 | 4 | 2 | 3 |   |   |   |   |   |   |   |   |   |   |   |
| 18  | 4 | 5 | 4 | 3 | 5 | 2 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 19  | 2 | 2 | 3 | 5 | 4 | 5 | 4 | 2 | 3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 20  | 4 | 5 | 3 | 3 | 4 | 5 | 1 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 21  | 1 | 3 | 2 | 3 | 4 | 5 | 3 | 3 | 5 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 22  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 23  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 24  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 25  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
Collaborative-Based Filtering

- A collection of \( n \) user \( u_i \) and a collection of \( m \) products \( p_j \)
- A \( n \times m \) matrix of ratings \( v_{ij} \), with \( v_{ij} = ? \) if user \( i \) did not rate product \( j \)
- Prediction for user \( i \) and product \( j \) is computed as

\[
v_{ij}^* = v_i + \sum_{u_{ik} \neq ?} u_{ik} (v_{kj} - v_k)
\]

Where, \( v_i \) is the average rating of user \( i \), \( K \) is a normalization factor such that the sum of \( u_{ik} \) is 1, and

\[
u_{ik} = \frac{\sum_j (v_{ij} - v_i)(v_{kj} - v_k)}{\sqrt{\sum_j (v_{ij} - v_i)^2 \sum_j (v_{kj} - v_k)^2}}
\]

- Where the sum (and averages) is over \( j \) s.t. \( v_{ij} \) and \( v_{kj} \) are not “?”.
**Example**

Users’ similarities: $u_{i5} = 0.5$, $u_{i8} = 0.5$, $u_{i9} = 0.8$

$$v^*_{ij} = v_i + K \sum_{v_{kj} \neq ?} u_{ik} (v_{kj} - v_k)$$

$$v^*_{ij} = 3.2 + 1/(0.5+0.5+0.8) \times [0.5 \times (4-4) + 0.5 \times (3-3.5) + 0.8 \times (5-3)]$$

$$= 3.2 + 1/1.8 \times [0 - 0.25 + 1.6] = 3.2 + 0.75 = 3.95$$
Proximity Measure: Cosine

- Correlation can be replaced with a typical Information Retrieval (IR) similarity measure: \textit{cosine}

\[
\cos(u_i, u_j) = \frac{\sum_{k=1}^{m} v_{ik} v_{jk}}{\sqrt{\sum_{k=1}^{m} v_{ik}^2 \sum_{k=1}^{m} v_{jk}^2}}
\]

- This has been shown to provide worse results by someone [Breese et al., 1998]
- But many uses cosine [Sarwar et al., 2000] and somebody reports that it performs better [Anand and Mobasher, 2005]

Most of these ratings are unknown, so are not used
Comparison: Pearson vs. Cosine

### Pearson

<table>
<thead>
<tr>
<th></th>
<th>user 1</th>
<th>user 2</th>
<th>user 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>p2</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>p3</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>p4</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>p5</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>p6</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>p7</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>p8</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

- User 2 ratings are those of user 1 incremented by 1

### Cosine

<table>
<thead>
<tr>
<th></th>
<th>user 1</th>
<th>user 2</th>
<th>user 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1</td>
<td>1,00</td>
<td>0,99</td>
<td>0,76</td>
</tr>
<tr>
<td>user 2</td>
<td>0,99</td>
<td>1,00</td>
<td>0,84</td>
</tr>
<tr>
<td>user 3</td>
<td>0,76</td>
<td>0,84</td>
<td>1,00</td>
</tr>
</tbody>
</table>

- User 3 has “opposite” preferences of user 1
Red and green pairs of vectors (users) have Pearson correlation = -1
(ratings inverted with respect to the “average” rating 2.5)

Red vectors have a “cosine” distance smaller than green (dashed)
vectors (more reasonable in this case)
Other Aggregation Function

