User-Centric Evaluation of a K-Furthest Neighbor Collaborative Filtering Recommender Algorithm

Alan Said
Technische Universität Berlin
alan@dai-lab.de

Ben Fields
Musicmetric (Semetric Ltd.)
me@benfields.net

Brijnesh J. Jain
Technische Universität Berlin
jain@dai-lab.de

Sahin Albayrak
Technische Universität Berlin
sahin@dai-lab.de

ABSTRACT

Collaborative filtering recommender systems often use nearest neighbor methods to identify candidate items. In this paper we present an inverted neighborhood model, k-*Furthest* Neighbors, to identify less ordinary neighborhoods for the purpose of creating more diverse recommendations. The approach is evaluated two-fold, once in a traditional information retrieval evaluation setting where the model is trained and validated on a split train/test set, and once through an online user study (N=132) to identify users' perceived quality of the recommender. A standard k-nearest neighbor recommender is used as a baseline in both evaluation settings. Our evaluation shows that even though the proposed furthest neighbor model is outperformed in the traditional evaluation setting, the perceived usefulness of the algorithm shows no significant difference in the results of the user study.

ACM Classification Keywords

H.3.3 Information Storage and Retrieval: Information Search and Retrieval - Information filtering, Retrieval models, Selection process; H.5.1 Multimedia Information Systems: Evaluation/methodology

General Terms

Algorithms, Design, Experimentation, Human Factors, Measurement

Author Keywords

Recommender Systems; Evaluation; Long Tail; Nearest Neighbors; Diversity; Usefulness; Collaborative Filtering; User-centric Evaluation

INTRODUCTION

In recent years the methods used in item-centric recommender systems have continued a steady increase in the accuracy of recommended items. Assuming the goal of a system is accurate prediction of ratings, this obviously represents progress. But it is worth asking, are we missing something? The way these systems are understood is explicitly defined via the way they are tested; focus on predictive accuracy can be sufficiently narrow as to miss the broader point. A system that recommends *items* to *users* needs to be tested to determine the rate of satisfaction of its users. Without measuring user satisfaction, there is no means to map the much more common automated measures (e.g. predictive accuracy) back to the users of an evaluated system [13].

The most direct way to do this is via user study. Specifically, with the goal in mind of examining effects that may be lost to standard accuracy measures, this paper conducts a comparative user study looking at two distinct approaches to item-based recommendation. Both of these approaches come in the form of the neighborhood model. The first of the these two models is k-nearest neighbors (kNN), where *neighborhoods* of items are created based on item rating history. Each user's neighborhood is used as a basis to generate future recommendations. kNN is one of the most prevalent algorithms used in item-based recommender systems and has been explored in depth by e.g. [4, 6, 8, 25].

While kNN is widespread and understood, it is not without problems. Among these are: lack of *diverse* recommendations, the maximal distribution of recommended items across the space of all possible items, and of the *obviousness* of recommendations. These are all aspects which are not captured in standard recommender system evaluation measures. It is with these deficits in mind that the second algorithm has been selected. Starting with the core idea of kNN, that of a neighborhood of users, k-furtherest neighbors (kFN) pulls kNN inside-out; this approach creates neighbors of maximally dissimilar users and then recommends items that a user's neighbors are most likely to dislike [23]. It is hypothesized that this double-negative style of similarity-based recommender will perform better in terms of both novelty and obviousness, while maintaining an acceptable level of predictive accuracy. In order to evaluate the actual, *perceived*, quality of the recommendations, the user study focuses less on traditional evaluation metrics, and instead examines the algorithms from a *user-centric* perspective.

CSCW'13, February 23–27, 2013, San Antonio, Texas, USA.

RELATED WORK

Two relevant channels of previous work provide background for this research. First, many existing approaches to increasing recommendation accuracy of recommender systems do so at the expense of diversity and novelty, and by increasing the computational cost of the resulting algorithms [12, 24]. Second, understanding the needs and opinions of your users (and how best to gather these) is critical to specifying an algorithm that adds significant value to a system from the user's perspective.

Accuracy vs. Diversity

One of the key reasons why lack of diversity, though long noticed in recommender systems, remains a problem, is tied to the observation that algorithms modified to increase diversity have as a byproduct lower predictive accuracy.

The broad justifications of the importance of diversity are outlined in [26]. Here diversity is held up in contrast to *similarity* as a counter aim of recommender systems. The paper proposes and evaluates some case-based reasoning (CBR) approaches to recommendation that emphasize diversity. While successful in its stated goals, the underlying CBR approach itself can be difficult to scale and quite complex. Further, the dichotomy set up in this work of similarity versus diversity can be seen as false if the task being considered is 'recommend the items a user will like'. The goal is neither to find items that are similar to each other or spread over the item space, rather to satisfy a user.

This idea of diversity tradeoffs not really being tradeoffs at all is furthered in [29]. In this work a *hybrid* of multiple algorithms is used to increase both predictive accuracy and diversity in a single system. While the output of such a system is excellent, it is not without cost. The effective complexity of such a system is quite high, being the combination of the underlying pieces of the hybrid model. However, this system does achieve its aims and in cases where the complexity is less important (e.g. domains with small datasets) it presents an effective solution.

Approaches which, like ours, try to raise the perceived values at the cost of standard accuracy metrics include [30] which defines the *intra-list similarity* metric as a means of measuring diversity in recommendations. Similarly [19] defines the metrics *unexpectedness* and *unexpectedness_r* for measuring the level of diversity a recommendation algorithm achieves. More recently, the Auralist framework [28], successfully creates a *serendipitous*, *diverse* and *novel* hybrid recommendation engine by combining several algorithms, similar to [29] the cost of this approach is the complicated underlying model.

