Part II

Evaluating performance in recommender systems

If you cannot measure it, you cannot improve it.
William Thomson (Lord Kelvin)

Chapter 3

Evaluation of recommender systems

The evaluation of recommender systems has been, and still is, the object of active research in the field. Since the advent of the first recommender systems, recommendation performance has been usually equated to the accuracy of rating prediction, that is, estimated ratings are compared against actual ratings, and differences between them are computed by means of the *mean absolute error* and *root mean squared error* metrics. In terms of the effective utility of recommendations for users, there is however an increasing realisation that the quality (precision) of a ranking of recommended items can be more important than the accuracy in predicting specific rating values. As a result, precision-oriented metrics are being increasingly considered in the field, and a large amount of recent work has focused on evaluating top-N ranked recommendation lists with the above type of metrics.

In this chapter we provide a survey of different evaluation metrics, protocols, and methodologies in the recommender systems field. In Section 3.1 we provide a preliminary overview of how recommender systems are evaluated, presenting the main (online and offline) evaluation protocols and dataset partitioning methods. Next, in Section 3.2 we present the most common evaluation metrics, classified into error-based and precision-based metrics, and in Section 3.3 we describe different dataset partition strategies used in the experimental configurations. Finally, in Section 3.4 we present some evaluation datasets which are commonly used by the research community, and that were used in the experimental work of this thesis.

3.1 Introduction

The evaluation of recommender systems has been a major object of study in the field since its earliest days, and is still a topic of ongoing research, where open questions remain (Herlocker et al., 2004; Shani and Gunawardana, 2011). Two main evaluation protocols are usually considered (Gunawardana and Shani, 2009): *online* and *offline*. In this thesis we focus on offline evaluation, which lets compare a wide range of candidate algorithms at a low cost (Shani and Gunawardana, 2011). For a review of the different tasks and protocols for online recommendation evaluation, see (Shani and Gunawardana, 2011), (Pu et al., 2012), and (Kohavi et al., 2009).

Drawing from methodological approaches common to the evaluation of classification, machine learning and information retrieval algorithms, offline recommender system evaluation is based on holding out from the system a part of the available knowledge of user likes (test data), leaving the rest (training data) as input to the algorithm, and requiring the system to predict such preferences, so that the goodness of recommendations is assessed in terms of how the system's predictions compare to the withheld known preferences. In the dominant practice, this comparison has been oriented to measure the accuracy of rating prediction, computing error-based metrics. However, in terms of the effective utility of recommendations for users, there is an increasing realisation that the quality (precision) of the ranking of recommended items can be more important than the accuracy (error) in predicting specific rating values. As stated in (Herlocker et al., 2004), the research community has moved from the *annotation in context* task (i.e., predicting ratings) to the *find good items* task (i.e., providing users with a ranked list of recommended items), which better corresponds to realistic settings in working applications where recommender systems are deployed. As a result, precision-oriented metrics are being increasingly considered in the field. Yet there is considerable divergence in the way such metrics are applied by different authors, as a consequence of which the results reported in different studies are difficult to put in context and be compared.

In the classical formulation of the recommendation problem, user preferences for items are represented as numeric ratings, and the goal of a recommendation algorithm consists of predicting unknown ratings based on known ratings and, in some cases, additional information about users, items, and the context. In this scenario, the accuracy of recommendations has been commonly evaluated by measuring the error between predicted and known ratings, using metrics such as the Mean Absolute Error (MAE), and the Root Mean Squared Error (RMSE). Although dominant in the literature, some authors have argued this evaluation methodology is detrimental to the field since the recommendations obtained in this way are not the most useful for users (McNee et al., 2006). Acknowledging this, recent work has evaluated top-N ranked recommendation lists with precision-based metrics (Cremonesi et al., 2010;

McLaughlin and Herlocker, 2004; Jambor and Wang, 2010b; Bellogín et al., 2011b), drawing from evaluation well studied methodologies in the Information Retrieval field.

Precision-oriented metrics measure the amount of relevant and non-relevant retrieved (recommended) items. A solid body of metrics, methodologies, and datasets has been developed over the years in the Information Retrieval field. Recommendation can be naturally stated as an information retrieval task: users have an implicit need with regards to a space of items which may serve the user's purpose, and the task of the recommender system is to select, rank and present the user a set of items that may best satisfy her need. The need of the user and the qualities or reasons why an item satisfies it cannot be observed in full, or described in an exact and complete way, which is the defining characteristic of an information retrieval problem, as opposed to data retrieval tasks or logical proof. It is thus natural to adapt relevance-based Information Retrieval evaluation methodologies here, which mainly consist of obtaining manual relevance labels of recommended items with respect to the user's need, and assessing, in different ways, the amount of relevant recommended items.

