Running head: COLLABORATIVE FILTERING IN REAL ESTATE SEARCHES

Collaborative Filtering in Real Estate Searches

Bryan P. Blue

NOVA Southeast University

Abstract

Collaborative filtering and item based collaborative filtering can provide automated recommendations in a real time environment such as the WEB. Ways of collecting ratings, both explicitly and implicitly are reviewed as well as different functions to compute similarity between users or items. This is followed by a brief prediction function explanation. This information is then used to propose a possible use of collaborative filtering in real estate search engines.

Introduction

The amount of information on the web can be overwhelming for even the most seasoned user. Even within a given sub-domain, say books for sale, the magnitude of choices can leave a user bewildered. Another problem with web-based purchases is the lack of human interaction with a (supposedly) knowledgeable sales person that can help make recommendations based on what they know about the products as well as what other people are buying. To this end, a technology was developed to address this need called collaborative filtering.

Collaborative Filtering uses the concept that consumers of products and information will have similar wants and needs as other consumers of those products. Collaborative Filtering is a process that calculates values that can qualify these wants and needs, and then make recommendations based on those values. (Francis, Kaiser, Sanders, White, 2002). This process is also referred to as nearest-neighbor, or user-based collaborative filtering. Today's proliferation of information that is available electronically, compounded with the large volume of customers gives rise to making a real-time, quality recommendation difficult.

Most major sites offer some variation of personalization. These can be explicit settings you can control using an *account* feature or *profile*, but are increasingly relying on implicit

choices of the customer while they are within the site. Through the use of collaborative filtering, a higher level of personalization of an Internet web site can be achieved. (Buono, Costabile, Guida, Piccinno, Tesoro, n.d.).

The overall process can be broken down into a few generalized steps. Collect a large set of different customer preferences on distinct items. A similarity metric is then used to find a subset of customers that have preferences that are similar to the customer needing advice. An average of the subset is calculated and then used along with a preference function to recommend alternatives that the advisee has not even expressed an opinion. (Heylighen, 2001). In this way, a top-n recommendation list can be created for the customer.

A large problem facing collaborative filtering is how to reliably collect preferences. In order to ensure that a usable result is given, the set of collected preferences needs to be large, in the order of thousands or larger. This means that there is a critical mass of preferences that need to be collected before any usable information can be generated. The question becomes, how do you get customers to give you these ratings? (Heylighen, 2001).

Getting customers to manually enter preferences while using a web site is not practical. Customers are reluctant to fill out surveys while shopping or browsing when they will initially receive no benefit from it. This is especially the case before you hit the critical mass of preferences and cannot generate a reliable recommendation. One way around this is the use of implicit ratings.

Explicit vs. Implicit Rating

Most people are familiar with explicit ratings. Many people use them everyday. They range from lengthy book and movie reviews to the quick and to the point star rating system or top 10 lists. This approach is used in the computer world as well, but requires a physical

response from the customer to qualify an item as to its' importance or interest to the customer. As this cannot always be expected, a new form of gathering ratings has been devised based on implicit ratings.

Implicit ratings remove the effort cost of the customer to explicitly rate an item. Implicit rating data can come from many sources. Below is a list of possible sources and notes about them.

Customer Action	**Notes**
Purchase	Buys item and at what price
Assess	Explicitly evaluates or recommends (yes/no)
Repeated Use	Number of times used/Number of links to
Save/Print	Moves document to personal storage
Mark	Add to favorites or links to document
Examine/Read	Time customer spends on page
Associate	Returns in search but never looks at/clicks on
Query	Associate with terms from query

Implicit ratings may solve the issue of evaluation cost to the advisor, but introduce a possibly new problem of over evaluation. Since almost every customer action can be viewed as a rating, it theoretically could be interpreted and stored. The problem with this is that it would lead to an overwhelming amount of data. Also, the true value of each of these ratings would be far less meaningful than an explicit rating. (Nicholes, D., n.d.). Because of this, the merits of recording every action cannot be justified. When building an implicit rating system, care must be taken to include only those actions that would yield the most reliable and useful ratings.