There are of course multiple ways of considering diversity. While in the remainder of this paper we take diversity in terms of fixed-in-time snapshots of our dataset, diversity is considered via user ratings over time in [16]. The resulting model of diversity in time-ordered sequence highlights user-behavioral patterns of great interest when constructing a recommender system. The work reveals a number of insights into the effects of user behavior on recommended item diversity, such as an inverse relationship between profile size (i.e. items rated by a user) and the diversity of items recommended to a user.

User-Centricity and Satisfaction

When considering metrics for evaluation of a recommender system, keeping the user central and focusing on user satisfaction, is vital. Bringing people into the equation does complicate methods and evaluation though, and raises a number of issues.

One early way that recommender systems' users were explicitly considered was with regard to trust and privacy [15]. This work formalized various issues around trust in recommender systems especially concerning privacy in sharing preferences. Perhaps most relevantly, the paper considered how users may be prone to increase instances of explicit bias in the expression of ratings when they do not trust a recommender.

A means to increase a user's trust in a system is to optimize the utility or usefulness of the recommendations a system offers to a user. Previous research has concluded that the reliance on predictive accuracy as the most critical measure is detrimental to the overall utility of a recommender system [18]. Thus it makes a great deal of sense to consider a recommender using a more rich and diverse set of metrics, especially focused on the user.

This presents another problem though. If we give up a reliance on standard automatic measures, what evaluation should be done and why? The *ResQue* framework is one such solution [20]. This framework presents a unified and extremely rigorous approach to user-driven evaluation, though it is time intensive for participants and may be overly costly for focused hypothesis testing.

NEIGHBORHOOD MODELS

Commonly memory-based collaborative filtering models utilize the k-nearest neighbor approach to identify candidate items [7]. This approach uses a neighborhood of similar users to identify and recommend items to users. Neighborhoods are created by, for each user, finding users within a certain *similarity*. Each user's neighborhood contains the k users' who's similarity is highest, an abstract outline of the algorithm is shown in Algorithm 1. Recommendations are then created by iterating over all the items rated by the user's neighbors which have not been seen by the user, and estimating the preference of each item by averaging all neighbors' rating values for each item multiplied with each neighbor's similarity value. Ultimately, the highly rated items of the most similar neighbors are recommended. The similarity function for neighbor identification can be freely selected.

Algorithm 1: The k-nearest neighbor algorithm

Input: set $\mathcal{U}$ of users
Input: number k of neighbors
Output: k neighbors for each user $u \in \mathcal{U}$
foreach $u \in \mathcal{U}$ **do**
  foreach $u' \in \mathcal{U} \setminus \{u\}$ **do**
    $s_{u,u'} = \text{similarity}(u,u')$;
  select k neighbors $u' \neq u$ for u with largest $s_{u,u'}$

Common similarity measures used for neighborhood creation are cosine similarity and Pearson product-moment correlation coefficient, when working with data containing ratings.

The Person correlation is calculated by comparing the rating values of two users' co-rated items,

$$\mathcal{P}(u,v) = \frac{\sum_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in \mathcal{I}_{uv}} (r_{vi} - \bar{r}_v)^2}} \quad (1)$$

where u and v are the two users, I_{uv} the intersection of user u's and v's rated items, and r the rating score, e.g. 4 on a scale from 1 to 5.

For binary co-occurrence data, common similarity measures include the Jaccard and Tanimoto coefficients [10, 11].

Nearest & Furthest Neighbors

Nearest neighborhood methods have been frequently used in information retrieval and information theory for many years [9], with satisfactory results. However, due to an inherent popularity bias in consumer-item interaction data [17], it is possible that nearest neighbor models recommend items which are merely popular and not explicitly personalized for the user [21,22]. Further, Bourke et al. found that the method of selecting nearest neighbors has little effect on the perceived quality of the recommender system itself [6].

Inverting the k-nearest neighbor algorithm into a k-furthest neighbor algorithm could serve as a means to mitigate the popularity bias and increase diversity, without adding more complexity (compared to a k-nearest neighbor algorithm) to a recommender system.

The motivation behind the k-furthest neighbor approach is simple. Due to the inherent bias towards popular items in the consumption patterns of people, and the positive rating bias of popular items [17,22], basic similarity measures can falsely create similarities solely based on popular items, instead of on actual taste [22]. In order to mitigate this effect, the furthest neighbor approach is used. The concept is based on an inverted similarity model, which finds items disliked by those least similar to a given user. This has the benefit of not being sensitive to bias induced by highly rated popular (i.e. obvious or non-novel) items. The approach first identifies the most dissimilar users to each user, to create *dissimilar* neighborhoods. In every neighborhood it then identifies the most disliked items and recommends these. This double negation creates a more diverse, yet personalized, set of recommendations.

Inverting Similarity Metrics

In order to create neighborhoods of the least similar users, a dissimilarity measure must be defined. Dissimilarity cannot simply be expressed by negating or inverting traditional similarity measures, as this would potentially create neighborhoods of completely unrelated, or disjoint, users. In order for a dissimilarity measure to work accurately, we propose the introduction of the following two constraints: (1) two users who share no co-rated items can be considered dissimilar by metrics such as the cosine similarity and the Pearson correlation. However, this dissimilarity is not based on an actual disagreement in taste, rather just on the two users not having an intersection of rated items. A similarity measure like this could result in extremely large neighborhoods of completely disjoint users. In order to circumvent these "false" dissimilarities, we propose a *minimum co-rated items* constraint. This constraint means that in order for two users to be considered dissimilar, they must both have co-interacted with a number of items. The constraint could be extended to treat popular items differently (e.g. lower weights for popular items). This, or similar, constraints are commonly used in kNN-based recommendation approaches as well in order to heighten (and strengthen) the level of similarity between prospective neighbors.
The second constraint, (2) the *opposed ratings* constraint, is introduced to secure an actual dissimilarity and is data-specific and relates to a rating scale. Consider this example:

