Accuracy of Similarity Measures in Recommender Systems

Seth Sorensen
Department of Computer Science
University of Minnesota, Morris
soren585@morris.umn.edu

ABSTRACT

As the volume of information available to Internet users continues to grow at unprecedented rates, systems that are able to filter out and present relevant information in a meaningful way become more and more important. Such systems are referred to as recommender systems. The purpose of a recommender system (also referred to as a recommendation system) is to predict the rating that a user of the system would assign to an item (e.g. movie, song, book), and recommend to that user the items with the highest predicted ratings. The designers of recommender systems face a tremendous challenge in making systems that scale well for large sets of data, returning accurate predictions in a timely manner. One possible approach to increasing the accuracy of a predicted rating is through the use of similarity functions. Similarity functions are used in recommender systems to combine a multitude of ratings into a single value that represents the similarity between two users or items. This paper presents a few commonly used similarity functions and looks at some recent research in the field of recommender systems that aims to determine which measures of similarity result in the most accurate recommendations.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: [Relevance feedback]; G.1.2 [**Approximation**]: [Least squares approximation]

General Terms

Performance, Algorithms

Keywords

recommender systems, aggregation functions, similarity measures

.

1. INTRODUCTION

The purpose of a recommender system (also referred to as a recommendation system) is to predict the rating that a user of the system would assign to an item (e.g. movie, song, book), and recommend to that user the items with the highest predicted ratings. This collection of recommended items is much more likely to interest the user, resulting in a more gratifying experience. A satisfied user translates to a valuable customer for the provider of the system. This sort of relationship benefits both parties, and acts as motivation for the development of better recommender systems.

In most recommender systems recommendations need to be calculated in real time, meaning functions used in the system must run quickly on large data sets while still returning accurate predictions. The system must also handle what is known as the cold-start problem. When a user or item is first added to the system very little is known of them, making comparisons to other users or items already in the system very difficult. Any given recommender system consists of various elements, and each element must be designed with these goals in mind.

One possible approach to increasing the accuracy of predicted ratings is through the use of a similarity function. Similarity functions are used in recommender systems to combine a multitude of ratings into a single value that represents the similarity between two users or items. This similarity measure is then used to represent the degree of influence that one of the two items or users will have on calculating the predicted rating of the other. In this paper I will focus on the role that similarity functions play in calculating predicted ratings in a recommender system.

Section 2 of this paper describes the general structure of a recommender system as well as the different types of recommender systems, how they differ from each other, and the situations in which one might be used in place of another. In section 3, I describe some methods commonly used in recommender systems to calculate the degree of similarity between two users or items. In section 4 I present a study conducted by Spertus *et al.* that compares the accuracy of various similarity measures by recommending to users of the Orkut social network a set of "communities" that might interest them, and recording which communities are viewed and which communities the user ends up joining. In section 5 I look at a study conducted by Lathia *et al.* that aims to compare the accuracy of various measures of similarity using a set of movie ratings, collected from the movie recommendation site MovieLens.

2. BACKGROUND

2.1 Recommender Systems

In order to calculate the predicted rating of some item d for a user u, denoted by $R(u, d)$, a recommender system must first acquire information regarding the preferences of the user, commonly in the form of some set of ratings related to either user u, item d, or both (e.g. ratings of other items by user u or ratings of item d by other users); these ratings may have been collected explicitly as user input or implicitly through observation of user activity, such as one user viewing the same movie a large number of times (implying a high rating). A system that is still in its infancy may not have much, if any user information already recorded, meaning that the system must forgo any personalized recommendations until sufficient data has been collected, and in the meantime either recommend items based on some other criteria or make no recommendations.

Once the appropriate data has been collected, the system can begin to calculate predicted ratings and make recommendations accordingly. The two primary types of recommender system, distinguished by which data is used in the calculation of the predicted rating, are collaborative filtering and content-based.