Similarity Computation

Once ratings have been accumulated, there is still the problem of determining similarity. Typically, a customer is represented as a N-dimensional vector with N being the number of distinct items being evaluated. This is characteristically a very sparsely populated vector for any given customer. Each element of the vector represents a rating for that item by the customer. The values are positive for purchased or positively reviewed items and negative for each negatively rated item. To compensate for best selling items that have many positive reviews, the vector is usually multiplied by the inverse frequency of each product. This has the effect of making lesser known, but positively reviewed items more prominent. Together, the group of customers' preferences for items makes up an m x n customer-item rating matrix.

There are many different ways to compute similarity between customers. One common method is by using the cosine of the angle between two customer vectors. If you have two customer vectors $\vec{A}$ and $\vec{B}$, then a similarity function can be defined as follows.

$$sim(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\left\|\vec{A}\right\| * \left\|\vec{B}\right\|}$$

Once a set of customers is determined to be similar, suggestions can be made based on items the related customers have already ranked as positive or have purchased.

This approach works fine for small quantities of items, but can slow down for large numbers of customers or items. It has been shown that using this form of computation, you can expect an overall performance of $O(M + N)$ where M is the number of customers and N is the number of items per customer. There are ways that can help reduce the overall data size that is used during the computation, but any reduction will invariably also reduce the recommendation quality. (Linden, Smith, York, 2003).

This algorithm also slows down for customers that have purchased many items. The more rated items that are contained in a given customers row, the more matches can be made to other customers. This in turn slows down the recommendation process for your most active customers. This is not a very positive aspect. (Karypis, Konstan, Riedl, Sarwar, 2001).

User-based collaborative filtering also suffers from the problem of scarcity. Recommendations may be poor or impossible for customers of sites that require recommendations of items from inventories that are quite large. If you had an inventory of 2,000,000 books, most customers will have purchased less than 1% of those titles (far less than 20,000 books). Therefore, user-based collaborative filtering may not be able to find any accurate match at all if nobody else has purchased those same books. (Karypis, Konstan, Riedl, Sarwar, 2001).

Item-based Collaborative Filtering

For large amounts of data, traditional collaborative filtering falls apart due to real-time computational constraints. Much of the computational work can be done offline by using item-based collaborative filtering. In this scenario, unlike traditional collaborative filtering, similarities between items are computed first, not between customers. These results are then used to find items that are similar to other items that the customer has liked in the past.

Item-based similarity computation

For item-based collaborative filtering the similarity between items needs to be calculated. Once this is accomplished, the results can be used to find the most similar items. In general, two items are chosen, i and j and then all users who have rated both items are found. From this data set, S_{ij}, the similarity between them is computed. Three popular similarity functions for item-based collaborative filtering follow.

Cosine-based Similarity

Cosine-based similarity uses the notion of the two items as being vectors of the customer user-space. Similarity can be computed using the cosine of the angle between the two item vectors, just as in user based collaborative filtering, the difference being that columns are now evaluated instead of rows. (Karypis, Konstan, Ridel, Sarwar, 2001). Given the *m* x *n* (user x item) rating matrix in Figure 1, the similarity between the two shaded items *i* and *j* can be calculated using the formula:

$$sim(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \bullet \vec{j}}{\left\|\vec{i}\right\|_2 * \left\|\vec{j}\right\|_2}$$

Correlation-based Similarity

Another similarity function is based on the Pearson-r correlation $corr_{i,j}$. For this to maintain accuracy, only occurrences where users have rated both *i* and *j* should be used. This is shown as the highlighted pairs in Figure 1. Each set of pairs where both items were rated is grouped into a set named *U*. $\overline{R_i}$ represents the average rating of the *i*-th item. (Karypis, Konstan, Ridel, Sarwar, 2001). The correlation based function then becomes:

$$sim(i, j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_i})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_j})^2}}$$