Suppose that R is a set of ratings r_{ui} submitted by users $u \in U$ for items $i \in I$. Ratings may take values from the discrete set $\{1, 2, 3, 4, 5\}$ of rating scores. Typically, ratings are known only for few user-item pairs.
If u_a's set of rated items is (i_a,i_b,i_c,i_d) rated $(1, 3, 4, 1)$ respectively and u_b's set of rated items is (i_a,i_b,i_c,i_e) rated $(5, 3, 2, 2)$ respectively, then the intersection of items (i_a, i_b, i_c) creates a basis for a dissimilarity calculation. (i_a, i_c) are rated on the opposite end of the rating scale, i_b is however rated in the middle of the rating scale and has no antipode. This rating, as well as those for i_d and i_e, are thus not taken into consideration when calculating a dissimilarity.

Given this opposed ratings constraint, the minimum co-rated items constraint does not consider items rated with the middle value of the rating scale, as the rating cannot be inverted. This rating-based constraint should be modeled according to the rating scale used in the dataset

or system. For a more precise dissimilarity, the matching of opposing ratings could be performed on a per-user level. This would be done by first identifying the personal rating scale of each user and subsequently matching this onto other users' personalized rating scales (e.g. some users may only use ratings from each end of the rating scale, others may predominantly use ratings from the middle, while others use the full rating scale meaning the matching of 2 to 4 might not be a true match for all users).

Applying this to the Pearson correlation as presented in Eq. (1), the similarity would be calculated as:

$$\acute{\mathcal{P}}(u,v) = \begin{cases} \mathcal{P}(u,v') & : \text{ if } |\mathcal{I}^*_{uv'}| \geq m \\ 0 & : \text{ otherwise} \end{cases}$$

where m is the number of minimum co-rated items, v' contains the inverted ratings of user v according to Eq. (2), and $\mathcal{I}^*_{uv'}$ is the list of co-rated items, excluding items rated with the "middle" score as specified by the opposed ratings constraint, i.e. (i_a, i_c) in the example above.

Each of user v's rating r_{vi} is inverted according to the following process

$$\acute{r}_{vi} = r_{max} - r_{vi} + r_{min} \tag{2}$$

where r_{max} is the highest possible score in the rating scale, r_{vi} is user v's rating on item i and r_{min} is the lowest possible score in the rating scale. In the example above u_b's ratings would be $r_{u_b,i_a} = 5 - 5 + 1 = 1$ and $r_{u_b,i_c} = 5 - 2 + 1 = 4$ turning the rating values 5 and 2 to 1 and 4 respectively.

A similar approach could be applied to many other recommendation scenarios, e.g. in a Singular Value Decomposition-based recommender [24], the ratings of the candidate user could be inverted prior to training the recommender, followed by recommending the items which, by the recommender, are deemed as least suitable for the user.

EXPERIMENTS & EVALUATION

To evaluate the quality of the furthest neighbor model, the Movielens[1] 10 million ratings dataset was used. Two sets of evaluation experiments were conducted: (1) a *traditional* setup using a portion of the ratings for training the model, and a portion used for validation, and (2) a *user-centric* evaluation method based on an online study evaluating the *perceived usefulness* of recommended items and users' overall satisfaction with the system.

Both the nearest neighbor and the furthest neighbor recommenders were implemented using the Apache Mahout[2] framework. Additionally, a random recommender, not taking the users' ratings into consideration when finding candidate items was also implemented using the same framework.

To create the neighborhoods, the Pearson correlation was used for the nearest neighbor approach. The furthest neighbor approach was created by inverting the Pearson correlation and applying the minimum co-rated and opposed ratings constraints as described in the previous section. The minimum co-rated items constraint was set to 5 based on small empirical experiments, as well as not to exclude users with few ratings. It should be noted that the higher this number is, the more the calculated dissimilarity accurately reflects the actual difference in taste, however it also minimizes the number of possible neighbors. In the nearest neighbor scenario, the minimum co-rated constraint was left to the default value set in the Mahout recommender library, 2. However, as the nearest neighbor approach requires two users to have *similar* taste, even a lower number should reflect some level of similarity.

The neighborhood size for both approaches was set to 30. Even though a larger neighborhood (up to approximately 100 neighbors [10]) can result in better performance in terms of accuracy metrics, the neighborhood size was kept low to minimize memory usage and keep the recommendation process fast for the survey.

An additional baseline recommender, a random recommender, not taking the users' ratings into consideration, was used as well. The baseline algorithm, a normal k-nearest neighbor algorithm, was chosen to show the duality of both approaches – thus showing the gain in diversity compared to a traditional recommendation setting. The random recommender was used to show that effect of the furthest neighbor algorithm were not random.

As mentioned in the previous section, similar inverted similarities could potentially be applied to other recommendation algorithm. Our choice of the k-nearest neighbor algorithm was based on it's illustrative qualities as well as on hardware limitations. Experiments with matrix factorization-based recommendation approaches required 5 to 7 minutes for retraining of the recommender. The participants in the study would need to wait for their recommendation for this period of time, which would likely have detrimental effects on the number of users who completed the study.

Dataset

The Movielens 10 million ratings dataset, provided by the Grouplens research group, contains $10,000,054$ ratings by $69,878$ users on $10,677$ movies. The dataset additionally contains the titles and production years of movies.

Using the titles, additional information such as actors, directors, plot, etc., was collected from the Internet Movie Database[3]. This data was used to display additional information about the movies in the survey.

