Probabilistic Neighborhood Selection in Collaborative Filtering Systems

Panagiotis Adamopoulos and Alexander Tuzhilin

Department of Information, Operations and Management Sciences

Leonard N. Stern School of Business, New York University

{padamopo,atuzhili}@stern.nyu.edu

ABSTRACT

This paper presents a novel probabilistic method for recommending items in the neighborhood-based collaborative filtering framework. For the probabilistic neighborhood selection phase, we use an efficient method for weighted sampling of k neighbors without replacement that also takes into consideration the similarity levels between the target user and the candidate neighbors. We conduct an empirical study showing that the proposed method alleviates the over-specialization and concentration biases in common recommender systems by generating recommendation lists that are very different from the classical collaborative filtering approach and also increasing the aggregate diversity and mobility of recommendations. We also demonstrate that the proposed method outperforms both the previously proposed user-based k-nearest neighbors and k-furthest neighbors collaborative filtering approaches in terms of item prediction accuracy and utility-based ranking measures across various experimental settings. This accuracy performance improvement is in accordance with the ensemble learning theory.

General Terms

Algorithms, Design, Experimentation, Performance

Keywords

Collaborative Filtering, Recommender Systems, k-NN algorithm, Probabilistic Neighborhood Selection, k-PN, Sampling, Item Accuracy, Diversity and Mobility Measures

"I don't need a friend who changes when I change and who nods when I nod; my shadow does that much better."

- Plutarch, 46 - 120 AD

1. INTRODUCTION

Although a wide variety of different types of recommender systems (RSs) has been developed and used across several domains over the last 20 years [9], the classical user-based k-NN collaborative filtering (CF) method still remains one of the most popular and prominent methods used in the recommender systems community [31].

Even though the broad social and business acceptance of RSs has been achieved, one of the key under-explored dimensions for further improvement is the usefulness of recommendations. Common recommenders, such as CF algorithms, recommend products based on prior sales and ratings. Hence, they tend not to recommend products with limited historical data, even if these items would be rated favorably. Thus, these recommenders can create a rich-get-richer effect for popular items while this *concentration bias* can prevent what may otherwise be better consumer-product matches [19]. At the same time, common RSs usually recommend items very similar to what the users have already purchased or liked in the past [1]. However, this *over-specialization* of recommendations is often inconsistent with sales goals and consumers' preferences.

Aiming at alleviating the important problems of over-specialization and concentration bias and enhancing the usefulness of collaborative filtering RSs, we propose to generate recommendation lists based on a probabilistic neighborhood selection approach. In particular, we aim at providing personalized recommendations from a wide range of items in order to escape the obvious and expected recommendations, while avoiding predictive accuracy loss.

In this paper, we present a certain variation of this classical k-NN method in which the estimation of an unknown rating of the user for an item is based not on the weighted averages of the k most similar (nearest) neighbors but on k probabilistically selected neighbors. The key intuition for this *probabilistic nearest neighbors* (k-PN) collaborative filtering method, instead of the nearest neighbors, is two-fold. First, using the neighborhood with the most similar users to estimate unknown ratings and recommend candidate items, the generated recommendation lists usually consist of known items with which the users are already familiar. Second, because of the multi-dimensionality of users' tastes, there are many items that the target user may like and are unknown to the k most similar users to her/him. Thus, we propose the use of probabilistic neighborhood selection in order to alleviate the aforementioned problems and move beyond the limited focus of rating prediction accuracy.

To empirically evaluate the proposed approach, we conducted an experimental study and showed that the proposed probabilistic neighborhood selection method outperforms both the standard user-based CF and the k-furthest neighbor [48, 46] approaches by a wide margin in terms of item prediction accuracy measures, such as precision, recall, and the F-measure, across various experimental settings. This performance improvement is due to the reduction of covariance among the selected neighbors and is in accordance with the ensemble learning theory that we employ in the neighborhood-based collaborative filtering framework. Finally, we demonstrate that this performance improvement is also combined with further enhancements in terms of other popular performance measures, such as catalog coverage, aggregate diversity, and mobility.

In summary, the main contributions of this paper are:

- We proposed a new neighborhood-based method (k-PN) as an improvement of the standard

k-NN approach.

- We formulated the classical neighborhood-based collaborative filtering method as an ensemble method, thus, allowing us to show the potential suboptimality of the k-NN approach in terms of predictive accuracy.
- We empirically showed that the proposed method outperforms, by a wide margin, the classical collaborative filtering algorithm and practically illustrated its suboptimality in addition to providing a theoretical justification of this empirical observation.
- We showed that the proposed k-PN method alleviates the common problems of over-specialization and concentration bias of recommendations in terms of various popular metrics and a new metric that measures the mobility of recommendations.
- We identified a particular implementation of the k-PN method that performs consistently well across various experimental settings.

2. RELATED WORK

Since the first collaborative filtering systems were introduced in the mid-90's [24, 44, 36], there have been many attempts to improve their performance focusing mainly on rating prediction accuracy [16, 39]. Common approaches include rating normalization [16], similarity weighting of neighbors [47], and neighborhood selection, using top-N filtering, threshold filtering, or negative filtering [27, 16]. Besides, several of the methods that have been proposed use a probabilistic approach. For instance, [35] partitioned the items into groups and made predictions for users considering the Gaussian distribution of user ratings and [55] used a data selection scheme based on probabilistic active learning to actively query users and select subsets of user profiles in order to improve the "cold start" problem.

Even though the rating prediction perspective is the prevailing paradigm in recommender systems, there are other perspectives that have been gaining significant attention in this field [31] and try to alleviate the problems pertaining to the narrow rating prediction focus [2]. This narrow focus has been evident in laboratory studies and real-world online experiments, which indicated that higher predictive accuracy does not always correspond to the higher levels of user-perceived quality or to increased sales [41, 29, 30, 15]. Two of the most important problems related to this narrow focus of many RSs that have been identified in the literature and hinder the user satisfaction are the over-specialization and concentration bias of recommendations.

Pertaining to the problem of over-specialization, [22] provided empirical evidence that indeed consumers prefer diversity in ranking results. This problem is often practically addressed by injecting randomness in the recommendation procedure [10], filtering out items which are too similar to items the user has rated in the past [11], or increasing the diversity of recommendations [56]. Interestingly, [48, 46] presented an inverted neighborhood model, k-furthest neighbors, to identify less ordinary neighborhoods for the purpose of creating more diverse recommendations by recommending items disliked by the least similar users.

Studying the concentration bias of recommendations, [30] compared different RS algorithms with respect to aggregate diversity and their tendency to focus on certain parts of the product spectrum and showed that popular algorithms may lead to an undesired popularity boost of already popular items. Finally, [19] showed that this concentration bias, in contrast to the potential goal of RSs to promote long-tail items, can create a rich-get-richer effect for popular products leading to a subsequent reduction in profits and sales diversity and suggested that better RS designs which limit popularity effects and promote exploration are still needed.

3. MODEL

Collaborative filtering (CF) methods produce user specific recommendations of items based on patterns of ratings or usage (e.g. purchases) without the need for exogenous information about either items or users [43]. Hence, in order to estimate unknown ratings and recommend items to users, CF systems need to relate two fundamentally different entities: items and users.

3.1 Neighborhood Models

User-based neighborhood recommendation methods predict the rating $r_{u,i}$ of user u for item i using the ratings given to i by users most similar to u, called nearest neighbors and denoted by $\mathcal{N}_i(u)$. Taking into account the fact that the neighbors can have different levels of similarity, $w_{u,v}$, and considering the k users v with the highest similarity to u (i.e. the standard user-based k-NN collaborative filtering approach), the predicted rating is:

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum\limits_{v \in \mathcal{N}_i(u)} w_{u,v} * (r_{v,i} - \bar{r}_v)}{\sum\limits_{v \in \mathcal{N}_i(u)} |w_{u,v}|}, \qquad (1)$$

where $\bar{r}_u$ is the average of the ratings given by user u.

However, the ratings given to item i by the nearest neighbors of user u can be combined into a single estimation using various combining (or aggregating) functions [9]. Examples of combining functions include majority voting, distance-moderated voting, weighted average, adjusted weighted average, and percentiles [7].

In the same way, the neighborhood used in estimating the unknown ratings and recommending items can be formed in different ways. Instead of using the k users with the highest similarity to the target user, any approach or procedure that selects k of the candidate neighbors can be used, in principle.

Algorithm 1 summarizes the user-based k-nearest neighbors (k-NN) collaborative filtering approach using a general combining function and neighborhood selection approach.

In this paper, we propose a novel k-NN method (k-PN) using probabilistic neighborhood selection

ALGORITHM 1: k-NN Recommendation Algorithm

```
Input: User-Item Rating matrix R
Output: Recommendation lists of size $l$

$k$: Number of users in the neighborhood of user $u$, $\mathcal{N}_i(u)$
$l$: Number of items recommended to user $u$

for each user $u$ do
    for each item $i$ do
        Find the $k$ users in the neighborhood of user $u$, $\mathcal{N}_i(u)$;
        Combine ratings given to item $i$ by neighbors $\mathcal{N}_i(u)$;
    end
    Recommend to user $u$ the top-$l$ items having the highest predicted rating $\hat{r}_{u,i}$;
end
```

that also takes into consideration the similarity levels between the target user and the n candidate neighbors.

3.2 Probabilistic Neighborhood Selection

For the probabilistic neighborhood selection phase of the proposed algorithm, following [54] and [18], we suggest an efficient method for weighted sampling of k neighbors without replacement that also takes into consideration the similarity levels between the target user and the population of n candidate neighbors. In particular, the set of candidate neighbors at any time is described by values $\{w_1^{'}, w_2^{'}, \dots, w_n^{'}\}$. In general, $w_i^{'} = w_i$ if the i user/item is still a candidate for selection, and $w_i^{'} = 0$ if it has been selected in the neighborhood and, hence, removed from the set of candidates. Denote the sum of the probabilities of the remaining candidate neighbors by $S_j = \sum_{i=1}^{j} w_i^{'}$, where $j = 1, \dots, n$, and let $Q = S_n$ be the sum of the $\{w_i\}$ of the remaining candidates.

In order to draw a neighbor, we choose x with uniform probability from $[0, Q]$ and we find l such that $S_{l-1} \leq x \leq S_l$. Then, we add j to the neighborhood and remove it from the set of candidates while we set $w_j^{'} = 0$. After a neighbor has been selected, this neighbor is in principle no longer available for later selection.

This method can be easily implemented using a binary search tree having all n candidate neighbors as leaves with values $\{w_1, w_2, \dots, w_n\}$, while the value of each internal node of the tree is the sum of the values of the corresponding immediate descendant nodes. This sampling method requires $O(n)$ initialization operations, $O(k \log n)$ additions and comparisons, and $O(k)$ divisions and random number generations [54].[1] The suggested method can be used with any valid probability distribution including the empirical distribution derived based on the user/item similarity.

Algorithm 2 summarizes the method used for efficient weighted sampling without replacement [54].