Adjusted Cosine Similarity

The cosine method of computing similarity does not take into effect the differences in the rating scale values between different users when applied to item-based filtering. The adjusted cosine function accomplishes this by subtracting the user average from each rated pair. Only occurrences where users have rated both *i* and *j* should be used. Each set of pairs where both items were rated is grouped into a set named *U*. $\overline{R_u}$ represents the average of the u-th user's

rating. (Karypis, Konstan, Ridel, Sarwar, 2001). The adjusted cosine function can the be expressed as:

$$sim(i, j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_u})^2}}$$

Test Results

Karypis, Konstan, Ridel, Sarwar (2001) ran experiments on various test data to determine the overall performance of each of the three similarity functions. From their test results, they evaluated the Mean Absolute Error (MAE). From this, it was found that the adjusted cosine calculation yielded the lowest MAE. This was most likely due to adjusting for the user-average in the computation. Both pure cosine and correlation functions produced nearly similar results with both having higher MAE values. From this, it may be most advantageous to use the adjusted cosine similarity function to yield the best recommendations for a customer.

Prediction

After computing the similarities of the items, a prediction of what the customer will want to see next still needs to be performed. At this point, the customers' preferences need to be evaluated and a prediction generated. Two possible options are described.

Weighted Sum

For a given customer *u* and item *i*, a prediction value is determined by computing the sum of the users ratings that are similar to item *i*. Each of the users ratings is weighted by the similarity value between items *i* and *j* given by $S_{i,j}$. (Karypis, Konstan, Ridel, Sarwar, 2001). This gives a prediction function of:

$$P_{u,i} = \frac{\sum_{N}(S_{i,N} * R_{u,N})}{\sum_{N}(|S_{i,N}|)}$$

Figure 2 illustrates this graphically.

Regression

Regression may also be used to calculate a prediction that may be more accurate. Here, instead of using the exact rating value as in the weighted sum prediction, a modified $R_{u,N}$ value based on linear regression is used. A complete description of this goes beyond the scope of this paper. See Karypis, Konstan, Ridel, Sarwar, 2001 for a more information.

Amazon.com

Amazon.com has one of the most widely known purchased based implicit rating systems in use today. They use a different approach to making recommendations. Instead of using traditional collaborative filtering to find similar customers, they use a variation of item based collaborative filtering in what they call item-to-item collaborative filtering to find similar items. (Linden, Smith, York, 2003).

Amazon.com compares similarity between items based on customers that have purchased them. The algorithm is iterative and can be calculated offline. It can be described with the following pseudocode:

For each item in product catalog I_1

    For each customer C who purchased I_1

        For each item I_2 purchased by customer C

            Record that a customer purchased I_1 and I_2

    For each item I_2

        Compute the similarity between I_1 and I_2

Here, the similarity function is based on a vector for an item N that has M customers that have purchased this item. The similarity function is based on the cosine method, and does

become very time consuming. This algorithm is $O(N^2M)$ in the worst case, although it has been shown that since most customers do not have many purchases, it runs in O(NM). (Linden, Smith, York, 2003). This creates a similar-items table that can be computed offline.

This similar-items table can then be quickly searched for recommendations in real time as most of the computation has been done ahead of time. This is particularly important on the web; customers can be given recommendations on each page as the customer clicks from item to item incurring little additional delay.

Real Estate Applications

Little has been written about applying item based collaborative filtering to the real estate industry. There is a wealth of data contained within each property database imbedded in most real estate search engine sites. Most of this data is not being taken advantage of. An explanation of traditional sites and then a proposed application of collaborative filtering are given here.

Traditional Overview

Traditional real estate search engines and web sites require the buyer to make explicit choices about the types of housing they are interested in. This is typically in the form of the number of bedrooms, number of baths, location and price. The problem with this type of interface is that it is next to impossible to have the user qualify all possible features about a house of interest to the buyer without creating a custom search page that is so large and overwhelming in its size that it would become a deterrent to it's use. Once the basic information about the query is entered, the buyer can then click a search button to have properties meeting these minimum requirements shown to them. Usually this list is sorted by price, not by interest to the buyer.

