CS 124/LINGUIST 180
From Languages to Information
Dan Jurafsky
Stanford University

Recommender Systems & Collaborative Filtering

Slides adapted from Jure Leskovec
Recommender Systems

Customer X
- Buys Metallica CD
- Buys Megadeth CD

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

Search

Recommendations

Items

Examples:

- Amazon.com
- Pandora
- StumbleUpon
- del.icio.us
- Netflix
- Movielens
- Last.fm
- Google News
- YouTube
- Xbox Live

Products, web sites, blogs, news items, …
From Scarcity to Abundance

Shelf space is a scarce commodity for traditional retailers
  ◦ Also: TV networks, movie theaters,…

Web enables near-zero-cost dissemination of information about products
  ◦ From scarcity to abundance

More choice necessitates better filters
  ◦ Recommendation engines
  ◦ How Into Thin Air made Touching the Void a bestseller:
    http://www.wired.com/wired/archive/12.10/tail.html
Sidenote: The Long Tail

---

**The Documentary Niche Gets Richer**

More than 40,000 documentaries have been released, according to the Internet Movie Database. Of those, Amazon.com carries 40 percent, Netflix stocks 3 percent, and the average Blockbuster just 2 percent.

---

Sources: Erik Brynjolfsson and Jeffrey Hu, MIT, and Michael Smith, Carnegie Mellon; Barnes & Noble; Netflix; RealNetworks

Source: Chris Anderson (2004)
Physical vs. Online

Read http://www.wired.com/wired/archive/12.10/tail.html to learn more!

SLIDES ADAPTED FROM JURE LESKOVEC
Types of Recommendations

**Editorial and hand curated**
- List of favorites
- Lists of “essential” items

**Simple aggregates**
- Top 10, Most Popular, Recent Uploads

**Tailored to individual users**
- Amazon, Netflix, ...
Formal Model

\[ X = \text{set of Customers} \]

\[ S = \text{set of Items} \]

**Utility function** \( u: X \times S \rightarrow R \)

- \( R = \text{set of ratings} \)
- \( R \) is a totally ordered set
- e.g., 0-5 stars, real number in \([0,1]\)
# Utility Matrix

<table>
<thead>
<tr>
<th></th>
<th>Avatar</th>
<th>LOTR</th>
<th>Matrix</th>
<th>Pirates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>1</td>
<td>0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>0.5</td>
<td></td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>0.2</td>
<td>1</td>
<td></td>
<td>0.4</td>
</tr>
<tr>
<td>David</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SLIDES ADAPTED FROM JURE LESKOVEC
Key Problems

(1) Gathering “known” ratings for matrix
   ◦ How to collect the data in the utility matrix

(2) Extrapolate unknown ratings from known ones
   ◦ Mainly interested in high unknown ratings
     ◦ We are not interested in knowing what you don’t like but what you like

(3) Evaluating extrapolation methods
   ◦ How to measure success/performance of recommendation methods
(1) Gathering Ratings

**Explicit**
- Ask people to rate items
- Doesn’t work well in practice – people can’t be bothered
- Crowdsourcing: Pay people to label items

**Implicit**
- Learn ratings from user actions
  - E.g., purchase implies high rating
- What about low ratings?
(2) Extrapolating Utilities

**Key problem:** Utility matrix $U$ is *sparse*
- Most people have not rated most items
- **Cold start:**
  - New items have no ratings
  - New users have no history

**Three approaches to recommender systems:**
1. Content-based
2. Collaborative
3. Latent factor based
Content-based Recommender Systems
Content-based Recommendations

**Main idea:** Recommend items to customer $x$ similar to previous items rated highly by $x$

**Example:**

**Movie recommendations**
- Recommend movies with same actor(s), director, genre, ...

**Websites, blogs, news**
- Recommend other sites with “similar” content
Plan of Action

Item profiles

Red
Circles
Triangles

User profile

match

build

likes

recommend
Item Profiles

For each item, create an **item profile**

Profile is a set (vector) of features

- **Movies:** author, genre, director, actors, year...
- **Text:** Set of “important” words in document

How to pick important features?

