A Probabilistic Reformulation of Memory-Based Collaborative Filtering – Implications on Popularity Biases

Rocío Cañamares
Universidad Autónoma de Madrid
C/ Fco. Tomás y Valiente 11, Madrid, Spain
rocio.cannamares@uam.es

Pablo Castells
Universidad Autónoma de Madrid
C/ Fco. Tomás y Valiente 11, Madrid, Spain
pablo.castells@uam.es

ABSTRACT

We develop a probabilistic formulation giving rise to a formal version of heuristic k nearest-neighbor (kNN) collaborative filtering. Different independence assumptions in our scheme lead to user-based, item-based, normalized and non-normalized variants that match in structure the traditional formulations, while showing equivalent empirical effectiveness. The probabilistic formulation provides a principled explanation why kNN is an effective recommendation strategy, and identifies a key condition for this to be the case. Moreover, a natural explanation arises for the bias of kNN towards recommending popular items. Thereupon the kNN variants are shown to fall into two groups with similar trends in behavior, corresponding to two different notions of item popularity. We show experiments where the comparative performance of the two groups of algorithms changes substantially, which suggests that the performance measurements and comparison may heavily depend on statistical properties of the input data sample.

CCS CONCEPTS

• **Information systems** → Recommender Systems; Collaborative Filtering; Probabilistic Retrieval Models; Evaluation of Retrieval Results

KEYWORDS

Recommender systems, collaborative filtering, probabilistic models, popularity, algorithmic bias, evaluation

ACM Reference format:

R. Cañamares and P. Castells. 2007 A Probabilistic Reformulation of Memory-Based Collaborative Filtering – Implications on Popularity Biases. In *Proc. of International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, August 2017 (SIGIR 2017)*, 10 pages.

1 INTRODUCTION

Nearest-neighbor algorithms (kNN) are probably the most popular and widely-known approach to collaborative filtering (CF) [1,16]. They can be said to be the first technique to be explicitly documented as a strategy to implement a proper recommender system as we understand it today [17]. kNN was further popularized for being the core approach in the pioneering large-scale deployment of recommendation technologies in e-commerce by Amazon [15]. They have been shown to have competitive empirical effectiveness and are still widely used today. Newer CF approaches, notably ones based on matrix factorization [12], have been reported to outperform kNN in raw recommendation accuracy, yet neighbor-based methods closely follow the former in this task. Furthermore, the kNN approach provides tradeoff advantages such as easier-to-explain outputs to end-users, a straightforward easy-to-understand formulation scheme, and ease of implementation [16].

A current limitation of the kNN scheme is its heuristic nature – heuristic is in fact a name by which the kNN approach is often referred to [1]. The kNN formulation was borrowed from classification and regression, but does not arise from a formal justification. Its use is supported by intuition ("users with similar behavior in the past should have similar behavior in the future") and empirical effectiveness, but the latter has not been explained in a principled way [14]. Furthermore, different variants of the kNN scheme have been used by different authors with often differing results, and there has not been a clear basis to decide upon which alternative is best or when, other than by practical comparative effectiveness [6].

In this paper we take on to develop a probabilistic formulation for nearest-neighbor recommendation. A probabilistic basis has obvious well-understood advantages for algorithm development, such as a design justification, a deeper understanding of the algorithm behavior and properties, and enabling variations and extensions of the scheme. The development we propose results in a formulation that closely matches the structure of traditional heuristic kNN. As we shall see, different independence assumptions in our scheme lead to user-based, item-based, normalized and non-normalized variants. We report experiments showing the probabilistic version is roughly equivalent in performance to the corresponding heuristic formulations. We further find the probabilistic basis enables explanations for some of the properties of the kNN approach, such as its effectiveness, the advantages of one variant over the other, and connections between kNN and popularity.

The rest of the paper is organized as follows. We start recalling the basics and notation for the recommendation task and the kNN approach. Then in section 3 we develop the proposed probabilistic formulation. We analyze the connections between kNN and popularity in section 4, after which in section 5 we contrast the theoretical analysis with empirical observations on different datasets. We discuss related work in the section following that, after which we end with a brief summary and some conclusions.

2 THE NEAREST NEIGHBOR COLLABORATIVE FILTERING APROACH

The recommendation task is widely known and needs little introduction today [18]. In summary, a recommender system receives as input data collected from observed user-item interactions, which can be abstracted to a set of user / item / timestamp records (for so-called implicit data directly obtained from natural user activity), or user / item / rating value (for so-called explicit preference feedback provided by users). Given this input, the recommender system's task is to predict unobserved user-item preferences, and thereupon deliver a set of ranked items (typically different for each user) that satisfy the needs of end-users. More detailed descriptions and discussion of the recommendation problem can be found elsewhere (see e.g. [1] for a popular introduction).

Following a common convention in recommender systems research, we shall denote by $r(u,i)$ the observed rating by a user u for an item i, and by $\hat{r}(u,i)$ the system score (which may or may not aim to be literally a rating prediction) by which an item is selected and ranked for recommendation to a user. We shall use the convention $r(u,i) = 0$ to indicate $r(u,i)$ is unknown to the system −note that by this we are implying 0 is not an allowed rating value; the reader may find this shall bring a slight simplification in notation along the way, with no generality loss. Finally, we shall use the symbols $\mathcal{U}$ and $\mathcal{I}$ to refer to, respectively, the set of users and the set of items involved in the recommendation problem.

The recommendation task can be refined in different ways [10], most notably including a) predicting rating values as accurately as possible, and b) delivering an item ranking that satisfies as much as possible the target users as viewed in an information retrieval perspective (i.e. returning as many relevant items as possible, as early as possible in the ranking). In this paper we take on the latter perspective, which is understood today to be more closely related to real recommendation scenarios [6].

The nearest-neighbor collaborative filtering strategy (also known as memory-based, or heuristic) is one among hundreds of solutions to the recommendation problem. It can be said to be the first proper approach envisioned for the task [17] and probably remains the most popular. It builds on the intuition that users with similar tastes in the past may enjoy similar choices in the future. The most common simple materialization of this principle has taken the following structure (see e.g. [16]):

$$\hat{r}(u,i) = C \sum_{v \in N_k[u]} sim(u,v)\, r(v,i) \tag{1}$$

where $sim(u,v)$ is a similarity function that estimates how much the tastes of two users look alike, $N_k[u]$ is a neighborhood of k users who are most "suitable" for contributing to the aggregated advice for u, and C is a normalizing constant. Neighborhood "suitability" is commonly defined as being among the k most similar users to u in terms of $sim(u,v)$, though neighbor selection is a modular problem open to alternative approaches aiming to improve the resulting recommendations.

By the partial symmetry of the item and user spaces it also makes sense to consider an item-based kNN variant based on the similarity between items:

$$\hat{r}(u,i) = C \sum_{j \in N_k[i]} sim(i,j)\, r(u,j) \tag{2}$$

The similarity functions can be defined in innumerous ways, the most popular being the cosine function and Pearson correlation. For instance, the cosine similarity can be defined as:

$$sim(u,v) = \frac{\sum_{j\in\mathcal{I}} r(u,j)\, r(v,j)}{\sqrt{\sum_{j\in\mathcal{I}} r(u,j)^2}\sqrt{\sum_{j\in\mathcal{I}} r(v,j)^2}} \tag{3}$$

$$sim(i,j) = \frac{\sum_{v\in\mathcal{U}} r(v,i)\, r(v,j)}{\sqrt{\sum_{v\in\mathcal{U}} r(v,i)^2}\sqrt{\sum_{v\in\mathcal{U}} r(v,j)^2}} \tag{4}$$

From this basic scheme many variations arise for which, again, we refer the reader e.g. to the survey in [16] for a good starting point. For the sake of our research, the above straightforward versions are sufficiently well-behaved and appropriately simple as a frame of reference for our proposed development.