3.3 Theoretical Foundation

In this section we discuss the theoretical framework under which the proposed method can gen-

[1]In the special case of equal probabilities for all the candidate neighbors or random sampling, an efficient method requires $O(k)$ operations.

ALGORITHM 2: Weighted Sampling Without Replacement

```
Input: Initial weights $\{w_1, \ldots, w_n\}$ of candidates for neighborhood $\mathcal{N}_i(u)$
Output: Neighborhood of user $u$, $\mathcal{N}_i(u)$
$k$: Number of users in the neighborhood of user $u$, $\mathcal{N}_i(u)$
$L(v)$: The left-descendent of node $v$
$R(v)$: The right-descendent of node $v$
$G_v$: The sum of weights of the leaves in the left subtree from node $v$
$Q$: The sum of weights of the nodes in the binary tree
Build binary search tree with $n$ leaves labeled $1, 2, \ldots, n$;
Assign to leaves corresponding values $w_1, w_2, \ldots, w_n$;
Associate values $G_v$ with internal nodes;
Set $Q = \sum_{v=1}^{n} w_v$;
Set $\mathcal{N}_i(u) = \emptyset$;
for $j \leftarrow 1$ to $k$ do
    Set $C = 0$;
    Set $v =$ the root node;
    Set $\mathcal{D} = \emptyset$;
    Select $x$ uniformly from $[0, Q]$;
    repeat
        if $x \leq G_v + C$ then
            Set $\mathcal{D} = \mathcal{D} \uplus \{v\}$;
            Move to node/leaf $L(v)$;
        else
            Set $C = C + G_v$;
            Move to node/leaf $R(v)$;
        end
    until a leaf is reached;
    Set $\mathcal{N}_i(u) = \mathcal{N}_i(u) \uplus \{v\}$;
    for each node $d \in \mathcal{D}$ do
        Set $G_d = G_d - w_v$;
    end
    Set $Q = Q - w_v$;
    Set $w_v = 0$;
end
```

erate recommendations orthogonal to the standard k-NN approach without significantly reducing, and even increasing, the predictive accuracy showing that similar but diverse neighbors should be used. It should be obvious by now that selecting the neighborhoods using an underlying probability distribution, instead of just selecting deterministically the k nearest neighbors, can result in very different recommendations from those generated based on the standard k-NN approach. For the sake of brevity, in the following paragraphs we focus on the effect of also selecting neighbors other than the k nearest candidates on the predictive accuracy of the proposed approach.

For the predictive tasks of a recommender system, we should construct an estimator $f(\mathbf{x}; \mathbf{w})$ that approximates an unknown target function $g(x)$ given a set of N training samples $z^N = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_i \in \mathcal{R}^d$, $y \in \mathcal{R}$, and $\mathbf{w}$ a weight vector; z^N is a realization of a random sequence $Z^N = \{Z_1, \ldots, Z_N\}$ whose i-th component consists of a random vector $Z_i = (X_i, Y_i)$ and, thus, each $\mathbf{z}_i$ is an independent and identically distributed (i.i.d) sample from an unknown joint distribution $p(\mathbf{x}, y)$. We assume that there is a functional relationship between the training pair $\mathbf{z}_i = (\mathbf{x}_i, y_i)$: $y_i = g(\mathbf{x}_i) + \epsilon$, where ϵ is the additive noise with zero mean

($E\{\epsilon\} = 0$) and finite variance ($Var\{\epsilon\} = \sigma^2 < \infty$).

Since the estimate $\hat{\mathbf{w}}$ depends on the given z^N, we should write $\hat{\mathbf{w}}(z^N)$ to clarify this dependency. Hence, we should also write $f(\mathbf{x}; \hat{\mathbf{w}}(z^N))$; however for simplicity we will write $f(\mathbf{x}; z^N)$ as in [21]. Then, introducing a new random vector $Z_0 = (X_0, Y_0) \in \mathcal{R}^{d+1}$, which has a distribution identical to that of Z_i, but is independent of Z_i for all i, the generalization error ($GErr$), defined as the mean squared error averaged over all possible realizations of Z^N and Z_0,

$$GErr(f) = E_{Z^N}\left\{E_{Z_0}\{[Y_0 - f(X_0; Z^N)]^2\}\right\},$$

can be expressed by the following "bias/variance" decomposition [21]:

$$GErr(f) = E_{X_0}\{Var\{f|X_0\} + Bias\{f|X_0\}^2\} + \sigma^2.$$

However, using ensemble estimators, instead of a single estimator f, we have a collection of them: $f_1, f_2, \ldots, f_k$, where each f_i has its own parameter vector $\mathbf{w}_i$ and k is the total number of estimators. The output of the ensemble estimator for some input $\mathbf{x}$ can be defined as the weighted average of outputs of k estimators for $\mathbf{x}$:

$$f_{ens}^{(k)}(\mathbf{x}) = \sum_{m=1}^{k} \alpha_m f_m, \tag{2}$$

where, without loss of generality, $\alpha_m > 0$ and $\sum_{m=1}^{k} \alpha_m = 1$.

Following [53], the generalization error of this ensemble estimator is:

$$GErr(f_{ens}^{(k)}) = E_{X_0}\left\{Var\{f_{ens}^{(k)}|X_0\} + Bias\{f_{ens}^{(k)}|X_0\}^2\right\} + \sigma^2,$$

which can also be expressed as:

$$\begin{aligned} GErr(f_{ens}^{(k)}) = E_{X_0}\Bigg\{ & \Bigg[\sum_{m=1}^{k} a_m^2 \underset{Z_{(m)}^N}{E}\left[\left(f_m - \underset{Z_{(m)}^N}{E}(f_m)\right)^2\right] + \\ & \sum_m \sum_{i \neq m} a_m a_i \underset{Z_{(m)}^N, Z_{(i)}^N}{E}\left\{\left[f_m - \underset{Z_{(m)}^N}{E}(f_m)\right]\left[f_i - \underset{Z_{(i)}^N}{E}(f_i)\right]\right\}\Bigg] + \\ & \left[\sum_{m=1}^{k} a_m \underset{Z_{(m)}^N}{E}(f_m - g)\right]^2 \Bigg\} + \sigma^2, \end{aligned}$$

where the term $E_{Z_{(m)}^N, Z_{(i)}^N}\left\{\left[f_m - E_{Z_{(m)}^N}(f_m)\right]\left[f_i - E_{Z_{(i)}^N}(f_i)\right]\right\}$ corresponds to the pairwise covariance of the estimators m and i, $Cov\{f_m, f_i|X_0\}$.

The results can also be extended to the following equation:

$$GErr(f_{ens}^{(k)}) = E_{X_0}\left\{\sum_{m=1}^{k}\sum_{l=1}^{k} a_m^* a_l^* R_{ml}\right\},$$

where $a_i^* = \sum_j R_{ij}^{-1} / \sum_l \sum_j R_{lj}^{-1}$ and denotes the optimal weight that minimizes the generalization error of the ensemble estimator given in (2), and R_{ij}^{-1} indicates the i, j component of the inverse matrix of R. The i, j component of matrix R is given by:

$$R_{ij} = \begin{cases} Var\{f_i|X_0\} + Bias\{f_i|X_0\}^2, & \text{if } i = j \\ Cov\{f_i, f_j|X_0\} + Bias\{f_i|X_0\}Bias\{f_j|X_0\}, & \text{otherwise.} \end{cases}$$

Hence, in addition to the bias and variance of the individual estimators (and the noise variance), the generalization error of an ensemble also depends on the the covariance between the individuals; an ensemble is controlled by a three-way trade-off. Thus, if f_i and f_j are positively correlated, then the correlation increases the generalization error, whereas if they are negatively correlated, then the correlation contributes to a decrease in the generalization error.

In the context of neighborhood-based collaborative filtering methods in recommender systems, we can think of the i^{th} (most similar to the target user) neighbor as corresponding to a single estimator f_i that simply predicts the rating of this specific neighbor. Thus, reducing the aggregated pairwise covariance of the neighbors (estimators) can decrease the generalization error of the model; on the same time, it may increase the bias or variance of the estimators and the generalization error as well. Hence, one way to reduce the covariance is not to restrict the k estimators only to the k nearest (most similar) neighbors but to use also other candidate neighbors (estimators).[2,3]

4. EXPERIMENTAL SETTINGS

To empirically validate the k-PN method presented in Section 3.1 and evaluate the generated recommendations, we conduct a large number of experiments on "real-world" data sets and compare our results to different baselines. For an apples-to-apples comparison, the selected baselines include the user-based k-nearest neighbors (k-NN) collaborative filtering approach which is the standard neighborhood-based method that we promise to improve in this study and has been found to perform well also in terms of other performance measures, besides the classical accuracy metrics [13, 15, 4, 5], and generates recommendations that suffer less from over-specialization and concentration biases [16, 30].

4.1 Data Sets

The data sets that we used are the MovieLens [14] and the MovieTweetings [17] as well as a snapshot from Amazon [40]. The RecSys HetRec 2011 MovieLens (ML) data set [14] is an extension of a data set published by [25], which contains personal ratings and tags about movies, and consists of 855,598 ratings (from 1-5) from 2,113 users on 10,197 movies. Moreover, the MovieTweetings (MT) data set is described in [17] and consists of ratings on movies that were contained in well-

[2]Let $r_{u,i}$ and $r_{u,j}$ the correlation of target user u and candidate neighbors i and j respectively, then the correlation $r_{i,j}$ of neighbors i and j is bounded by the following expression: $r_{u,i}r_{u,j} - \sqrt{1-r_{u,i}^2}\sqrt{1-r_{u,j}^2} \leq r_{i,j} \leq r_{u,i}r_{u,j} + \sqrt{1-r_{u,i}^2}\sqrt{1-r_{u,j}^2}$.

[3]For a formal argument why the proposed probabilistic approach can result in very different recommendations from those generated based on the standard k-NN approach and how the item predictive accuracy can be affected, a 0/1 loss can be used in the context of classification ensemble learning with the (highly) rated items corresponding to the positive class. For a rigorous derivation of the generalization error in ensemble learning using the bias-variance-covariance decomposition and a 0/1 loss function see [45, 52].

structured tweets on Twitter. Owing to the extreme sparsity of the data set, we decided to condense the data set in order to obtain more meaningful results from collaborative filtering algorithms. In particular, we removed items and users with fewer than 10 ratings. The resulting data set contains 12,332 ratings (from 0-10) from 839 users on 836 items. Finally, the Amazon (AMZ) data set is described in [40] and consists of reviews of fine foods during a period of more than 10 years. After removing items with fewer than 10 ratings and reviewers with fewer than 25 ratings each, the data set consists of 15,235 ratings (from 1-5) from 407 users on 4,316 items.