- **TF-IDF** (Term frequency * Inverse Doc Frequency)
  - Term ... Feature
  - Document ... Item
Content-based Item Profiles

<table>
<thead>
<tr>
<th></th>
<th>Melissa McCarthy</th>
<th>Actor A</th>
<th>Actor B</th>
<th>...</th>
<th>Johnny Depp</th>
<th>Comic Genre</th>
<th>Spy Genre</th>
<th>Pirate Genre</th>
<th>Avg Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movie X</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3α</td>
</tr>
<tr>
<td>Movie Y</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4α</td>
</tr>
</tbody>
</table>

- Maybe there is a scaling factor $\alpha$ between binary and numeric features
- Or maybe $\alpha=1$

$$\text{Cosine(Movie X, Movie Y)} = \frac{2 + 12\alpha^2}{\sqrt{5 + 9\alpha^2} \sqrt{5 + 16\alpha^2}}$$
User Profiles

Want a vector with the same components/dimensions as items:

- Could be 1s representing user purchases
- Or arbitrary numbers from a rating

User profile is aggregate of items:

- Average (weighted?) of rated item profiles
Sample user profile

- Items are movies
- Utility matrix has 1 if user has seen movie
- 20% of the movies user U has seen have Melissa McCarthy
- \( U["\text{"Melissa McCarthy"}"] = 0.2 \)

<table>
<thead>
<tr>
<th>Melissa McCarthy</th>
<th>Actor A</th>
<th>Actor B</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>User U</td>
<td>0.2</td>
<td>.005</td>
<td>0</td>
</tr>
</tbody>
</table>
Prediction

- User and item vectors have the same components/dimensions!
- So just recommend the items whose vectors are most similar to the user vector!

- Given user profile $\mathbf{x}$ and item profile $\mathbf{i}$,
- estimate $u(x, i) = \cos(x, i) = \frac{x \cdot i}{||x|| \cdot ||i||}$
Pros: Content-based Approach

+: No need for data on other users
  ◦ No cold-start or sparsity problems

+: Able to recommend to users with unique tastes

+: Able to recommend new & unpopular items
  ◦ No first-rater problem

+: Able to provide explanations
  ◦ Can provide explanations of recommended items by listing content-features that caused an item to be recommended
Cons: Content-based Approach

→: Finding the appropriate features is hard
  ◦ E.g., images, movies, music

→: Recommendations for new users
  ◦ How to build a user profile?

→: Overspecialization
  ◦ Never recommends items outside user’s content profile
  ◦ People might have multiple interests
  ◦ Unable to exploit quality judgments of other users
Collaborative Filtering
Harnessing quality judgments of other users
Collaborative Filtering

Consider user $x$

Find set $N$ of other users whose ratings are “similar” to $x$’s ratings

Estimate $x$’s ratings based on ratings of users in $N$
Finding Similar Users

Let $r_x$ be the vector of user $x$'s ratings

**Jaccard similarity measure**
- **Problem:** Ignores the value of the rating

**Cosine similarity measure**
- $\text{sim}(x, y) = \cos(r_x, r_y) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$
- **Problem:** Treats missing ratings as “negative”

\[ r_x = [\ast, _, _, *, ***,] \]
\[ r_y = [\ast, _, ***, **, _] \]
9.3.1 Measuring Similarity

The first question we must deal with is how to measure similarity of users or items from their rows or columns in the utility matrix. We have reproduced Fig. 9.1 here as Fig. 9.4. This data is too small to draw any reliable conclusions, but its small size will make clear some of the pitfalls in picking a distance measure. Observe specifically the users A and C. They rated two movies in common, but they appear to have almost diametrically opposite opinions of these movies. We would expect that a good distance measure would make them rather far apart. Here are some alternative measures to consider.

![Utility Matrix](image)

<table>
<thead>
<tr>
<th></th>
<th>HP1</th>
<th>HP2</th>
<th>HP3</th>
<th>TW</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

**Intuitively we want:** $\text{sim}(A, B) > \text{sim}(A, C)$  
**Jaccard similarity:** $\frac{1}{5} < \frac{2}{4}$  
**Cosine similarity:** $0.386 > 0.322$  
Considers missing ratings as “negative”
CHAPTER 9. RECOMMENDATION SYSTEMS

9.3.1 Measuring Similarity

The first question we must deal with is how to measure similarity of users or items from their rows or columns in the utility matrix. We have produced Fig. 9.1 here as Fig. 9.4. This data is too small to draw any reliable conclusions, but its small size will make clear some of the pitfalls in picking a distance measure. Observe specifically the users A and C. They rated two movies in common, but they appear to have almost diametrically opposite opinions of these movies. We would expect that a good distance measure would make them rather far apart.

