Recommender Systems

UBCF and IBCF

What are Recommender Systems?

- Systems which help the users to recommend the items for consuming (purchase – items, view – movie/picture)
- The Recommender System can be non-personalized or personalized
- Non-personalized:
  - Rating, Reviews etc.
- Personalized:
  - Content-Based, User-Based collaborative filterings etc.

Non-Personalized Recommendations

- Movies, Books, Videos etc are rated, viewed, some comments +ve/-ve are passed
- These evaluations can differ from person to person e.g. If any video is rated high, not necessarily that it will be liked by each and every user who views it
- Non-personalized evaluations get us the popularity of the content (book, video etc.)

Personalized Recommendations

- You get some recommendations relevant to your preferences and your taste
- No other user might be getting the same recommendations which you might be getting
- We can implement personalized recommendations by using:
  - Content-Based filtering
  - User-Based Collaborative filtering
  - Item-Based Collaborative filtering

Content-Based Filtering

- Content-Based filtering deals with the text analysis terms like TF-IDF
- It is a technique in which we profile each item by a list of terms and profile a user based on a list of the same terms and find the item vectors nearest to the user vector

User-Based Collaborative Filtering (UBCF)

- User-Based CF is a algorithm which tries to mimics word-of-mouth by analysing rating data from many individuals.
- The assumption is that users with similar preferences will rate items similarly.
- Thus missing ratings for a user can be predicted by first finding a neighbourhood of similar users and then aggregate the ratings of these users to form a prediction.

Elements in UBCF

- The neighborhood of any two users is defined in terms of similarity of ratings between the users
- Either we can take some k nearest neighbor users or all users within a given similarity threshold
- Pearson’s Correlation Coefficient or Cosine Similarity can be used as similarity measures

Similarity Measures

- Karl Pearson's Correlation Coefficient

$-1 \le \rho \le 1$

$$\rho_{uv} = \frac{Cov(r_{ui}, r_{vi})}{\sigma_u \sigma_v}$$

M	U_1	U_2
I	3	3
II	2	4
III	1	1
IV	5	2

- Cosine Similarity

$$Cosine_{uv} = \frac{\sum r_{ui} r_{vi}}{\sqrt{\sum r^2{}_{ui}}\sqrt{\sum r^2{}_{vi}}} = \frac{3(3)+2(4)+1(1)+5(2)}{\sqrt{3^2+2^2+1^2+5^2}\sqrt{3^2+4^2+1^2+2^2}}$$

Where $r_{ui}: ratings\ given\ by\ user\ u\ to\ item\ i$

$r_{vi}: ratings\ given\ by\ user\ v\ to\ item\ i$

Sane's STATS Academy of Statistics

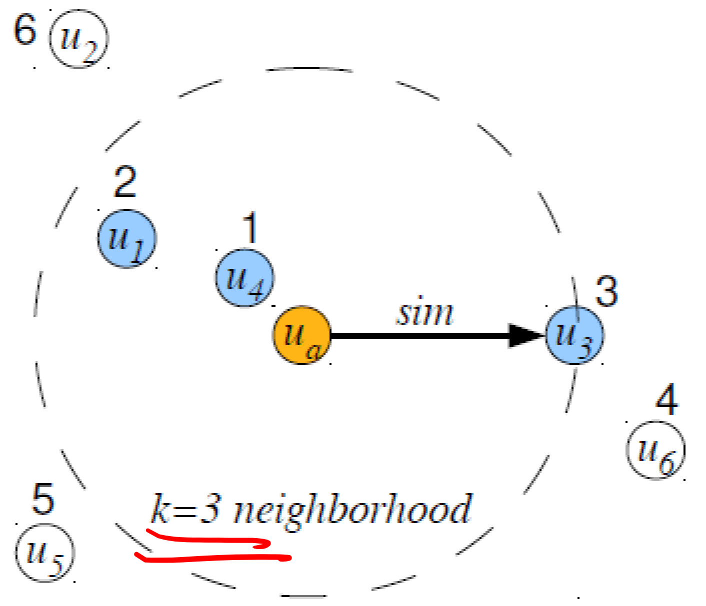
Steps in UBCF

- Given active user, find the k nearest neighbor users from the ratings matrix based on any chosen similarity measure
- Aggregate the ratings of k nearest neighbor users found to form a predicted rating for the active user. Mean of the ratings can be taken in this case

UBCF Example

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	?	4.0	4.0	2.0	1.0	2.0	?	?
u_2	3.0	?	?	?	5.0	1.0	?	?
u_3	3.0	?	?	3.0	2.0	2.0	?	3.0
u_4	4.0	?	?	2.0	1.0	1.0	2.0	4.0
u_5	1.0	1.0	?	?	?	?	?	1.0
u_6	?	1.0	?	?	1.0	1.0	?	1.0
u_a	?	?	4.0	3.0	?	1.0	?	5.0
$\hat{r}_a$	3.5	4.0			1.3		2.0	