4.2 Experimental Settings

Using the ML, MT, and AMZ data sets, we conducted a large number of experiments and compared the results against the standard user-based k-NN approach. In order to test the proposed approach of probabilistic neighborhood selection under various experimental settings, we use different sizes of neighborhoods ($k \in \{20, 30, \ldots, 80\}$) and different probability distributions ($\mathcal{P} \in \{\text{normal, exponential, Weibull, folded normal, uniform}\}$) with various specifications (i.e. location and scale parameters) and the empirical distribution of user similarity, described in Table 1; the uniform distribution is used in order to compare the proposed method against randomly selecting neighbors. Also, we use two furthest neighbor models (k-FN) [48, 46]; the second furthest neighbor model (k-FN$_2$) employed in his study corresponds to recommending the least liked items of the neighborhoods instead of the most liked ones (k-FN$_1$). In addition, we generate recommendation lists of different sizes ($l \in \{1, 3, 5, 10, 20, \ldots, 100\}$). In summary, we used 3 data sets, 7 different sizes of neighborhoods, 12 probability distributions, and 13 different lengths of recommendation lists, resulting in $3,276$ experiments in total.

For the probabilistic neighborhood selection, we use an efficient method for weighted sampling [54] of k neighbors without replacement that also takes into consideration the similarity levels between the target user and all the candidate neighbors. In order to estimate the initial weights of the procedure described in Section 3.2, we use the probability density functions illustrated in Table 1.[4] Without loss of generality, in order to take into consideration the similarity weights of the neighbor, the candidates can be ordered and re-labeled such that $w_1 \geq w_2 \geq \ldots \geq w_n$. Then, the weight for each candidate can been generated using its rank and a probability density function. For instance, using the Weibull probability distribution (i.e. W_1 or W_2), the weight of the most similar candidate is $w_1 = \frac{\mu}{\lambda}\left(\frac{1}{\lambda}\right)^{\mu-1} e^{-(1/\lambda)^{\mu}}$, where n is the total number of all candidate neighbors.[5] In contrast to the deterministic k-NN and k-FN approaches, depending on the parameters of the employed probability

[4]The density function of the folded normal distribution shown in Table 1 can also be expressed as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\left[e^{\left(-\frac{(-x-\mu)^2}{2\sigma^2}\right)} + e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}\right]$$

where $\mu = \theta\sigma$ [34, 26].

[5]For continuous probability distributions, the cumulative distribution function can also be used such as $w_i = F(i + 0.5) - F(i - 0.5)$ or $w_i = F(i) - F(i - 1)$.

Table 1: Probability Distributions and Density Functions for Neighborhood Selection.

Label	**Probability Distribution**	**Probability Density Function (weights)**	**Location and Shape Parameters**	
k-NN	-	$\begin{cases} 1/k, \text{if } x \leq n-k \\ 0, \text{otherwise} \end{cases}$	-	
E	Empirical Similarity	$w_x / \sum_{i=1}^{n} w_i$	-	
U	Uniform	$1/n$	-	
N_1	Normal	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\mu = 0$	$\sigma = (0.25/15.0)n$
N_2			$\mu = (2.0/15.0)n$	$\sigma = (0.5/15.0)n$
Exp_1	Exponential	$\lambda e^{-\lambda x}$	$\lambda = 1/k$	
Exp_2			$\lambda = 2/k$	
W_1	Weibull	$\frac{\mu}{\lambda} \left(\frac{x}{\lambda}\right)^{\mu-1} e^{-(x/\lambda)^{\mu}}$	$\mu = 0.25$	$\lambda = n/20$
W_2			$\mu = 0.50$	$\lambda = n/20$
FN_1	Folded normal	$\frac{\sqrt{2}}{\sigma\sqrt{\pi}} e^{-\frac{\theta^2}{2}} e^{-\frac{x^2}{2\sigma^2}} \cosh\left(\frac{\theta x}{\sigma}\right)$	$\theta = 1$	$\sigma = k$
FN_2			$\theta = 1$	$\sigma = k/2$
k-FN	-	$\begin{cases} 1/k, \text{if } x \geq n-k \\ 0, \text{otherwise} \end{cases}$	-	

density function, this candidate neighbor may or may not have the highest weight.[6] Figure 1 shows the likelihood of sampling each candidate neighbor using different probability distributions for the MovieLens data set and $k = 80$ and Figure 2 shows the sampled neighborhoods for a randomly selected target user using the different probability distributions; the candidate neighbors for each target user and item in the x axis are ordered based on their similarity to the target user with 0 corresponding to the nearest (i.e. most similar) candidate.

In all the conducted experiments, in order to measure the similarity among the candidate neighbors, we used the Pearson correlation.[7] Also, we used significance weighting as in [27] in order to penalize for similarity based on few common ratings and filtered any candidate neighbors with zero weight [16]. For the similarity estimation of the candidates in the k-furthest neighbor algorithm, we used the approach described in [48, 46]. Besides, we use the standard combining function as in Eq. (1).[8] In addition, we used a holdout validation scheme in all of our experiments with 80/20 splits of data to the training/test part in order to avoid overfitting. Finally, the evaluation of the various approaches in each experimental setting is based on users with more than k candidate neighbors where k is the corresponding neighborhood size; if a user has k or less available candidate neighbors then the same neighbors are always selected and the results for the specific user are in principal identical for all the examined approaches, apart from the inverse k-FN (k-FN$_2$) method.

[6]For a probabilistic furthest neighbors model the candidates can be ordered in reverse similarity order such that $w_1 \leq w_2 \leq \ldots \leq w_n$.

[7]Similar results were obtained using the cosine similarity.

[8]Similar results were also obtained using a combining function without a first-order bias approximation: $\hat{r}_{u,i} = \sum_{v \in \mathcal{N}_i(u)} w_{u,v} r_{v,i} / \sum_{v \in \mathcal{N}_i(u)} |w_{u,v}|$. Any differences are explicitly discussed in the following section.

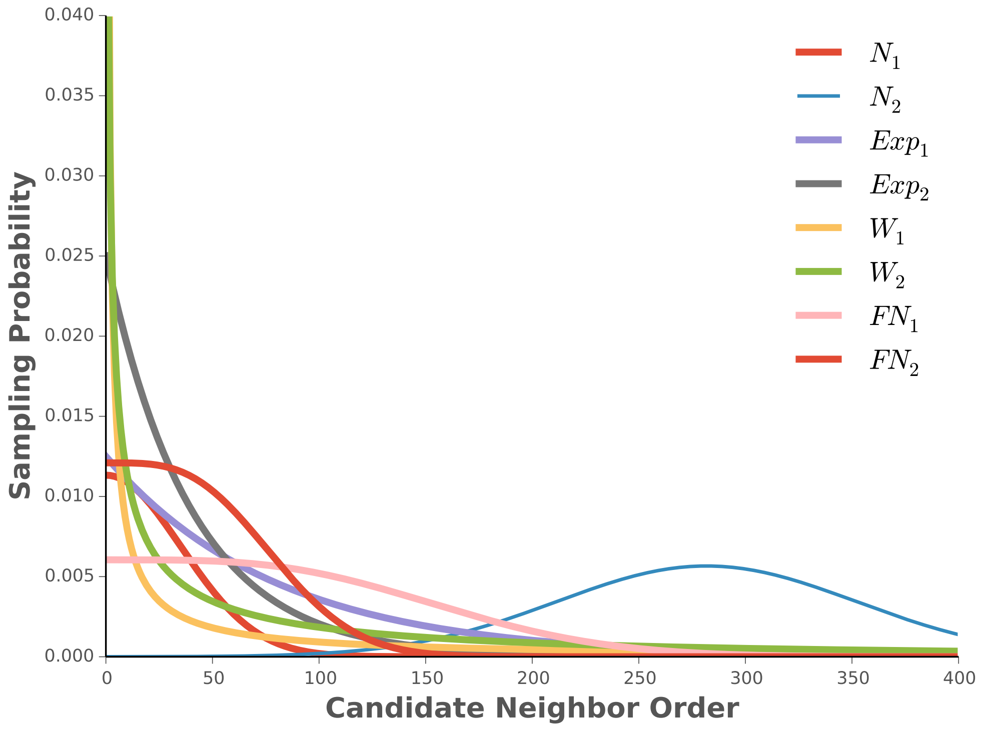


Figure 1: Sampling probability for the nearest candidate neighbors using different probability distributions for the MovieLens (ML) data set.

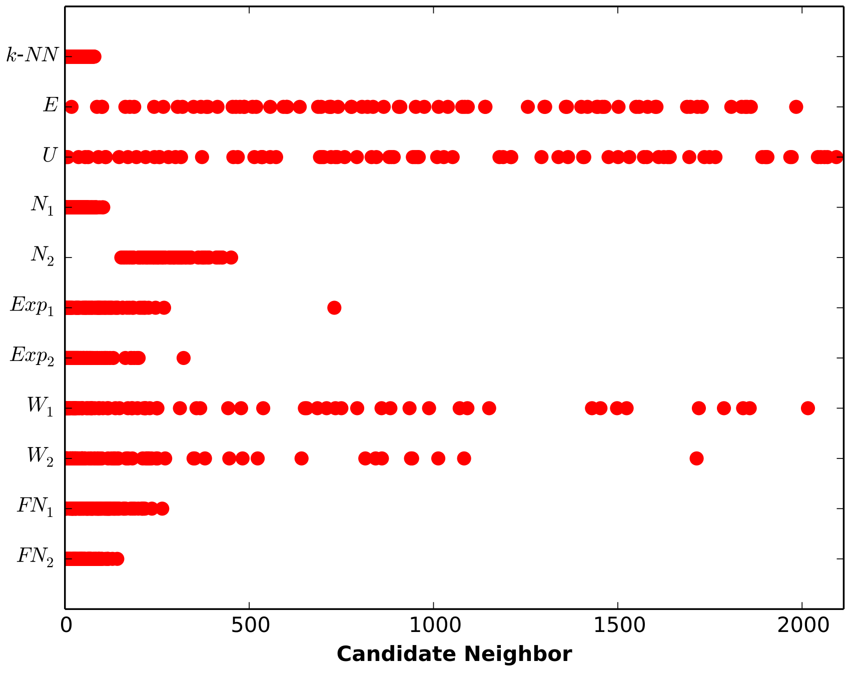
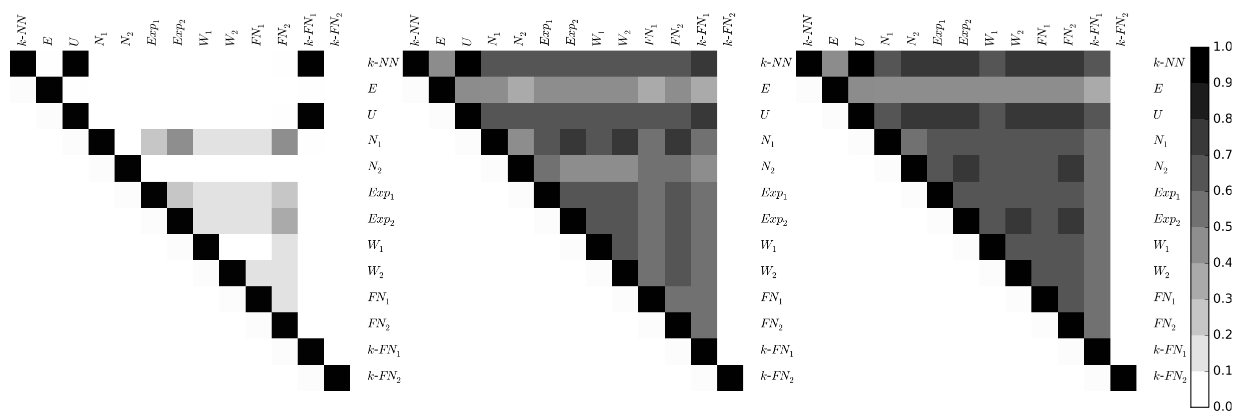


Figure 2: Sampled Neighborhoods using the different probability distributions for the MovieLens data set.



