UNIVERSITÀ DEL SALENTO

Recommendation Systems

Recommendation Systems

Recommendation System regards **applications** that involve predicting user *responses to options*

Examples:

- Recommending news articles
- Product suggestions on online retailers

Types:

- **Content-based**: examine properties of the items recommended
- **Collaborative filtering**: systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users

Applications

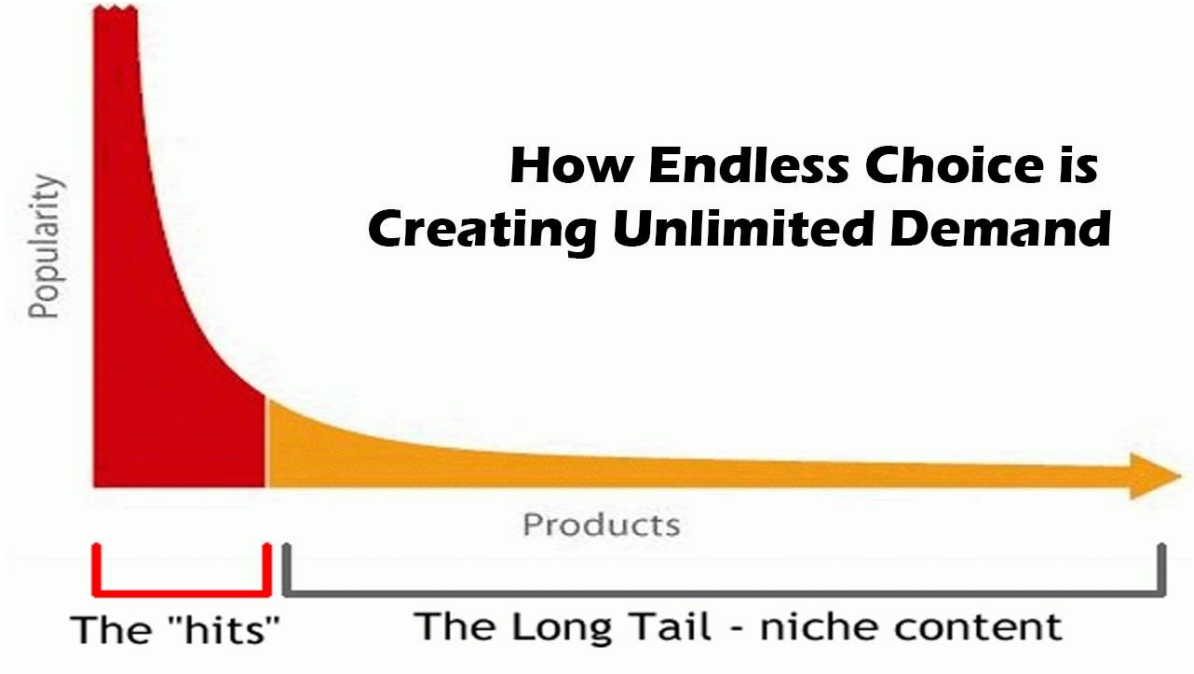
- **Product recommendations**: The most important use of recommendation systems is at online retailers (e.g., Amazon).
  - Vendors strive to present each returning user with some suggestions for products.
  - Suggestions are based on the purchasing decisions made by similar customers (or other techniques)
- **Movie recommendations: online** platforms offers their customers recommendations of movies they might like. These recommendations are based on ratings provided by users, (e.g., Netflix Prize)
- **News Articles recommendations**: News services have attempted to identify articles of interest to readers, based on the articles that they have read in the past.

The Long Tail Phenomenon

The distinction between the **physical** and **online** worlds has been called the long tail phenomenon

- **Online** vendors can offer everything
- **Physical** stores: only most popular items
- ***Long tail*** = niche items with low individual demand but high aggregate demand

The long-tail phenomenon forces online institutions to **recommend items** to **individual users**



Utility Matrix: model for Recommendation Systems

Users vs. Items: In a recommendation-system application, there are *two classes of entities*, which we shall refer to as **users** and **items**.

Utility Matrix: gives for each user-item pair, a value that represents what is known about the degree of **preference** of that user for that item.

Rows = users, Columns = items

Goal: predict missing entries (user preferences)

- it is not necessary to predict every blank entry in a utility matrix.
- it is **only necessary** to discover some entries in each row that are **likely to be high**

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Populating the Utility Matrix

A **utility matrix** is essential for recommendation systems, but **gathering data is challenging**.

Explicit Feedback

- Ask users to rate items (e.g., movies, news, videos).
- Limitations: low participation and potential bias.

Implicit Feedback

- Infer preferences from user behavior (e.g., purchases, views).
- Binary feedback: 1 = like, 0 = no data (not a negative rating).
- Even browsing activity can indicate interest.

Content-Based Systems

Content-based recommendations suggests to user U the items similar to the ones U has already rated in a positive way.

