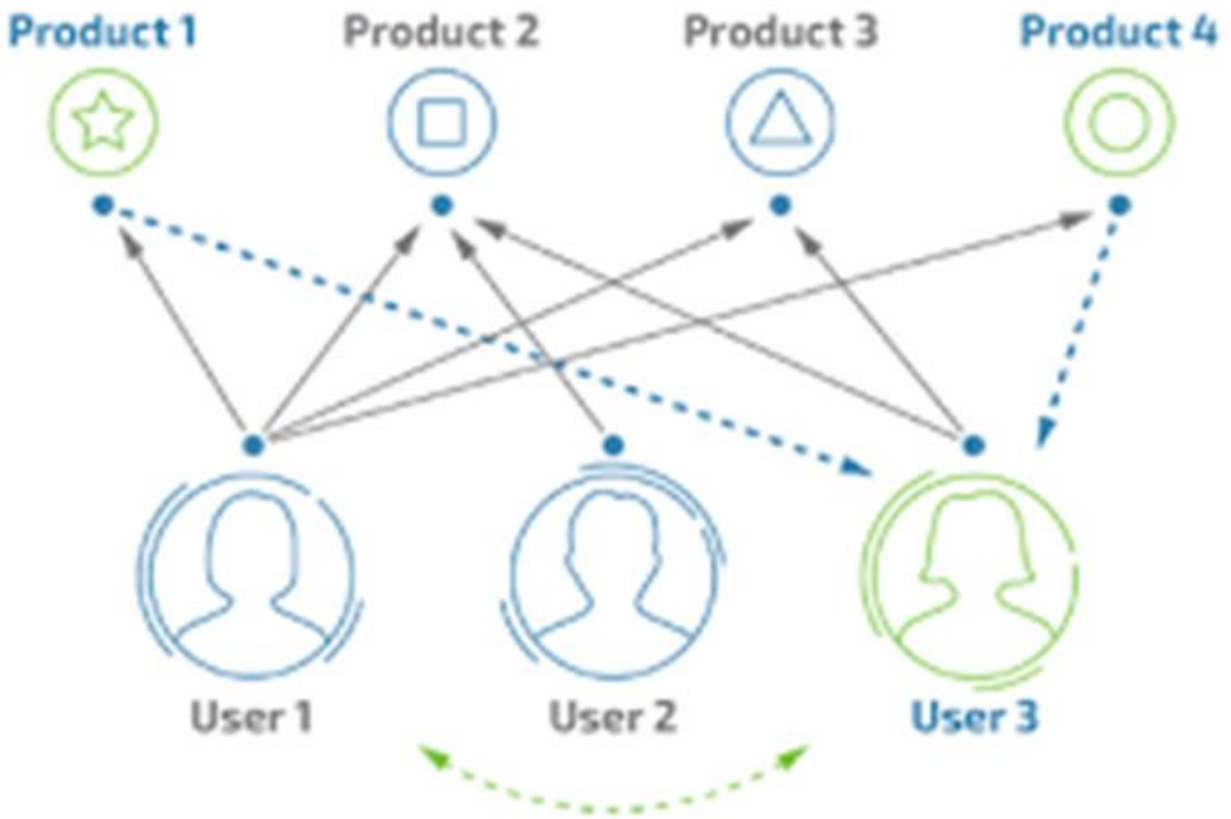
Collaborative Recommendation (Part A)

Topic 4A

Collaborative Recommendation

- The main idea is to exploit information about the past behavior or the opinions of an existing user community for predicting which items the current user of the system will most probably like or be interested in.
- "Birds of the same feather flock together": A collaborative recommendation algorithm usually works by searching a large group of people and finding a smaller set with tastes similar to the target user. It looks at other things they like and combines them to create a ranked list of suggestions.
- There are several different ways of deciding which people are similar and combining their choices to make a list.
- Pure collaborative approaches take a matrix of given user-item ratings as the only input and typically produce the following types of output: (1) a numerical prediction indicating to what degree the current user will like or dislike a certain item and (2) a list of n recommended items.
- Two alternative approaches: User-based approach vs. Item-based approach