Many real estate search systems allow the user to create a profile and then store properties of interest. Often, there is an option to save the search criteria within the profile as well. They buyer can then choose to have new properties that they have not seen e-mailed to them as they appear in the system.

The buyer can print out a flyer about each listing they are interested in. These are usually a printout that contains all the details about the property as well as a picture of the home. The sites also supply e-mail links to the listing agent, allowing the buyer to contact the agent for more information about the property. All of these actions can be viewed as an invaluable source of implicit rating data.

Possible Collaborative Filtering Application

The buyers' profile and list of selected favorite homes could be used to enhance the search for a home. In addition to the explicitly stated price, bedrooms and baths that were entered into the search, typically there is also a large list of features that could also be evaluated. By reviewing this listing data, an overall set of likes and dislikes of the buyer could be inferred.

Implicit ratings can be pulled from the profile as well as actions such as *print a flyer*. The following list shows some of the buyer actions that can be used to trigger the recording of an implicit rating and how the rating could be interpreted.

Buyer Action

- Print a flyer of the property (positive)
- Save property to *favorites* (positive)
- E-mail agent for more information (positive)
- Buyer never visits detail page of property that is shown (negative)

All descriptive features of the properties for a given buyer that have been recorded due to one of the above actions can be evaluated for pertinence using collaborative filtering techniques. Each action that triggers an implicit rating can be recorded into a matrix where the items being rated would be the features of those properties. Each feature would represent a column and each row would represent a property. The resulting matrix could look similar to Figure 3. For space limitations, only 4 features are shown, but in reality could be quite large.

This matrix could then be averaged, or other statistical functions applied, in a column wise fashion. The result could then be interpreted as a single rating vector of an *ideal home* for that buyer. The idea is that the more properties a buyer reviews, the more likely it will be that certain features will start to form a pattern. The more homes that have a swimming pool in their results would increase the rating of the feature swimming pool. This in turn would be interpreted as find more homes for the buyer, but try to find ones with swimming pools first. All of this could be collected without ever having the buyer explicitly rate each feature.

A matrix containing other buyers' ideal home vectors could be stored as well. In this way, a similarity calculation could be performed as in the item based collaborative filtering process. A sample buyers' ideal home rating matrix is shown here.

Buyers Ideal Home	Above Ground Pool	In Ground Pool	Porch	Patio	1 story	2 story	Ocean	Gulf
1	8	72		33	100			
2			13	56	13	87		
3	10	85			90		13	80
4	5	95	24	34	86		50	50
5						100		
…								

This would help in the suggesting process for handling similar features. If a buyer had only happened upon homes that had above ground pools, that would not necessarily rule out that

they are not interested in in-ground pools, hot tubs, or other uniquely different but similar water features. As a result of the item based collaborative filtering, properties with these similar ratings would get weighed more heavily. A top-n list of features that the buyer is most likely interested in would be the result of collaborative filtering.

A subset of the entire listing database would first be generated based on the query results from the quantitative information that was explicitly given by the buyer. These are usually values that cannot change due to social and economic limitations of the buyer. Then, the top-n features could be used with this subset of listings to order them by most relevant by features. This should have the effect of helping to present homes that are most similar to the properties that have been implicitly rated as positive.

Conclusion

Item-based collaborative filtering has many advantages over user-based collaborative filtering, the largest being response time. Although it is in use in many large e-commerce sites today, collaborative filtering still has yet to be used in locations outside of the entertainment area. When properly applied, it may be of great use to any e-commerce site that needs to help the customer make a decision.

Figures

	1	2	3	*i*		*j*		n-1	n
1				R		R			
2				-		R			
3									
u				R		R			
m-1				R		R			
m				R		-			

Figure 1. Customer/Item rating matrix. Rows represent customers, columns represent particular items. Here vecotors *i* and *j* are shown. R represents a rated item *n* by user *m*. Adapted from Karypis, Konstan, Ridel, Sarwar, 2001.