Collaborative filtering recommender systems operate on the concept that users who have exhibited similar interests in the past will tend to rate items in a similar fashion. The predicted rating of an item d for some user u is calculated by collecting a predetermined number k of u's *nearest neighbors* and aggregating the ratings that they have provided for d in a way that reflects the degree of similarity between u and those neighbors. The nearest neighbors of u are the users that exhibit the highest degree of similarity to u, and are commonly determined using the *k-Nearest Neighbor* approach.

Content-based recommender systems are ideal for situations in which a user is likely to be interested in an entire category of items, such as a genre of music. For instance, a Pandora user listening to Led Zeppelin might next like to hear a song by Pink Floyd. The methods of calculation in content-based systems are very similar to those used in a collaborative filtering system, however instead of user similarities, ratings are based on item similarities. The degree of similarity between some item d_i previously rated by user u, and the item d for which the rating is being calculated determines the importance of $R(u, d_i)$ in calculating $R(u, d)$.

Other types of recommender systems include demographic, knowledge-based, and hybrid systems. A demographic recommender system bases predicted ratings on the demographic to which the user belongs. Knowledge-based systems use information explicitly provided by the user to provide recommendations that are specific to the user's immediate needs. An example of a knowledge-based recommender system is the advanced search often available for library catalogs. A hybrid system is one that incorporates components of two or more different types of recommender system into one combined system. [5]

2.2 Weighted Arithmetic Mean

The weighted arithmetic mean (WAM) is an averaging function that is commonly used in recommender systems to calculate a predicted rating based on a set of input ratings, where each input rating is assigned a weight representing its level of importance in calculating the overall rating. An input with a relatively large weight will cause the calculated rating to be more similar to itself than if it were assigned a smaller weight.

In a collaborative filtering recommender system, a common implementation of the prediction function, $R(u, d)$, uses the similarity of user u and u's nearest neighbors, u_j, as weights in the weighted arithmetic mean:

$$R(u, d) = \sum_{j=1}^{k} sim(u, u_j) R(u_j, d) \tag{1}$$

where $sim(u, u_j)$ is a function that calculates the degree of similarity between u and one of its neighbors, u_j, and $\sum_{j=1}^{k} sim(u, u_j) = 1$ [2]. Similarity functions are discussed further in Section 3. $R(u_j, d)$ is the rating that user u_j has explicitly provided for item d.

A common form of the prediction function for a content-based system, which closely resembles Equation 1 is:

$$R(u, d) = \sum_{j=1}^{k} sim(d, d_j) R(u, d_j) \tag{2}$$

Again we assume that $\sum_{j=1}^{k} sim(u, u_j) = 1$. Equation 2 is nearly identical to Equation 1, however rather than iterating through the nearest neighbors of u, it iterates through the items that have been previously rated by u. The weights are the degree of similarity between each item and the item whose predicted rating is being calculated.

3. SIMILARITY FUNCTIONS

Similarity functions are commonly used in recommender systems to measure the degree of similarity between two items or users. This degree of similarity then acts as a weight in an averaging function such as the weighted arithmetic mean described in Section 2.2. As an example, suppose we have a collaborative filtering recommender system using a weighted arithmetic mean to calculate the predicted rating of an item d for some user u:

$$R(u, d) = \sum_{j=1}^{k} sim(u, u_j) R(u_j, d) \tag{3}$$

For the sake of simplicity we assume that the weights are normalized, meaning that $\sum_{j=1}^{k} sim(u, u_j) = 1$. A result of this assumption is that the range of the similarity function must be [0, 1]; the function will however still yield the same results as if the weights were not normalized and we instead divided each resulting weight by the sum of the weights. [11] A special case of the weighted arithmetic mean is where each of the weights is equal to $1/k$, in which case it is just the arithmetic mean. [2]

With a simple example we can demonstrate the utility of the similarity function. Suppose there exist users u_1, u_2, and u_3, and an item d, with $R(u_1, d) = 10$, $R(u_2, d) = 2$, $R(u_3, d) = 1$, and the similarity function returns weights of 0.8, 0.1, and 0.1 respectively. Under these conditions, Equation 3 results in $R(u, d) = 8.3$, whereas the unweighted arithmetic mean results in $R(u, d) = 4.33$, a 48% and nearly 4 point difference. In this example the value returned by the weighted mean is much closer to the rating provided by the user most similar to user u than the value returned by the unweighted mean, evidence that an accurate similarity

function can significantly benefit the overall accuracy of a recommendation system.

