Scalable Hierarchical Recommendations Using Spatial Autocorrelation

Ayushi Dalmia, Joydeep Das, Prosenjit Gupta, Subhashis Majumder, Debarshi Dutta

Information Overloading

"We are drowning in information but starved for knowledge"

- Information Filtering Systems have been designed in order to deal with this information overload.
- Recommender Systems - a subclass of Information Filtering System

Recommender System : Introduction

- Apply statistical and knowledge discovery techniques to the problem of making Product Recommendation.

Recommender System : Introduction

- Apply statistical and knowledge discovery techniques to the problem of making Product Recommendation.
- It receives information from a customer about which products he/she is interested in, and recommends products that are likely to fit his/her needs.

Recommender System : Introduction

- Apply statistical and knowledge discovery techniques to the problem of making Product Recommendation.
- It receives information from a customer about which products he/she is interested in, and recommends products that are likely to fit his/her needs.
- Today, recommender systems are deployed on hundreds of different e-commerce websites, serving millions of customers.

Recommender System : Approaches

1. Content Based

Recommender System : Approaches

1. Content Based
2. Collaborative Filtering

Recommender System : Approaches

1. Content Based
2. Collaborative Filtering
3. Demographic Approaches

Recommender System : Approaches

1. Content Based
2. Collaborative Filtering
3. Demographic Approaches
4. Knowledge Based Approaches

Recommender System : Approaches

1. Content Based
2. Collaborative Filtering
3. Demographic Approaches
4. Knowledge Based Approaches
5. Hybrid Approaches

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
   - Find user user or item item similarity based on the user rating

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
    - Find user user or item item similarity based on the user rating
    - Find a subset of users, having similar tastes and preferences to that of active user

Collaborative Filtering

Harness the *"Collective Intelligence"*

1. User Based Collaborative Filtering
   - Find user user or item item similarity based on the user rating
   - Find a subset of users, having similar tastes and preferences to that of active user
   - Offer recommendations based on the subset of users

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
    - Find user user or item item similarity based on the user rating
    - Find a subset of users, having similar tastes and preferences to that of active user
    - Offer recommendations based on the subset of users
2. Item Based Collaborative Filtering

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
   - Find user user or item item similarity based on the user rating
   - Find a subset of users, having similar tastes and preferences to that of active user
   - Offer recommendations based on the subset of users
2. Item Based Collaborative Filtering
   - Find the list of items rated by the active user

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
    - Find user user or item item similarity based on the user rating
    - Find a subset of users, having similar tastes and preferences to that of active user
    - Offer recommendations based on the subset of users
2. Item Based Collaborative Filtering
    - Find the list of items rated by the active user
    - Find a subset of items similar to the items already rated by the user

Collaborative Filtering

Harness the "*Collective Intelligence*"

1. User Based Collaborative Filtering
   - Find user user or item item similarity based on the user rating
   - Find a subset of users, having similar tastes and preferences to that of active user
   - Offer recommendations based on the subset of users
2. Item Based Collaborative Filtering
   - Find the list of items rated by the active user
   - Find a subset of items similar to the items already rated by the user
   - Offer recommendations based on the subset of items

Collaborative Filtering

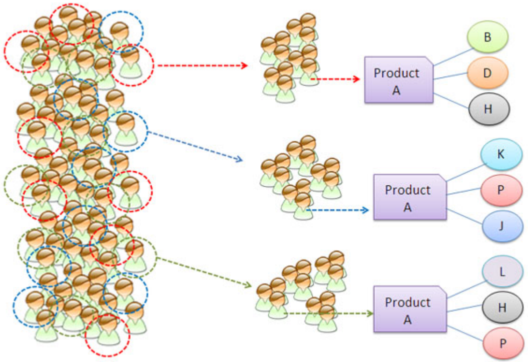

Figure: Collaborative Filtering

Challenges: Scalability

- Finding similarity among n users is $O(n^2)$ process

Challenges: Scalability

- Finding similarity among n users is $O(n^2)$ process
- If n is large, this leads to a quadratic blow up. Item Based Collaborative Filtering is an option but the number of items can also be substantially large.