(a)



(b)

(a) Rating matrix and estimated ratings for active User

(b) User Neighbourhood formation

Michael Hahsler (2017). recommenderlab: Lab for Developing and Testing Recommender Algorithms. R package version 0.2-2. http://lyle.smu.edu/IDA/recommenderlab/

Sane's STATS Academy of Statistics

Item Based Collaborative Filtering

- Item-based CF is a approach which produces recommendations based on the relationship between items inferred from the rating matrix.
- The assumption behind this approach is that users will prefer items that are similar to other items they like.
- The model-building step consists of calculating a similarity matrix containing all item-to-item similarities using a given similarity measure.

Elements in IBCF

- Similarity matrix is to be calculated containing item to item similarity with any chosen similarity measure
- This similarity matrix is of order $n\ x\ n$ with n as number of items
- This similarity matrix is to be reduced to $n\ x\ k$ with $k \ll n$ for each item, where k is the number of most similar items

Steps in IBCF

- Calculate the item to item similarity matrix for all the items
- Choose some k top rated associated items for each item i in the similarity matrix
- For an active user u_a , consider only those items for which its rating is available and calculate the weighted average of the ratings of the items associated with the items of the user u_a

Example of IBCF

	i1	i2	i3	i4	i5	i6	i7	i8		
i1	-	0.1	0	0.3	**0.2**	0.4	0	0.1	-	
i2	0.1	-	0.8	0.9	0	0.2	0.1	0	0.0	
i3	0	0.8	-	0	**0.4**	0.1	0.3	**0.5**	4.6	=(4*0.4+5*0.5)/(0.4+0.5)
i4	**0.3**	0.9	0	-	0	0.1	0	**0.2**	3.2	=(2*0.3+5*0.2)/(0.3+0.2)
i5	**0.2**	0	0.4	0	-	0.1	0.2	0.1	-	-
i6	**0.4**	0.2	0.1	0.3	0.1	-	0	0.1	2.0	=(2*0.4)/0.4
i7	0	0.1	0.3	0	**0.2**	0	-	0	4.0	=(4*0.2)/0.2
i8	**0.1**	0	0.5	0.2	0.1	0.1	0	-	-	
Ua	2	-	-	-	4	-	-	5		

From the above calculations, we can conclude that item i3 is best suited to be recommended for the user Ua

Need for Normalization

- Many times the ratings are known to be matter of personal bias
- e.g. It may happen that any user may be inclined to give more magnitude of rating to the item than the other.
- Consider a case in which for a scale 1-10, some users u_l have a tendency of rating the items higher (4-10) whereas users like u_m have a tendency of rating the items lower (1-7). In this case, we won't be getting the true "top" ratings just by merely averaging them

Normalization

- We can solve such problem with help of scaling or normalization
- There can be various ways of doing this:
  - Standard Scaling
  - Min Max Scaling
- This operation brings all the user ratings on a common platform

Collaborative filtering with binary data

- Many times user ratings are not available for many products ?
- In this case, only usage behavior can be analyzed 0
- One can record as to what any customer has purchased but we don't know why other products weren't purchased 1
- The reasons can be
  - Customer does not need the product now
  - Customer does not know about the product (which can be a good recommendation for him/her)
  - Customer does not like the product (which cannot serve as a good recommendation)

Similarity Measure

- For 0 / 1 kind of binary data, we can use Jaccard's Index
- $Jaccard\ Index(X,Y) = \dfrac{No.of\ times\ both\ the\ Users\ who\ have\ response\ 1\ for\ both\ X\ and\ Y}{No.of\ times\ both\ the\ Users\ who\ have\ response\ 1\ \ for\ either\ of\ X\ or\ Y\ or\ both}$
- Where X and Y are the set of the items with a 1 in user profiles

Jaccard Index

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9
U_a	1	0	1	1	0	0	0	1	0
U_b	1	1	0	1	0	1	0	1	1
U_c	1	0	0	0	0	1	0	0	0

$$Jaccard(a,b) = \frac{3}{7}$$

$$Jaccard(b,c) = \frac{2}{6}$$

$$Jaccard(a,c) = \frac{1}{5}$$

Evaluation of Recommender Algorithms

- The data of ratings (U x I) is divided in parts of training and validation
- Rows corresponding to training are used to build a recommender model
- With each user u_a in validation set is considered as active user, recommendations are created after withholding some items from the profile of user u_a
- Then the comparison in ratings is done for the withheld items by any accuracy measure like RMSE, MAE etc.
- Splitting, bootstrapping as well as k-folds cross-validation can be done with this data of ratings