Recommendation tasks and the available data for their evaluation, nonetheless, have specific characteristics, which introduce particularities with respect to mainstream experience in the Information Retrieval field. In common information retrieval experimental practice, driven to a significant extent by the TREC campaigns (Voorhees and Harman, 2005), relevance knowledge is typically assumed to be (not far from) complete – mainly because in the presence of a search query, relevance is simplified to be a user-independent property. However, in recommender systems it is impractical to gather complete preference information for *each* user in a system. In datasets containing thousands of users and items, only a fraction of the items that users like is generally known. The unknown rest are, for evaluation purposes, assumed to be non-relevant. This is a source of – potentially strong – bias in the measurements depending on how unknown relevance is handled. In the next chapter we cover in detail these problems, along with an analysis of the different experimental design alternatives available in the literature.

In the reminder of this chapter we present some of the most common evaluation metrics. We classify them into error-based and precision-based metrics, accounting for the two tasks previously described – rating prediction and item ranking, respectively. After that, we describe the main methodologies used in the area to partition datasets and to select the candidate items in the latter task. Finally, we introduce the datasets used in this thesis to evaluate different recommendation algorithms.

3.2 Evaluation metrics

The evaluation of recommender systems should take into account the goal of the system itself (Herlocker et al., 2004). For example, in (Herlocker et al., 2004) the authors identify two main user tasks: *annotation in context* and *find good items*. In these tasks the users only care about errors in the item rank order provided by the system, not the predicted rating value itself. Based on this consideration, researchers have started to use precision-based metrics to evaluate recommendations, although most works also still report error-based metrics for comparison with state of the art approaches. Moreover, other authors, such as Herlocker and colleagues (Herlocker et al., 2004), encourage considering alternative performance criteria, like the novelty of the suggested items and the item coverage of a recommendation method. We describe the above types of evaluation metrics in the subsequent sections.

3.2.1 Error-based metrics

A classic assumption in the recommender systems literature is that a system that provides more accurate predictions will be preferred by the user (Shani and Gunawardana, 2011). Although this has been further studied and refuted by several authors (McNee et al., 2006; Cremonesi et al., 2011; Bollen et al., 2010), the issue is still worth being analysed.

Traditionally, the most popular metrics to measure the accuracy of a recommender system have been the **Mean Absolute Error** (MAE), and the **Root Mean Squared Error** (RMSE):

$$\mathrm{MAE} = \frac{1}{|\mathrm{Te}|} \sum_{(u,i)\in \mathrm{Te}} |\tilde{r}(u,i) - r(u,i)| \tag{3.1}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\mathrm{Te}|} \sum_{(u,i)\in \mathrm{Te}} \big(\tilde{r}(u,i) - r(u,i)\big)^2} \tag{3.2}$$

where $\tilde{r}$ and r denote the predicted and real rating, respectively, and Te corresponds to the test set. The RMSE metric is usually preferred to MAE because it penalises larger errors.

Different variations of these metrics have been proposed in the literature. Some authors **normalise MAE** and **RMSE** with respect to the maximum range of the ratings (Goldberg et al., 2001; Shani and Gunawardana, 2011) or with respect to the expected value if ratings are distributed uniformly (Marlin, 2003; Rennie and Srebro, 2005). Alternatively, **per-user** and **per-item average errors** have also been proposed in order to avoid biases from the error (or accuracy) on a few very frequent users or items (Massa and Avesani, 2007a; Shani and Gunawardana, 2011). For instance, the user-average MAE is computed as follows:

$$\text{uMAE} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{|\text{Te}_u|}\sum_{i\in\text{Te}_u}|\tilde{r}(u,i) - r(u,i)| \tag{3.3}$$

A critical limitation of these metrics is that they do not make any distinction between the errors made on the top items predicted by a system, and the errors made for the rest of the items. Furthermore, they can only be applied when the recommender predicts a score in the allowed range of rating values. Because of that, log-based, and some content-based and probabilistic recommenders cannot be evaluated in this way, since $\tilde{r}(u,i)$ would represent a probability or, in general, a preference score. Hence, these methods can only be evaluated by measuring the performance of the generated ranking using precision-based metrics.

3.2.2 Precision-based metrics

These metrics can be classified into three groups: metrics that only use one ranking, metrics that compare two rankings (typically, one of them is a reference or ideal ranking), and metrics from the Machine Learning field.