Here are some alternative measures to consider.

<table>
<thead>
<tr>
<th></th>
<th>HP1</th>
<th>HP2</th>
<th>HP3</th>
<th>TW</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

- **Problem with cosine**: 0 acts like a negative review
  - C really loves SW
  - A hates SW
  - B just hasn’t seen it
- Another problem: we’d like to normalize for raters
  - D rated everything the same; not very useful
Modified Utility Matrix: subtract the means of each row

<table>
<thead>
<tr>
<th></th>
<th>HP1</th>
<th>HP2</th>
<th>HP3</th>
<th>TW</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>HP1</th>
<th>HP2</th>
<th>HP3</th>
<th>TW</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2/3</td>
<td></td>
<td></td>
<td>5/3</td>
<td>-7/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1/3</td>
<td>1/3</td>
<td>-2/3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>-5/3</td>
<td>1/3</td>
<td>4/3</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

- Now a 0 means no information
- And negative ratings means viewers with opposite ratings will have vectors in opposite directions!
Modified Utility Matrix:
subtract the means of each row

<table>
<thead>
<tr>
<th></th>
<th>HP1</th>
<th>HP2</th>
<th>HP3</th>
<th>TW</th>
<th>SW1</th>
<th>SW2</th>
<th>SW3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2/3</td>
<td></td>
<td></td>
<td>5/3</td>
<td>-7/3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1/3</td>
<td>1/3</td>
<td>-2/3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>-5/3</td>
<td>1/3</td>
<td>4/3</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \cos(A,B) = \frac{(2/3) \times (1/3)}{\sqrt{(2/3)^2 + (5/3)^2 + (-7/3)^2} \sqrt{(1/3)^2 + (1/3)^2 + (-2/3)^2}} = 0.092 \]

\[ \cos(A,C) = \frac{(5/3) \times (-5/3) + (-7/3) \times (1/3)}{\sqrt{(2/3)^2 + (5/3)^2 + (-7/3)^2} \sqrt{(-5/3)^2 + (1/3)^2 + (4/3)^2}} = -0.559 \]

Now A and C are (correctly) way further apart than A,B
Cosine after subtracting mean

Turns out to be the same as Pearson correlation coefficient!!!

Cosine similarity is correlation when the data is centered at 0

- Terminological Note: subtracting the mean is zero-centering, not normalizing (normalizing is dividing by a norm to turn something into a probability), but the textbook (and in common use) we sometimes overload the term “normalize”
Finding Similar Users

Let \( r_x \) be the vector of user \( x \)'s ratings

**Cosine similarity measure**
- \( \text{sim}(x, y) = \cos(r_x, r_y) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||} \)
- **Problem**: Treats missing ratings as “negative”

**Pearson correlation coefficient**
- \( S_{xy} \) = items rated by both users \( x \) and \( y \)

\[
\text{sim}(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \cdot \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}
\]

\( r_x, r_y \) as points:
- \( r_x = \{1, 0, 0, 1, 3\} \)
- \( r_y = \{1, 0, 2, 2, 0\} \)

\( r_x = [*, _, _, *, **] \)
\( r_y = [*, _, **, **, _] \)
Rating Predictions

From similarity metric to recommendations:

Let $r_x$ be the vector of user $x$’s ratings

Let $N$ be the set of $k$ users most similar to $x$ who have rated item $i$

Prediction for item $i$ of user $x$:

\[ r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi} \]

\[ r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}} \]

- Many other tricks possible...

Shorthand:

\[ s_{xy} = sim(x, y) \]
Item-Item Collaborative Filtering

So far: User-user collaborative filtering

Another view: Item-item

- For item $i$, find other similar items
- Estimate rating for item $i$ based on ratings for similar items
- Can use same similarity metrics and prediction functions as in user-user model

$$r_{xi} = \frac{\sum_{j \in N(i;x)} S_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} S_{ij}}$$

$s_{ij} \ldots$ similarity of items $i$ and $j$
$r_{xj} \ldots$ rating of user $x$ on item $i$
$N(i;x) \ldots$ set of items rated by $x$ similar to $i$
Item-Item CF ($|N|=2$)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- unknown rating
- rating between 1 to 5

SLIDES ADAPTED FROM JURE LESKOVEC
### Item-Item CF (|N| = 2)

#### Users

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>?</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- Estimate rating of movie 1 by user 5