To each **item,** it associates an **item profile** (record or collection of records representing the item)

To each **user,** it associates a **user profile** (derived by the utility matrix)

		[0 1 0 1 1 0 1]	[0 1 1 0 1 0 1]	[0 0 0 1 0 0 1]	[1 1 0 1 1 0 1]	[1 1 0 0 1 1 1]	[0 1 0 1 1 0 1]	[0 1 0 1 1 0 1]
		HP1	HP2	HP3	TW	SW1	SW2	SW3
[0 0.2 0.4 0.21 0.4 0.1 0.7]	A	4			5	1		
[0 0.5 0.5 0.33 0.4 0.1 0.7]	B	5	5	4				
[0 0.2 0.4 0.1 0.7 0.9 0.1]	C				2	4	5	
[0 0.1 0.4 0.21 0.4 0.1 0.7]	D		3					3

Content-Based Systems: Item Profiles

An **item profile** is a record or collection of records representing important characteristics of that item

Example: features of a movie that might be relevant to a recommendation system

- The **set of actors** of the movie. Some viewers prefer movies with their favorite actors.
- The **director**. Some viewers have a preference for the work of certain directors.
- The **year** in which the movie was made. Some viewers prefer old movies; others watch only the latest releases.
- The **genre** or **general** type of movie. Some viewers like only comedies, others dramas or romances.

Item profile: a vector of actors, then imagine that there is a component for each actor, with 1 if the actor is in the movie, and 0 if not

Example: features of a document that might be relevant to a recommendation system

- Words that characterize the documents (TF-IDF)

Item profile: a vector of 0's and 1's, where a 1 represented the occurrence of a high-TF.IDF word in the document

Content-Based Systems: Item Profiles

Example: Suppose the only features of movies are the set of actors and the average rating. Consider two movies with five actors each.

- Consider two movies with five actors each
- Two of the actors are in both movies.
- One movie has an average rating of 3 and the other an average of 4.

$$\begin{array}{ccccccccc} 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 3\alpha \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 4\alpha \end{array}$$

Since cosine distance of vectors is not affected by components in which both vectors have 0, we need not worry about the effect of actors that are in neither movie.

The last component shown represents the average rating. We have shown it as having an unknown scaling factor α.

Cosine between vectors = $$\frac{2+12\alpha^2}{\sqrt{25+125\alpha^2+144\alpha^4}}$$

α =1 → Cosine = 0.816

α =2 → Cosine = 0.940

We cannot tell which value of α is "right," but we see that the choice of scaling factor for numerical features affects our decision about how similar items are

Content-Based Systems: User Profiles

User profile is a **vectors** with the *same components of* **item profile** that describe the user's preferences

We have the **utility matrix** representing the connection between users and items

The best estimate is some **aggregation** of the **profiles of items** the **user likes**

- If the utility matrix has only 1's, the natural aggregate is the **average** of the components of the vectors representing the item profiles (in which the utility matrix has 1 for that user)

Example 9.3: Suppose items are movies, represented by Boolean profiles with components corresponding to actors. Also, the utility matrix has a 1 if the user has seen the movie and is blank otherwise. If 20% of the movies that user U likes have Julia Roberts as one of the actors, then the user profile for U will have 0.2 in the component for Julia Roberts. □

- If the utility matrix has a value (e.g raiting 1-5), then we can **weight the vectors** representing the **profiles of items** by the **utility value**

Example 9.4: Consider the same movie information as in Example 9.3, but now suppose the utility matrix has nonblank entries that are ratings in the 1–5 range. Suppose user U gives an average rating of 3. There are three movies with Julia Roberts as an actor, and those movies got ratings of 3, 4, and 5. Then in the user profile of U, the component for Julia Roberts will have value that is the average of $3-3$, $4-3$, and $5-3$, that is, a value of 1.

On the other hand, user V gives an average rating of 4, and has also rated three movies with Julia Roberts (it doesn't matter whether or not they are the same three movies U rated). User V gives these three movies ratings of 2, 3, and 5. The user profile for V has, in the component for Julia Roberts, the average of $2-4$, $3-4$, and $5-4$, that is, the value $-2/3$. □

User Profiles & Recommendations

We have **profile vectors** for both **users** and **items**,

We can estimate the degree to which a **user** would **prefer** an **item**

Items with low cosine distance to user profile

- Given user profile x and item profile i, estimate
  - $u(\boldsymbol{x},\boldsymbol{i}) = \cos(\boldsymbol{x},\boldsymbol{i}) = (\boldsymbol{x}\cdot\boldsymbol{i})/(||\boldsymbol{x}||\cdot||\boldsymbol{i}||)$

Collaborative Filtering

Collaborative filtering focus on the **similarity** of the **user ratings** for two items

Users are **similar** if their **vectors** are **close** according to some distance measure (e.g. Jaccard or cosine distance)

Recommendation for **user U** is made by looking at **users** that are **most similar** to **U**