Metrics based on one ranking

Examples of these metrics are precision, recall, normalised discounted cumulative gain, mean average precision, and mean reciprocal rank. Each of these metrics captures the quality of a ranking from a slightly different angle. More specifically, **precision** accounts for the fraction of recommended items that are relevant, whereas **recall** is the fraction of the relevant items that has been recommended. Both metrics are inversely related, since an improvement in recall typically produces a decrease in precision. They are typically computed up to a ranking position or cutoff k, being denoted as $P@k$ and $R@k$, and defined as follows (Baeza-Yates and Ribeiro-Neto, 2011):

$$P@k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{|\text{Rel}_u@k|}{k} \tag{3.4}$$

$$R@k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{|\text{Rel}_u@k|}{|\text{Rel}_u|} \tag{3.5}$$

where Rel_u represents the set of relevant items for user u, and $\text{Rel}_u@k$ is the number of relevant recommended items up to position k.

Recall has also been referred to as **hit-rate** in (Deshpande and Karypis, 2004). Hit-rate has also been defined as the percentage of users with at least one correct recommendation (Bellogín et al., 2012), corresponding to the **success** metric (or **first relevant score**), as defined by TREC (Tomlinson, 2005).

Furthermore, the **mean average precision** (MAP) metric provides a single summary of the user's ranking by averaging the precision figures obtained after each new relevant item is obtained, as follows (Baeza-Yates and Ribeiro-Neto, 2011):

$$\text{MAP} = \frac{1}{|\mathcal{U}|}\sum_{u}\frac{1}{|\text{Rel}_u|}\sum_{i\in\text{Rel}_u}\text{P@rank}(u,i) \tag{3.6}$$

where $\text{rank}(u,i)$ outputs the ranking position of item i in the user's u list; hence, precision is computed at the position where each relevant item has been recommended.

Normalised discounted cumulative gain (nDCG) uses graded relevance that is accumulated starting at the top of the ranking and may be reduced, or discounted, at lower ranks (Järvelin and Kekäläinen, 2002):

$$\text{nDCG} = \frac{1}{|\mathcal{U}|}\sum_{u}\frac{1}{\text{IDCG}_u^{p_u}}\sum_{p=1}^{p_u}\text{f}_{\text{dis}}\big(\text{rel}\big(u,i_p\big),p\big) \tag{3.7}$$

where the discount function $f_{\text{dis}}\big(rel\big(u,i_p\big),p\big)$ is usually defined as $f_{\text{dis}}(x,y) = (2^x-1)/\log(1+y)$ or simply $f_{\text{dis}}(x,y) = x/\log y$ if $y>1$, $f_{\text{dis}}(x,y)=x$ otherwise, depending on the emphasis required on retrieving highly relevant items (Croft et al., 2009). IDCG_u^k denotes the score obtained by an ideal or perfect ranking for user u up to position k, which acts as a normalisation factor in order to compare different users and datasets. Besides, p_u denotes the maximum number of items evaluated for each user; which is typically assumed to be a cutoff k, the same for all the users. In that situation, this metric is denoted as $\text{nDCG@}k$.

Using a different discount function, the **rank score** or **half-life utility** metric (Breese et al., 1998; Herlocker et al., 2004; Huang et al., 2006) can be obtained as follows:

$$\text{HL} = 100\left(\sum_{u}\text{HL}_u^{\max}\right)^{-1}\sum_{u}\text{HL}_u\,;\quad \text{HL}_u = \sum_{p=1}^{p_u}\frac{\max\big(\tilde{r}\big(u,i_p\big)-d,0\big)}{2^{(p-1)/(\alpha-1)}} \tag{3.8}$$

where d is the default ranking, and α is the half-life utility that represents the rank of the item on the list such that there is a 50% chance that the user will view that item. In (Breese et al., 1998) the authors use a value of 5 in their experiments, and note that they did not obtain different results with a half-life of 10.

Mean reciprocal rank (MRR) favours rankings whose first correct result occurs near the top ranking results (Baeza-Yates and Ribeiro-Neto, 2011). It is defined as follows:

$$\text{MRR} = \sum_{u}\frac{1}{s_r(u)} \tag{3.9}$$

where $s_r(u)$ is a function that returns the position of the first relevant item obtained for user u. This metric is similar to the **average rank of correct recommendation** (ARC) proposed in (Burke, 2004) and to the **average reciprocal hit-rank** (ARHR) defined in (Deshpande and Karypis, 2004).

It is important to note that since its early days, there has been a concern in the Information Retrieval field for the value and validity of the standard precision and recall metrics in interactive contexts (Su, 1992; Belkin and Croft, 1992). Nonetheless, precision-based metrics such as precision and recall, and more in general, metrics that measure the quality of the item ranking returned by a recommender have been frequently used in the field, despite they often lead to uncomparable results (Bellogín et al., 2011a).