Challenges: Recommendation Quality

User Satisfaction is important

- Recommendation quality needs to improve

Challenges: Recommendation Quality

User Satisfaction is important

- Recommendation quality needs to improve
- Requires more similar users

Challenges: Recommendation Quality

User Satisfaction is important

- Recommendation quality needs to improve
- Requires more similar users
- Increases quadratic complexity associated with Collaborative Filtering

Proposed Approach: Idea

- Preference Neighbourhood suggests users from a spatial region prefer items (e.g., movies, destinations) that are noticeably different from items preferred by users from other, even adjacent regions.

Proposed Approach: Idea

- Preference Neighbourhood suggests users from a spatial region prefer items (e.g., movies, destinations) that are noticeably different from items preferred by users from other, even adjacent regions.
- Build a Spatially-Aware Recommendation System, which will take location into consideration while finding similar users

Proposed Approach: Idea

- Preference Neighbourhood suggests users from a spatial region prefer items (e.g., movies, destinations) that are noticeably different from items preferred by users from other, even adjacent regions.
- Build a Spatially-Aware Recommendation System, which will take location into consideration while finding similar users
- Decompose the entire user space into regions of spatially similar users, apply Collaborative Filtering on each of these regions

Proposed Approach: Idea

- Preference Neighbourhood suggests users from a spatial region prefer items (e.g., movies, destinations) that are noticeably different from items preferred by users from other, even adjacent regions.
- Build a Spatially-Aware Recommendation System, which will take location into consideration while finding similar users
- Decompose the entire user space into regions of spatially similar users, apply Collaborative Filtering on each of these regions

Proposed Approach: Idea

- Partition a region with n users into k partitions with nearly equal sizes
- Overall time required for collaborative filtering is proportional to $k.(n/k)^2 = (n^2/k^2).k = n^2/k$
- Achieve a k order speed up by dividing the users space into k partitions
- Maintain recommendation quality while enhancing scalability

Proposed Approach: Preliminaries

Quadtree:

- A tree data structure in which each internal node has exactly four children.
- Partitions a two-dimensional space by recursively subdividing it into four quadrants or regions.
- Features:
  - They decompose space into adaptable cells.
  - Each cell (or bucket) has a maximum capacity. When maximum capacity is reached, the bucket splits.
  - The tree directory follows the spatial decomposition of the Quadtree.

Proposed Approach: Preliminaries

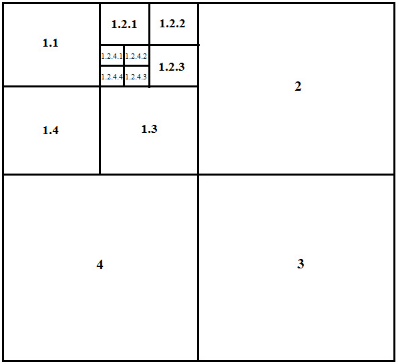

Figure: Division of the sample space into regions by recursively subdividing it into four sub-regions

Proposed Approach: Preliminaries

Spatial Autocorrelation

- The first law of geography says **"Everything is related to everything else, but near things are more related than distant things."**
- Measures the co-variance of properties within geographic space.
- Geary's index (c): Measures the similarity of i's and j's attributes, c_{ij}, which can be calculated as follows:

$$c_{ij} = (z_i - z_j)^2$$

where z_i and z_j are the values of the attribute of interest for objects i and j.