Studies regarding the relative accuracy of various similarity functions have resulted in conflicting opinions as to which one is the "best" [4, 7]. In Sections 4 and 5 I will present two studies, carried out by Spertus *et al.* and Lathia *et al.* respectively, that compare the accuracy of predicted ratings calculated using various similarity measures. In this section I will present a few of the more common approaches to calculating similarity in recommender systems: cosine similarity, Pearson correlation, and least squares [1].

3.1 Cosine Similarity

Given two users, u and u_i, we seek to define the function $sim(u, u_i)$ such that the resulting values range from 0 to 1 and reflect the degree to which u and u_i agree. Supposing that u and u_i are actually two n-dimensional vectors consisting of ratings previously provided by each respective user for items d_i, $i \epsilon \{1, 2, ..., n\}$, we can then measure the similarity of the two users by the magnitude of the angle between these vectors, where an angle of 0 indicates identical vectors. The greater the angle, the less similar the two users are. This behavior is actually the reverse of that which we seek in our similarity function, so instead we will use the cosine of the angle. For simplicity we will assume that all ratings are positive, the result of which is that the maximum angle between the two vectors is $\pi/2$. We now have a function whose range is [0, 1], with a 0 indicating the lowest degree of similarity possible and a 1 the highest.

The cosine of the angle between two vectors is calculated using the equation

$$cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} \tag{4}$$

where A and B are the two vectors, $A \cdot B$ is the dot product of A and B, and $\|A\|$ is the L2 norm of A. The L2 norm, also known as the Euclidean norm, is calculated using:

$$\|A\| = \sqrt{\sum_{i=1}^{n} a_i^2} \tag{5}$$

where a_i is the ith component of the vector A.

In [3], Jacobi *et al.* describe an analogous approach for calculating the similarity, referred to by Jacobi *et al.* as the Commonality Index (CI), of two items A and B based on N_A, N_B, and N_{AB}: the number of customers who have purchased item A, the number who have purchased item B, and the number who have purchased both item A and item B, respectively:

$$CI(A, B) = \frac{N_{AB}}{\sqrt{N_A \times N_B}} \tag{6}$$

A standard form of Equation 6 using set notation is obtained by treating A and B instead as the sets of users that have purchased items a and b respectively:

$$CI(a, b) = \frac{|A \bigcap B|}{\sqrt{|A| \cdot |B|}} \tag{7}$$

Equations 6 and 7 are equivalent, and both are analogous to Equation 4, but in terms of sets rather than vectors.[7]

3.2 Pearson's Correlation

The Pearson product-moment correlation coefficient (Pearson's correlation), as used in statistics, is a measure of dependence between two random variables. The method for obtaining the correlation between random variables $\mathbf{X}$ and $\mathbf{Y}$ is

$$corr(\mathbf{X}, \mathbf{Y}) = \frac{Cov(\mathbf{X}, \mathbf{Y})}{\sigma_{\mathbf{X}} \sigma_{\mathbf{Y}}} \tag{8}$$

where $Cov(\mathbf{X}, \mathbf{Y})$ is the covariance of $\mathbf{X}$ and $\mathbf{Y}$, and σ_X is the standard deviation of $\mathbf{X}$. [9] Substituting u and u_i in for $\mathbf{X}$ and $\mathbf{Y}$, we can calculate Pearson's correlation for u and u_i by treating them as random variables with possible values of $R(u, d_j)$ and $R(u_i, d_j)$ respectively, where each outcome is equally probable.