(a) MovieLens (b) MovieTweetings (c) Amazon

Figure 3: Overlap of recommendation lists of size $l = 10$ for the different data sets.

Evaluating the different approaches based on all the users in the data set yields similar results exhibiting the same patterns with reduced magnitude of differences.

5. RESULTS

The aim of this study is to demonstrate, by a comparative analysis of our method and both the standard baseline and the k-furthest neighbor approaches in different experimental settings, that the proposed method indeed effectively generates recommendations that are very different from the classical collaborative filtering systems having the potential to alleviate the over-specialization and concentration problems while it performs well in terms of the classical accuracy metrics.

Given the number and the diversity of experimental settings, the presentation of the results constitutes a challenging problem. A reasonable way to compare the results across the different experimental settings is by computing the relative performance differences and discussing only the most interesting dimensions. Detailed results about all the conducted experiments are included in [6].

5.1 Orthogonality of Recommendations

In this section, we examine whether the proposed approach finds and recommends different items than those recommended by the standard recommenders and, thus, whether it can alleviate the over-specialization problem [48]. In particular, we investigate the overlap of recommendations (i.e. the percentage of items that belong to both recommendation lists) between the classical neighborhood-based collaborative filtering method and the various specifications (i.e. E, N_1, N_2, Exp_1, Exp_2, W_1, W_2, FN_1, and FN_2) of the proposed approach described in Table 1. Fig. 3 presents the results obtained by applying our method to the MovieLens, MovieTweetings, and Amazon data sets. The values reported are computed as the average overlap over seven neighborhood sizes, $k \in \{20, 30, \ldots, 80\}$, for recommendation lists of size $l = 10$.

As Fig. 3 demonstrates, the proposed method generates recommendations that are very different from the recommendations provided by the classical k-NN approach. The more different recommendations were achieved using the empirical distribution of user similarity and the inversed k-furthest neighbors approach [46]. In particular, the average overlap across all the proposed probability distributions, neighborhoods, and recommendation list sizes was 14.87%, 64.17%, and 64.64% for the ML, MT, and AMZ data sets, respectively; the corresponding overlap using only the empirical distribution was 2.79%, 44.82%, and 39.80% for the different data sets. Hence, it is worth to note that not only the k-FN approach but also the proposed probabilistic method resulted in orthogonal recommendations to the standard k-NN method. Besides, for the more sparse data sets (i.e. MovieTweetings, Amazon), the recommendation lists exhibit greater overlap, since there are proportionally less candidate neighbors available to sample from and, thus, the neighborhoods tend to be more similar. This is also depicted on the experiments using one of the standard probability distributions but the empirical distance of candidate neighbors, the uniform distribution, and the deterministic k-FN approach. Similarly, recommendation lists of smaller size resulted in even smaller overlap among the various methods. Moreover, the experiments conducted using the U and k-FN$_1$ approaches resulted in recommendations very different from the recommendations provided by the classical k-NN approach only when the first-order bias approximation was not used in the combining function of the ratings. In general, without the first-order bias approximation the average overlap was further reduced by 58.70%, 19.69%, and 52.18% for the ML, MT, and AMZ data sets, respectively. As one would expect, the experiments conducted using the same probability distribution (e.g. Exp_1 and Exp_2) result in similar performance. To determine statistical significance, we have tested the null hypothesis that the performance of each of the methods is the same using the Friedman test. Based on the results, we reject the null hypothesis with $p < 0.0001$. Performing post hoc analysis on Friedman's Test results, all the specifications of the proposed approach (apart from the cases of the N_2, W_2, Exp, and FN specifications for the AMZ data set) significantly outperform the k-FN$_1$ method in all the data sets. The difference between the empirical distribution and k-FN$_2$ are not statistically significant for any data set.

Nevertheless, even a large overlap between two recommendation lists does not imply that these lists are the same. For instance, two recommendation lists might contain the same items but in reverse order. In order to further examine the orthogonality of the generated recommendations, we measure the rank correlation of the generated lists using the Spearman's rank correlation coefficient [51], which measures the Pearson correlation coefficient between the ranked variables. In particular, we use the top 100 items recommended by method i and examine the correlation ρ_{ij} in the rankings generated by methods i and j for those items; ρ_{ij} might be different from ρ_{ji}. Fig. 4 shows the average ranking correlation over seven neighborhood sizes, $k \in \{20, 30, \ldots, 80\}$, using the the Spearman's rank correlation coefficient ρ_{ij} (i corresponds to the row index and j to the column index) for the MovieTweetings data set (i.e. the data set that exhibits the largest overlap). The correlation between the classical neighborhood-based collaborative filtering method and the probabilistic approach with the empirical distribution using the top 100 items recommended by the

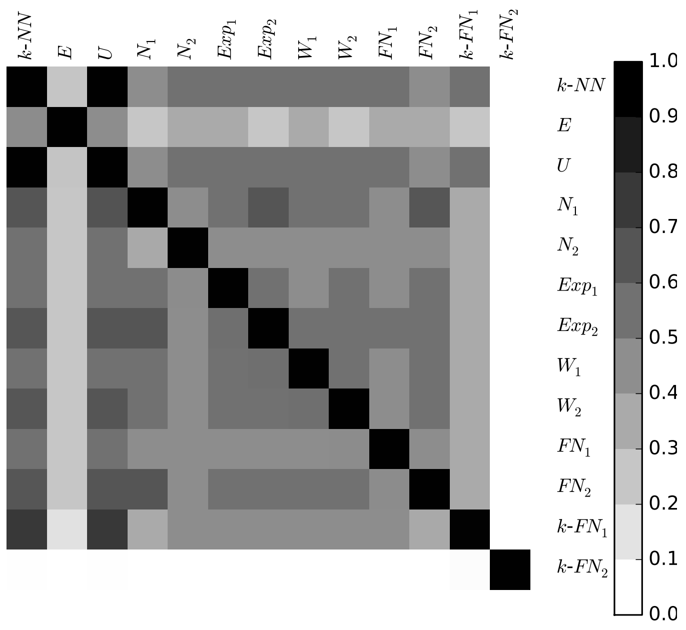
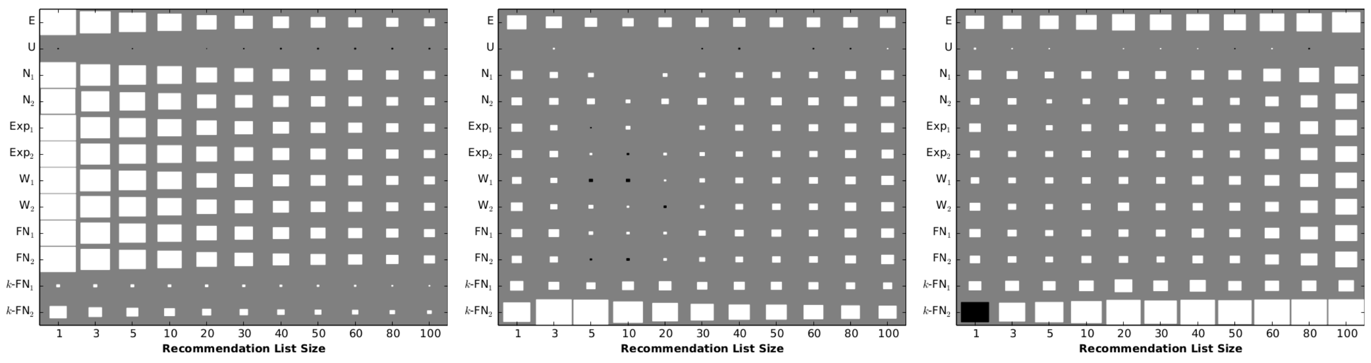


Figure 4: Spearman's rank correlation coefficient for the MovieTweetings data set.



(a) MovieLens (b) MovieTweetings (c) Amazon

Figure 5: Increase in aggregate diversity performance for the different data sets and recommendation list sizes.

k-NN method is 22.21%. As Figs. 3b and 4 illustrate, even though some specifications may result in recommendation lists that exhibit significant overlap, the ranking of the recommended items is not strongly correlated.

5.2 Comparison of Diversity

In this section we investigate the effect of the proposed method on coverage and aggregate diversity, two important metrics [49] which in combination with the rest of the measures discussed in this study show whether the proposed approach can alleviate the over-specialization and concentration bias problems of CF systems. The results obtained using the *catalog coverage* metric [28, 20] are equivalent to those using the *diversity-in-top-N* metric for aggregate diversity [8]; henceforth, only one set of results is presented. Fig. 5 presents the results obtained by applying our method to the

ML, MT, and AMZ data sets. In particular, the Hinton diagram in Fig. 5 shows the percentage increase/decrease of the average number of items in the catalog that are ever recommended to users compared to the k-NN baseline for each probability distribution and recommendation lists of size $l \in \{1, 3, 5, 10, 30, 40, 50, 60, 80, 100\}$ over seven neighborhood sizes, $k \in \{20, 30, \ldots, 80\}$. Positive and negative values are represented by white and black squares, respectively, and the size of each square represents the magnitude of each value; for each dataset the maximum size corresponds to the maximum percentage difference observed in the conducted experiments.

Fig. 5 demonstrates that the proposed method in most cases performs better than both the standard user-based k-NN and the k-FN methods. The more diverse recommendations were achieved using the empirical distribution of user similarity and the inverse k-furthest neighbors approach (k-FN$_2$). In particular, the average aggregate diversity across all the probability distributions, neighborhoods, and recommendation list sizes was 22.10%, 46.09%, and 13.52% for the ML, MT, and AMZ data sets, respectively; the corresponding aggregate diversity using only the empirical distribution was 24.20%, 50.55%, and 17.04% for the different data sets. Furthermore, the performance was increased both in the experiments where the k-NN method, because of the specifics of the particular data sets, resulted in low aggregate diversity (e.g. Amazon) or high diversity performance (e.g. MovieTweetings). In addition, the experiments conducted using the same probability distribution exhibit very similar performance also for this metric. As one would expect, in most cases the aggregate diversity increased whereas the magnitude of the difference in performance decreased with increasing recommendation list size l. Without using the first-order bias approximation in the combining function, the standard k-NN method resulted in higher aggregate diversity and catalog coverage but the proposed approach still outperformed the classical algorithm in most of the cases by a narrower margin; using the inverse k-FN method (k-FN$_2$) without the first-order bias approximation resulted in decrease in performance for the Amazon data set.