A point of variation in the scheme will nonetheless get our attention, namely the normalizing constant C. Most of the literature (including popular surveys [1,16]) reports taking $C = 1/\sum_{v\in N_k[u] \wedge r(u,i)>0} |sim(u,v)|$ in order to make of equation 1 a weighted sum of ratings with weights adding to 1 (and analogously for equation 2). However it has been recently found that $C = 1$ is generally more effective at the item ranking task (see e.g. [2,6,22]). In this paper we will propose a formal basis for both options, and we will find that each of them leads to quite different theoretical and empirical implications, which we shall formally characterize.

3 PROBABILISTIC kNN

We derive a probabilistic formulation from a formalization of the user-item interaction dynamics as a random process in a sampling space, as follows. Along with the $\mathcal{U}$ and $\mathcal{I}$ spaces, let us consider the set $\mathcal{T}$ of all time instants where some user interacts with (consumes, purchases, rates, choses, etc.) some item. Taking the time points in $\mathcal{T}$ as (theoretically) fine-grained as needed (e.g. real numbers), we can assume there is only a user-item pair interaction at each $t \in \mathcal{T}$, that is, two users cannot access an item at the exact same time. This is just a condition that facilitates having well-defined probability distributions, as we shall see. Actually, assigning an arbitrary unique ID to each interaction would do as well for our purposes −time is just a convenient and particularly meaningful one. We consider $\mathcal{T}$ includes all interactions, both observed and non-observed by the system. We can also consider that $\mathcal{T}$ includes both past and future interactions. It is not a problem for us to introduce such an abstract construct, as it will just as well serve our theoretical purposes. We may also consider for a moment an ideal situation in which users have full omniscient knowledge of the whole item space and always choose items in full awareness of how much they would benefit (enjoy, gain, etc.) from each item, therefore choosing the item that maximizes their satisfaction at that time.

We consider the user choices are not deterministic and may depend on the occasion, therefore it makes sense to describe them by a probability distribution over items (for each user) to represent this uncertainty. For this purpose, over the set $\mathcal{T}$ we define two functions $U\colon \mathcal{T} \to \mathcal{U}$ and $I\colon \mathcal{T} \to \mathcal{I}$. For a given $t \in \mathcal{T}$, $U(t)$, which we shall abbreviate as U_t, is the user who had the interaction at time t, and $I(t)$, which we shall abbreviate as I_t, is the item that U_t interacted with. We can consider U and I as random variables for which $p(U_t = u)$ and $p(I_t = i)$ are well-defined (categorical) probability distributions over $u \in \mathcal{U}$ and $i \in \mathcal{I}$ respectively.

Now assume we have a target user u for whom we are to produce a recommendation. Say $t \in \mathcal{T}$ is the next time u will consume some item (either spontaneously by his own initiative, or induced by the system delivering a recommendation, or by any other means). Defining t this way means that $U_t = u$. Upon this formal scheme, we may define the recommendation task as estimating the probability for each item to be the one the user would choose at that time. That is, we wish to estimate $p(I_t = i|U_t = u)$. If we can do that, an optimal recommendation for u (which maximizes user satisfaction in expectation) is ranking items by decreasing value of this probability. In other words, we should take $p(I_t = i|U_t = u)$ as the ranking function (ranking score) for recommendation, to optimize the chance to satisfy a user who browses our recommendation top-down.

Upon this initial basis, we shall derive formulations of user-based and item-based kNN by introducing a second user (or, respectively, item) –the neighbor– as a second random variable over which we shall marginalize the ranking function, as we show next.

3.1 User-Based kNN

Let $t' < t$ be a time when the target item I_t (whichever that item happens to be) was interacted with by some other user v. Applying the law of total sum we can marginalize the objective function as follows:

$$\begin{aligned} & p(I_t = i|U_t = u) \\ & \quad \sim \sum_{v \in \mathcal{U}} p(I_{t'} = i|U_{t'} = v) p(U_{t'} = v|U_t = u, I_{t'} = I_t) \end{aligned} \tag{5}$$

where we apply some independence assumptions: first, $p(I_t = i|U_t = u) \sim p(I_t = i|U_t = u, I_{t'} = I_t)$ considering that $I_{t'} = I_t$ does not provide much meaningful information in the absence of any further condition related to time t' –the equality just states a condition as vague as the fact that some undefined user chose item I_t at some undefined moment t' in the past. Second, given that we do not have any direct evidence of the preference of the target user u for target items i (otherwise we would not consider i for recommendation to u), and assuming we do have some direct observation of other users v interacting with i, we have removed $U_t = u$ from the conditional part of the first term in the summation:

$$\begin{aligned} p(I_t = i|U_t = u, U_{t'} = v, I_{t'} = I_t) &\sim p(I_t = i|U_{t'} = v, I_{t'} = I_t) \\ = p(I_{t'} = i|U_{t'} = v, I_{t'} = I_t) &\sim p(I_{t'} = i|U_{t'} = v) \end{aligned} \tag{6}$$

The second term in equation 5, $p(U_{t'} = v|U_t = u, I_{t'} = I_t)$, denotes the probability that v has chosen sometime an item that u will also pick. It can be rewritten as:

$$\begin{aligned} & p(U_{t'} = v|U_t = u, I_{t'} = I_t) \\ & \qquad = \frac{p(U_{t'} = v, U_t = u, I_{t'} = I_t)}{\sum_{w \in \mathcal{U}} p(U_{t'} = w, U_t = u, I_{t'} = I_t)} \end{aligned} \tag{7}$$

3.2 Estimation from Observed Data

Now let us neglect the variations in user tastes over time. This is a very strong assumption, but one that basic non context-sensitive recommender systems make to generate their predictions: user behavior consistency over time.

If we do that, then we may estimate the above distributions based on observed data as follows. Assume the observations we have are a set of recorded interactions between users and items. This can be represented as a sample of triples $\mathcal{F} \subset \mathcal{U} \times \mathcal{I} \times \mathcal{T}$ so that $(u, i, t) \in \mathcal{F}$ means that u has been observed consuming i at time t. We denote by $r(u, i) = |\{(u, i, t) \in \mathcal{F}\}|$ the number of times u has been observed to pick i. As a frequency, $r(u, i)$ follows a multinomial distribution resulting from the categorical probability that the user will pick the item in question at any given time, that is, in expected value:

$$r(u, i) \Big/ \sum_{j \in \mathcal{I}} r(u, j) \sim p(I_t = i|U_t = u) \tag{8}$$

which provides an estimate for the distribution in equation 6.

This scheme, and the derivations we will develop hereupon, can be generalized to $r(u, i)$ denoting explicit numeric ratings also, to the extent that it is reasonable to assume that rating values reflect (are a monotonically increasing function of, or more strictly, are proportional to) the expected frequency with which a user would pick an item. To that extent, we will henceforth occasionally (and intentionally) refer to $r(u, i)$ as a "rating".