[1] `http://www.grouplens.org/system/files/ml-10m-README.html`

[2] `http://mahout.apache.org/`

[3] `http://www.imdb.com`

Traditional Evaluation

A traditional *offline* evaluation approach was used to assess the recommendation quality of the furthest neighbor approach. Offline in this sense meaning not interacting with users during evaluation, i.e. a data-centric or system-oriented evaluation. For this purpose, the above mentioned Movielens dataset was split into training and validation sets which were subsequently used to train and validate the furthest neighbor as well as the nearest neighbor model for comparative purposes.

For each user with at least $2 \times N$ ratings, precision and recall accuracy at N were evaluated for $N = \{5, 10, 100, 200\}$. In each of the cases 20% of the ratings of each user in the dataset were included. For the validation, only movies having been rated above each user's average rating plus 0.5 of the user's standard deviation of rating scores were considered. Thus making sure only positively rated items were treated as true positive recommendations. The remaining ratings were included in the training set.

User-Centric Evaluation: A Movie Recommendation Study

In order to make a real-life estimate of the perceived quality of the furthest neighbor approach, a user study was performed. The study consisted of two steps, first participants were asked to rate a minimum of 10 movies from a page showing 100 randomly selected movies out of the 500 most rated movies in the Movielens dataset, shown in Fig. 1. Due to the large number of movies in the dataset (more than $10,000$), we chose randomly among the 500 most popular movies in order to show the participants movies which they would most likely be familiar with[4].

Having rated at least 10 movies, recommendations based on each participant's ratings were generated and a page containing the top 10 recommendations and a set of questions was shown, see Fig. 2. The recommender engine providing the survey with recommendations was similar to the one used in offline evaluation. The difference being that the complete Movielens dataset was used for training as this scenario did not include any offline evaluation. The time to generate recommendations for one user, using either of the neighborhood-based algorithms, was under one minute for users having rated a large portion of the 100 movies shown; this response time was considerably lower for those only having rated the minimum allowed number of movies (10). The random recommender did not need any noticeable time for generating recommendations, to create the illusion of personalized recommendations, one of the neighborhood-based

[4]In an earlier version of the survey, which was tested on a small number of users, random movies from the whole set of movies were shown instead. The vast majority of the reactions collected from these early testers were that most of the movies were completely unknown to them, thus we chose to present a random set from the 500 most popular movies instead.

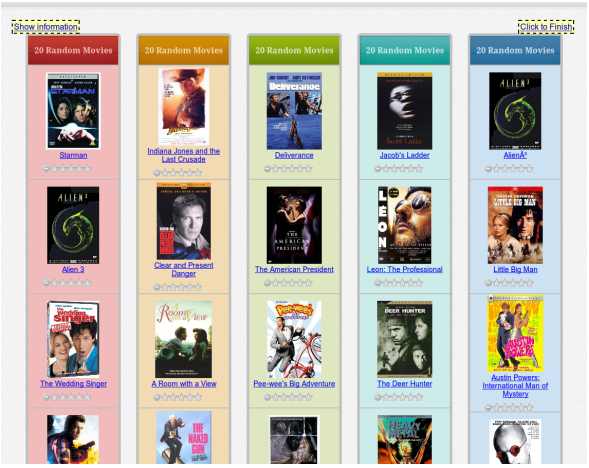


Figure 1. The movie rating page where participants were asked to rate movies. The page contains a random selection of 100 of the 500 most popular movies in the Movielens dataset.

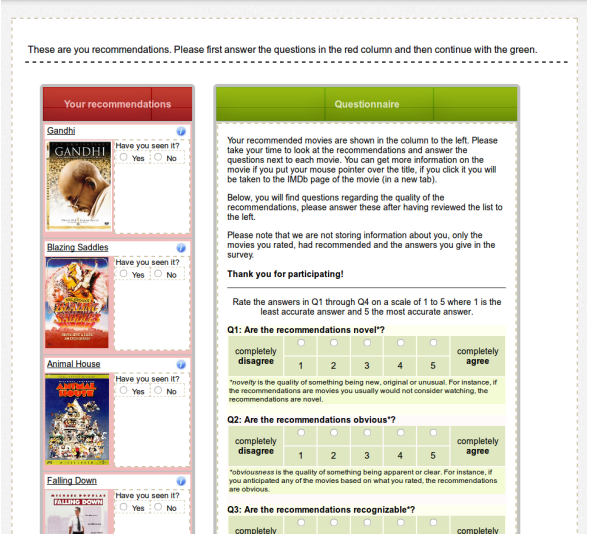


Figure 2. The questions presented to the participants of the survey after having rated a minimum of 10 ratings.

algorithms was trained (but not used) in parallel to serve as a timer.

The selection of algorithm to be used for each participant was proportional to a randomized variable. For 40% of the participants, the recommendations were to be based on the traditional k-nearest neighbors approach. Another 40% of the participants were to be given recommendations based on the k-furthest neighbors approach, and finally 20% of the participants were to be presented with randomized recommendations, not taking their initial ratings into consideration.

Questionnaire

For each of the 10 recommended movies (shown in the leftmost column in Fig. 2), participants were asked to answer the following questions:

Have you seen the movie?

if Yes: Please rate it (5 star rating)

if No:

1. Are you familiar with it? (y/n)
2. Would you watch it? (y/n)

Additionally, participants were asked to answer a set of 8 question regarding the complete set of recommended items. The questions were:

1. Are the recommendations novel?
2. Are the recommendations obvious?
3. Are the recommendations recognizable?
4. Are the recommendations serendipitous?
5. Are the recommendations useful?
6. Pick the movie you consider the best recommendation
7. Pick the movie you consider the worst recommendation
8. Would you use this recommender again?