In terms of statistical significance, using the Friedman test and performing post hoc analysis, the differences among the employed baselines (i.e. k-NN, k-FN$_1$, and k-FN$_2$) and all the proposed specifications are statistically significant ($p < 0.001$) for the ML data set. For the MT and AMZ data sets, all the proposed specifications (i.e. E, N_1, N_2, Exp_1, Exp_2, W_1, W_2, FN_1, and FN_2) significantly outperform the k-NN algorithm; the empirical distribution significantly outperforms also the k-FN$_1$ method. Besides, the k-FN$_2$ method significantly outperforms the Exp_1, Exp_2, W_1, W_2, FN_1, and FN_2 but not the E and N specifications for the MT and AMZ data sets.

5.3 Comparison of Dispersion and Mobility

In order to conclude whether the proposed approach alleviates the concentration biases, the generated recommendation lists should also be evaluated for the inequality across items using the Gini coefficient [23]. In particular, the Gini coefficient was on average improved by 6.81%, 3.67%, and 1.67% for the ML, MT, and AMZ data sets, respectively; the corresponding figures using only the empirical distribution were 7.48%, 6.73%, and 3.45% for the different data sets. The more uniformly distributed recommendation lists were achieved using the empirical distribution of user similarity and the inverse k-furthest neighbors approach. Moreover, the larger the size of the recommendation lists, the larger the improvement in the Gini coefficient. Similarly, without using the first-order bias approximation in the rating combining function the average dispersion was further improved by 6.48%, 6.83%, and 20.22% for the ML, MT, and AMZ data sets, respectively. As we can conclude, in the recommendation lists generated from the proposed method, the number of times an item is recommended is more equally distributed compared to the classical k-NN method. In terms of statistical significance, using the Friedman test and performing post hoc analysis, all the proposed specifications (apart from the N_1, Exp_2, and FN_2 for the MT data set and the N_2, Exp_2 for the AMZ data set) significantly outperform the k-NN and k-FN$_1$ methods ($p < 0.001$). The empirical distribution also significantly outperforms the k-FN$_2$ method for the ML data set; the differences are not statistically significant for the other data sets.

However, simply evaluating the recommendation lists in terms of dispersion and inequality does not provide any information about the mobility of the recommendations (i.e. whether popular or "long tail" items are more likely to be recommended) since these metrics do not consider the prior state of the system. Hence, in order to provide more evidence on whether the proposed approach could solve the concentration bias problem, we employed a mobility measure [12, 50] M to assess whether the proposed recommender system approach follows or changes the prior popularity of items when recommendation lists are generated. Thus, we define M, which equals the proportion of items that is mobile (e.g. changed from popular in terms of number of ratings to "long tail" in terms of recommendation frequency), as follows:

$$M = 1 - \sum_{i=1}^{K} \pi_i \rho_{ii}$$

where the vector π denotes the initial distribution of each of the K (popularity) categories and ρ_{ii} the probability of staying in category i, given that i was the initial category.[9] A score of zero denotes no change (i.e. the number of times an item is recommended is proportional to the number of ratings it has received) whereas a score of one denotes that the recommender system recommends only the "long tail" items (i.e. the number of times an item is recommended is proportional to the inverse of the number of ratings it has received).

In the conducted experiments, based on the 80-20 rule or Pareto principle [42], we use two categories, labeled as "head" and "tail", where the former category contains the top 20% of the items (in terms of ratings or recommendations frequency) and the latter category the remaining 80% of the items. The experimental results demonstrate that the proposed method generates recommendation lists that exhibit in most cases better mobility compared to the k-NN and k-FN methods. In particular, the performance was increased by 0.91%, 0.95%, and 0.19% for the ML, MT, and AMZ data sets, respectively; the corresponding overlap using only the empirical distribution was 1.29%, 1.46%, and 0.45% for the different data sets. We also note that recommendation lists

[9]The proposed mobility score can be easily adapted in order to differentiate the direction of change and the magnitude.

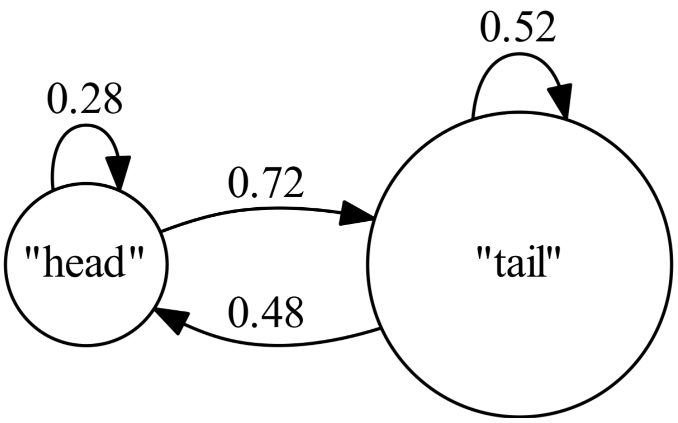


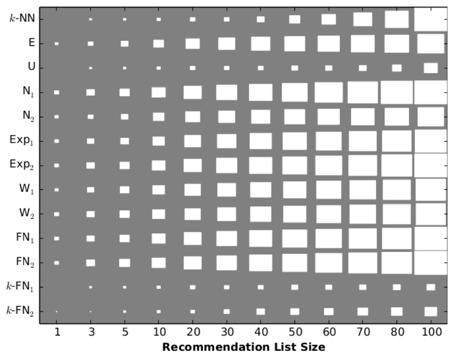
Figure 6: Mobility of recommendations for the MT data set.

of larger size resulted in even larger improvements on average. Similarly, without the first-order bias approximation the average mobility was further increased by 0.69%, 0.53%, and 3.28% for the ML, MT, and AMZ data sets, respectively. Fig. 6 shows the transition probabilities of each category for recommendation lists of size $l = 100$ using the empirical distribution of similarity and the MovieTweetings data set. In terms of statistical significance, all the proposed specifications significantly outperform the k-NN method ($p < 0.005$). Similarly, the proposed specifications (apart from the cases of N_1, Exp_2, and FN_2 for the ML data set) significantly outperform also the k-FN$_1$ method for the ML and MT data sets. However, for the AMZ data set the proposed approach significantly underperforms the k-FN$_2$ method.

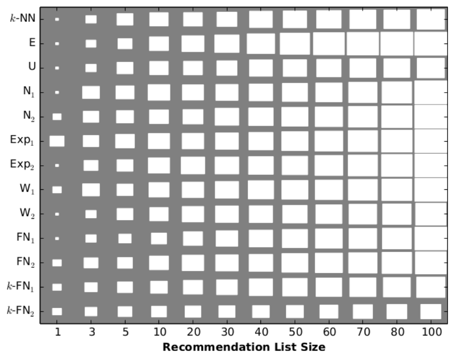
5.4 Comparison of Item Prediction

Apart from alleviating the concentration bias and over-specialization problems in CF systems, the proposed approach should also perform well in terms of predictive accuracy. Thus, the goal in this section is to compare the proposed method with the standard baseline methods in terms of traditional metrics for *item prediction*, such as the F$_1$ score. Figs. 7 and 8 present the results obtained by applying the proposed method to the MovieLens (ML), MovieTweetings (MT), and Amazon (AMZ) data sets. The values reported in Fig. 8 are computed as the average performance over seven neighborhood sizes, $k \in \{20, 30, \ldots, 80\}$, using the F$_1$ score for recommendation lists of size $l = 10$. The Hilton diagram show in Figure 7 presents the relative F$_1$ score for recommendation lists of size $l \in \{1, 3, 5, 10, 20, 30, 40, 50, 60, 70, 80, 100\}$; the size of each white square represents the magnitude of each value with the maximum size corresponding to the maximum value achieved in the conducted experiments for each data set. Similar results were also obtained using as positive instances only the highly rated items (i.e. items rated above the average rating or above the 80% of the rating scale) in the test set.

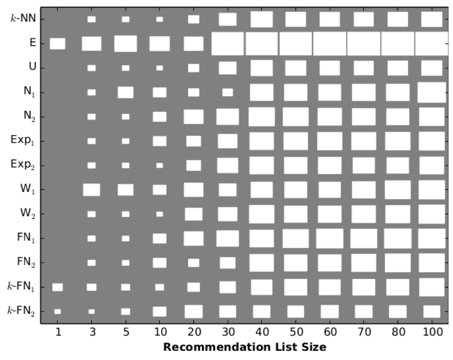
Figs. 7 and 8 demonstrate that the proposed method outperforms the standard user-based k-NN method and the k-FN approach in most of the cases. The most accurate recommendations were gen-



(a) MovieLens

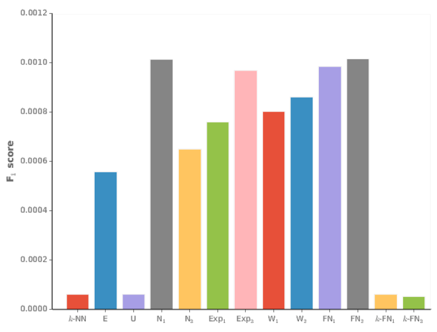


(b) MovieTweetings

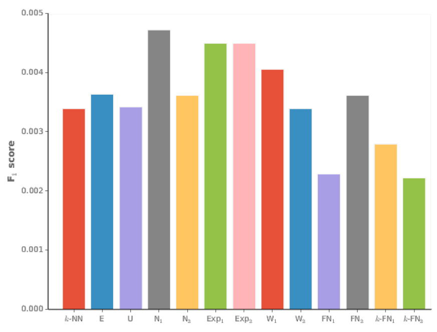


(c) Amazon

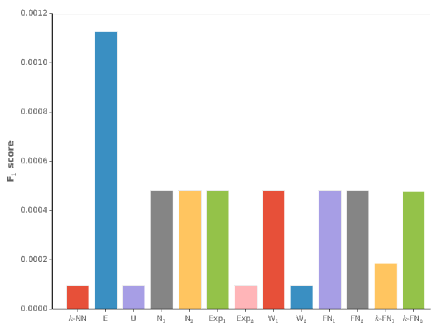
Figure 7: Item prediction performance for the different data sets.



(a) MovieLens



(b) MovieTweetings



(c) Amazon

Figure 8: Item prediction performance for the different data sets and recommendation lists of size $l = 10$.

erated using the empirical distribution of user similarity, the normal or the exponential distribution. In particular, the average F_1 score across all the proposed probability distributions, neighborhoods, and recommendation list sizes was 0.0018, 0.0050, and 0.0010 for the ML, MT, and AMZ data sets, respectively; the corresponding performance using only the empirical distribution was 0.0015, 0.0055, and 0.0022 for the different data sets resulting on average in a 4-fold increase. Besides, without using the first-order bias approximation in the rating combining function, the proposed approach outperformed in most of the cases the classical k-NN algorithm and the k-FN method by a wider margin. Furthermore, we should note that the performance was increased across various experimental specifications, including different sparsity levels, neighborhood sizes, and recommendation list lengths. This performance improvement is due to the reduction of covariance among the selected neighbors and is in accordance with the ensemble learning theory that we introduce in the neighborhood-based collaborative filtering framework in Section 3.3.