As for equation 7, the chances that v and u pick the same item at any two (arbitrary) different points in time can be estimated by the number of times this was observed to occur in our sample $\mathcal{F}$:

$$\begin{aligned} p(U_{t'} = v, U_t = u, I_{t'} = I_t) &= \sum_{j \in \mathcal{I}} p(U_{t'} = v, U_t = u, I_{t'} = I_t = j) \\ &\sim \frac{1}{|\mathcal{F}|^2} \sum_{j \in \mathcal{I}} |\{(t_1, t_2) \in \mathcal{T}^2 | (u, j, t_1) \in \mathcal{F} \wedge (v, j, t_2) \in \mathcal{F}\}| \\ &= \frac{1}{|\mathcal{F}|^2} \sum_{j \in \mathcal{I}} |\{t_1 \in \mathcal{T} | (u, j, t_1) \in \mathcal{F}\}| |\{t_2 \in \mathcal{T} | (v, j, t_2) \in \mathcal{F}\}| \\ &= \frac{\sum_{j \in \mathcal{I}} r(u, j) r(v, j)}{\left(\sum_{w \in \mathcal{U}} \sum_{j \in \mathcal{I}} r(w, j)\right)^2} \end{aligned} \tag{9}$$

Substituting equation 9 into 7 we get:

$$p(U_{t'} = v|U_t = u, I_{t'} = I_t) \sim \frac{\sum_{j \in \mathcal{I}} r(u, j) r(v, j)}{\sum_{w \in \mathcal{U}} \sum_{j \in \mathcal{I}} r(u, j) r(w, j)} \tag{10}$$

And finally, replacing equations 8 and 10 into 5 we have:

$$\begin{aligned} & p(I_t = i|U_t = u) \\ & \sim \frac{1}{\sum_{w \in \mathcal{U}} \sum_{j \in \mathcal{I}} r(u, j) r(w, j)} \sum_{v \in \mathcal{U}} \frac{\sum_{j \in \mathcal{I}} r(u, j) r(v, j)}{\sum_{j \in \mathcal{I}} r(v, j)} r(v, i) \\ & \propto \sum_{\substack{v \in \mathcal{U} \\ v \neq u}} \frac{\sum_{j \in \mathcal{I}} r(u, j) r(v, j)}{\sum_{j \in \mathcal{I}} r(u, j) \sum_{j \in \mathcal{I}} r(v, j)} r(v, i) \end{aligned} \tag{11}$$

where we have intentionally added $\sum_{j \in \mathcal{I}} r(u, j)$ in the denominator (as it is a constant for a fixed target user u, thus preserving the ranking) to make our point next. We also add the (redundant) condition $v \neq u$ because we should have $r(u, i) = 0$, i.e. no previous recorded interaction for any i we may consider as recommendation for u.

As we now can see, equation 11 defines a user-based kNN recommender where the similarity function looks quite like a cosine (as in equation 3), only using the L_1 norm instead of L_2 in the cosine similarity. We have thus just shown that the ranking function of a common user-based kNN algorithm can be described as the computation of a probability. The specific variant matches the heuristic kNN scheme defined by equations 1 plus 3 with $C = 1$, that is, no normalization [2,6,22].

We may now want to restrict the set of users v to a subset of neighbors, or not −this would just be an ad-hoc refinement of the probabilistic scheme. In our experiments, similarly to heuristic kNN, we shall consider neighbor selection based on the highest values of the weights in the rating sum in equation 11, which correspond to $p(U_t = u, I_{t'} = I_t | U_{t'} = v)$, for each target user u.

Moreover, we found in our experiments that smoothing the probability estimates slightly improves empirical results. Specifically we tested Dirichlet smoothing [25] on the equation 11 weights as estimates of $p(U_t = u, I_{t'} = I_t | U_{t'} = v)$, using $\sum_{w\in\mathcal{U}} \sum_{j\in\mathcal{J}} r(u,j) r(w,j) \,/\, \sum_{w\in\mathcal{U}} \sum_{j\in\mathcal{J}} r(w,j) \propto p(U_t = u, I_{t'} = I_t)$ as a Dirichlet prior estimate.

3.3 Normalized Variant

In equation 5, nothing prevents us from restricting the sum over users to essentially any condition on v, as far as the sum over v adds to one. In particular we can take the condition that v has been seen interacting with i in the system:

$$
\begin{aligned}
&p(I_t = i | U_t = u) \\
&\quad \sim \sum_{v\in\mathcal{U}} p(I_{t'} = i | U_{t'} = v, r(v,i) > 0) \\
&\qquad\qquad\qquad p(U_{t'} = v | U_t = u, I_{t'} = I_t, r(v,i) > 0)
\end{aligned}
$$

With this variation, it is easy to see that equation 11 remains almost the same but, quite importantly, a constant C appears:

$$
p(I_t = i | U_t = u) \sim C \sum_{v\in\mathcal{U}} \frac{\sum_{j\in\mathcal{J}} r(u,j) r(v,j)}{\sum_{j\in\mathcal{J}} r(u,j) \sum_{j\in\mathcal{J}} r(v,j)} r(v,i)
$$

with $C = 1/\sum_{w\in\mathcal{U}:r(w,i)>0} \sum_{j\in\mathcal{J}} r(u,j) r(w,j)$. This is a normalized variant since the numerators of the weights in the sum of ratings $r(v,i)$ above now sum to 1, similar in structure to the heuristic normalized kNN [1,16] (except for $\sum_{j\in\mathcal{J}} r(v,j)$ in the denominator being left out of the normalization). Neighbor selection and smoothing can be introduced in a similar way as in the non-normalized variant.

3.4 Item-Based kNN

The analysis of user-based kNN can be developed in an item-oriented version as well. For this purpose, we marginalize $p(I_t = i | U_t = u)$ by neighbor items j, and we apply analogous developments to the ones detailed in section 3.2 for the user-based variant, only we start by inverting the target user and item in the initial ranking function:

$$
\begin{aligned}
&p(I_t = i | U_t = u) \propto_u p(I_t = i) p(U_t = u | I_t = i) \\
&\quad \sim p(I_t = i) \sum_{j\in\mathcal{J}} p(I_{t'} = j | I_t = i, U_{t'} = U_t) \\
&\qquad\qquad\qquad p(U_t = u | I_t = i, I_{t'} = j, U_{t'} = U_t)
\end{aligned}
\tag{12}
$$

Now similarly to the user-based version, we consider the independence assumption between target user and item for lack of observations:

$$
p(U_t = u | I_t = i, I_{t'} = j, U_{t'} = U_t) \sim p(U_{t'} = u | I_{t'} = j)
$$

We rewrite again the conditional pairwise item dependence as:

$$
p(I_{t'} = j | I_t = i, U_{t'} = U_t) = \frac{p(I_{t'} = j, I_t = i, U_{t'} = U_t)}{\sum_{k\in\mathcal{J}} p(I_{t'} = k, I_t = i, U_{t'} = U_t)}
$$

Applying equivalent model estimations on observed data, we get:

$$
\begin{aligned}
&p(I_t = i | U_t = u) \\
&\quad \propto \left(\sum_{v\in\mathcal{U}} r(v,i) \right) \sum_{\substack{j\in\mathcal{J} \\ j\neq i}} \frac{\sum_{v\in\mathcal{U}} r(v,i) r(v,j)}{\sum_{k\in\mathcal{J}} \sum_{v\in\mathcal{U}} r(v,i) r(v,k)} \cdot \frac{r(u,j)}{\sum_{v\in\mathcal{U}} r(v,j)} \\
&\quad \propto C \sum_{\substack{j\in\mathcal{J} \\ j\neq i}} \frac{\sum_{v\in\mathcal{U}} r(v,i) r(v,j)}{\sum_{v\in\mathcal{U}} r(v,j)} r(u,j)
\end{aligned}
$$

with $C = \sum_{v\in\mathcal{U}} r(v,i) / \sum_{j\in\mathcal{J}} \sum_{v\in\mathcal{U}} r(v,i) r(v,j)$.

Comparing this to equations 2 plus 4, we see we have obtained a formulation that has analogies to the heuristic item-based kNN. One difference is that the $\sum_{v\in\mathcal{U}} r(v,i)$ term is missing in the denominator here, and what is more, it is present in the numerator through C. We see on the other hand that the denominator in C somewhat balances this (for instance if $\sum_{j\in\mathcal{J}} r(v,j)$ were constant on v, then C would be constant on i).

On the other hand, as in user-based, we may consider the normalized version by conditioning on the observation of some interaction between the target user u and the neighbors j, which results in a refinement of the normalizing constant: $C = \sum_{v\in\mathcal{U}} r(v,i) / \sum_{j\in\mathcal{J}:r(u,j)>0} \sum_{v\in\mathcal{U}} r(v,i) r(v,j)$. And neighborhoods and smoothing can be analogously introduced.

