Group Recommendation: Semantics and Efficiency
Another well-known Example (IMDb)
Recommendations: Application Field

- Sales
  (recommend items that the user might be interested to buy)

- Entertainment
  (recommend a good movie or a good restaurant)
Motivation for group recommendations

- In most cases: recommendations for a single user

- But: for entertainment purposes the users often form groups (movies, restaurants)

- How can we create good recommendations for a group of users?
Outline

- Recommendation for individuals
- Group recommendation
- Experiments
- Conclusion
Outline

- Recommendation for individuals
- Group recommendation
- Experiments
- Conclusion
Individual Recommendation: Item based

- In order to estimate the user’s opinion about a new item $i$, use items $i'$ similar to the user’s previously highly rated items.

\[
\text{relevance}(u, i) = \sum_{i' \in I} \text{ItemSim}(i, i') \times \text{rating}(u, i')
\]

- $I$: set of all items
- $u$: current user
- $\text{ItemSim}(i, i')$: How similar $i, i'$ are (e.g., cosine similarity)
Individual Recommendation: User based (Collaborative filtering)

- In order to estimate the user’s opinion about a new item, use the opinion of people who share the user’s interests, for this item.

\[
\text{relevance}(u,i) = \sum_{u' \in U} \text{UserSim}(u,u') \times \text{rating}(u',i)
\]

- U: set of users
- UserSim(u,u'): How similar u, u' are
Outline

- Recommendation for individuals
- Group recommendation
- Experiments
- Conclusion
Group Recommendation

- Basic approaches
- The consensus function
- The TA Algorithm
- The monotonicity issue
- Full and Partial Materialization Algorithms
- Sharpening the Thresholds
Basic Approaches

- Preference Aggregation
  group members’ prior ratings are aggregated into a single virtual user profile. Recommendations are made for the virtual user.

- Score Aggregation
  each member’s individual recommendations are computed and merged into a single list for the group, where the score of each item is aggregated from individual recommendations.
### Preference Aggregation (Example)

<table>
<thead>
<tr>
<th>Known Movie</th>
<th>Mary</th>
<th>Chris</th>
<th>Helen</th>
<th>Virtual User</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="lion_king.jpg" alt="Image" /></td>
<td>0.7</td>
<td>0.2</td>
<td>0.5</td>
<td>0.46</td>
</tr>
<tr>
<td><img src="cinderella.jpg" alt="Image" /></td>
<td>0.6</td>
<td>0.5</td>
<td>0.6</td>
<td>0.56</td>
</tr>
<tr>
<td><img src="pocahontas.jpg" alt="Image" /></td>
<td>0.3</td>
<td>0.7</td>
<td>0.9</td>
<td>0.63</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unseen Movies</th>
<th>Virtual User</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="lord_of_the_rings.jpg" alt="Image" /></td>
<td>0.7</td>
</tr>
<tr>
<td><img src="mulan.jpg" alt="Image" /></td>
<td>0.9</td>
</tr>
<tr>
<td><img src="spiderman.jpg" alt="Image" /></td>
<td>0.3</td>
</tr>
</tbody>
</table>
Score Aggregation Functions

- **Average**
  
  maximize the average of group members’ scores for an item.
  But: ignores “veto” votes

- **Least Misery**
  
  maximize the lowest score among all group members.
  But: misses the item that is liked by all members except one
Score Aggregation (Example)

<table>
<thead>
<tr>
<th></th>
<th>Mary</th>
<th>Chris</th>
<th>Helen</th>
<th>Avg</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.66</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>0.8</td>
<td>0.4</td>
<td>0.53</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>0.2</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
</tr>
</tbody>
</table>
Consensus Function (1)

Better:

- For each item calculate two components:
  - Relevance $rel(G, i)$
  - Disagreement $dis(G, i)$
Consensus Function (2)

- **Group Relevance:**
  - **Average:** \( rel(G,i) = \frac{1}{|G|} \sum_{u \in G} relevance(u,i) \)
  - **Least Misery:** \( rel(G,i) = \text{Min}(relevance(u,i)) \)

- **Group Disagreement:**
  - **Average Pair-wise Disagreement:**
    \[
    dis(G,i) = \frac{2}{|G|(|G| - 1)} \sum_{(u,v) \in G} (|relevance(u,i) - relevance(v,i)|)
    \]
  - **Disagreement Variance:**
    \[
    dis(G,i) = \frac{1}{|G|} \sum_{u \in G} (relevance(u,i) - \text{mean})^2
    \]
Consensus Function (3)