To determine the statistical significance of the previous findings, we have tested using the Friedman test the null hypothesis that the performance of each of the methods is the same. Based on the results, we reject the null hypothesis with $p < 0.0001$. Performing post hoc analysis on Friedman's Test results, in most of the cases (i.e. 86.42% of the experimental settings) the proposed approach significantly outperforms the employed baselines and in the remaining cases the differences are not statistically significantly. In particular, the differences between the traditional k-NN and each one of the proposed variations (apart from the case of the FN_1 specification for the MT data set) are statistically significant for all the data sets; similar results were also obtained for the differences among the proposed approach and the k-FN models.

5.5 Comparison of Utility-based Ranking

Further, in order to better assess the quality of the proposed approach, the recommendation lists should also be evaluated for the ranking of the items that present to the users, taking into account the rating scale of the selected data sets. In principal, since all items are not of equal relevance/quality to the users, the relevant/better items should be identified and ranked higher for presentation. Assuming that the utility of each recommendation is the rating of the recommended item discounted by a factor that depends on its position in the list of recommendations, in this section we evaluate the generated recommendation lists based on the normalized Cumulative Discounted Gain (nDCG) [32], where positions are discounted logarithmically.

The highest performance was again achieved using the empirical distribution of user similarity, the normal or the Weibull distribution. In particular, the average increase of the nDCG score across all the examined probability distributions, neighborhoods, and recommendation list sizes was 100.06%, 20.05%, and 89.85% for the ML, MT, and AMZ data sets, respectively; the corresponding increase using only the empirical distribution was 117.65%, 23.01%, and 383.99% for the different data sets resulting on average in a 2-fold increase. As for the F_1 metric, without using the first-order bias approximation in the rating combining function, the proposed approach outperformed in most of the cases the classical k-NN algorithm and the k-FN methods by an even wider margin.

In terms of statistical significance, using the Friedman test and performing post hoc analysis, the differences among the employed baselines and all the proposed specifications (apart from the FN_1 for the MT data set and the N_1, Exp_2, W_2, and FN_2 for the AMZ data set) are statistically significant ($p < 0.001$).

6. DISCUSSION AND CONCLUSIONS

In this paper, we present a novel method for recommending items based on probabilistic neighborhood selection in collaborative filtering systems. We illustrate the practical implementation of the proposed method in the context of memory-based collaborative filtering systems adapting and improving the standard user-based k-nearest neighbors (k-NN) approach. In the proposed variation of the classical k-NN collaborative filtering method, the neighborhood selection is based on an underlying probability distribution, instead of just the k neighbors with the highest similarity level to the target user. For the probabilistic neighborhood selection (k-PN), we use an efficient method for weighted sampling of k neighbors without replacement that also takes into consideration the similarity levels between the target user and all the candidate neighbors. In addition, we conduct an empirical study showing that the proposed method generates recommendations that are very different from the classical collaborative filtering approach and alleviates the over-specialization and concentration problems. We also demonstrate that using probability distributions which sample mainly from the nearest neighbors and also some further neighbors, the proposed method outperforms, by a wide margin in most cases, both the standard user-based k-nearest neighbors and the k-furthest neighbors approaches in terms of both item prediction accuracy and utility-based ranking measures, such as the F-measure and the normalized discounted cumulative gain (nDCG), across various experimental settings. These results are also in accordance with the ensemble learning theory that we employ in the neighborhood-based collaborative filtering framework. Besides, we show that the performance improvement is not achieved at the expense of other popular performance measures, such as catalog coverage, aggregate diversity, and recommendation dispersion and the proposed metric of mobility.

Following the proposed approach and providing personalized recommendations from a wide range of items, we can further enhance the usefulness of collaborative filtering RSs. In particular, avoiding obvious and expected recommendations [4, 5] while maintaining high predictive accuracy levels, we can alleviate the common problems of over-specialization and concentration bias that often characterize the CF algorithms. Besides, building such a recommender system, we also have the potential to further increase user satisfaction and engagement and offer a superior experience to the users by providing them with non-obvious and high quality recommendation lists that fairly match their interests and they will remarkably like [2].

Furthermore, the generated recommendations should be useful not only for the users but for the businesses as well. The proposed approach exhibits a potential positive economic impact based on (i) the direct effect of increased sales and enhanced customer loyalty through offering more useful for the users recommendations from a wider range of items, enabling them to find relevant items that are harder to discover, and making the users familiar with the whole product catalog, and (ii) the indirect effect of recommending items from the long tail and not focusing mostly on bestsellers that usually exhibit higher marginal costs and lower profit margins because of acquisition costs and licenses as well as increased competition. This potential economic impact, which should be empirically verified and precisely quantified, is a topic of future research.

Moreover, the proposed method can be further extended and modified in order to sample k neighbors from the x nearest candidates, instead of all the available users, and combined with additional rating normalization and similarity weighting schemes [33] beyond those employed in this study. Also, apart from the user-based and item-based k-NN collaborative filtering approaches, other popular methods that can be easily extended with the use of probabilistic neighborhood selection (k-PN), in order to allow us to generate both accurate and novel recommendations, include Matrix Factorization approaches [37, 38]. Besides, this approach can be further extended to the popular methods of k-NN classification and regression in information retrieval (IR).

As a part of the future work, we would like to propose a novel approach in collaborative filtering recommender systems directly optimizing the generalization error rate derived in Section 3.3. Finally, we would also like to conduct live experiments with real users in a traditional on-line retail setting as well as in a platform for massive open on-line courses [3].

7. REFERENCES

[1] Z. Abbassi, S. Amer-Yahia, L. V. Lakshmanan, S. Vassilvitskii, and C. Yu. Getting recommender systems to think outside the box. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 285–288, New York, NY, USA, 2009. ACM.

[2] P. Adamopoulos. Beyond Rating Prediction Accuracy: On New Perspectives in Recommender Systems. In *Proceedings of the seventh ACM conference on Recommender systems*, RecSys '13. ACM, 2013.

[3] P. Adamopoulos. What Makes a Great MOOC? An Interdisciplinary Analysis of Student Retention in Online Courses. In *Proceedings of the 34th International Conference on Information Systems*, ICIS 2013, 2013.

[4] P. Adamopoulos and A. Tuzhilin. On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected. In *DiveRS 2011 - ACM RecSys 2011 Workshop on Novelty and Diversity in Recommender Systems*, RecSys 2011. ACM, Oct. 2011.

[5] P. Adamopoulos and A. Tuzhilin. On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected. *Working Paper: CBA-13-03, New York University*, 2013. http://ssrn.com/abstract=2282999.

[6] P. Adamopoulos and A. Tuzhilin. Probabilistic Neighborhood Selection in Collaborative Filtering Systems. *Working Paper: CBA-13-04, New York University*, 2013. http://hdl.handle.net/2451/31988.

[7] P. Adamopoulos and A. Tuzhilin. Recommendation Opportunities: Improving Item Prediction Using Weighted Percentile Methods in Collaborative Filtering Systems. In *Proceedings of the seventh ACM conference on Recommender systems*, RecSys '13. ACM, 2013.

[8] G. Adomavicius and Y. Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *Knowledge and Data Engineering, IEEE Transactions on*, 24(5):896 –911, may 2012.

[9] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[10] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[11] D. Billsus and M. J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10(2-3):147–180, Feb. 2000.

[12] R. Boudon, R. Boudon, R. Boudon, and R. Boudon. *Mathematical structures of social*

mobility, volume 12. Elsevier Amsterdam, 1973.

[13] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.

[14] I. Cantador, P. Brusilovsky, and T. Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys '11. ACM, 2011.

[15] P. Cremonesi, F. Garzotto, S. Negro, A. V. Papadopoulos, and R. Turrin. Looking for "good" recommendations: A comparative evaluation of recommender systems. In *Human-Computer Interaction–INTERACT 2011*, pages 152–168. Springer, 2011.

[16] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[17] S. Dooms, T. De Pessemier, and L. Martens. MovieTweetings: a Movie Rating Dataset Collected From Twitter. In *Workshop on Crowdsourcing and Human Computation for Recommender Systems, CrowdRec at RecSys '13*, 2013.

[18] R. Fagin and T. G. Price. Efficient calculation of expected miss ratios in the independent reference model. *SIAM Journal on Computing*, 7(3):288–297, 1978.

[19] D. Fleder and K. Hosanagar. Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Manage. Sci.*, 55(5):697–712, May 2009.

[20] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 257–260. ACM, 2010.

[21] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

[22] A. Ghose, P. Ipeirotis, and B. Li. Designing ranking systems for hotels on travel search engines by mining user-generated and crowd-sourced content. *Marketing Science*, 2012.

[23] C. Gini. Concentration and dependency ratios (in Italian). *English translation in Rivista di Politica Economica*, 87:769–789, 1909.

[24] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[25] GroupLens research group, 2011. http://www.grouplens.org.

[26] W. Gui, P. Chen, and H. Wu. A folded normal slash distribution and its applications to non-negative measurements. *Journal of Data Science*, 2012.

[27] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237. ACM, 1999.

[28] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative

filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004.

[29] D. Jannach and K. Hegelich. A case study on the effectiveness of recommendations in the mobile internet. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 205–208. ACM, 2009.

[30] D. Jannach, L. Lerche, F. Gedikli, and G. Bonnin. What recommenders recommend–an analysis of accuracy, popularity, and sales diversity effects. In *User Modeling, Adaptation, and Personalization*, pages 25–37. Springer, 2013.

[31] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender systems: an introduction.* Cambridge University Press, 2010.

[32] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, Oct. 2002.

[33] R. Jin, J. Y. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 337–344. ACM, 2004.

[34] N. Johnson. The folded normal distribution: Accuracy of estimation by maximum likelihood. *Technometrics*, 4(2):249–256, 1962.

[35] B. M. Kim and Q. Li. Probabilistic model estimation for collaborative filtering based on items attributes. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 185–191. IEEE Computer Society, 2004.

[36] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[37] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[38] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data*, 4(1):1:1–1:24, Jan. 2010.

[39] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 145–186. Springer, 2011.

[40] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the seventh ACM conference on Recommender systems*, RecSys '13. ACM, 2013.

[41] S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *Proceedings of CHI '06*, pages 1097–1101, New York, NY, USA, 2006. ACM.

[42] V. Pareto. Manual of political economy tr. by ann s. schwier. 1927.

[43] F. Ricci and B. Shapira. *Recommender systems handbook.* Springer, 2011.

[44] E. Rich. User modeling via stereotypes. *Cognitive science*, 3(4):329–354, 1979.

[45] F. Roli and G. Fumera. Analysis of linear and order statistics combiners for fusion of imbalanced classifiers. In *Proceedings of the Third International Workshop on Multiple Classifier Systems*, MCS '02, pages 252–261, London, UK, UK, 2002. Springer-Verlag.

[46] A. Said, B. Fields, B. J. Jain, and S. Albayrak. User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm. In *Proceedings of the ACM 2013 conference on Computer Supported Cooperative Work*. ACM, 2013.