4 POPULARITY BIASES

Aiming to analyze the generic trends within the kNN formulations, we examine what course these algorithms follow when users' tastes (respectively item trends in the item-based variants) are pairwise independent.

4.1 User-Based Bias

In the user-based variant, user independence means $p(U_{t'} = v | U_t = u, I_{t'} = I_t) \sim p(U_{t'} = v)$ for all v and u, whereby equation 5 becomes:

$$
\begin{aligned}
p(I_t = i | U_t = u) &\sim \sum_{v\in\mathcal{U}} p(I_{t'} = i | U_{t'} = v) p(U_{t'} = v) \\
&= p(I_{t'} = i) \sim \frac{\sum_{v\in\mathcal{U}} r(v,i)}{\sum_{w\in\mathcal{U}} \sum_{j\in\mathcal{J}} r(w,j)} \propto \sum_{v\in\mathcal{U}} r(v,i)
\end{aligned}
\tag{13}
$$

Thus by this independence assumption the ranking function $p(I_t = i | U_t = u)$ is estimated as proportional to the number (or the sum) of ratings of i, that is, the item's popularity. We also see that under this assumption we are approximating $(I_t = i | U_t = u) \sim p(I_{t'} = i)$, i.e. the probability that a random user picks the item i, which is a natural understanding of the notion of item popularity.

This means that whereas user-based kNN takes an exact decomposition of $p(I_t = i | U_t = u)$ by the law of total probability, popularity (as per equation 5) represents an inexact approximation to $p(I_t = i | U_t = u)$ by an additional independence assumption. To the extent that in some case we actually had $p(U_{t'} | U_t, I_{t'} = I_t) \sim p(U_{t'})$, popularity would start becoming a fair estimate for $p(I_t = i | U_t = u)$. In other words, kNN degrades to popularity-based recommendation when user tastes are independent from each other. In the absence of any particular pairwise bias in $p(U_{t'} | U_t, I_{t'} = I_t)$, user-based kNN simply ranks items by the number of ratings they have (or their sum).

If on the contrary user tastes are not pairwise independent, we have $p(U_{t'}|U_t, I_{t'} = I_t) \neq p(U_{t'})$, and equation 13 becomes a worse and worse approximation of equation 5 as the conditional distribution diverges from the user prior. In such situations kNN, as a more exact development of $p(I_t = i|U_t = u)$, should perform better than popularity, as the formal inaccuracy in the probabilistic development can be expected to translate into corresponding errors in the predictions thereupon.

Formal imprecisions are however not the only source of prediction inaccuracies. Another fundamental one is the approximate estimation of distributions from observed data. To this respect, popularity only uses the estimate $p(I_{t'} = i) \propto \sum_{v\in\mathcal{U}} r(v,i)$ as per equation 8, whereas kNN introduces, in addition, the estimations $p(I_{t'} = i|U_t = v) \propto r(v,i)$ by equation 8 and $p(U_{t'} = v|U_t = u, I_{t'} = I_t) \propto \sum_{j\in\mathcal{J}} r(u,j) r(v,j)$ by equation 10. If the estimates applied in kNN happen to be less reliable than the ones in popularity, then the estimation inaccuracy could outdo the formal accuracy to the point that kNN may perform worse than popularity. This might happen, as we shall see, in data sparsity situations, to which kNN appears to be more vulnerable than popularity. This is to be expected, since the estimation of popularity is one-dimensional (involving a single random variable i), whereas the estimates of kNN are two-dimensional (involving two variables v and i, u and v, respectively), and therefore more vulnerable to sparsity.

In sum, the probabilistic formulation thus shows why there is some degree of structural relation between kNN and popularity, and identifies a key factor (user dependence) for kNN to be (more) effective (than popularity).

4.2 Normalized Variant Bias

The pairwise user independence in the normalized version would mean $p(U_{t'} = v|U_t = u, I_{t'} = I_t, r(v,i) > 0) \sim p(U_{t'} = v|r(v,i) > 0)$, which for this variant yields:

$$\begin{aligned} &p(I_t = i|U_t = u) \\ &\quad \sim \sum_{v\in\mathcal{U}} p(I_{t'} = i|U_{t'} = v, r(v,i) > 0) p(U_{t'} = v|r(v,i) > 0) \\ &\quad = \sum_{v\in\mathcal{U}} p(I_{t'} = i, U_{t'} = v|r(v,i) > 0) \\ &\quad \sim \sum_{v\in\mathcal{U}} r(v,i) \Bigg/ \sum_{\substack{v\in\mathcal{U} \\ r(v,i)>0}} \sum_{j\in\mathcal{J}} r(v,j) \end{aligned}$$

If $\sum_{j\in\mathcal{J}} r(v,j)$ are not too different for the set of raters v of each item i (i.e. each item has a similar mix of highly and moderately active users), then $\sum_{v\in\mathcal{U}:r(v,i)>0} \sum_{j\in\mathcal{J}} r(v,j)$ is approximately proportional to the number of users who interacted with i, and $p(I_t = i|U_t = u)$ is then similar to the average rating of item i.

We thus remarkably see that the normalized user-based kNN becomes not similar to the sum of rating values of the target item, but to its average rating. This hints a potentially fundamental difference in the behavior of the normalized and non-normalized variants which can show up when the distribution of the sum and the average of item ratings diverge.

4.3 Item-Based Bias

If we assume pairwise item independence, i.e. $p(I_t|I_{t'}, U_{t'} = U_t) \sim p(I_t)$ then equation 12 becomes:

$$p(I_t = i|U_t = u) \sim \sum_{j\in\mathcal{J}} p(I_t = i) p(I_{t'} = j|U_{t'} = u) = p(I_t = i)$$

That is, item-based kNN becomes ranking by popularity, thus showing a similar popularity bias as in the user-based variant: item-based kNN degrades to popularity when the divergence of the inter-item distribution from the item prior is weak.

It is easy to see that the normalized item-based variant is equally related to popularity, and not to the average rating as its normalized user-based counterpart. This is because $p(I_t = i|r(u,j) > 0) \sim p(I_t = i)$, while $p(I_{t'} = j|U_{t'} = u, r(u,j) > 0)$ simply adds to 1 over j.

5 EMPIRICAL OBSERVATION

In order to test and quantify empirically the trends and theoretical analysis derived in the previous sections, we run a series of experiments on publicly available data. For this purpose we use data from the movie and music domains provided in the MovieLens 1M,[1] Netflix,[2] and Last.fm 1K[3] datasets. We show in Table 1 the volumetric details of the three datasets.

MovieLens is perhaps the most widely used dataset in the recommender systems research literature. It includes ratings for movies in a 1-5 scale by users of the MovieLens application. The Netflix dataset contains data of similar nature collected from Netflix subscriptors, and was released in 2006 in the Netflix Prize contest. The Last.fm dataset was collected by O. Celma [5] and includes records of music tracks played by users on Last.fm. The recorded data for each play action includes the user ID, track, artist and timestamp. For our experiments we just aggregate this data into user / artist / playcount triplets.

5.1 General Performance

The first question we aim to check is whether the probabilistic kNN formulations are as effective as the traditional versions. For this purpose we implement the probabilistic user-based (PUB), normalized user-based (nPUB), and item-based variants (PIB, nPIB) developed along the previous sections. We take the implementations of the heuristic kNN algorithms, as described by equations 1-4, provided in the RankSys[4] public library, including user-

Table 1: Dataset characteristics.

	Nr. users	Nr. items	Nr. ratings
MovieLens 1M	6,040	3,706	1,000,209
Netflix	480,189	17,770	100,480,507
Last.fm	992	174,091	898,073

Table 2: Neighborhood size k in the kNN configuration on each dataset ($k = \infty$ indicates all items are taken as neighbors).