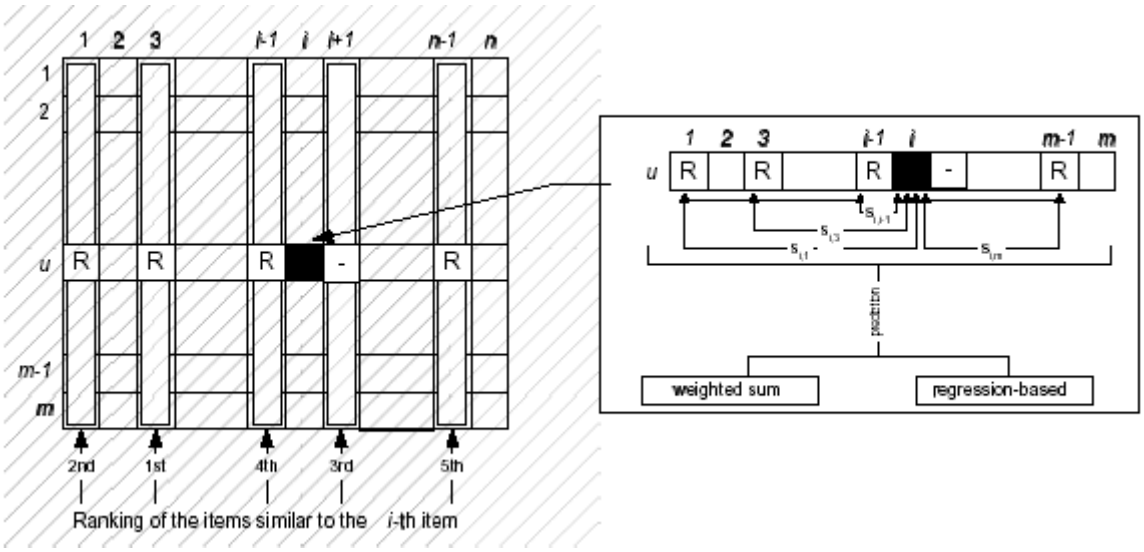

Figure 2. Overview of the item-based collaborative filtering algorithm. The prediction process is shown for 5 neighbors. (Sarwar, Karypis, Konstan, Riedl, 2001).

Property ID	Swimming Pool	Porch	2-story	1-story
19288487	1	0	1	0
23948583	1	0	0	1
39348548	0	0	1	0
Ideal Home Rating (%)	66	0	66	33

Figure 3. Possible buyer rating matrix. Each row is a unique property; each column is a qualitative feature of the property.

References

Buono, P., Costabile, M., Guida, S., Piccinno, A., Tesoro, G. (n.d.). Integrating user data and collaborative filtering in a web recommendation system. *Dipartimeno di Informatica, Universita di Bari, Italy*. Retrieved September 2, 2003, from http://wwwis.win.tue.nl/ah2001/papers/costabile.pdf

Francis, J., Kaiser, L., Sanders, S., White, S. (2002, October 23). Collaborative filtering. Retrieved August 20, 2003, from http://www4.ncsu.edu/~jlfranci/CollaborativeFilteringReport.doc

Heylighen, F., (2001, January 31). Collaborative filtering. Retrieved August 15, 2003, from http://pespmc1.vub.ac.be/COLLFILT.html

Karypis, G., Konstan, J., Riedl, J., Sarwar, B. (2001, May 5). Item-based collaborative filtering recommendation algorithms. *University of Minnesota*. Retrieved from http://www.cs.umn.edu/Research/GroupLens/papers/pdf/www10_sarwar.pdf

Linden, G., Smith, B., York, J. (2003, January). Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*. Retrieved August 25, 2003 from http://dsonline.computer.org/0301/d/w1lind.htm

Nichols, D. (n.d.). Implicit rating and filtering. Retrieved August 15, 2003, from http://www.ercim.org/publication/ws-proceedings/DELOS5/nichols.pdf