[47] A. Said, B. J. Jain, and S. Albayrak. Analyzing weighting schemes in collaborative filtering: cold start, post cold start and power users. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 2035–2040. ACM, 2012.

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

[49] G. Shani and A. Gunawardana. Evaluating recommendation systems. *Recommender Systems Handbook*, 12(19):1–41, 2011.

[50] A. Shorrocks. Income inequality and income mobility. *Journal of Economic Theory*, 19(2):376–393, 1978.

[51] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):pp. 441–471, 1987.

[52] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection science*, 8(3-4):385–404, 1996.

[53] N. Ueda and R. Nakano. Generalization error of ensemble estimators. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 90–95. IEEE, 1996.

[54] C.-K. Wong and M. C. Easton. An efficient method for weighted sampling without replacement. *SIAM Journal on Computing*, 9(1):111–113, 1980.

[55] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel. Probabilistic memory-based collaborative filtering. *Knowledge and Data Engineering, IEEE Transactions on*, 16(1):56–69, 2004.

[56] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 22–32. ACM, 2005.

APPENDIX

A. ORTHOGONALITY OF RECOMMENDATIONS

Table 2: Recommendation List Overlap for the MovieLens data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
E	0.023618	0.029383	0.030350	0.024373	0.025736	0.033923
U	1.000000	0.998648	0.987506	0.994037	0.993510	0.951956
N_1	0.075384	0.083564	0.068461	0.062572	0.064530	0.068360
N_2	0.030356	0.045839	0.047096	0.039522	0.038891	0.043741
Exp1	0.064183	0.080035	0.065296	0.056070	0.059030	0.062051
Exp2	0.070696	0.083943	0.067534	0.060206	0.062192	0.066197
W_1	0.045929	0.057562	0.048671	0.041034	0.043609	0.049763
W_2	0.047777	0.060807	0.053465	0.046168	0.048964	0.053364
FN_1	0.058391	0.072422	0.061139	0.052958	0.055922	0.059822
FN_2	0.071057	0.083510	0.067034	0.059678	0.061477	0.065411
k-FN_1	0.932955	0.914042	0.940646	0.953672	0.957523	0.925060
k-FN_2	0.007820	0.021202	0.022000	0.028049	0.019120	0.012307

Table 3: Recommendation List Overlap for the MovieTweetings data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
E	0.356235	0.393340	0.440539	0.484813	0.506766	0.507518
U	0.991939	0.992761	0.996261	0.997459	0.998091	0.996717
N_1	0.586323	0.612999	0.648849	0.659846	0.664346	0.631611
N_2	0.574418	0.587484	0.601078	0.592829	0.585793	0.528291
Exp1	0.594758	0.611744	0.641992	0.646005	0.646492	0.606485
Exp2	0.598922	0.626255	0.652424	0.655622	0.661077	0.622078
W_1	0.589900	0.615959	0.643690	0.656167	0.655816	0.615431
W_2	0.592088	0.616988	0.645584	0.657343	0.656685	0.617908
FN_1	0.591875	0.599743	0.625584	0.631387	0.630335	0.581871
FN_2	0.592729	0.616506	0.643608	0.653232	0.655255	0.618087
k-FN_1	0.670937	0.709653	0.757535	0.808060	0.831620	0.864437
k-FN_2	0.003096	0.004118	0.005992	0.012164	0.019039	0.032655

Table 4: Recommendation List Overlap for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
E	0.321458	0.387693	0.446873	0.426609	0.421664	0.383464
U	0.960376	0.984311	0.992510	0.995101	0.994642	0.992772
N_1	0.632804	0.672832	0.666023	0.620295	0.578305	0.496725
N_2	0.657378	0.708096	0.714015	0.639556	0.599205	0.510404
Exp1	0.663727	0.708240	0.704363	0.651985	0.603150	0.518858
Exp2	0.659965	0.709320	0.715483	0.647315	0.602282	0.515271
W_1	0.650088	0.691760	0.698649	0.632451	0.592569	0.513235
W_2	0.651969	0.692767	0.709575	0.652442	0.607232	0.516341
FN_1	0.661611	0.703418	0.703514	0.652351	0.607106	0.523333
FN_2	0.662904	0.700396	0.704556	0.641427	0.599885	0.511996
k-FN_1	0.558260	0.615977	0.697104	0.785151	0.788080	0.805297
k-FN_2	0.000000	0.000000	0.000000	0.000365	0.002334	0.004847

B. COMPARISON OF DIVERSITY

Table 5: Aggregate Diversity for the MovieLens data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.001464	0.002483	0.004536	0.013320	0.024888	0.057427
E	0.089219	0.116074	0.168449	0.278853	0.350088	0.449498
U	0.001464	0.002454	0.004536	0.012947	0.022635	0.053150
N_1	0.084167	0.117423	0.176228	0.304086	0.373211	0.482252
N_2	0.074220	0.102955	0.151182	0.259276	0.324396	0.422026
Exp1	0.081971	0.112959	0.166482	0.283289	0.351610	0.455125
Exp2	0.081167	0.114983	0.175884	0.303799	0.370800	0.475592
W_1	0.087497	0.118701	0.173200	0.281810	0.353476	0.452757
W_2	0.084755	0.115672	0.169324	0.275968	0.347261	0.449283
FN_1	0.082301	0.113433	0.168765	0.285054	0.357495	0.454134
FN_2	0.081296	0.114797	0.172583	0.293494	0.362604	0.468244
k-FN_1	0.002483	0.003918	0.006904	0.018429	0.030228	0.060987
k-FN_2	0.017052	0.023066	0.034749	0.064374	0.084540	0.115471

Table 6: Aggregate Diversity for the MovieTweetings data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.202324	0.265892	0.365003	0.509911	0.574163	0.669515
E	0.250684	0.302119	0.405844	0.588517	0.668831	0.816986
U	0.202666	0.265892	0.365003	0.509569	0.574163	0.669856
N_1	0.213602	0.272727	0.365003	0.525120	0.606459	0.766063
N_2	0.217191	0.279563	0.370984	0.544258	0.632946	0.778708
Exp1	0.211210	0.265721	0.370813	0.522215	0.609023	0.770335
Exp2	0.214798	0.266917	0.363807	0.519993	0.606630	0.755639
W_1	0.210014	0.262987	0.360731	0.521189	0.606972	0.764525
W_2	0.210697	0.270506	0.366200	0.516576	0.608510	0.766405
FN_1	0.221975	0.269651	0.367396	0.530075	0.620984	0.768455
FN_2	0.213431	0.265208	0.362611	0.523753	0.608681	0.768455
k-FN_1	0.222830	0.285202	0.410287	0.557245	0.635167	0.716336
k-FN_2	0.487355	0.582194	0.677204	0.768797	0.809125	0.871839

Table 7: Aggregate Diversity for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.055475	0.073282	0.091983	0.140275	0.158414	0.166656
E	0.073017	0.097081	0.128161	0.201046	0.229909	0.293393
U	0.055508	0.073315	0.091983	0.140342	0.158381	0.166656
N_1	0.061234	0.078247	0.100159	0.154475	0.182014	0.252681
N_2	0.058983	0.074904	0.096253	0.146597	0.170727	0.237819
Exp1	0.058818	0.075963	0.099000	0.152092	0.176652	0.238084
Exp2	0.058454	0.077122	0.097908	0.148881	0.171654	0.239739
W_1	0.058056	0.077353	0.097776	0.149609	0.177678	0.237488
W_2	0.058487	0.076592	0.097743	0.151331	0.174864	0.239474
FN_1	0.059314	0.077155	0.098405	0.150801	0.175659	0.241791
FN_2	0.058917	0.076791	0.099662	0.147789	0.172680	0.241229
k-FN_1	0.064941	0.083675	0.109658	0.167218	0.182808	0.202337
k-FN_2	0.091454	0.129121	0.174732	0.281113	0.326923	0.398550

C. COMPARISON OF DISPERSION AND MOBILITY

Table 8: Recommendation Dispersion for the MovieLens data set.

Setting	Recommendation List Size					
	3	5	10	30	50	100
k-NN	0.999679	0.999450	0.998949	0.996881	0.994765	0.988882
E	0.967092	0.956073	0.934667	0.886077	0.854847	0.801690
U	0.999679	0.999451	0.998949	0.996883	0.994806	0.989327
N_1	0.975081	0.965689	0.945760	0.904081	0.877641	0.830113
N_2	0.975244	0.965997	0.948003	0.906370	0.879857	0.833867
Exp1	0.974450	0.965457	0.946198	0.905185	0.879987	0.833044
Exp2	0.975618	0.965858	0.945082	0.902877	0.876529	0.829812
W_1	0.971111	0.960503	0.938843	0.894461	0.867748	0.819284
W_2	0.972434	0.962484	0.942437	0.900043	0.872987	0.825401
FN_1	0.974613	0.965388	0.946352	0.906139	0.879784	0.833630
FN_2	0.975907	0.966667	0.946240	0.905342	0.878576	0.831412
k-FN_1	0.999657	0.999403	0.998904	0.996801	0.994692	0.989252
k-FN_2	0.998154	0.997797	0.997002	0.994832	0.992714	0.987585

Table 9: Recommendation Dispersion for the MovieTweetings data set.

Setting	Recommendation List Size					
	3	5	10	30	50	100
k-NN	0.936491	0.921943	0.893003	0.835938	0.797022	0.740282
E	0.914480	0.893939	0.855011	0.775761	0.727419	0.630710
U	0.936539	0.921869	0.893044	0.835888	0.796956	0.740215
N_1	0.926762	0.910941	0.880962	0.814093	0.772384	0.689430
N_2	0.923981	0.901400	0.861974	0.778002	0.727768	0.625284
Exp1	0.926365	0.909449	0.875794	0.803821	0.761713	0.671532
Exp2	0.926511	0.910044	0.879680	0.810912	0.769215	0.684036
W_1	0.928135	0.911360	0.879245	0.808777	0.765059	0.677172
W_2	0.926040	0.909344	0.877952	0.809122	0.765394	0.677760
FN_1	0.925456	0.906975	0.872679	0.796588	0.749795	0.654710
FN_2	0.927283	0.911119	0.878965	0.809969	0.766119	0.679624
k-FN_1	0.925930	0.910815	0.881207	0.821557	0.782738	0.728067
k-FN_2	0.824085	0.788323	0.760065	0.708830	0.678347	0.602940

Table 10: Recommendation Dispersion for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.982802	0.978695	0.975836	0.964118	0.957721	0.949920
E	0.970337	0.962897	0.954176	0.925977	0.914727	0.881766
U	0.982791	0.978702	0.975842	0.964112	0.957729	0.949916
N_1	0.978029	0.972209	0.965668	0.945471	0.932307	0.902952
N_2	0.979921	0.975423	0.970189	0.951467	0.940828	0.911624
Exp1	0.979136	0.974036	0.967180	0.947252	0.934497	0.908355
Exp2	0.980009	0.974278	0.968784	0.948724	0.936771	0.910333
W_1	0.979809	0.974026	0.967706	0.947657	0.934137	0.908228
W_2	0.979119	0.973779	0.968358	0.947738	0.935747	0.907739
FN_1	0.979014	0.973539	0.967682	0.948595	0.936134	0.907853
FN_2	0.979636	0.974129	0.967699	0.948789	0.937418	0.908246
k-FN_1	0.978394	0.974276	0.970528	0.956788	0.949965	0.942488
k-FN_2	0.960722	0.939343	0.918239	0.899212	0.900390	0.883906