	HUB	PUB	HIB	PIB	nHUB	nPUB	nHIB	nPIB
MovieLens 1M	50	50	100	100	10	20	10	40
Netflix	100	100	100	50	10	10	10	100
Last.fm	100	500	∞	∞	10	10	10	∞

[1] http://grouplens.org/datasets/movielens/1m

[2] http://www.netflixprize.com

[3] http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html

[4] http://ranksys.org

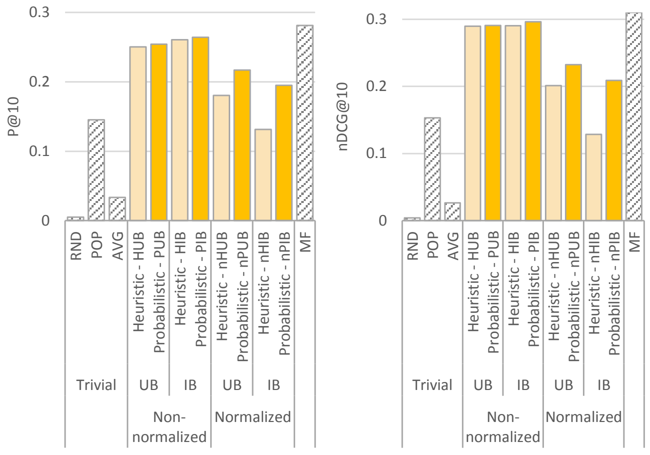

Figure 1: Comparative performance in the MovieLens 1M dataset. The metric value bars for heuristic and probabilistic versions of each variant are shown next to each other for ease of comparison. We use a darker color for the probabilistic versions, and a streaked color pattern for the algorithms other than kNN. The differences between the probabilistic and heuristic versions of non-normalized variants (PUB > HUB, PIB > HIB) are not statistically significant. All other pairwise comparisons are significant (Student's two-tailed t-test at $p < 0.001$).

based (HUB), item-based (HIB), and normalized (nHUB, nHIB) variants as well. We shall show here the results with the overall most effective configuration of the heuristic variants, which uses the cosine similarity. We select the best neighborhood k for each kNN variant (in terms of P@10) by grid search starting with steps of 10 in the 10-100 interval, then steps of 100 in 100-1,000, and so forth. For all the normalized variants, we set a minimum number of 5 neighbor ratings for an item to be recommended to a target user. Table 2 shows the neighborhood size settings for each kNN variant on each dataset. We found Dirichlet smoothing [25] slightly improves the probabilistic algorithms on Netflix and MovieLens, with $\mu = 100$ on MovieLens and $\mu = 200$ on Netflix for PUB and nPUB; and $\mu = 200$ on MovieLens, $\mu = 20{,}000$ on Netflix for PIB and nPIB. We preprocess the rating values for all purposes by subtracting a relevance threshold (3 in MovieLens and Netflix, and 0 for Last.fm playcounts) and truncating the difference at zero.

As a frame of reference, we include trivial baseline recommendations: popularity-based (POP), average (AVG) and random (RND). In the datasets that consider rating values representing dislike (namely MovieLens and Netflix), we consider popularity defined as the sum of "positive" ratings (i.e. rating values above the relevance threshold), which provides a more effective and sensible recommendation than the total number of ratings (we obtain similar results with the number –instead of the sum– of positive ratings). For the average rating to work as a recommendation, one needs to smooth it, otherwise the top average values are taken up by items with very few ratings, which make for very poor recommendations. We have observed that a simple additive smoothing [25] is effective enough. Equivalently, in the results we report here, we simply require the items to have a minimum of 5 ratings to be recommended by average rating.

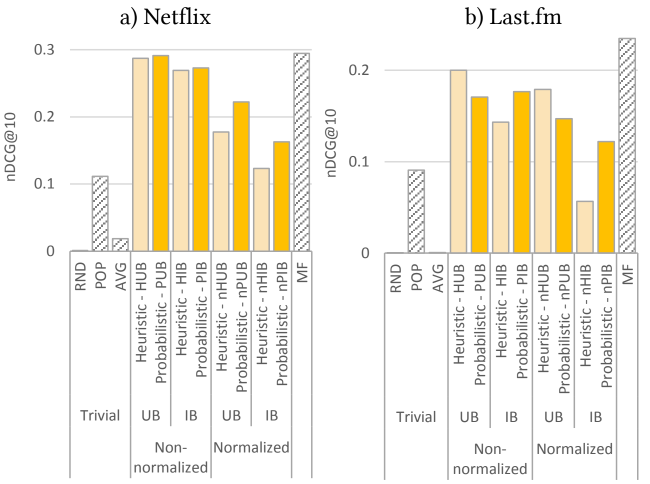

Figure 2: Comparative performance in the Netflix and Last.fm datasets. We use similar color codes as used in Fig. 1. All the pairwise comparisons are statistically significant (Student's two-tailed t-test at $p < 0.001$).

As a top-performing reference, we also include a matrix factorization approach (MF) proposed in [12] and implemented in RankSys, informally tuning the parameter values based on previously reported configurations [12,22] and our own experience with well-behaving values for this algorithm, finally taking $k = 20$ factors, $\alpha = 1$, and $\lambda = 0.1$, with 20 iterations on all datasets, except $k = 50$ on Netflix.

We test the effectiveness of the algorithms by splitting the rating data into training and test sets. Training ratings are given as input data for the recommendation algorithms, whereas positive ratings in the test set are taken as positive relevance judgments in the computation of the metrics. Test rating values indicating dislike (values below the relevance threshold) are taken as non-relevant judgments, and so are unrated items (as the equivalent of unjudged documents in IR tasks). In all three datasets we randomly sample 20% of the data for testing, and leave the remaining 80% for training.

Fig. 1 shows the results on the MovieLens dataset in terms of precision and nDCG. We can see that the probabilistic and heuristic non-normalized versions perform comparably well. PUB and PIB show slightly better results than their heuristic counterparts, but the differences are not statistically significant in this experiment. We also see that the normalized variants do not perform as well as the non-normalized ones. This goes along with the lower performance of average rating compared to popularity. Even though we are not aware of an explicit comparison and discussion on this point in the literature, we can assert that normalized kNN variants were devised for the rating value prediction task, to be evaluated with error metrics such as RMSE [10]. And this can account for their inferior performance at item ranking, a different task from the one they were originally designed for.

Matrix factorization shows to be the best system, confirming prior accounts that superior accuracy can be achieved by matrix factorization approaches [12]. But this is so by a rather small difference with respect to the best performing kNN algorithms. We also confirm the non-negligible effectiveness of recommendation by plain popularity reported in prior work [6], which in this experiment is as much as about half as effective as the best performing algorithm.

