# Recommendation System

* Tradition recommendation system 🡪 exploitation only
* Better recommender system is to introduce the concept of exploration
  + E.g. recommend books to you when you have been always only looking at movies
  + Maybe including sparse item to collect information on them, if not random
* Dating apps uses reciprocal recommendation system
  + Users who ‘like’ each other will rank higher in the recommendation system
* 2 classical types of recommendation methods – collaborative filtering and content-based filtering
  + Usually these two methods are applied in combination

## Collaborative filtering

* User-based collaborative filtering
  + Make use of users’ history
    - E.g. rating, review, bookmark history
  + If User A and User B seen item A-C and User B has seen item D, recommendation system will recommend item D to User A
  + Problems will arise in for a completely new website (cold start problem). Because, there is no user history data
    - How to overcome? Randomly recommend until a history is built
  + Utility matrix is used
    - Rows 🡪 users
    - Columns 🡪 items

|  |  |  |  |
| --- | --- | --- | --- |
|  | Item 1 | Item 2 | Item 3 |
| User A | 1 |  | 4 |
| User B |  | 2 |  |

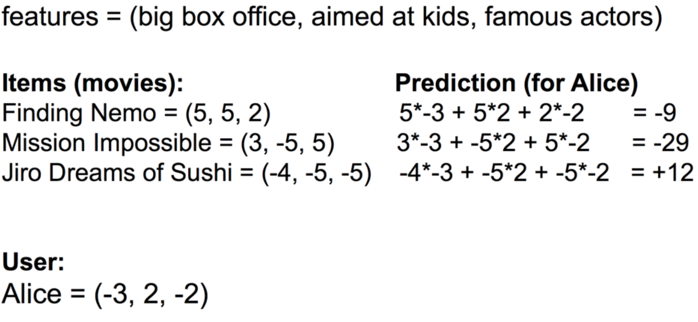
* + To find users most similar with User A, a similarity metric is used
    - Angle of 0 degree (same direction) cos0 = 1🡪 perfectly similar (User A and User B have purchased the same item and rated it similarly)
    - Angle of 90 degrees (orthogonal) cos90 = 0 🡪 perfectly dissimilar (User A and User B have not purchased the same item)
    - Angles of 180 degrees (opposite direction) cos180 = -1 🡪 means that User A and User B have purchased the same item and rated it differently)
  + By inputting 0 to fill the missing values, we have indicated strong negative sentiment for the missing ratings and thus agreement where there is none. We should instead represent that with a neutral value. We can do this by mean centering the values at zero.
    - Cal the mean rating of User A then use it to subtract from every individual rating. We then do the same for all other users, subtracting their mean ratings from each of their ratings
    - 0 will then be neutral
  + Can use cos similarity to predict unknown values
  + Problems?
    - Faced with sparsity problem 🡪 many empty boxes
      * Due to many products being available and users only purchase a few products
      * Solution? Augment with more data (social network data)
      * Clustering does not solve the problem because users in the same cluster will have similar purchasing behavior, i.e. have bought the same items and have not bought the same items
      * There is no good solution to the sparsity problem
    - Frequently-liked items will necessarily have users who like all kinds of other items. So, recommendations based on frequently-liked items may be inaccurate.
    - If user have 5 likes, adding 1 like changes total number of likes by 20%. Whereas, user that have 2000 likes, add 1 like is very minimal. The first situation is very prone to skewing
      * New websites face this problem
      * Solution? New websites ask you to choose what you prefer to view. E.g. Netflix
* Item-based collaborative filtering

|  |  |  |  |
| --- | --- | --- | --- |
|  | User A | User B | User C |
| Item 1 | 5 |  | 3 |
| Item 2 | 3 |  | 1 |
| Item 3 |  | 4 |  |

* + In item-based filtering, we are trying to find similarities across items rather than users
  + Just as in user-based filtering, we need to center our values by row

## Content-based filtering

* Items usually have metadata about them
  + E.g. author, year of release, etc.
* The recommendation system will recommend based on the metadata of items
  + E.g. User A like horror movies, recommended system will recommend you horror movies. Instead of finding another user who is similar to you, it will find a product that you like
* The items are broken down into "feature baskets". These are the characteristics that represent the item. The idea is that if you like the features of song X, then finding a song that has similar characteristics will tell us that you're likely to like it as well.
  + E.g. item 🡪 movie, features 🡪 (big box office, length of movie, famous actors, aimed at kids)



## Comparison

* Collaborative Filtering
  + Pros:
    - No need to hand craft features
  + Cons:
    - Needs a large existing set of ratings (cold-start problem)
    - Sparsity occurs when the number of items far exceeds what a person could purchase

#### Content-based Filtering

#### Pros:

#### No need for a large number of users

#### Cons:

#### Lacks serendipity

#### May be difficult to generate the right features

#### Hard to create cross-content recommendations (different feature spaces)

#### In fact, the best solution -- and the one most likely in use in any large-scale, production system is a combination of both of these. This is known as a **hybrid system**. By combining the two systems, you can get the best of both worlds