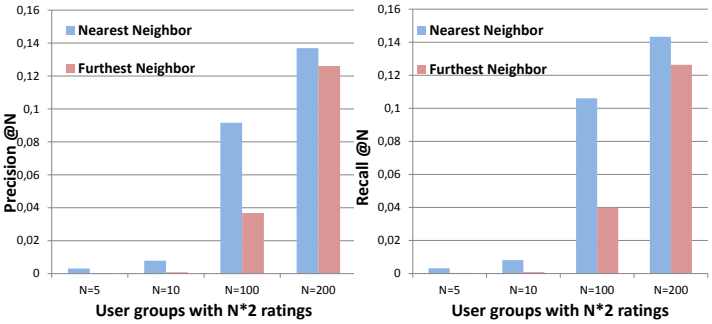
The participants were asked to answer question 1 through 5 by stating the level of agreement, from strongly disagree (1) to strongly agree (5). A short description was given for the terms *novel*, *obvious*, *recognizable* and *serendipitous* in order to mitigate erroneous answers based on misunderstanding of the question. Additionally a field for comments was placed on the bottom of the survey.

Submission of the answers was only possible after all questions had been answered.

The questionnaire was designed to create a truthful picture of users' perceived usefulness of the recommendations, considering aspects such as choice overload [5], construction of questions [13] and similar perception-related concepts [11, 27].

The link to the survey was circulated on social media sites (e.g. Facebook, Twitter, Google+) during the last two weeks of March 2012. People from several professions (computer science professionals, lawyers, business management professionals) were asked to circulate the link in their networks in order to gain participants from several communities, thus attempting to minimize any biasing effects which could surface in answers from a homogeneous community. The data used in this paper was collected in early April 2012, by then a total of 132 participants had completed the survey.

Out of the 132 participants in the study, 47 (36%) were presented with recommendations based on the nearest neighbor model, 43 (33%) based on the furthest neighbor model and 42 (32%) based on the random recommender (due to the randomized seeding of the algorithm selection, the percentages are somewhat different from the intended 40%, 40%, 20% distribution).



(a) Precision@N (b) Recall@N

Figure 3. The Precision@N and Recall@N values for users with $2 \times N$ ratings for the furthest and nearest neighbor models for the traditional offline evaluation approach.

Knijnenburg et al. claim that "at least 20 users" per condition should be adequate for being able to mine statistically sound data from a user study [14], indicating the amount of participants in our study should be sufficient.

No demographic data except for location (reverse lookup via IP address) was collected. The participants of the survey came from a total of 19 countries on 4 continents. Participants from Germany, Ireland and Sweden were most prominent, in descending order.

RESULTS & DISCUSSION

The furthest neighbor approach is outperformed by its nearest neighbor counterpart in terms of precision and recall when comparing the performance in an offline evaluation scenario. However, when comparing the results obtained from the user-centric evaluation, the outcomes are different.

Traditional Evaluation Results

For small values of N, nearest neighbor outperforms furthest by a factor of 8.4 in terms of precision(3.14×10^{-3} and 3.75×10^{-4} for $N = 5$ respectively). For larger values the difference in performance grows smaller, with the nearest neighbor approach outperforming the furthest neighbor approach by 8.6% at $N = 200$. Fig. 3(a) shows the values for all four evaluated Ns. Performance in terms of recall is almost identical to precision, as shown in Fig. 3(b). The results of the random recommender are not shown as they were several orders of magnitude lower than both neighborhood-based recommenders.

The nearest neighbor approach outperforms the furthest one in traditional information retrieval metrics. However, the recommenders seem to be almost entirely disjoint in terms of recommended items, as shown in Table 1, i.e. users get completely different items recommended from each of the neighborhood-based recommender algorithms.

	$N = 5$	$N = 10$	$N = 100$	$N = 200$
items	42	75	7,925	27,295
%	0.0013	0.0012	0.74	3.2

Table 1. The absolute number of intersecting items for the nearest and furthest neighbor recommender and the percentage of intersecting items from the whole set of recommended items (i.e. $N \times |U|$).

The almost complete orthogonality of recommended items indicates that both approaches are complementary, and if optimally combined into one, the resulting precision value would be the sum of the precision of both approaches. A recommender engine which would be able to create an ensemble recommender by estimating the true positive recommendations in each algorithm prior to recommendation would in this case serve as an optimal neighborhood algorithm. Ensembles are however commonly built by combining the intersecting items from both sets of recommendations, which becomes infeasible in this scenario.

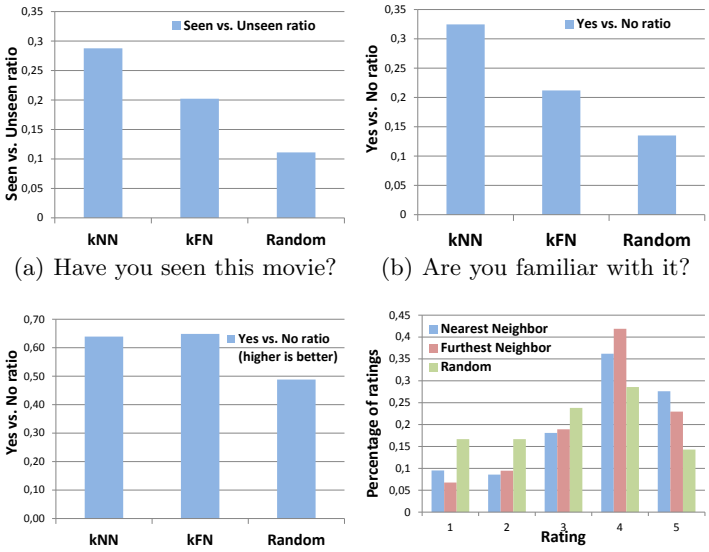
User-Centric Results

As stated earlier, participants were presented with recommendations from either a nearest neighbor recommender, a furthest neighbor recommender, or a random recommender. The results of all three approaches are presented throughout this section. All t-test values presented are either for two-tailed t-tests (when only two distributions are compared), or ANOVA testing using two-tailed t-tests with Bonferroni correction ($n = 2$ for two comparisons) [2].