3.2.1 Covariance

The covariance of two random variables is a measure of the similarity in their behavior, calculated with

$$Cov(\mathbf{X}, \mathbf{Y}) = E[(\mathbf{X} - E[\mathbf{X}])(\mathbf{Y} - E[\mathbf{Y}])] \tag{9}$$

$E[\mathbf{X}]$ represents the *expected value*, or average of $\mathbf{X}$. Extending the concept of covariance to our collaborative filtering recommender system example described in Equation 3, the covariance of user u and some user u_i is

$$Cov(u, u_i) = \frac{1}{n} \sum_{j=1}^{n} (R(u, d_j) - \overline{R(u)})(R(u_i, d_j) - \overline{R(u_i)}) \tag{10}$$

where d_j is one of n items that have been rated by both u and u_i, and $\overline{R(u)}$ is the average rating by user u of all n items d_j. What Equation 10 represents is the degree to which user u and user u_i tend to rate items similarly.

3.2.2 Standard Deviation

The standard deviation of a random variable describes the amount of variation between the average value and the actual values of that random variable, and is calculated using:

$$\sigma_{\mathbf{X}} = \sqrt{E[(\mathbf{X} - E[\mathbf{X}])^2]} \tag{11}$$

Again extending this concept to the example illustrated by Equation 3, we find the standard deviation of the ratings provided by user u with:

$$\sigma_u = \sqrt{\frac{1}{n} \sum_{j=1}^{n} (R(u, d_j) - \overline{R(u)})^2} \tag{12}$$

n, u, d_j, and $\overline{R(u)}$ all have the same representations as in Equation 10. The standard deviation for user u_i is calculated in the exact same way replacing u with u_i.

3.2.3 Correlation Coefficient

Substituting Equations 10 and 12 into Equation 8 we now have

$$corr(u, u_i) = \frac{\sum_{j=1}^{n} (R(u, d_j) - \overline{R(u)})(R(u_i, d_j) - \overline{R(u_i)})}{\sqrt{\sum_{j=1}^{n} (R(u, d_j) - \overline{R(u)})^2} \sqrt{\sum_{j=1}^{n} (R(u_i, d_j) - \overline{R(u_i)})^2}} \tag{13}$$

[6] The result is a value in the range $[-1, 1]$, where a 1 represents perfect agreement and a -1 represents perfect disagreement. We would like Equation 13 to have a range of $[0, 1]$ so that we can use it as our similarity function. One option would be to shift and scale all resulting values by adding 1 and dividing by 2, however [6] suggests simply discarding negative correlations as they "are generally believed to not be valuable in increasing prediction accuracy".

3.3 Least Squares

The method of least squares is a mathematical procedure used for determining the line of best fit for a given set of points (x_i, y_i), $i \in \{1, 2, ..., n\}$. The procedure consists of calculating a function $f(x)$ that minimizes the sum

$$\sum_{i=1}^{n} r_i^2 \tag{14}$$

where $r_i = y_i - f(x_i)$ and is referred to as the *residual error*. f is then the function describing the line of best fit.[10] The method for minimizing the sum is beyond the scope of this paper, but is described in [8].

Assuming again that we have a prediction function of the same form as Equation 3, we can adapt the method of least squares for use in calculating the weights represented by $sim(u, u_j)$. Given a set of items $D = \{d_1, ..., d_q\}$ that have already been rated by both u and u's k nearest neighbors, we aim to find a function $(sim(u, u_j))$ that minimizes the residual error between the actual rating and the predicted rating of d_i by user u. Since all of the items in D have already been rated by both u and u_j, we already know the actual rating $R(u, d_i)$; this will be substituted in for y_i in the residual error equation. The predicted rating of d_i by user u is just Equation 3 so this will be substituted into the residual error equation for $f(x_i)$, resulting in:

$$minimize \sum_{i=1}^{q} [R(u, d_i) - \sum_{j=1}^{k} sim(u, u_j) R(u_j, d_i)] \tag{15}$$

with the restrictions that $sim(u, u_j) \geq 0$, and $\sum_{j=1}^{k} sim(u, u_j) = 1$. Using the same methods as are used in calculating the function f in the method of least squares, the partial function $sim(u, u_j)$ can be calculated at the points u_j, $j \in \{1, 2, ..., k\}$, and the resulting values are used as weights in Equation 3 to calculate the predicted rating of item d by user u. [2]

3.4 Remarks

As previously mentioned, multiple studies have been performed with the purpose of determining the most accurate of similarity measures in a recommender system. [7, 4] In Sections 4 and 5, I will discuss two studies that compare the accuracy of various similarity measures, including Cosine Similarity and the Pearson correlation coefficient.