Birds of the same feather flock together

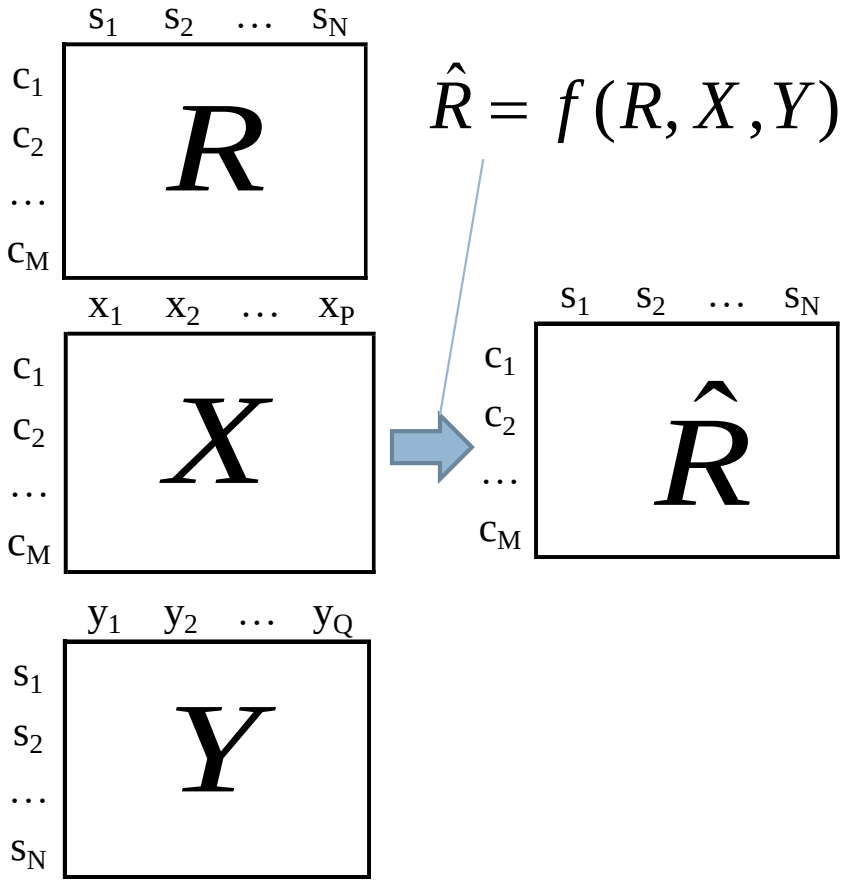


Collaborative Recommender Systems Fundamentals

- Two-dimensional: users and items
- Utility of an item to a user revealed by a *single* rating
  - Rating can be binary or multi-scaled
  - Transactions as implicit unary ratings
- Recommendations to *individual* users

Recommender System Fundamentals

- Input
  - Rating matrix *R*:
    - r_{ij}: rating user c_i assigns to item s_j
    - *R* is a *sparse* matrix
  - User attribute matrix *X*: x_{ij} – *attribute* x_j *of user* c_i
  - Item attribute matrix *Y*: y_{ij} – *attribute* y_j *of item* s_i
- Output
  - Predicted interaction matrix (predicted utility) $\hat{R}$



User-based Nearest Neighbor Recommendation

- Given a ratings database and the ID of the current user as an input, this approach identifies other users (nearest neighbors) that had similar preferences to those of the active user in the past. Then, for every product p that the active user has not yet seen, a prediction is computed based on the ratings for p made by the peer users.
- Assumptions
  - If users had similar tastes in the past, they will have similar tastes in the future.
  - User preferences remain stable and consistent over time.

First Example

- Table 2.1

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

* 5 means strongly liked on a 1-5 scale

Should we recommend Item5 to Alice?
-Compute Pearson's correlation coefficient
-Find similar users
-Make predictions for Item 5

Pearson's Correlation Coefficient

- PCC, commonly denoted by *r*, takes values from +1 (perfect positive correlation) to -1 (perfect negative correlation).
- PCC between two variables is computed as,

$$\boldsymbol{sim(a,b)} = \frac{\sum_{\boldsymbol{p}\in\boldsymbol{P}}(\boldsymbol{r_{a,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{a}})(\boldsymbol{r_{b,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{b}})}{\sqrt{\sum_{\boldsymbol{p}\in\boldsymbol{P}}(\boldsymbol{r_{a,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{a}})^{\boldsymbol{2}}}\sqrt{\sum_{\boldsymbol{p}\in\boldsymbol{P}}(\boldsymbol{r_{b,p}} - \bar{\boldsymbol{r}}_{\boldsymbol{b}})^{\boldsymbol{2}}}}$$

a, b : users

$r_{a,p}$: rating of user a for item p

P : set of items, rated both by a and b

First Example - PCC

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- PCC between Alice & User1

$$\frac{(5-\overline{r_a})*(3-\overline{r_b})+(3-\overline{r_a})*(1-\overline{r_b})+\cdots+(4-\overline{r_a})*(3-\overline{r_b}))}{\sqrt{(5-\overline{r_a})^2+(3-\overline{r_a})^2+\cdots}\sqrt{(3-\overline{r_b})^2+(1-\overline{r_b})^2+\cdots}}=0.85$$

Predicting the Ratings

- Determine nearest neighbors based on the correlation coefficients
- Make a prediction using a formula such as

$$pred(a, p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a, b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a, b)} \qquad (2.3)$$

- Select the items with the highest prediction values in the recommendation list.