Table 11: Recommendation Mobility for the MovieLens data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.500213	0.500301	0.500550	0.501447	0.501610	0.500927
E	0.506426	0.507050	0.507755	0.508672	0.507900	0.506094
U	0.500213	0.500301	0.500550	0.501547	0.502418	0.502413
N_1	0.502977	0.503113	0.503779	0.503666	0.503718	0.502865
N_2	0.506540	0.507904	0.509946	0.510982	0.511352	0.510443
Exp1	0.504142	0.504782	0.505690	0.506239	0.506395	0.505970
Exp2	0.503897	0.504050	0.504620	0.504252	0.504187	0.503901
W_1	0.505258	0.506087	0.506106	0.506913	0.507023	0.506032
W_2	0.504853	0.505560	0.506578	0.507196	0.507748	0.507004
FN_1	0.504855	0.505813	0.505852	0.506967	0.507178	0.506464
FN_2	0.502581	0.503967	0.504776	0.504383	0.503989	0.504169
k-FN_1	0.500358	0.500527	0.500922	0.502400	0.503749	0.505755
k-FN_2	0.502398	0.503137	0.504629	0.508045	0.510106	0.512892

Table 12: Recommendation Mobility for the MovieTweetings data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.503138	0.500375	0.500047	0.496122	0.503089	0.510768
E	0.502513	0.499893	0.497389	0.508719	0.518313	0.530964
U	0.503007	0.500325	0.500149	0.496212	0.503398	0.510768
N_1	0.500493	0.500703	0.499784	0.503324	0.515638	0.521355
N_2	0.502418	0.500718	0.502725	0.504436	0.512921	0.523414
Exp1	0.503176	0.500469	0.501296	0.503626	0.514547	0.522153
Exp2	0.503116	0.500986	0.501483	0.502894	0.515638	0.521692
W_1	0.501999	0.499628	0.499851	0.502857	0.515031	0.522515
W_2	0.501866	0.498883	0.499977	0.502175	0.516448	0.522439
FN_1	0.502651	0.499398	0.499918	0.503833	0.515208	0.523444
FN_2	0.501792	0.499663	0.500852	0.503866	0.514776	0.521769
k-FN_1	0.501854	0.497423	0.495348	0.499400	0.508471	0.515355
k-FN_2	0.518735	0.521162	0.516566	0.509188	0.509101	0.505164

Table 13: Recommendation Mobility for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.499598	0.499158	0.498078	0.497871	0.496201	0.501670
E	0.499391	0.499462	0.500508	0.497607	0.497620	0.511461
U	0.499523	0.499213	0.498091	0.497861	0.496137	0.501717
N_1	0.500288	0.500035	0.499025	0.499275	0.497201	0.502813
N_2	0.499946	0.499846	0.499196	0.499269	0.498228	0.502098
Exp1	0.500072	0.499485	0.498665	0.498520	0.497631	0.502634
Exp2	0.500198	0.499759	0.498981	0.498874	0.497672	0.501325
W_1	0.500066	0.499459	0.498653	0.499275	0.498053	0.501600
W_2	0.499826	0.499219	0.499267	0.499747	0.497762	0.502966
FN_1	0.500370	0.499923	0.499164	0.500039	0.497249	0.501974
FN_2	0.499829	0.499597	0.498890	0.498975	0.497721	0.502392
k-FN_1	0.499323	0.497506	0.496719	0.497643	0.499386	0.506676
k-FN_2	0.501157	0.506011	0.506838	0.510855	0.513287	0.517692

D. COMPARISON OF ITEM PREDICTION

Table 14: F_1 score for the MovieLens data set.

Setting	Recommendation List Size					
	3	5	10	30	50	100
k-NN	0.000023	0.000032	0.000061	0.000132	0.000625	0.006718
E	0.000132	0.000274	0.000556	0.001533	0.002368	0.004036
U	0.000023	0.000032	0.000061	0.000118	0.000202	0.001001
N_1	0.000341	0.000514	0.001012	0.002398	0.003662	0.006911
N_2	0.000136	0.000252	0.000648	0.001512	0.002370	0.003912
Exp1	0.000237	0.000356	0.000758	0.002031	0.003241	0.005759
Exp2	0.000333	0.000487	0.000967	0.002425	0.003392	0.006390
W_1	0.000226	0.000435	0.000802	0.001878	0.002978	0.005222
W_2	0.000308	0.000509	0.000860	0.001906	0.002926	0.005086
FN_1	0.000300	0.000490	0.000983	0.002324	0.003453	0.005949
FN_2	0.000381	0.000601	0.001015	0.002260	0.003444	0.006292
k-FN_1	0.000023	0.000032	0.000061	0.000118	0.000209	0.000346
k-FN_2	0.000002	0.000025	0.000052	0.000159	0.000381	0.000808

Table 15: F_1 score for the MovieTweetings data set.

Setting	Recommendation List Size					
	3	5	10	30	50	100
k-NN	0.001042	0.002537	0.003388	0.003949	0.005483	0.007540
E	0.000950	0.001957	0.003627	0.005932	0.008705	0.011757
U	0.001042	0.002537	0.003415	0.004043	0.005446	0.007438
N_1	0.002833	0.003272	0.004715	0.004948	0.006354	0.010392
N_2	0.002386	0.002883	0.003609	0.005598	0.006909	0.010377
Exp1	0.002386	0.003534	0.004493	0.005237	0.006966	0.010285
Exp2	0.001938	0.002537	0.004493	0.005621	0.006547	0.010212
W_1	0.002833	0.002537	0.004051	0.005439	0.006845	0.010255
W_2	0.001042	0.002495	0.003388	0.005247	0.005998	0.009299
FN_1	0.001042	0.001498	0.002282	0.004694	0.006390	0.010146
FN_2	0.001938	0.002191	0.003609	0.005078	0.006774	0.009549
k-FN_1	0.001544	0.002233	0.002788	0.004307	0.006027	0.008070
k-FN_2	0.001382	0.001886	0.002214	0.002609	0.002897	0.004044

Table 16: F_1 score for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.000152	0.000130	0.000095	0.000838	0.001171	0.001225
E	0.001030	0.001407	0.001127	0.002949	0.002975	0.003488
U	0.000152	0.000130	0.000095	0.000838	0.001221	0.001225
N_1	0.000152	0.000657	0.000481	0.000277	0.001512	0.002276
N_2	0.000152	0.000130	0.000481	0.001399	0.001866	0.001802
Exp1	0.000152	0.000130	0.000481	0.001047	0.001680	0.001900
Exp2	0.000152	0.000130	0.000095	0.001212	0.001527	0.001684
W_1	0.000769	0.000657	0.000481	0.000838	0.001450	0.001959
W_2	0.000152	0.000130	0.000095	0.001025	0.001527	0.001967
FN_1	0.000152	0.000130	0.000481	0.001399	0.001912	0.002092
FN_2	0.000152	0.000130	0.000481	0.000651	0.001527	0.001892
k-FN_1	0.000226	0.000192	0.000187	0.000719	0.001460	0.001772
k-FN_2	0.000074	0.000188	0.000478	0.000760	0.000749	0.000758

E. COMPARISON OF UTILITY-BASED RANKING

Table 17: normalized Discounted Cumulative Gain for the MovieLens data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.304075	0.296816	0.377174	0.331410	0.446068	0.782338
E	0.869749	0.801486	0.782000	0.787776	0.825025	0.937583
U	0.304075	0.296816	0.377809	0.327922	0.361710	0.712368
N_1	0.831997	0.808763	0.807143	0.799080	0.843317	0.939936
N_2	0.803426	0.823698	0.801642	0.788407	0.823351	0.937855
Exp1	0.653975	0.721381	0.740922	0.777835	0.815083	0.926142
Exp2	0.751713	0.764668	0.759618	0.766304	0.808081	0.928331
W_1	0.766838	0.753325	0.742316	0.803725	0.825046	0.931653
W_2	0.840853	0.837526	0.809513	0.810525	0.842339	0.944436
FN_1	0.718876	0.758183	0.745903	0.785626	0.822951	0.931357
FN_2	0.790887	0.798450	0.771482	0.807938	0.834869	0.940199
k-FN_1	0.315889	0.313537	0.413037	0.367578	0.449218	0.708598
k-FN_2	0.005941	0.095978	0.136643	0.178816	0.271173	0.574583

Table 18: normalized Discounted Cumulative Gain for the MovieTweetings data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.550549	0.455055	0.539540	0.604974	0.703573	0.867139
E	0.588233	0.683739	0.757816	0.743558	0.788773	0.914791
U	0.550549	0.455055	0.536109	0.617187	0.710376	0.869461
N_1	0.629669	0.580871	0.658780	0.674039	0.735630	0.894556
N_2	0.819458	0.702065	0.724943	0.736969	0.792015	0.921057
Exp1	0.749127	0.778405	0.769666	0.752475	0.768560	0.920481
Exp2	0.550224	0.667754	0.742592	0.741440	0.788129	0.915420
W_1	0.876019	0.743846	0.805453	0.767668	0.804665	0.931699
W_2	0.467869	0.578842	0.606639	0.689895	0.731227	0.889291
FN_1	0.437036	0.556066	0.556860	0.619271	0.702454	0.871238
FN_2	0.881449	0.763428	0.741233	0.764979	0.807412	0.928810
k-FN_1	0.811184	0.796170	0.735325	0.692052	0.762447	0.914969
k-FN_2	0.428829	0.447308	0.426425	0.447731	0.498487	0.772569

Table 19: normalized Discounted Cumulative Gain for the Amazon data set.

Setting	**Recommendation List Size**					
	3	5	10	30	50	100
k-NN	0.068518	0.050614	0.034307	0.138484	0.240079	0.433335
E	0.449127	0.415998	0.300097	0.322464	0.387439	0.682673
U	0.068518	0.050614	0.034307	0.143570	0.255439	0.438872
N_1	0.068518	0.171536	0.116270	0.074153	0.220848	0.514214
N_2	0.068518	0.050614	0.102098	0.189390	0.328765	0.516811
Exp1	0.068518	0.050614	0.102098	0.157188	0.293581	0.528331
Exp2	0.068518	0.050614	0.034307	0.126381	0.221085	0.458022
W_1	0.448612	0.331386	0.224620	0.231668	0.331904	0.589357
W_2	0.068518	0.050614	0.034307	0.132985	0.224928	0.496493
FN_1	0.068518	0.050614	0.081544	0.176184	0.279528	0.521485
FN_2	0.068518	0.050614	0.072370	0.096220	0.229657	0.481677
k-FN_1	0.108598	0.080221	0.063438	0.117251	0.248285	0.477642
k-FN_2	0.054299	0.067075	0.139260	0.227271	0.275825	0.372995