Proposed Approach: Preliminaries

Spatial Autocorrelation

A locational similarity w_{ij} is used in the calculation of Geary's index, where $w_{ij} = 1$ if i and j share a common boundary, and $w_{ij} = 0$ if not. Geary's index is expressed as follows:

$$c = \frac{\sum_i \sum_j w_{ij} c_{ij}}{2 \sum_i \sum_j w_{ij} \sigma^2}$$

where σ^2 is the variance of the attribute z values, or

$$\sigma^2 = \frac{\sum_i (z_i - \bar{z})^2}{(n-1)}$$

and

$$\bar{z} = \frac{\sum_i^n z_i}{n}$$

Proposed Approach: Preliminaries

Collaborative Filtering

- Based on the principle of finding a subset of users who have similar taste and preferences to that of the active user
- Given an active user u, compute her n similar users $\{u_1, u_2, \cdots, u_n\}$ and predict u's preference based on the preferences of $\{u_1, u_2, \cdots, u_n\}$.
- Users who agreed on the past tend to agree in the future also

Proposed Approach: Outline

Offline Processing: Decomposition

Input: A set of users(u) with the item ratings(r) and location

↓

Compute the spatial autocorrelation of the users

↓

Decompose the user space into Quadtree regions based on the location and spatial autocorrelation

Online Processing: Recommendation

Input: Active User for which recommendation is needed

↓

Map the user to the corresponding Quadtree region

↓

Find top10 similar users for this user based on Pearson Correlation Coefficient

↓

Select top5 highly rated items for each of these top10 users, giving a total of topset of 50 items

↓

Recommend top10 items from the topset by averaging the rating of the items in the region

Proposed Approach: Outline

Table: 1. Splitting Criteria

No. of users in the region	Correlation value	Look ahead (one level)	Split
High	High	N/A	Y
High	Low	N/A	Y
Low	High	N/A	N
Low	Low	positive	Y
		negative	N

Algorithm: Decomposition

Algorithm Quadtree_Decomposition

Step 1: Represent user location (city) as coordinates (longitude-latitude).

Step 2: Find the spatial autocorrelation value of the entire region (level-0 of the tree).

Step 3: Build the tree using splitting criteria.

Step 3.1: If correlation is good and number of users in the region is low (below the threshold limit), we do not split the region.

Step 3.2: If the number of users in a region is high (above the threshold limit), then irrespective of the correlation value we split the region.

Step 3.3: If both the number of users and correlation value of a region is low (below threshold limit), we apply the *look ahead* criteria, and consequently split(or do not split).

Step 4: Repeat steps 3 and 4 for each of the regions (tree nodes) as long as the splitting criterion is met.

Algorithm: Recommendation

Algorithm Recommend_Item
Step 1: Select a user for recommendation.
Step 2: Identify the location (longitude and latitude) of the user.
Step 3: Map the user in the exact region (node) of the Quadtree according to his/her location.
Step 4: Find a subset of users in the region who share similar preferences for items with the active user. Select *top-10* similar users.
Step 5: Select top 5 highly rated item from each of these *top-10* users to from a *top_set* of 50 items.
Step 6: Recommend *top-10* items from the *top_set* by averaging the rating of the items in the region.

Experimentation: Dataset

Book-Crossing Dataset

- 278,858 users
- 1,149,780 ratings
- 271,379 books
- Scale: 1-10

MovieLens Dataset

- 6,040 users
- 1,000,209 ratings
- 3,900 movies
- Scale: 1-5

Experimentation: Parameters

- User Threshold u_1 and u_2
- Correlation Threshold C_T
- Item Threshold f
- n_1 and n_2 represent the minimum and maximum number of users in a region respectively

Results: Decomposition

Tables 2 and 3 represent the average correlation values across all the regions using different threshold values. We can observe that the decomposition algorithm produces better results (in terms of correlation values) when the fraction of items(f) is less.

Table: 2. Summary of Spatial Decomposition with various threshold values [*BookCrossing Data*]

n_1	n_2	CT	f	No. of Regions	% $<$ 0.75 (Average)	% $<$ 1.0 (Average)
1000	3000	0.5	0.250	12	98.65	98.87
1000	3000	0.5	0.125	12	99.07	99.33
1000	5000	0.5	0.250	8	99.03	99.22
1000	5000	0.5	0.125	8	99.29	99.55

Results: Decomposition

Table: 3. Summary of Spatial Decomposition with various threshold values [*MovieLens Data*]

n_1	n_2	CT	f	No. of Regions	% < 0.75 (Average)	% < 1.0 (Average)
500	1000	0.5	0.250	19	29.38	54.59
500	1000	0.5	0.125	19	31.64	50.87
1000	3000	0.5	0.250	4	22.38	49.99
1000	3000	0.5	0.125	4	27.46	53.37

Results: Recommendation

Tables 4 and 5 represent the Mean Absolute Error (MAE), Root Mean Square Error (RMSE) Recall metric averaged over all the regions.