First Example - Prediction

- Predicted Rating for Alice - Item 5
  - Assuming that the nearest neighbors are User 1 (Similarity 0.85), User 2 (Similarity 0.7),

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

$$pred(a, p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a, b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a, b)}$$

$$= 4 + 1/(0.85 + 0.7) * (0.85 * (3 - 2.4) + 0.70 * (5 - 3.8)) = 4.87$$

Unresolved Issues

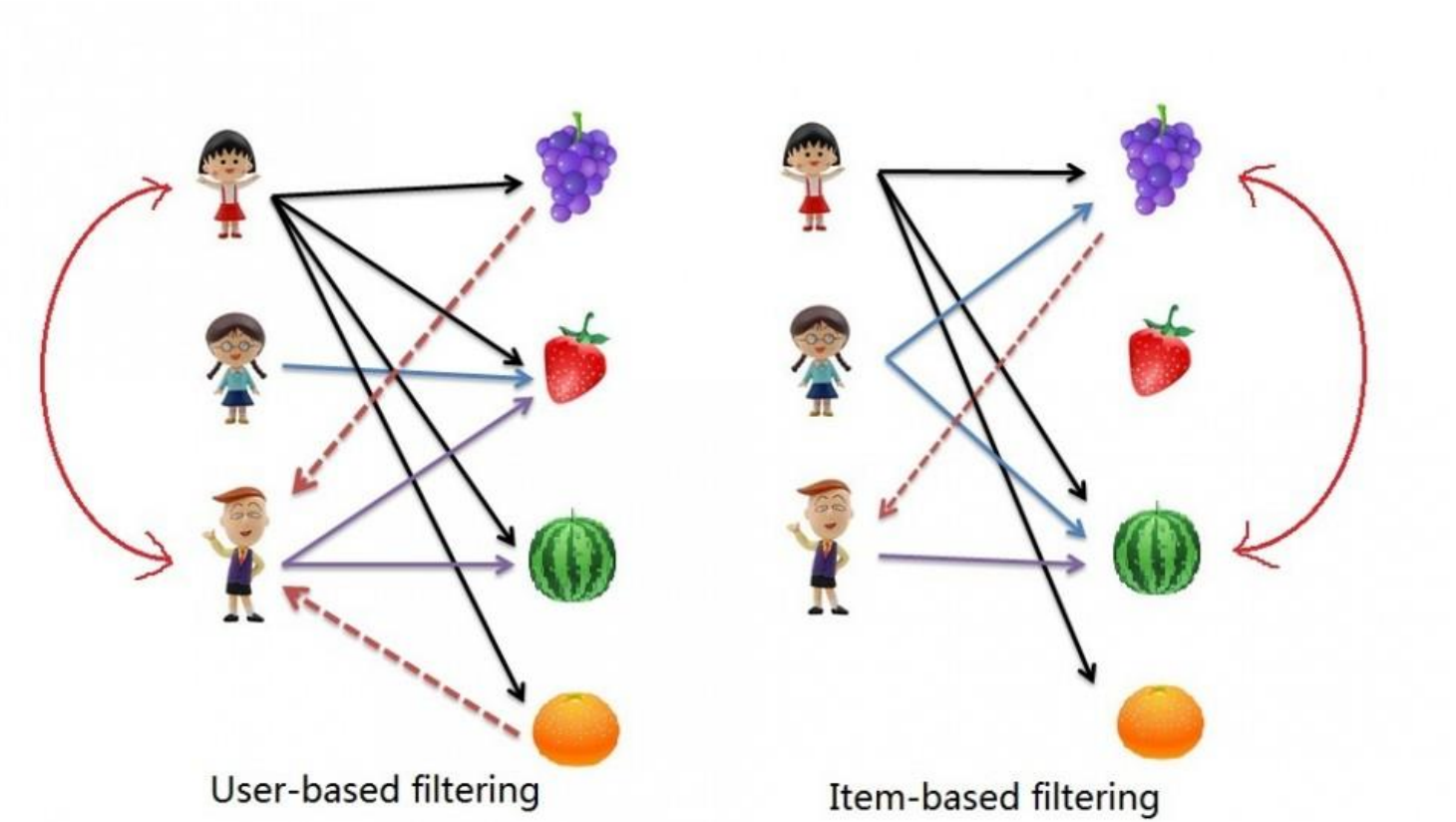
- Computational complexity
- Sparsity of rating matrix
- Alternative similarity measures
  - The Pearson coefficient outperforms other measures of comparing users
  - For item-based recommendation techniques, the cosine similarity measure outperforms the Pearson correlation metric.