**SLIDES ADAPTED FROM JURE LESKOVEC**
Item-Item CF ($|N|=2$)

Neighbor selection:
Identify movies similar to movie 1, rated by user 5

Here we use Pearson correlation as similarity:
1) Subtract mean rating $m_i$ from each movie $i$
   
   $m_1 = (1+3+5+5+4)/5 = 3.6$
   
   row 1: [-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]

2) Compute cosine similarities between rows
### Item-Item CF ($|N|=2$)

Compute similarity weights:

\[ s_{1,3} = 0.41, \quad s_{1,6} = 0.59 \]
## Item-Item CF ($|N|=2$)

**Predict by taking weighted average:**

$$r_{1,5} = \frac{(0.41 \times 2 + 0.59 \times 3)}{(0.41 + 0.59)} = 2.6$$

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$
### Item-Item vs. User-User

<table>
<thead>
<tr>
<th></th>
<th>Avatar</th>
<th>LOTR</th>
<th>Matrix</th>
<th>Pirates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>1</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>0.5</td>
<td>0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>0.9</td>
<td>1</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td>1</td>
<td>0.4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- In practice, it has been observed that item-item often works better than user-user
- **Why?** Items are simpler, users have multiple tastes
Pros/Cons of Collaborative Filtering

+ Works for any kind of item
  ◦ No feature selection needed

- Cold Start:
  ◦ Need enough users in the system to find a match

- Sparsity:
  ◦ The user/ratings matrix is sparse
  ◦ Hard to find users that have rated the same items

- First rater:
  ◦ Cannot recommend an item that has not been previously rated
  ◦ New items, Esoteric items

- Popularity bias:
  ◦ Cannot recommend items to someone with unique taste
  ◦ Tends to recommend popular items
Hybrid Methods

**Implement two or more different recommenders and combine predictions**
- Perhaps using a linear model

**Add content-based methods to collaborative filtering**
- Item profiles for new item problem
- Demographics to deal with new user problem
# Evaluation

<table>
<thead>
<tr>
<th>users</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>movies</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SLIDES ADAPTED FROM JURE LESKOVEC
# Evaluation

![Matrix diagram with users and movies]

## Test Data Set

<table>
<thead>
<tr>
<th>Users</th>
<th>Movies</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>![Cells with question marks]</td>
</tr>
<tr>
<td>3</td>
<td>![Cells with question marks]</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The matrix represents a test data set for evaluating user-movie interactions.
Evaluating Predictions

Compare predictions with known ratings

- **Root-mean-square error (RMSE)**
  \[ \sqrt{\sum_{x_i} (r_{xi} - r_{xi}^*)^2} \]
  where \( r_{xi} \) is predicted, \( r_{xi}^* \) is the true rating of \( x \) on \( i \)

- **Rank Correlation:**
  - Spearman’s *correlation* between system’s and user’s complete rankings

SLIDES ADAPTED FROM JURE LESKOVEC
Problems with Error Measures

Narrow focus on accuracy sometimes misses the point
  - Prediction Diversity
  - Prediction Context
  - Order of predictions

In practice, we care only to predict high ratings:
  - RMSE might penalize a method that does well for high ratings and badly for others
There’s No Data like Mo’ Data

Leverage all the data
  ◦ Simple methods on large data do best

Add more data
  ◦ e.g., add IMDB data on genres

More data beats better algorithms
Latent Factor Models (like SVD)

- The Color Purple
- The Princess Diaries
- The Lion King
- Ocean’s 11
- Braveheart
- Lethal Weapon
- Dumb and Dumber

- Serious
- Escapist
- Geared towards females
- Geared towards males

SLIDES ADAPTED FROM JURE LESKOVEC
Famous Historical Example: The Netflix Prize

**Training data**
- 100 million ratings, 480,000 users, 17,770 movies
- 6 years of data: 2000-2005

**Test data**
- Last few ratings of each user (2.8 million)
- Evaluation criterion: root mean squared error (RMSE)
- Netflix Cinematch RMSE: 0.9514

**Competition**
- 2700+ teams
- $1 million prize for 10% improvement on Cinematch
- BellKor system won in 2009. Combined many factors
  - Overall deviations of users/movies
  - Regional effects
  - Local collaborative filtering patterns
  - Temporal biases
Summary on Recommendation Systems

• The Long Tail
• Content-based Systems
• Collaborative Filtering
• Latent Factors