Table: 4. Summary of Recommendation Results with various Threshold Values [*BookCrossing Data*]

n_1	n_2	CT	f	No. of Regions	*Recall* (Average)	*MAE* (Average)	*RMSE* (Average)
1000	3000	0.5	0.250	12	0.9530	0.7632	0.8544
1000	3000	0.5	0.125	12	0.9175	0.7620	0.8980
1000	5000	0.5	0.250	8	0.9540	0.7775	0.8797
1000	5000	0.5	0.125	8	0.9125	0.7779	0.9363

Results: Recommendation

Table: 5. Summary of Recommendation Results with various Threshold Values [*MovieLens Data*]

n_1	n_2	CT	f	No. of Regions	*Recall* (Average)	*MAE* (Average)	*RMSE* (Average)
500	1000	0.5	0.250	19	0.8824	0.4435	0.6147
500	1000	0.5	0.125	19	0.9041	0.4447	0.6274
1000	3000	0.5	0.250	4	0.8731	0.4941	0.6856
1000	3000	0.5	0.125	4	0.9008	0.4982	0.6956

Results: Time Complexity

Tables 6 and 7 depicts the time complexity for a given set of parameters.

Table: 6. Results of Recommendation [*BookCrossing Data*] [$n_1 = 1000$; $n_2 = 5000$; $CT = 0.5$; $f = 0.125$]

Region No.	*Total Users*	*MAE*	*RMSE*	*Recall*	*Time (mins)*	*Cum. Time(m)*
1.1	1594	0.7514	0.8756	0.926	98.54	3199.45
1.2	5265	0.8312	0.1.022	0.9259	**2987.43**	
1.3	857	0.6823	0.9234	0.9415	9.78	
1.4	1935	0.7623	0.9123	0.8996	103.7	
1.2.1	222	0.8812	1.134	0.8571	0.14	**584.67**
1.2.2	353	0.7882	0.8765	0.9091	0.55	
1.2.3	1615	0.8143	0.9232	0.911	68.38	
1.2.4	3075	0.7123	0.8234	0.9297	515.6	
	Average	**0.7779**	**0.9363**	**0.9125**		

Results: Time Complexity

Table: 7. Results of Recommendation [*BookCrossing Data*] [$n_1 = 1000$; $n_2 = 3000$; $CT = 0.5$; $f = 0.25$]

Region No.	*Total Users*	*MAE*	*RMSE*	*Recall*	*Time (mins)*	*Cum. Time(m)*
1.1	1594	0.7613	0.8379	0.9225	111.5	3458.22
1.2	5265	0.8123	0.9834	0.9444	**3221.33**	
1.3	857	0.6776	0.7381	0.9346	10.78	
1.4	1935	0.7806	0.8786	0.9599	114.61	
1.2.1	222	0.8822	0.9929	0.9953	0.21	**665.27**
1.2.2	353	0.7884	0.8961	0.997	0.62	
1.2.3	1615	0.8023	0.9103	0.9515	79.75	
1.2.4	3075	0.7156	0.8007	0.9297	**584.69**	
1.2.4.1	653	0.6304	0.6944	0.944	6.33	**96.76**
1.2.4.2	125	0.7514	0.7929	0.9833	0.5	
1.2.4.3	759	0.7962	0.8846	0.9358	8.28	
1.2.4.4	1538	0.7598	0.8425	0.9377	81.65	
	Average	**0.7632**	**0.8544**	**0.953**		

Conclusion

- Employed a splitting technique to first divide the locations based on the correlation value
- Experimental analysis using real datasets shows that our model is efficient and scalable.
- Provides quality recommendations and also minimizes the computations