Consider user U

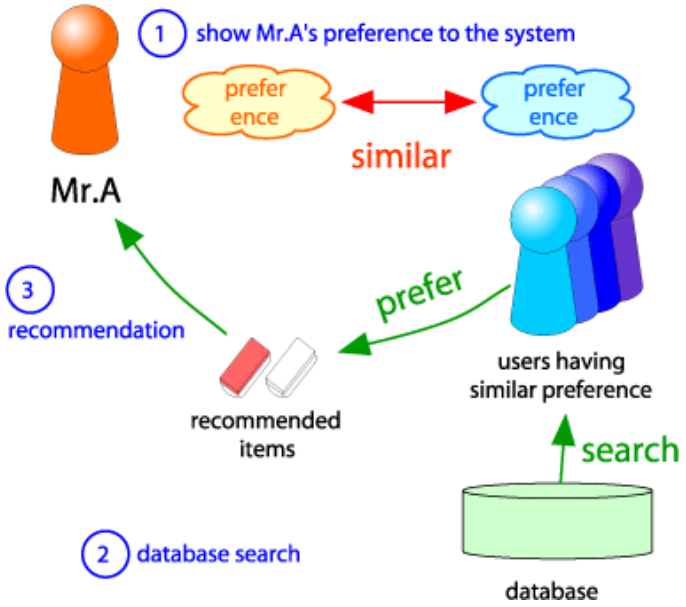
- Find set N of other **users** whose ratings are "**similar**" to U's ratings
- Estimate U's ratings based on ratings of users in N

- Use utility matrix directly

- Similarity between users or items

- Normalize ratings

- Predict based on nearest neighbors



Measuring Similarity

How to measure similarity of users or items from their rows or columns in the utility matrix ?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Jaccard Distance: ignores values in the matrix and focus only on the sets of items rated.

Cosine Distance: treats blanks as a 0 value. (it has the effect of treating the lack of a rating as more similar to disliking)

Example:

$$\text{CosDist (A,B)} = \frac{4 \times 5}{\sqrt{4^2 + 5^2 + 1^2}\sqrt{5^2 + 5^2 + 4^2}} = 0.380$$

$$\text{CosDist (A,C)} = \frac{5 \times 2 + 1 \times 4}{\sqrt{4^2 + 5^2 + 1^2}\sqrt{2^2 + 4^2 + 5^2}} = 0.322$$

Measuring Similarity

How to measure similarity of users or items from their rows or columns in the utility matrix ?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Rounding the Data: eliminate the apparent similarity between movies a user rates highly and those with low scores by rounding the ratings.

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	1			1			
B	1	1	1				
C					1	1	
D		1					1

Jaccard distance between A and B is 3/4, while between A and C it is 1; i.e., C appears further from A than B does, which is intuitively correct. Applying cosine distance to allows us to draw the same conclusion.

Measuring Similarity

How to measure similarity of users or items from their rows or columns in the utility matrix ?

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Normalizing Ratings: subtracting from each rating the average rating of that user turns:

- low ratings into negative numbers
- high ratings into positive numbers

With cosine distance:

- users with opposite views of the movies they viewed in common will have vectors opposite directions
- users with similar opinions about the movies rated in common will have a relatively small angle

Measuring Similarity

Normalizing Ratings: subtracting from each rating the average rating of that user turns:

Example:

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

- D's ratings have effectively disappeared
- cosine of the angle between A and B: $\frac{(2/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(1/3)^2+(1/3)^2+(-2/3)^2}} = 0.092$
- cosine of the angle between A and C: $\frac{(5/3)\times(-5/3)+(-7/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(-5/3)^2+(1/3)^2+(4/3)^2}} = -0.559$

The Duality of Similarity

Any of the **techniques** for **finding similar users** can be used on **columns** of the utility matrix to **find similar items**

- we can use information about users to recommend items.
- items tend to be classifiable in simple terms
- it is easier to discover items that are similar

Predicting the value of the utility-matrix entry for user U and item I:

- find the n users (for some predetermined n) most similar to U
- average their ratings for item I (only the n similar users who have rated I) better to normalize the matrix first

Dually, we can use item similarity to estimate the entry for user U and item I.

- find the m items most similar to I
- take the average rating of the ratings that U has given

The Duality of Similarity

Note that whichever approach to estimating entries in the utility matrix we use, it is not sufficient to find only one entry. In order to recommend items to a user U , we need to estimate every entry in the row of the utility matrix for U , or at least find all or most of the entries in that row that are blank but have a high estimated value. There is a tradeoff regarding whether we should work from similar users or similar items.

• If we find similar users, then we only have to do the process once for user U . From the set of similar users we can estimate all the blanks in the utility matrix for U . If we work from similar items, we have to compute similar items for almost all items, before we can estimate the row for U.

• On the other hand, item-item similarity often provides more reliable information, because of the phenomenon observed above, namely that it is easier to find items of the same genre than it is to find users that like only items of a single genre.

Whichever method we choose, we should precompute preferred items for each user, rather than waiting until we need to make a decision. Since the utility matrix evolves slowly, it is generally sufficient to compute it infrequently and assume that it remains fixed between recomputations.