- $u_{ik}$ is the similarity of user $u_i$, and $u_k$
- A $n \times m$ matrix of ratings $v_{ij}$, with $v_{ij} = ?$ if user $i$ did not rate product $j$
- Prediction for user $i$ and product $j$, is computed as (K is a normalization factor):

$$v_{ij}^* = v_i + K \sum_{v_{kj} \neq ?} u_{ik} (v_{kj} - v_k)$$

$$v_{ij}^* = K \sum_{v_{kj} \neq ?} u_{ik} v_{kj}$$

$$v_{ij}^* = \frac{1}{N} \sum_{u_k \in U(u_i)} v_{kj} \quad \text{U}(u_i) \text{ is a neighbor of users similar to } u_i \text{ - of size } N$$
The goal of a RS: service provider

- **Increase the number of sold items**
  - Because the recommended items are likely to suit the user's needs and wants

- **Sell more diverse items**
  - Using a RS the user can select items that might be hard to find without a precise recommendation

- **Increase the user satisfaction**
  - The user will find the recommendations interesting, relevant, and would enjoy using the system

- **Increase user fidelity**
  - A user should be loyal to a Web site which, when visited, recognizes the old customer and treats him as a valuable visitor

- **Better understand what the user wants**
  - Build a user profile that can be used in several personalization tasks (e.g., direct marketing).
The Long Tail

- Netflix (catalog of over 80,000 movie titles) rents a large volume of less popular movies in addition to the substantial business it does renting hits.

- The Long Tail: the economic model in which the market for non-hits (typically large numbers of low-volume items) could be significant and sometimes even greater than the market for big hits (typically small numbers of high-volume items).
The goal of a RS: users

- Find some good Items
- Find all good items
- Annotation in context
- Recommend a sequence
- Recommend a bundle
- Just browsing

- Find credible recommender
- Improve the profile
- Express self
- Help others
- Influence others.
Evaluating Recommender Systems

- The majority focused on system’s accuracy in supporting the “find good items” user’s task
- Assumption: if a user could examine all the available items, she could place them in a ordering of preference
  1. Measure how good is the system in predicting the exact rating value (value comparison)
  2. Measure how well the system can predict whether the item is relevant or not (relevant vs. not relevant)
  3. Measure how close the predicted ranking of items is to the user’s true ranking (ordering comparison).
How Has Been Measured

- **Split** the available data (so you need to collect data first!), i.e., the user-item ratings into two sets: *training* and *test*

- **Build a model** on the training data
  - For instance, in a nearest neighbor (memory-based) CF simply put the ratings in the training in a separate set

- **Compare the predicted ...**
  - *rating on each test item* (user-item combination) with the true rating stored in the test set
  - *recommendations* with the really good recommendations (*what are they?*)
  - *ranking* with the true ranking (*what is this?*)

- You need a *metric* to compare the *predicted* rating (or recommendation or ranking) with the *true* rating (or recommendation or ranking).
Splitting the data

Does not work

- **items**
- **users**

- **train**
- **test**
Splitting the data

Does not work

users

items

\[ 
\text{train} \quad \text{test} 
\]
Splitting the data

It works!

<table>
<thead>
<tr>
<th>users</th>
<th>items</th>
</tr>
</thead>
<tbody>
<tr>
<td>train</td>
<td>test</td>
</tr>
</tbody>
</table>
Accuracy: Comparing Values

- Measure **how close** the predicted ratings are to the true user ratings (for all the ratings in the test set)

- **Predictive accuracy (rating):** **Mean Absolute Error (MAE)**, \( p_i \) is the **predicted** rating and \( r_i \) is the **true** one:

  \[
  MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N}
  \]

- It may be less appropriate for tasks such as Find Good Items – because people look only to top rated items

- **Variation 1:** **Mean Squared Error** (take the square of the differences), **Root Mean Squared Error** (and then take the square root). These metrics emphasize large errors.