The results of the survey are either related to each of the movies which had been recommended, or to the whole list of recommended movies. Fig. 4 shows the results from the movie-related questions.

Considering whether the participants had seen the recommended movies or not, the ratio of seen vs. unseen was highest for the kNN approach, followed by the kFN, with the random recommender having the lowest ratio ($p < 0.1$), as shown in Fig. 4(a). Looking at this in the context of the ratings given to seen movies (Fig. 4(d)) per algorithm, we can conclude that the kFN algorithm indeed shows more unknown movies, whilst remaining an almost identical average rating score compared to kNN algorithm. This is interpreted as an indication of the kFN recommender being able to attain a comparable level of user satisfaction, with a higher level of diversity. The random recommender recommends more unseen movies, with the cost of a lower overall average rating.

Similar observations can be made for whether or not the participants were familiar (Fig. 4(b)) with the recommended movies ($p < 0.05$), and whether they would consider watching (Fig. 4(c)) the movie (showing a trend towards differences between both neighborhood approaches and random at $p < 0.1$).



(a) Have you seen this movie? (b) Are you familiar with it?

(c) Would you watch it? (d) Ratings of watched movies.

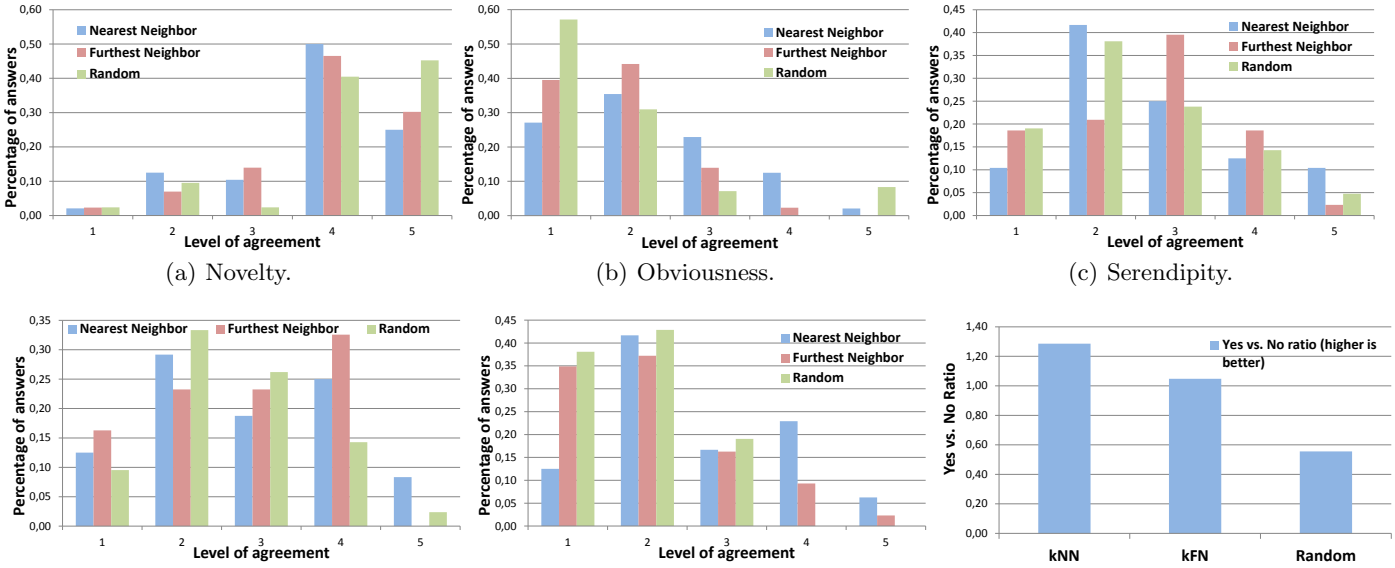
Figure 4. The ratio of yes versus no answers to the questions "Have you seen this movie?" (Fig. 4(a)) and "Are you familiar with it?" (Fig. 4(b)) both of which showing the diversity and non-obviousness of the recommended items. "Would you watch it?" (Fig. 4(c)), and the average ratings of recommended movies previously seen by the users (Fig. 4(d)) showing the general quality of the recommendations.

These observation, in conjunction with the the significance of tests (kNN/kFN vs. random $p < 0.05$, kFN vs. kNN $p > 0.1$ failing to reject the assumed null hypothesis of there not being a difference between the rating quality of both neighborhood-based recommenders) indicate that the overall quality of the kFN recommender seems to be on par with its kNN counterpart.

Looking at the results of the questions related to the full set of recommended movies (the leftmost column in Fig. 2), shown in Fig. 5, the trend continues. For instance, Fig. 5(a) shows that the perceived novelty of kFN matches that of kNN. Novelty, by it's nature implies new, or unheard of movies explaining the relatively high score of the random recommender. For obviousness, the distribution is almost opposite to that of novelty. The furthest neighbor model seems to be slightly less obvious than the nearest neighbor counterpart, high obviousness here being interpreted as something negative as obvious items could be found by the user without the help of a recommender system.