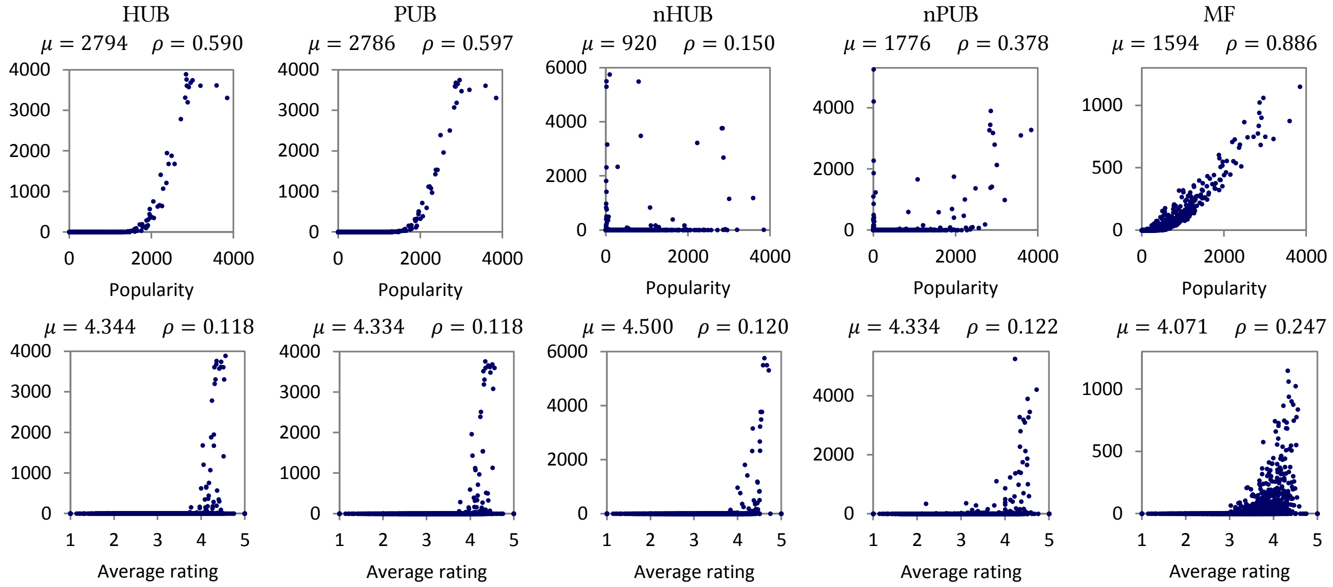

Figure 3: Popularity biases in recommendation algorithms on MovieLens 1M. Each point in the scatterplots corresponds to an item in the dataset; the x axis indicates its sum of positive ratings (top), and its average rating value (bottom); and the y axis indicates the number of users to whom the item is recommended in the top 10 by each algorithm. As further indication of the corresponding bias, on top of each plot we indicate a) as μ the global popularity –rounded to integers– (top) or rating value (bottom), averaged over the whole top 10 cutoff of each algorithm for all users, and b) the global Pearson correlation ρ between the x and y values. All kNN algorithms are shown here with $k = \infty$.

Complementing the MovieLens results, Fig. 2 shows similar trends on the two other datasets. For the sake of space we just show the nDCG results there (precision is very much in line with these). The comparison between the heuristic and probabilistic versions sometimes favors one and sometimes the other, by small differences, thus showing an overall equivalent empirical effectiveness. The generally better performance of the probabilistic version in the normalized variants on Netflix and MovieLens is partly achieved by the Dirichlet smoothing, though it brought no improvement on Last.fm. We lowered the minimum neighbor requirement to 2 for nHIB on Last.fm as it was suffering from too low coverage by the extremely long-tailed rating distribution over items. All in all, the normalized variants remain (with the exception of nHUB on Last.fm) systematically inferior to the non-normalized ones.

5.2 Popularity Biases

In order to check to what extent kNN looks alike or deviates from popularity, we visualize in Fig. 3 how much of each item popularity range is recommended in the top 10 of the ranking for each algorithm. The trends we show and discuss next are similar in the other two datasets. The details of the scatterplot display are described in the figure legend. To avoid the distorting effect of neighbor selection and show the biases more clearly, the plots are computed for the kNN algorithms taking all users (or items) as neighbors.

We see how HUB and PUB display a clear bias towards recommending popular items, confirming the trend formally analyzed in section 4. The bias may seem a trifle cleaner in the probabilistic version, as its structural connection to popularity is more direct. As context, the average popularity in the whole dataset is 261, and the global average rating is 3.581. The bias is clearly weaker in the normalized variants, which lean instead towards the average rating. We can see that the non-normalized variants also have a bias towards high average ratings, and even the normalized kNN have an, albeit weak, slight popularity bias. We can cast these as indirect biases, due to the correlation that actually exists in the dataset between the number of positive ratings and the average rating value. We show this in Fig. 7a, where we see that the average rating of the most popular items is very high. It has been hypothesized that this can be attributed to the propensity of people to rate items they like rather than ones they do not [4,19,20].

It is also interesting to see that the MF algorithm has a clear bias towards popular and highly rated items. This may hint that to some extent our analysis and findings on kNN could be generalized to other collaborative filtering approaches.

Finally, Fig. 4 confirms our theoretical findings regarding the popularity bias in item-based kNN, which in the non-normalized version is as strong as in the user-based counterpart. We also find a striking structural difference between the formal normalized item-based kNN and its heuristic counterpart. We confirm that –contrarily to the normalized user-based algorithm– the probabilistic item-based algorithms are indeed biased to popularity quite as much with or without normalization, while the normalization in nHIB completely does away with the popularity trend. This may also account for the somewhat worse performance of the heuristic normalized item-based variant we see in Fig. 1 and 2. The poor results of nHIB illustrate a weakness of heuristic compared to more principled approaches. Heuristics rely more heavily on trial, error and chance, and their success is hence harder to ensure beforehand.

5.3 A Second Look at Biases

Awareness of popularity biases in the data consumed by recommender systems and their evaluation has risen in the field in recent years. Researchers and practitioners have noticed the biases and are posing questions about it [3,4,8,13,19,20].

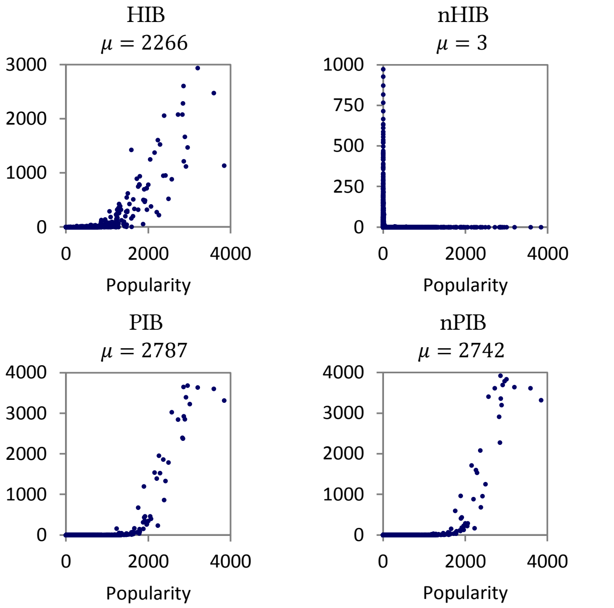

Figure 4: Popularity bias in item-based kNN on MovieLens 1M. The x and y axes and the μ average have the same meaning as in Fig. 3.

We may say nonetheless that the effects of the popularity biases on offline experiments have not been thoroughly researched and understood yet. It is therefore legitimate to wonder whether a biased evaluation might be suffering from some sort of distortion on the comparison of biased algorithms. The factors that can be identified to generate popularity biases include discovery distributions (bias in the means by which items reach users −or vice-versa), bias towards rating liked items (user behavior bias), and ultimately the actual user tastes (user preference bias) [4,19]. As a step in the direction of shedding light on this question, we set up to carry out an experiment with data missing at random where we remove the first two (artificial) sources of biases, isolating the true preference distribution as the only bias in the data.

For this purpose, we randomly sampled music tracks from a large database, Deezer,[5] containing over 30 million songs at the time of this experiment. Using this list of songs, we set up a survey on CrowdFlower[6] where we asked a number of users to rate 100 songs each, sampled uniformly at random from our set in such a way that each music track got around 100 ratings. The resulting dataset[7] includes 103,584 ratings by 1,054 users on 1,084 tracks.

A unique property of this dataset compared to all others available today is the absence of biases in the exposition of users to items. The discovery of items and the decision to rate them is forced rather than free. Even though the total number of ratings of each item is generated from a uniform distribution, the number of positive ratings is not uniform, as we show in Fig. 5, where we also see the rather flat distribution of the total number of ratings, in contrast to MovieLens (which is representative of the typical highly skewed distribution of the number of ratings over items in common datasets). In this dataset the kNN variants work best taking all neighbors (i.e. $k = \infty$) except $k = 200$ for nHIB. We use no Dirichlet smoothing and require no minimum number of neighbors in the normalized version, as we found these adjustments do not perceptibly improve the results on this dataset. We configure MF with $k = 5$, $\alpha = 10$, $\lambda = 500$, 20 iterations. To smooth the variance due to the smaller size of this dataset we average the metrics over 10 repetitions of the experiment (namely, of the random rating split, with 5-fold cross-validations each).