- So the Consensus Function will be:

\[ F(G,i) = w_1 \times rel(G,i) + w_2 \times (1 - dis(G,i)) \]

where \( w_1 + w_2 = 1.0 \)

- But: …
**Consensus Function (example)**

<table>
<thead>
<tr>
<th></th>
<th>Mary</th>
<th>Chris</th>
<th>Helen</th>
<th>Avg</th>
<th>Δ(M,C)</th>
<th>Δ(C,H)</th>
<th>Δ(H,M)</th>
<th>ΣΔ</th>
<th>CF(0.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.66</td>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td><img src="image2.png" alt="Image" /></td>
<td>0.4</td>
<td>0.8</td>
<td>0.4</td>
<td>0.53</td>
<td>0.4</td>
<td>0.4</td>
<td>0.0</td>
<td>0.8</td>
<td>0.625</td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td>0.4</td>
<td>0.2</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
<td>0.8</td>
<td>0.56</td>
</tr>
</tbody>
</table>
Q: How to calculate the CF?
A: TA algorithm

- **Input:**
  for each user a list of items, sorted in descending score order (relevance order), together with the score for each item

- **Output:**
  the top-k items according to some aggregating function
TA algorithm

while there are items in lists
	Scan lists;
	Consider item $I$ at position $pos_i$ list $L_i$;
	high_i = s(u_i, I);
	if $I \notin top - k$ then
		look up $s_v(I)$ in all lists $L_v$ with $v \neq i$;
		score(I) = aggr\{s_v(I), v = 1...m\};
	if score(I) > min - k then
		add $I$ to top - $k$ and remove min - score $I$;

threshold = aggr\{high_v, v = 1...m\};
	if threshold $\leq$ min - $k$ then exit;
TA in action (k=2, aggr=Sum) - in general

- Depth=1
TA in action (k=2, aggr=Sum) - in general

- Depth=1

<table>
<thead>
<tr>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>I78: 0.9</td>
<td>I64: 0.9</td>
<td>I10: 0.7</td>
<td>2.5</td>
</tr>
<tr>
<td>I23: 0.8</td>
<td>I23: 0.6</td>
<td>I78: 0.5</td>
<td></td>
</tr>
<tr>
<td>I10: 0.8</td>
<td>I10: 0.6</td>
<td>I64: 0.3</td>
<td></td>
</tr>
<tr>
<td>I1: 0.7</td>
<td>I12: 0.2</td>
<td>I99: 0.2</td>
<td></td>
</tr>
<tr>
<td>I88: 0.2</td>
<td>I78: 0.1</td>
<td>I34: 0.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I78</td>
<td>1.5</td>
</tr>
</tbody>
</table>
TA in action (k=2, aggr=Sum)- in general

- Depth=1

<table>
<thead>
<tr>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>I78: 0.9</td>
<td>I64: 0.9</td>
<td>I10: 0.7</td>
<td>2.5</td>
</tr>
<tr>
<td>I23: 0.8</td>
<td>I23: 0.6</td>
<td>I78: 0.5</td>
<td></td>
</tr>
<tr>
<td>I10: 0.8</td>
<td>I10: 0.6</td>
<td>I64: 0.3</td>
<td></td>
</tr>
<tr>
<td>I1: 0.7</td>
<td>I12: 0.2</td>
<td>I99: 0.2</td>
<td></td>
</tr>
<tr>
<td>I88: 0.2</td>
<td>I78: 0.1</td>
<td>I34: 0.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I78</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>I64</td>
<td>1.2</td>
</tr>
</tbody>
</table>
TA in action (k=2, aggr=Sum)- in general

- Depth=1

<table>
<thead>
<tr>
<th></th>
<th>U1</th>
<th></th>
<th>U2</th>
<th></th>
<th>U3</th>
<th></th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I78:0.9</td>
<td></td>
<td>I64:0.9</td>
<td></td>
<td>I10:0.7</td>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>I23:0.8</td>
<td></td>
<td>I23:0.6</td>
<td></td>
<td>I78:0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I10:0.8</td>
<td></td>
<td>I10:0.6</td>
<td></td>
<td>I64:0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I1 :0.7</td>
<td></td>
<td>I12:0.2</td>
<td></td>
<td>I99:0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I88:0.2</td>
<td></td>
<td>I78:0.1</td>
<td></td>
<td>I34:0.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I10</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>I78</td>
<td>1.5</td>
</tr>
</tbody>
</table>
TA in action (k=2, aggr=Sum) - in general