Item-based Nearest Neighbor Recommendation

- User-based CF approaches are difficult to use in large-scale e-commerce sites.
- Large e-commerce sites often implement item-based recommendation, which is more apt for offline preprocessing.
- The main idea is to compute predictions using the similarity between items and not the similarity between users.

Item-based Nearest Neighbor Recommendation

- User-based approach vs. Item-based approach



Cosine Similarity Measure

- It has been shown that cosine similarity produces the most accurate results for item-based recommendation.
- Cosine similarity is a measure of similarity between two vectors by measuring the cosine of the angle between them (the result of the Cosine function is equal to 1 when the angle is 0, and less than 1 when the angle is of any other value).
- The similarity between two vectors A and B are computed as

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

- The basic cosine measure does not take the differences in the average rating behavior of the users into account, which can be addressed by using the adjusted cosine measure, which subtracts the user average from the ratings as follows:

$$sim(a,b) = \frac{\sum_{u \in U} (r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U} (r_{u,a} - \overline{r_u})^2} \sqrt{\sum_{u \in U} (r_{u,b} - \overline{r_u})^2}}$$

First Example – Cosine Similarity

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

$$sim(I5, I1) = \frac{3*3+5*4+4*3+1*1}{\sqrt{3^2+5^2+4^2+1^2}*\sqrt{3^2+4^2+3^2+1^2}} = 0.99$$

$$\frac{0.6*0.6+0.2*1.2+(-0.2)*0.80+(-1.8)*(-1.8)}{\sqrt{(0.6^2+0.2^2+(-0.2)^2+(-1.8)^2}*\sqrt{0.6^2+1.2^2+0.8^2+(-1.8)^2}} = 0.80$$

Predicting the Ratings

- After the similarities between the items are determined, we can predict the rating for user *u* for a product *p* as follows:

$$pred(u, p) = \frac{\sum_{i \in ratedItems(u)} sim(i, p) * r_{u,i}}{\sum_{i \in ratedItems(u)} sim(i, p)} \qquad (2.9)$$

Preprocessing data

- To make item-based recommendation algorithms applicable for large scale e-commerce sites, the item similarity matrix that describes the pairwise similarity of all catalog items can be precomputed offline.
- At run time, a prediction for a product p and user u is made by determining the items that are most similar to I and by building the weighted sum of u's ratings for these items in the neighborhood.
- The number of neighbors to be taken into account is limited to the number of items that the active user has rated, which is relatively small.
- In principle, such an offline precomputation is also possible for user-based approaches. Still, in practical scenarios the number of overlapping ratings for two users is relatively small, which means that a few additional ratings may quickly influence the similarity value between users.
- The item similarities are much more stable.

Ratings

- A rating is a single score that reveal the utility of an item to a user.
- It can be explicit or implicit.
- Ratings can be binary or multi-scaled.
- Explicit ratings require additional efforts from the users.
- Implicit ratings can be generated when a customer buys an item or browses an item.
- Implicit ratings are commonly less accurate.

Data Sparsity and the Cold-Start Problem

- In real-world applications, the rating matrices tend to be very sparse.
  - Utilizing demographic information about the users as well to find similar users
  - Combining user similarities and item similarities
- The cold-start problem can be viewed as a special case of the sparsity problem.
  - Utilize hybrid approaches
  - Ask for a gauge set of user ratings

Memory-based vs. Model-based

- The traditional user-based technique is memory-based because the original rating database is held in memory and used directly for generating the recommendations.
- In model-based techniques, the raw data are first processed offline and only the precomputed or "learned" model is required to make predictions.
- Memory-based approaches are more precise, but face scalability problems.

Model-Based Approaches

- Matrix factorization/latent factor models
- Association rule mining
- Probabilistic recommendation approaches