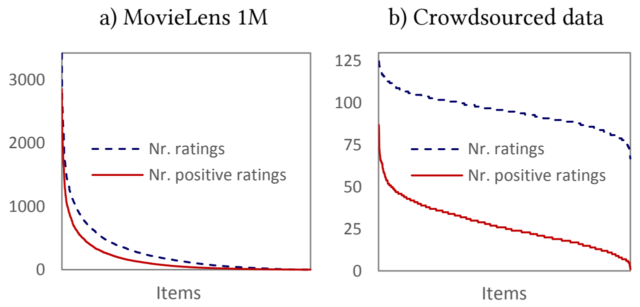

Figure 5: Total and positive rating distributions in MovieLens (left) vs. our crowdsourced dataset (right). The items in the x axis are ordered by decreasing number of ratings for the blue dashed line, and by decreasing number of positive ratings for the red continuous line.

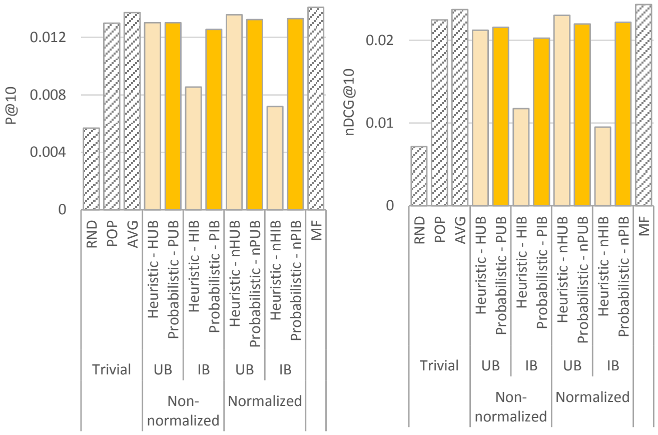

Figure 6: Comparative performance on the crowdsourced dataset. All pairwise comparisons between any two item-based variants are statistically significant (Student's two-tailed t-test at $p < 0.001$), whereas they are not significant among user-based variants (except for nHUB > nPUB in nDCG@10).

Fig. 6 shows the results. We see that the difference between all algorithms gets considerably reduced, including the random recommendation which performs at about 1/3 as well as the best recommender. This can be explained by the absence of bias in the total number of ratings over items. Yet the bias in the number of positive ratings (albeit less pronounced than in typical datasets) enables a better than random performance by popularity recommendation.

We also see that in these conditions popularity becomes a difficult baseline to overcome. We believe this may be due to the low "local" density of positive ratings. The total rating density of about 10% is higher than it is in all the other datasets we have reported results for (4.5% in MovieLens, 1.2% in Netflix, 0.5% in Last.fm), but

[5] https://www.deezer.com
[6] https://www.crowdflower.com
[7] The dataset is publicly available at http://ir.ii.uam.es/cm100k.

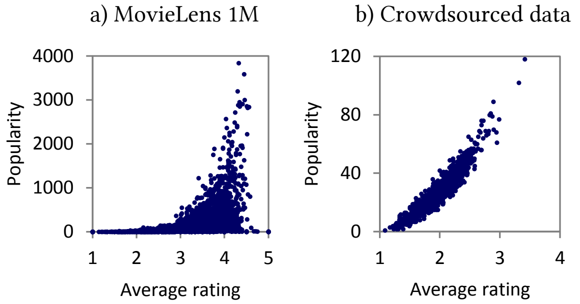

Figure 7: Average rating vs. popularity in MovieLens 1M (left), and the crowdsourced data (right). Each point in the scatterplot represents an item in dataset, the x axis is the average rating of the item, and the y value is the sum of positive rating values (popularity) of the item.

it is lower both in terms of positive preferences per user (~28 on average vs. e.g. ~95 in MovieLens) and per item (~27 vs. ~155 in MovieLens), and the resulting low pairwise overlapping between users and between items, further intensified by the uniform rating distribution. Still, we see several interesting phenomena. First, recommendation based on the average rating catches up in performance with the popularity-based recommendation. This makes sense, since in a uniform rating distribution, the number of positive ratings strongly correlates with the average rating value, as we see in Figure 7b. Yet it is quite a salient finding that in the absence of a bias in the number of ratings, the average rating is as effective as popularity, or more, in achieving decent recommendations.

On the other hand, following this trend, and quite remarkably, the normalized kNN variants, which we showed to go theoretically along with the average rating, now match or even slightly outperform their non-normalized counterparts. Fig. 8 further explains the good performance of the normalized algorithms: the normalized user-based versions retain a stronger popularity bias than they do on the biased datasets in section 5.2. In the absence of an item selection bias in the input data, popularity and average rating become almost equivalent signals, as we can see in Fig. 7b, and the normalized versions seem to more fully capture the remaining item popularity distribution as a meaningful and effective signal. We omit the plots for the item-based versions, but the pattern is similar to what we observed in MovieLens: the probabilistic variant retains its popularity bias in the normalized version, not so in the heuristic variant.

This qualitative disagreement in the comparative effectiveness of normalized vs. non-normalized variants with respect to the results we previously obtained in section 5.2 strike us as a hint that we should revise the observed results in offline experiments with the common biased datasets. The contrast in observations raises the question whether the low results of normalized kNN in common datasets is due to a truly poor performance, or to the bias in observations. The only definitive way to know for sure would be to obtain the missing relevance information in those datasets.

6 RELATED WORK

Many collaborative filtering methods have been proposed that build on a probabilistic basis [9,11] –and even on probabilistic IR models [21,23,24]–, but to the best of our knowledge none has been proposed that explains or results in the structure of a kNN scheme through a fully probabilistic development.

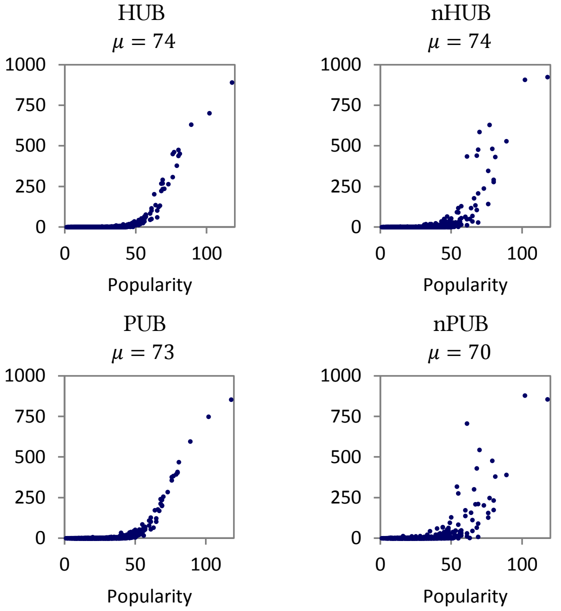

Figure 8: Popularity bias in user-based kNN on the crowdsourced dataset. The x and y axes and the μ average have the same meaning as in Fig. 3 and 4. As context, the global average popularity in this dataset is 26.

Notably for our purpose nonetheless, Deshpande and Karypis [7] explored the use of conditional probabilities between users and items in the role of the similarity function in kNN, but this was otherwise an isolated probabilistic piece in a heuristic scheme for the rest of the formulation. Later on Aiolli [2] also tested conditional probability in place of cosine in a heuristic scheme, and hinted at the structural connection between the cosine similarity and conditional probabilities. More recently in this line, Valcarce et al. [21] apply more sophisticated probabilistic IR models to rank and select neighbors, but then follow on into a cosine-based ranking function similar to equations 1-4. Random walk models [9] and Markov chains have also been used to build solutions upon probabilistic inspiration, though they have been generally developed into a mix of formal and heuristic aspects (e.g. by a heuristic computation of the transition probabilities).