- Depth=2

<table>
<thead>
<tr>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>I78: 0.9</td>
<td>I64: 0.9</td>
<td>I10: 0.7</td>
<td>2.5</td>
</tr>
<tr>
<td>I23: 0.8</td>
<td>I23: 0.6</td>
<td>I78: 0.5</td>
<td>1.9</td>
</tr>
<tr>
<td>I10: 0.8</td>
<td>I10: 0.6</td>
<td>I64: 0.3</td>
<td></td>
</tr>
<tr>
<td>I1: 0.7</td>
<td>I12: 0.2</td>
<td>I99: 0.2</td>
<td></td>
</tr>
<tr>
<td>I88: 0.2</td>
<td>I78: 0.1</td>
<td>I34: 0.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I10</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>I78</td>
<td>1.5</td>
</tr>
</tbody>
</table>
TA in action (k=2, aggr=Sum)- in general

- Depth=3

<table>
<thead>
<tr>
<th></th>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>178:0.9</td>
<td>164:0.9</td>
<td>110:0.7</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>123:0.8</td>
<td>123:0.6</td>
<td>178:0.5</td>
<td>1.9</td>
</tr>
<tr>
<td>3</td>
<td>110:0.8</td>
<td>110:0.6</td>
<td>164:0.3</td>
<td>1.7</td>
</tr>
<tr>
<td>4</td>
<td>11:0.7</td>
<td>112:0.2</td>
<td>199:0.2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>188:0.2</td>
<td>178:0.1</td>
<td>134:0.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>110</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>178</td>
<td>1.5</td>
</tr>
</tbody>
</table>
TA in action (k=2, aggr=Sum)- in general

- Depth=4 (Score(I78)<Threshold: STOP)

<table>
<thead>
<tr>
<th></th>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I78: 0.9</td>
<td>I64: 0.9</td>
<td>I10: 0.7</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>I23: 0.8</td>
<td>I23: 0.6</td>
<td>I78: 0.5</td>
<td>1.9</td>
</tr>
<tr>
<td></td>
<td>I10: 0.8</td>
<td>I10: 0.6</td>
<td>I64: 0.3</td>
<td>1.7</td>
</tr>
<tr>
<td></td>
<td>I1: 0.7</td>
<td>I12: 0.2</td>
<td>I99: 0.2</td>
<td>1.1</td>
</tr>
<tr>
<td></td>
<td>I88: 0.2</td>
<td>I78: 0.1</td>
<td>I34: 0.1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>ItemID</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I10</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>I78</td>
<td>1.5</td>
</tr>
</tbody>
</table>
The monotonicity issue (1)

- The correct early stopping of TA Algorithm is possible only when the aggregating function is monotone.
- The consensus function is comprised by two components: group relevance and group disagreement. Both of them should be monotone.
- Relevance (average, min) is monotone.
- Disagreement: ?
The monotonicity issue (2)

- Group Disagreement is not monotonic w.r.t. relevance lists

\[
dis(G,i) = \frac{2}{|G|(|G|-1)} \sum_{(u,v) \in G} |\text{relevance}(u,i) - \text{relevance}(v,i)|
\]
The monotonicity issue (2)

- Group Disagreement is not monotonic w.r.t. relevance lists

- Solution: maintain also pairwise disagreement lists

\[
dis(G, i) = \frac{2}{|G|(|G|-1)} \sum_{(u,v) \in G} (|\text{relevance}(u,i) - \text{relevance}(v,i)|)
\]
TA for Group Recommendation

- Relevance list for each user, with items sorted in descending order
- Disagreement lists for each pair of users, with items sorted in ascending order
- Aggregation Function = Consensus Function

<table>
<thead>
<tr>
<th>Rel(u1,i)</th>
<th>Rel(u2,i)</th>
<th>Dis(u1,u2,i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1: 0.5</td>
<td>I2: 0.5</td>
<td>I2: 0.0</td>
</tr>
<tr>
<td>I2: 0.5</td>
<td>I4: 0.4</td>
<td>I4: 0.1</td>
</tr>
<tr>
<td>I3: 0.4</td>
<td>I1: 0.3</td>
<td>I5: 0.1</td>
</tr>
<tr>
<td>I4: 0.3</td>
<td>I6: 0.3</td>
<td>I6: 0.1</td>
</tr>
<tr>
<td>I5: 0.3</td>
<td>I7: 0.3</td>
<td>I1: 0.2</td>
</tr>
<tr>
<td>I6: 0.2</td>
<td>I3: 0.2</td>
<td>I3: 0.2</td>
</tr>
<tr>
<td>I7: 0.1</td>
<td>I5: 0.2</td>
<td>I7: 0.2</td>
</tr>
</tbody>
</table>
Group Recommendation Algorithm with Fully Materialized Disagreement Lists