Generally, the differences between the neighborhood models and the random recommender show (at a minimum) trends towards being different, except for when it comes to serendipity (Fig. 5(c)) and novelty (Fig. 5(a)). We believe this is based on two factors: (1) the non-trivial meaning of the words (especially concerning serendipity for non-native English speakers[5], and (2) the

[5]In 2004 serendipity was voted the third most difficult English word to translate [3]



(a) Novelty. (b) Obviousness. (c) Serendipity.

(d) Usefulness. (e) Recognizability. (f) Would you use the system again?

Figure 5. The distributions of agreement levels of the non-movie related answers (i.e. those shown in the right column of Fig. 2) for each algorithm evaluated in the user study. In Fig. 5(a) to Fig. 5(e) the values $\{1, 2, 3, 4, 5\}$ correspond to {*most disagree*, ..., *most agree*} respectively. In Fig. 5(f) the bars correspond to the ratio of yes vs. no answers.

Alg.	rating	novel	obvious	recogn.	serend.	useful
kNN	3.64	3.83	2.27	2.69	2.71	2.69
kFN	3.65	3.95	1.79	2.07	2.65	2.63
Rand.	3.07	4.17	1.64	1.81	2.48	2.24

Table 2. The average rating given by users for movies they had seen which were recommended by the different algorithms as well as the average of the agreement levels for novelty, obviousness, recognizability, serendipity and usefulness respectively.

inherent difficulty of rating. something which is not known (which both novelty and serendipity at least in part cover). Analogously, the assumed null hypothesis of there not being a difference between kNN and kFN fails to be rejected within significant levels of p.

Regarding the level of agreement on the usefulness (Fig. 5(d)) across all the algorithms, Table 2 shows both neighborhood models performing very similarly, t-test comparisons of the distributions show a p value close to one for comparison between neighborhood models and $p = 0.16$ for the comparisons between random and each of the neighborhood models.

However, recognizability is interpreted differently depending on the algorithm (two-tail t-tests give $p < 0.05$ between nearest and furthest, $p < 0.01$ between nearest and random, and a non-significant $p = 0.19$ between furthest and random). It seems the furthest neighbor recommender recommends less known items than the nearest neighbor does, which confirms the results shown in Fig. 4(b). We believe this provides, at least in part, an explanation to the lower precision values in offline evaluation, as less known items are less likely to have been seen by users, and are thus less likely to appear in the validation set.

Finally, on the question of whether the participants would use a recommender system like the one evaluated, there is a slight bias towards the nearest neighbor recommender, i.e. Fig. 5(f). However, this goes in line with users' natural bias towards known items as well trust-related aspects, e.g. before recommending unknown items a system should establish trust with the user [1, 11]. Establishing trust requires repetitive use of a system, which was not possible in the scope of the conducted study.

We believe that given the reported results, precision accuracy levels alone cannot be used to accurately evaluate an approach which is intended to recommend diverse items, e.g. items from the long tail. The results of the user study seem to confirm that even though the nearest neighbor approach outperforms the furthest neighbor approach in terms of traditional accuracy metrics, the general perceived usefulness of the recommenders by the users does not seem to differ substantially. Even though some of the reported t-test p-values show significance within a large error margin ($p > 0.1$), there is a definite trend pointing to similar perceived usefulness of both neighborhood approaches when considering all reported results in context.

Due to the fact that the nearest and furthest neighbor recommender are practically orthogonal in terms of recommended items, as shown in Table 1, this type of evaluation can have a higher positive effect than it would have had for two algorithms which recommend a similar set

of items. In contrast, if comparing two recommendation algorithms that create less disjoint recommendation sets, i.e. by replacing the furthest neighbor approach with an algorithm recommending more "standard" items, it is reasonable to assume that the perceived usefulness would have a bias towards the algorithm that performs better (in terms of precision, recall, and similar metrics).

CONCLUSION

In this work we have presented an extensive analysis of the recommendation quality of both a nearest neighbor and a furthest neighbor algorithm. We evaluated the quality of these algorithms through traditional information retrieval metrics as well as through a user study in order to gain insight on the perceived usefulness of the algorithms. In terms of precision and recall, the nearest neighbor algorithm outperforms the furthest neighbor approach. The lists of recommended items are however almost completely disjoint due to the nature of the furthest neighbor approach, which recommends more non-obvious and diverse items. By analyzing the feedback obtained from a user study with more than 130 participants we have shown that a higher predictive accuracy of a recommender system, in this scenario, does not increase the perceived usefulness of the recommender system, from the users' perspective.

From the answers obtained in the study, we were able to recognize several aspects that point to the notion that *at least* similar satisfaction can be obtained with lower precision, e.g. proportionally the same amount of users claimed they would watch a movie recommended by the lower scoring furthest neighbor approach than a movie recommended by the nearest neighbor approach - Fig. 4(c). Similarly, the furthest neighbor recommender and nearest neighbor recommender received almost identical overall rating values, despite the differences in precision and recall, shown in Table 2.

ONGOING & FUTURE WORK

We believe that this model of evaluation of diverse and other non-obvious items needs to be analyzed in order to gain further insight on when traditional information retrieval accuracy metrics cannot express the quality of a recommender system accurately. For this purpose, we plan on extending the survey to include several more types of recommender algorithms and more questions. Specifically, we intend to study the perceived usefulness of similar algorithms (in terms of how recommendations are found), but which still perform differently in standard metrics. An extended study will be conducted within a multi-algorithm recommender system we have developed for user-centric evaluation. In this system we hope to mitigate any effects of the level of trust, or lack of trust, the users have towards the system by continuous interactions with the recommender system.