The concentration and popularity bias of collaborative filtering is a well-known issue that several researchers have pointed at and addressed in the field. To name a few, Fleder and Hosanagar [8] research the recursive dimension of the concentration biases of recommendations in their feedback loop. Cremonesi et al. [6] were to the best of our knowledge the first to report and discuss on the decent performance achieved by plain popularity-based recommendation. They proposed a simple trick to avoid it, consisting in excluding most popular items from the evaluation. Realizing the popularity biases that are commonly present in rating data streams, Steck [19,20] suggests biases are likely to lurk in the evaluation results drawn upon such conditions, and proposes new metrics aiming to cope with the biases, as well as modifications in collaborative filtering methods in the same direction. Bellogín et al. [3] similarly analyze the strong popularity biases that surface in IR evaluation methodologies when applied to recommendation. In prior work [4] we also addressed the question whether popularity reflects or contradicts the true distribution of user preference, and aimed to identify key factors that may determine the answer. Also recently, Jannach et al. [13] present a thorough study of concentration biases with respect to both the number of ratings and the average rating values. The reported study has a primary empirical orientation, testing and comparing different algorithms and configuration details, and proposing some bias-mitigating ideas.

Our work very much shares the concerns of such studies, which we address in a complementary direction: that of seeking theoretical explanations for the empirically observed and described phenomena, aiming to glimpse fundamental causes beneath them. We focus for this purpose on a narrower, specific recommender system approach, the kNN scheme.

7 CONCLUSIONS

We have proposed a fully probabilistic yet intuitive formalization for the nearest-neighbor collaborative filtering approach. The proposed formulation shows equivalent empirical effectiveness to the traditional heuristic formulations, and thus can serve to analyze properties of the kNN scheme on a more principled ground.

The probabilistic basis provides an explanation of the effectiveness of kNN at the same time that it relates the algorithm to popularity distributions, thus helping understand this well-known bias. The effectiveness of kNN relies on the pairwise statistical dependence between user behaviors, and can be expected to be most effective to the extent that the pairwise conditional distributions deviate from the prior collective behavior. Conversely, the more a user's tastes are as similar to any user as they are to the next, the more kNN behaves like a plain majority-based recommendation.

We further show that the theoretical characterizations align the algorithm variants along two fundamentally different trends: popularity and average rating. We report experiments showing that the comparison between algorithm variants (and normalized vs. non-normalized in particular) can change depending on statistical properties of the input data and the biases in the implicit or explicit item sampling in the data.

Our results suggest that further research would be needed to determine to what extent popularity biases are a helpful signal to produce effective recommendations, or a confounder that may distort offline results. Even a simple question as which of popularity and the average rating is a better signal would deserve further analysis at both the empirical and formal levels –further conclusions might be drawn on the corresponding kNN variants they are a trend within. The formal analysis of kNN we present here aims to be a step in this direction. Further research in this line might perhaps lead towards more generalized findings that are not exclusive of kNN but may be shared by collaborative filtering approaches.

ACKNOWLEDGMENTS

This work was supported by the national Spanish Government (grant nr. TIN2016-80630-P).

REFERENCES

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

[2] F. Aiolli. 2013. Efficient top-n recommendation for very large scale binary rated datasets. In *Proceedings of the 7th ACM Conference on Recommender Systems* (RecSys 2013). ACM, New York, NY, USA, 273–280.

[3] A. Bellogín, P. Castells, and I. Cantador. Statistical Biases in Information Retrieval Metrics for Recommender Systems. *Information Retrieval.* Springer, Netherlands, in press.

[4] R. Cañamares and P. Castells. 2014. Exploring social network effects on popularity biases in recommender systems. In *Proc of the 6th Workshop on Recommender Systems and the Social Web* (RSWeb 2014) at the *8th ACM Conference on Recommender Systems* (RecSys 2014). Foster City, CA, USA, October 2014.

[5] O. Celma. 2010. *Music Recommendation and Discovery in the Long Tail.* Springer-Verlag Berlin Heidelberg.

[6] P. Cremonesi, Y. Koren, and R. Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of the 4th ACM Conference on Recommender Systems* (RecSys 2010). ACM, New York, NY, USA, 39–46.

[7] M. Deshpande and G. Karypis. 2004. Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems* 22, 1 (Jan 2004). ACM, New York, NY, USA, 143– 177.

[8] D. Fleder and K. Hosanagar. 2009. Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management Science* 55, 5 (May 2009). Informs, Catonsville, MD, USA, 697–712.

[9] F. Fouss, A. Pirotte, J-M. Renders, and M. Saerens. 2007. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 19, 3 (March 2007). IEEE, Piscataway, NJ, USA, 355–369.

[10] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. 2004. Evaluating Collaborative Filtering .Recommender Systems. *ACM Transactions on Information Systems* 22, 1 (Jan. 2004). ACM, New York, NY, USA, 5–53.

[11] T. Hofmann. 2003. Collaborative filtering via Gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR 2003). ACM, New York, NY, USA, 259–266.

[12] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining* (ICDM 2008). IEEE Computer Society, Washington, DC, USA, 15–19.

[13] D. Jannach, L. Lerche, I. Kamehkhosh, and M. Jugovac. 2015. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction* 25, 5 (Dec. 2015). Kluwer Academic Publishers Hingham, MA, USA, 427–491.

[14] Y. Koren. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data* 4, 1 (Jan 2010), ACM, New York, NY, USA.

[15] G. Linden, B. Smith, and J. York. 2003. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (Jan. 2003). IEEE, Piscataway, NJ, USA, 76–80.

[16] X. Ning, C. Desrosiers, and G. Karypis. 2015. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In: *Recommender Systems Handbook, 2nd ed.*, F. Ricci, L. Rokach, and B. Shapira (Eds.). Springer, New York, NY, USA, Chapter 2, 37–76.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. T. Riedl. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work* (CSCW 1994). ACM, New York, NY, USA, 175–186.

[18] F. Ricci, L. Rokach, and B. Shapira (Eds.). 2015. Recommender Systems: Introduction and Challenges. In: *Recommender Systems Handbook, 2nd ed.*, F. Ricci, L. Rokach, and B. Shapira (Eds.). Springer, New York, NY, USA, Chapter 2, 1–34.

[19] H. Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD 2010). ACM, New York, NY, USA, 713–722.

[20] H. Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the 5th ACM Conference on Recommender Systems* (RecSys 2011). ACM, New York, NY, USA, 125–132.

[21] D. Valcarce, J. Parapar, and A. Barreiro. 2006. Language Models for Collaborative Filtering Neighbourhoods. In *Proceedings of the 38th European Conference on Information Retrieval* (ECIR 2016). LNCS Vol. 9626. Springer, Switzerland, 614–625.

[22] S. Vargas and P. Castells. 2014. Improving sales diversity by recommending users to items. In *Proceedings of the 8th ACM Conference on Recommender Systems* (RecSys 2014). ACM, New York, NY, USA, 145–152.

[23] J. Wang, S. Robertson, A. P. de Vries, and M. J. T. Reinders.2008. Probabilistic relevance ranking for collaborative filtering. *Information Retrieval* 11, 6 (Dec. 2008). Springer, Netherlands, 477–497.

[24] J. Wang, A. P. de Vries, M. J. T. Reinders. 2008. Unified relevance models for rating prediction in collaborative filtering. *ACM Transactions on Information Systems* 26, 3 (June 2008). ACM New York, NY, USA.

[25] C. Zhai and, J. D. Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems* 22, 2 (April 2004). ACM, New York, NY, USA, 179–14.