Require : Group G, consensus function F;
1: Retrieve relevance lists $IL_u$ for each user $u$ in group G;
2: Retrieve disagreement lists $DL_{(u,v)}$ for each user pair $(u, v)$ in group G;
3: Cursor $cur = getNext()$ moves across each relevance and disagreement lists;
4: while ($cur \neq$ NULL) do
5: Get entry $e = (i, r)$ at $cur$;
6: if not (inHeap(topKHeap, e)) then
7: if $\text{ComputeMaxScore}(e.i, e.r, F) \geq \text{topKHeap.kthscore}$ then
8: ComputeExactScore; Probe ILs to compute exact score of $e$ using $F$;
9: topKHeap.addToHeap(e.i, score);
10: else break;
11: end if
12: end if
12: $cur = getNext();$
14: end while
15: return topKList(topKHeap);
FM Algorithm: ComputeExactScore

- ComputeExactScore: random access (RA) on all other relevance lists to compute the score of an item $i$, using the input consensus function $F$.

$$F(G, i) = w_1 \times \frac{1}{|G|} \sum_{u \in G} r_u + w_2 \times (1 - \frac{2}{|G|(|G| - 1)} \sum_{(u,v) \in G} \Delta_{u,v})$$

- DLs are not necessary to compute the final result (disagreement can be computed from relevances). They are only used to compute the threshold (using ComputeMaxScore) and hopefully, enable early termination.
FM Algorithm: ComputeMaxScore

- ComputeMaxScore produces a new threshold value at each round, in order to provide an upper bound for the score of any item that has not yet been seen by the algorithm:

if \( r_u \) is the last relevance value, read on list \( IL_u \) for all \( u \in G \), and \( \Delta_{u,v} \) the last pairwise disagreement value, read on disagreement list \( DL_{u,v} \) for all \( u, v \in G \):

\[
F(G,i) \leq w_1 \times \frac{1}{|G|} \sum_{u \in G} r_u + w_2 \times (1 - \frac{2}{|G|(|G|-1)} \sum_{u,v \in G} \Delta_{u,v})
\]
RO Algorithm (Relevance lists Only)

- None of DLs available
- Consume less space
- No impact on ComputeExactScore
- But ComputeMaxScore: less tight threshold

\[ F(G, i) \leq w_1 \times \frac{1}{|G|} \sum_{u \in G} r_u + w_2 \]
PM Algorithm (Partial Materialization)

- Only some of the DLs are materialized
- Less space consumption than FM
- Let $M$ be the set of all pairs of users for which disagreement lists have been materialized. Threshold:

$$F(G, i) \leq w_1 \times \frac{1}{|G|} \sum_{u \in G} r_u + w_2 \times (1 - \frac{2}{|G|(|G| - 1)} \sum_{(u, v) \in M} \Delta_{u,v})$$

- But…
More DLs $\Rightarrow$ Faster Algorithm?

- If none of the top items in a DL are in the end in the top-k, each SA on this DL is pure overhead.

- A DL is providing the chance to tighten the threshold only if there is some skew in its items. DLs for users that have very similar or dissimilar relevances for all items, are not useful.

- Which DLs should be materialized as a preprocessing step?
DL Materialization

- Assume only groups of pairs of users
- Assume that we know the distribution $p(G)$, i.e., the probability that a given user group $G$ will be queried next (query log? New users?) and prune the most unlikely groups
- Run top-k algorithm both for FM and RO and calculate the times $t_{DL_{(u,v)}}(\{u,v\})$ and $t_{\emptyset}(\{u,v\})$
- Eliminate groups for which $t_{DL_{(u,v)}}(\{u,v\}) \geq t_{\emptyset}(\{u,v\})$
- Select the set $M$ of user pairs, so that it will maximize:
  \[
  \sum_{(u,v)\in M} p(\{u,v\}) \cdot (t_{\emptyset}(\{u,v\}) - t_{DL_{(u,v)}}(\{u,v\}))
  \]
- Once the DLs are materialized, they can be used at query processing time for bigger groups
DLs: A Chance for Sharpening the Thresholds

<table>
<thead>
<tr>
<th>Rel(u)</th>
<th>Rel(v)</th>
<th>Dis(u,v)</th>
</tr>
</thead>
<tbody>
<tr>
<td>i_u</td>
<td>i_v</td>
<td></td>
</tr>
<tr>
<td>i1: 0.5</td>
<td>i2: 0.5</td>
<td>i3: 0.2</td>
</tr>
</tbody>
</table>