4. ORKUT

In 2004 Google Inc. launched a social networking site called Orkut, named for its creator, Orkut Büyükökten, that allowed users to connect with similar users by joining or creating "communities" based on a shared interest. In 2005 Spertus, Sahami, and Büyükökten published a paper [7] detailing a study in which the accuracy of various similarity measures was compared by presenting Orkut users with a set of recommended communities, then tracking and recording the recommendations in which the users displayed interest. In this section I will first provide details on the design of the experiment described in [7], then discuss the results.

4.1 Experiment Design

4.1.1 Preparation

At the beginning of the experiment, the authors collected a set of data consisting of all pairs of the form (u, c), where c is a community to which at least 20 members belong, and u is a user that belongs to at least one such community. Next, the similarity between each pair of communities in the data set was calculated using six different similarity functions, including the L2 norm, described in Section 3.1, and the closely related L1 norm. The other four similarity measures used in the experiment will not be presented here, but are referred to as MI1, MI2, LDF, and LogOdds. The calculated similarities were recorded and used throughout the course of the experiment.

Each of the six similarity functions used in the experiment calculated similarities between two communities based on the number of overlapping members in the data set. For example, the similarity of two communities using the L2 norm would be calculated in exactly the same manner as the commonality index, as seen in Equation 7, except A and B, rather than the sets of users who have purchased the items being compared, are the sets of members in the two communities being compared.

4.1.2 Recommendation

Over the course of an 18 day period, any new users who visited the page of a community belonging to the previously collected data set were presented with recommendations (new users being users who had joined Orkut during the 18 day period). The recommendations would be for communities that were considered similar to the community being viewed. The authors refer to the community on whose page the recommendations were being displayed as the "base" community; I will use this terminology as well.

In order to determine which recommendations were to be displayed, the system would first select two similarity measures "in a deterministic manner so that a given user always saw the same recommendations for a given community". Then, for each of the selected similarity measures, the system would find up to 6 communities with the highest degree of similarity to the base community. There was a guarantee that the number of recommendations to be displayed for each chosen similarity measure would differ by at most 1. The two resulting sets of recommendations were then interleaved and displayed in rows of three, each recommendation containing the name of the community, a link to the community page, and possibly a picture associated with the community.

4.1.3 Observation

During the experiment, any time a set of recommendations was generated and one of the recommendations was clicked on, an observation on the behavior of the user was recorded. Either the user was a member of the base community or they were not, and in both cases, either the user was already a member of the recommended community, the user joined the recommended community after clicking the link, or the user did not join the community after clicking

Measures		Wins	Losses	Ties
L2	MI1	9499	6846	4050
L2	MI2	9576	6880	3821
L2	IDF	9539	6929	3761
L2	L1	9586	6822	3480
L2	LogOdds	11038	6097	2202

Table 1: Wins, losses, and ties for the L2 similarity measure when compared to 5 other similarity measures [7]

the link. The result is six different possible observations for every time a user clicks on a recommendation. These observations formed the data on which the authors based their conclusions.

4.2 Results

In the 18 day period over which the experiment was performed, over 900,000 users clicked on recommendations. The authors considered all cases in which a user was a non-member of the recommended community and clicked on the recommendation. If the user then joined the recommended community, it is referred to as a *conversion*.