- **Variation 2:** **Normalized MAE** – MAE divided by the range of possible ratings – allowing comparing results on different data sets, having different rating scales.
Relevant Recommendations: Precision and Recall

<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>$N_{rn}$</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>

- To compute P and R the rating scale must be binary – or one must transform it into a binary scale (e.g. items rated above 3 vs. those rated below)
- **Precision** is the ratio of relevant items selected by the recommender to the number of items selected ($N_{rs}/N_s$)
- **Recall** is the ratio of relevant items selected to the number of relevant ($N_{rs}/N_r$)
- **Precision and recall** are the most popular metrics for evaluating information retrieval systems.
Example – Complete Knowledge

- We assume to know the relevance of all the items in the catalogue for a given user.
- If you have ratings – consider relevant the items whose rating is above the average rating (e.g., 4 and 5).
- Assume that the orange portion is that recommended by the system.

Precision = 4/7 = 0.57
Recall = 4/9 = 0.44
Example – Incomplete Knowledge

- We do not know the relevance of all the items in the catalogue for a given user.
- The orange portion is that recommended by the system.

Precision: $4/7 = 0.57$ OR $4/6$?
Recall: $4/10 \leq R \leq 4/7$
$4/10$ if all unknown are relevant
$4/7$ if all unknown are irrelevant

Researchers typically say $P=4/6$ and $R=4/7$ (they are optimistic 😊)
F1

- Combinations of Recall and Precision such as $F_1$
- Typically systems with high recall have low precision and vice versa
- Same problems as before when knowledge is incomplete.

\[ F_1 = \frac{2PR}{P + R} \]

Selected

1
0
0
1
1
0
1
1
1
0
Precision recall for recommenders

- Relevant if the true rating is >= 4
- Retrieve all the items whose predicted rating is >= x (x=5, 4.5, 4, 3.5, ... 0)
- You get 10 points to plot
- Why precision is not going to 0? Exercise.
- What the 0.7 value represents?
Problems with Precision and Recall

- To compute them we must know what items are relevant and what are not relevant.
- Difficult to know what is relevant for a user in a recommender system that manages thousands/millions of products.
- May be easier for some tasks where, given the user or the context, the number of recommendable products is small – only a small portion could fit.
- Recall is more difficult to estimate (knowledge of all the relevant products).
- Precision is a bit easier – you must know what part of the selected products are relevant (you can ask to the user after the recommendation – but has not been done in this way – not many evaluations did involve real users).
Quality of the produced ranking: NDCG

- For a set of queries \( Q \) (users), let \( R(j, d) \) be the relevance score (rating) that human assessors (user) gave to document (item) at rank index \( d \) for query (user) \( j \).
- The ranking is computed by sorting the items by decreasing rating prediction.

\[
\text{NDCG}(Q, k) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} Z_{kj} \left( \sum_{m=1}^{k} \frac{2^{R(j,m)} - 1}{\log_2(1 + m)} \right)
\]

- Where \( Z_{kj} \) is a normalization factor calculated to make it so that a perfect ranking’s NDCG at \( k \) for query \( j \) is 1.
- For queries for which \( k' < k \) documents are retrieved, the last summation is done up to \( k' \).
- When rating knowledge is incomplete the items considered are only those evaluated by the user (test set).
Summary

- Illustrated the basic Collaborative Filtering recommendation method
- Illustrated different methods for neighbor formation and prediction computation
- Explained the role of a recommender system
- Illustrated some methods for measuring the performance of a RS
  - Exact rating prediction: mean square error (MAE)
  - Relevant: precision and recall
  - Normalized cumulative discounted gain
- Discussion on the precision/recall issues and tradeoff.
Questions

- How the collaborative filtering (CF) technique works?
- Can CF work on your PC if this is not networked?
- What are the advantages and disadvantages of CF?
- What are the methods used for computing the similarity of users?
- Could you imagine other similarity measures?
- What is the user model in a CF recommender system?
- Why a RS can help to sell the less popular items?
- How the CF method can take into account the fact that a rating is old and may not be relevant anymore?
- How to select those items for which to ask the user rating? A good item selection method.
- Is precision more important than recall in a recommender system?