- Thresholds from each list:
  - $0 \leq i_u \leq 0.5$
  - $0 \leq i_v \leq 0.5$
  - $0.2 \leq |i_u - i_v| \leq 1$

- ComputeMaxScore: $F(G,i) \leq (0.5 + 0.5) / 2 + (1 - 0.2 / 1) = 1.3$

- A tighter threshold: $F(G,i) \leq (0.5 + 0.3) / 2 + (1 - 0.2 / 1) = 1.2$

- But…
Outline

- Recommendation for individuals
- Group recommendation
- Experiments
- Conclusion
Experiments for quality evaluation

- Data Set: MovieLens 10M ratings data set (71,567 users, 10,681 movies, 10,000,054 ratings, scale rating: 0-5)
- Relevance (individual recommendation): Collaborative filtering
- Similarity measure:

\[
\text{sim}(u,u') = \frac{\left| \{ i, \ i \in I_u \land i \in I_{u'}, \ |\text{rating}(u,i) - \text{rating}(u',i)| \leq 2 \} \right|}{\left| \{ i, \ i \in I_u \lor i \in I_{u'} \} \right|}
\]
Recommendation mechanisms examined

- Average Relevance Only (AR)
- Least-Misery Relevance Only (MO)
- Consensus with Pairwise Disagreement (RP)
  - RP20 where disagreement weight=0.2
  - RP80 where disagreement weight=0.8
- Consensus with Disagreement Variance (RV)
  - RV20 where disagreement weight=0.2
  - RV80 where disagreement weight=0.8
User Collection Phase - Preferences Collection

- Recruit users from Amazon Mechanical Turk
- Create a *Popular Set* of movies (top-40 in terms of popularity) and a *Diversity Set* (top-20 in terms of user rating variance, but also within the top-200 in terms of popularity)
- Create 2 HITs, each with 40 movies and 50 users:
  - Similar HIT (40 movies from Popular Set)
  - Dissimilar HIT (20 movies from Popular and 20 from Diversity)
User Collection Phase – Group Formation

- Vary Group size: Small (3 persons) and large (8 persons) groups
- Vary Group Cohesiveness:
  - Similar: Users from Similar HIT and maximum summation of pairwise similarities
  - Dissimilar: Users from Dissimilar HIT and minimum summation of pairwise similarities
  - Random
Group Judgement Phase

- For each group generate group recommendations
- For each member of group generate individual recommendations with relevance values for each movie
- Create Group HIT: given, for each group member, the relevance value of each movie in the Group Recommendations Set, decide if the movie is a good recommendation
<table>
<thead>
<tr>
<th>Unseen movies</th>
<th>Estim Rating for User 1</th>
<th>Estim Rating for User 2</th>
<th>Estim Rating for User 3</th>
<th>Will you recommend it?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group judgement phase (example)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="Unseen movies" /></td>
<td>0.8</td>
<td>0.7</td>
<td>0.7</td>
<td>1</td>
</tr>
<tr>
<td><img src="image" alt="Unseen movies" /></td>
<td>0.8</td>
<td>0.7</td>
<td>0.3</td>
<td>1</td>
</tr>
<tr>
<td><img src="image" alt="Unseen movies" /></td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>1</td>
</tr>
<tr>
<td><img src="image" alt="Unseen movies" /></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
The Quality Measure

- To evaluate the recommendation strategies, we use the Discounted Cumulative Gain (DCG)
- For a 10-movie recommendation list:
  \[
  DCG_{10} = rel_1 + \sum_{i=2}^{10} \frac{rel_i}{\log_2(i)}
  \]
  
  If item at position \( i \) is a good recommendation, then \( rel_i = 1 \) else \( rel_i = 0 \)

- Normalize:
  \[
  NDCG_{10} = \frac{DCG_{10}}{DCG_{10}(perfect\ result)}
  \]
Quality Results (1)
Quality Results (2)
Quality Results (3)
Performance Results

- m=3
- Group Size=5
- 10 recommended items
Outline

- Recommendation for individuals
- Group recommendation
- Experiments
- Conclusion
Conclusion

- An approach that gives better results for large, dissimilar groups
- Tuning Problem (weights, m)
- PM: materializing in preprocessing time can be very time consuming (new ratings?)
- PM: depends on knowing the p(G) distribution
- Threshold Sharpening: nice, but a difficult optimization problem
- PM vs Threshold Sharpening
Thank you for your Attention

Questions?