In order to compare the relative accuracy of the 6 similarity measures, each conversion was recorded as a win, a loss, or a tie for each of the two similarity measures whose recommendations were being displayed when the conversion took place. Suppose two similarity measures are being used to display recommendations and the user clicks a recommendation and joins the recommended community. Of the two similarity measures whose recommendations were being displayed, the one with the joined community higher on its list of recommendations than the other receives a win, and the other receives a loss. If the two similarity measures rank the joined community equally then it is recorded as a tie.

Wins, losses, and ties were recorded for each pair of similarity measures. Table 1 shows a segment of the overall recorded results, comparing the L2 norm to the five other similarity measures used for this experiment. In the table, the number of wins refers to the number of times the measure on the left (in this case the L2 norm) outperformed the measure on the right, in the manner described previously. As illustrated in the table, the L2 norm outperformed every other similarity measure. Similarly, the MI1 outperformed every other similarity measure besides the L2 norm, the MI2 outperformed every other similarity measure besides the L2 norm and the MI1, and so on, resulting in a distinct order of similarity measures from best to worst: L2, MI1, MI2, IDF, L1, LogOdds.

5. MOVIELENS

A study published in 2008 by Lathia *et al.* compares the accuracy of various similarity measures by applying each measure to a set of movie ratings collected from a movie recommendation site called MovieLens. In this section I will briefly describe the different types of similarity measures that were considered, explain the calculations and observations that were made, and discuss the results of the study.

5.1 Similarity Measures

In the study, 7 similarity measures were compared; these measures will be referred to as: Co-Rated, Concordance, Pearson correlation coefficient (PCC), weighted-PCC, R(0.5,1.0), R(-1.0,1.0), and Constant(1.0). Each measure determines the similarity of two users in the data set, represented by a value in the range [-1.0, 1.0].

The first of the four measures is based on the idea that two users who have both rated the same item are similar to the degree that they both decided to consume and rate the same item. Lathia *et al.* present a simple function to compute the similarity of two users, a and b, based solely on the quantity of rated items:

$$sim(a,b) = \frac{|R_a \cap R_b|}{|R_a \cup R_b|} \tag{16}$$

R_a is the set of items that have been rated by user a.

The second similarity measure is concordance-based. The ratings provided by two users a and b for some item d are *concordant* if either both ratings are higher than the respective user's average rating, or both ratings are lower than the respective user's average rating. If one rating is higher than the user's average and one is lower, then the ratings are *discordant*. For this measure, the similarity of two users was calculated by subtracting the quantity of discordant ratings from the quantity of concordant ratings and dividing by the sum of the two quantities.

The third similarity measure considered in the study was the Pearson correlation coefficient (described in Section 3.2). The authors also included a variation of this measure called the weighted-PCC, however for the sake of space I will omit it from this discussion.

Lastly, the authors included three measures that make no attempt to measure the actual similarity of the two users. One of the measures returned random values in the range of [0.5, 1.0], signifying a high degree of similarity for every pair of users. The second measure returned random values in the range [-1.0, 1.0], signifying a completely random degree of similarity between the two users. The third measure always returned the value 1.0, signifying a perfect agreement between all users.

5.2 Experimental Analysis

To determine the accuracy of the measures described above, the data set of movie ratings was first divided into 5 sets of test data referred to as $u1$, $u2$, $u3$, $u4$, and $u5$. It is important to note that all of the ratings in the data set are on a scale of 0 to 5. Using a function similar to Equation 3 with $sim(u, u_j)$ replaced in turn by each of the seven similarity measures, a predicted rating was calculated for every rating already present in the test data set. Once all predicted ratings were calculated, the differences between the predicted ratings and the actual ratings were averaged and recorded. These averages are displayed in Table 2. For this part of the experiment, all predicted ratings were calculated using the entire set of users as u's k nearest neighbors. Table 3 shows the results of the same calculations, performed with varying values of k, in other words neighborhoods of varying sizes.