Furthermore, we believe that a similar approach could be applied to most recommendation algorithms in order to generate more diverse and non-obvious recommendations, i.e. by inverting the ratings of a user according to Eq. (2) and recommending the items found to be least likely to be liked by the user. This is however outside of the scope of this paper and left as a possible avenue for future work.

Additionally, we are investigating the possibilities of creating an ensemble recommender using both types of neighborhoods.

ACKNOWLEDGMENTS

The authors would like express their gratitude to all participants of the study and everyone who helped by distributing it in their communities, and to Bart Knijnenburg from UC Irvine and Michael Meder from TU Berlin for their help with conceptualization, implementation and evaluation of the user study.

REFERENCES

1. *FilmTrust: movie recommendations using trust in web-based social networks*, vol. 1 (2006).
2. Abdi, H. *Bonferroni and Sidak corrections for multiple comparisons*. Sage, 2007.
3. Avérous, Y. Untranslateable words - serendipity. *Journal of the Northern Californa Translators Association: Online Edition* (2005).
4. Bellogin, A., and Parapar, J. Using graph partitioning techniques for neighbour selection in user-based collaborative filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, RecSys '12, ACM (New York, NY, USA, 2012), 213–216.
5. Bollen, D., Knijnenburg, B. P., Willemsen, M. C., and Graus, M. Understanding choice overload in recommender systems. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, ACM (New York, NY, USA, 2010), 63–70.
6. Bourke, S., McCarthy, K., and Smyth, B. Power to the people: exploring neighbourhood formations in social recommender system. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, ACM (New York, NY, USA, 2011), 337–340.
7. Breese, J. S., Heckerman, D., and Kadie, C. *Empirical analysis of predictive algorithms for collaborative filtering*, vol. 461. San Francisco, CA, 1998.
8. Candillier, L., Jack, K., Fessant, F., and Meyer, F. State-of-the-art recommender systems. *Collaborative and Social Information Retrieval and AccessTechniques for Improved User Modeling* (2009), 1–22.
9. Cover, T., and Hart, P. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on 13*, 1 (january 1967), 21 –27.

10. Herlocker, J., Konstan, J. A., and Riedl, J. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr. 5*, 4 (Oct. 2002), 287–310.

11. Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. Evaluating collaborative filtering recommender systems. *Trans. Inf. Syst. 22*, 1 (2004).

12. Karypis, G. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, ACM (New York, NY, USA, 2001), 247–254.

13. Knijnenburg, B., Willemsen, M., Gantner, Z., Soncu, H., and Newell, C. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction 22* (2012).

14. Knijnenburg, B. P., Willemsen, M. C., and Kobsa, A. A pragmatic procedure to support the user-centric evaluation of recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, ACM (New York, NY, USA, 2011), 321–324.

15. Lam, S., Frankowski, D., and Riedl, J. Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In *Emerging Trends in Information and Communication Security*, G. Müller, Ed., vol. 3995 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2006, 14–29.

16. Lathia, N., Hailes, S., Capra, L., and Amatriain, X. Temporal diversity in recommender systems. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, ACM (New York, NY, USA, 2010), 210–217.

17. Marlin, B. M., and Zemel, R. S. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, ACM (New York, NY, USA, 2009), 5–12.

18. McNee, S. M., Riedl, J., and Konstan, J. A. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI '06 extended abstracts on Human factors in computing systems*, CHI EA '06, ACM (New York, NY, USA, 2006), 1097–1101.

19. Murakami, T., Mori, K., and Orihara, R. Metrics for evaluating the serendipity of recommendation lists. In *Proceedings of the 2007 conference on New frontiers in artificial intelligence*, JSAI'07, Springer-Verlag (Berlin, Heidelberg, 2008), 40–46.

20. Pu, P., Chen, L., and Hu, R. A user-centric evaluation framework for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, ACM (New York, NY, USA, 2011), 157–164.

21. Rafter, R., O'Mahony, M. P., Hurley, N. J., and Smyth, B. What have the neighbours ever done for us? a collaborative filtering perspective. In *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH*, UMAP '09, Springer-Verlag (Berlin, Heidelberg, 2009), 355–360.

22. Said, A., Jain, B. J., and Albayrak, S. Analyzing weighting schemes in collaborative filtering: Cold start, post cold start and power users. In *27th ACM Symposium On Applied Computing (SAC '12)*, ACM (New York, NY, USA, 2012).

23. Said, A., Kille, B., Jain, B. J., and Albayrak, S. Increasing diversity through furthest neighbor-based recommendation. In *Proceedings of the WSDM'12 Workshop on Diversity in Document Retrieval (DDR'12)* (2012).

24. Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. T. Application of dimensionality reduction in recommender system – a case study. In *WebKDD Workshop at the ACM SIGKKD* (2000).

25. Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. The adaptive web. Springer-Verlag, Berlin, Heidelberg, 2007, ch. Collaborative filtering recommender systems, 291–324.

26. Smyth, B., and McClave, P. Similarity vs. diversity. In *Case-Based Reasoning Research and Development*, D. Aha and I. Watson, Eds., vol. 2080 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2001, 347–361.

27. Swearingen, K., and Sinha, R. Beyond Algorithms: An HCI Perspective on Recommender Systems.

28. Zhang, Y. C., Séaghdha, D. O., Quercia, D., and Jambor, T. Auralist: introducing serendipity into music recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, ACM (New York, NY, USA, 2012), 13–22.

29. Zhou, T., Kuscsik, Z., Wakeling, J.-G. L. M. M. J. R., and Zhang, Y.-C. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Acadamy of Sciences of the United States of America (PNAS) 107*, 10 (2010), 4511–4515.

30. Ziegler, C.-N., McNee, S. M., Konstan, J. A., and Lausen, G. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, ACM (New York, NY, USA, 2005), 22–32.