5.3 Results

The data obtained from this study points to two main results. The first result is that the accuracy of the predicted ratings generally increases as the size of the neighborhood used in calculating the rating approaches the size of the community. The second result is that the choice of similarity measure to be used in calculating the predicted rating does

Dataset	Co-Rated	Concordance	PCC	R(0.5, 1.0)	R(-1.0, 1.0)	Constant(1.0)
u1	0.7718	0.7992	0.8073	0.7773	0.7812	0.7769
u2	0.7559	0.7825	0.7953	0.7630	0.7666	0.7628
u3	0.7490	0.7706	0.7801	0.7554	0.7563	0.7551
u4	0.7463	0.7666	0.7792	0.7534	0.7554	0.7531
u5	0.7501	0.7715	0.7824	0.7573	0.7595	0.7573
Average	0.7548	0.7781	0.7889	0.7613	0.7638	0.7610

Table 2: The average error of the predicted rating, ordered by dataset [4]

Neighborhood Size	Co-Rated	Concordance	PCC	R(0.5, 1.0)	R(-1.0, 1.0)	Constant(1.0)
1	0.9449	0.9492	1.1150	1.0665	1.0341	1.0406
10	0.8498	0.8355	1.0455	0.9595	0.9689	0.9495
30	0.7979	0.7931	0.9464	0.8903	0.8848	0.9108
50	0.7852	0.7817	0.9007	0.8584	0.8498	0.8922
100	0.7759	0.7728	0.8136	0.8222	0.8153	0.8511
153	0.7725	0.7727	0.7817	0.8053	0.8024	0.8243
229	0.7717	0.7771	0.7716	0.7919	0.8058	0.7992
459	0.7718	0.7992	0.8073	0.7773	0.7812	0.7769

Table 3: The average error of the predicted rating, ordered by neighborhood size, for $u1$ [4]

not affect the accuracy of the prediction when the neighborhood consists of the entire community.

6. CONCLUSION

Looking at two different studies, both aiming to determine which similarity measure results in the most accurate predicted rating, we find conflicting results. The results of the experiment conducted by Spertus *et al.* clearly indicated that predicted ratings are significantly more accurate when calculated using the L2 norm than when using any of the five other similarity measures included in the experiment. In the study performed by Lathia *et al.* however, the results indicated that the choice of similarity measure used in calculating the predicted rating plays little to no role in how accurate the ratings are, and in fact functions that returned random values performed better in some cases than actual similarity functions. Clearly there were significant differences in the two approaches seen in these two studies, such as the types of similarity functions being considered, the kinds of observations on which results were based, and the test data that was operated on, and this may explain the conflicting opinions regarding the utility of similarity functions in recommender systems.

7. ACKNOWLEDGMENTS

I would like to thank my advisor, Kristin Lamberty, for advising me and providing feedback on my paper, and Nic Mcphee and Elijah Mayfield for also providing feedback on my paper.

8. REFERENCES

[1] X. Amatriain, A. Jaimes*, N. Oliver, and J. M. Pujol. Data mining methods for recommender systems. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 39–71. Springer US, 2011.

[2] G. Beliakov, T. Calvo, and S. James. Aggregation of preferences in recommender systems. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 705–734. Springer US, 2011.

[3] J. A. Jacobi, E. A. Benson, and G. D. Linden. U.S. Patent 7,908,183 B2, 03 2011.

[4] N. Lathia, S. Hailes, and L. Capra. The effect of correlation coefficients on communities of recommenders. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 2000–2005, New York, NY, USA, 2008. ACM.

[5] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. Springer US, 2011.

[6] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 291–324. Springer Berlin / Heidelberg, 2007.

[7] E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the Orkut social network. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 678–684, New York, NY, USA, 2005. ACM.

[8] E. W. Weisstein. Least squares fitting. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/LeastSquaresFitting.html. [Online; accessed 3-October-2012].

[9] Wikipedia. Correlation and dependence — wikipedia, the free encyclopedia, 2012. [Online; accessed 14-October-2012].

[10] Wikipedia. Least squares — wikipedia, the free encyclopedia, 2012. [Online; accessed 3-October-2012].

[11] Wikipedia. Weighted mean — wikipedia, the free encyclopedia, 2012. [Online; accessed 4-October-2012].