Metrics based on two rankings

Additionally, specific metrics have been defined in the context of recommender evaluation that take as inputs two rankings (ideal vs estimated) instead of just one. A first example is the **normalised distance-based performance measure** (NDPM), used in (Balabanovic and Shoham, 1997), and proposed in (Yao, 1995). This metric compares two different weakly ordered rankings, and is formulated as follows (Herlocker et al., 2004; Shani and Gunawardana, 2011):

$$\text{NDPM} = \frac{1}{|\mathcal{U}|}\sum_{u}\frac{2C_u^{\text{con}} + C_u^{\text{tie}}}{2C_u} \tag{3.10}$$

where C_u is the number of pairs of items for which the real ranking (reference ranking using the ground truth) asserts an ordering, i.e., the items are not tied. Besides, C_u^{con} denotes the number of discordant item pairs between the method's ranking and the reference ranking, and C_u^{tie} represents the number of pairs where the reference ranking does not tie, but where the method's ranking does. This metric is comparable across datasets since it is normalised with respect to the worst possible scenario (denominator). Furthermore, it provides a perfect score of 0 to systems that correctly predict every preference relation asserted by the reference, and a worst score of 1 to methods that contradict every reference preference relation. Besides, a penalisation of 0.5 is applied when a reference preference relation is not predicted, whereas predicting unknown preferences (i.e., they are not ordered in the reference ranking) receives no penalisation.

As the previous metric, rank correlation metrics such as **Spearman**'s ρ and **Kendall**'s τ have also been proposed to directly compare the system ranking to a preference order given by the user. These correlation coefficients are later defined and analysed (Chapter 5). Here we only indicate that they provide scores in the range

of -1 to 1, where 1 denotes a perfect correlation between the two above rankings, and -1 represents an inverse correlation.

These two metrics, along with NDPM, suffer from the interchange weakness (Herlocker et al., 2004), that is, interchanges at the top of the ranking have the same weight that interchanges at the bottom of the ranking.

Metrics from Machine Learning

Finally, some other metrics from the Machine Learning literature have also been used, although they are not very popular. For instance, the **receiving operating characteristic** (ROC) curve and the **area under the curve** (AUC) have been used in (Herlocker et al., 1999), (Schein et al., 2001), (Schein et al., 2002), and (Rojsattarat and Soonthornphisaj, 2003), among others. Metrics based on the ROC curve provide a theoretically grounded alternative to precision and recall (Herlocker et al., 2004). The ROC model attempts to measure the extent to which an information filtering system can successfully distinguish between signal (relevant items) and noise. Starting from the origin of coordinates at (0,0), the ROC curve is built by considering, at each rank position, whether the item is relevant or not for the user; in the first case, the curve goes one step up, and in the second, one step right.

A random recommender is expected to produce a straight line from the origin to the upper right corner; on the other hand, the more leftwards the curve leans, the better is the performance of the system. These facts are related to the area under the ROC curve, a summary metric that is expected to be higher when the recommender performs better, where the expected value of a random recommender is 0.5, corresponding to a diagonal curve in the unit square.

In (Schein et al., 2001) the authors discriminate between the Global ROC (GROC) curve and the Customer ROC (CROC) curve, where the former assumes that only the most certain recommendations are made where some users may receive no recommendation at all; thus, the number of recommendations could be different for each user. The CROC curve is more realistic in the sense that every user receives the same amount of recommended items. However, for this curve a perfect recommender would not necessarily obtain an AUC of 1, and thus, it is required to compute the associated value of a perfect ROC curve in order to provide a fair comparison and normalise accordingly.

3.2.3 Other metrics

As different applications have different needs, additional characteristics of recommendations could be taken into consideration, and thus alternative metrics beyond accuracy and precision may be measured. In this context, it is important to understand and evaluate the possible trade-offs between these additional characteristics

and their effect on the overall recommendation performance (Shani and Gunawardana, 2011). For instance, some algorithms may provide recommendations with high quality or accuracy, but only for a small proportion of users or items, probably due to data sparsity. This effect can be quantified by measuring the **coverage** of the recommender system. Two types of coverage can be defined: *user coverage* (proportion of users to whom the system can recommend items) and *item or catalog coverage* (proportion of items the system can recommend). In (Shani and Gunawardana, 2011) two metrics are proposed for measuring item coverage: one based on the Gini's index, and another based on Shannon's entropy. In (Ge et al., 2010) the authors propose simple ratio quantities to measure such metrics, and to discriminate between the percentage of the items for which the system is able to generate a recommendation (*prediction coverage*), and the percentage of the available items that are effectively ever recommended (*catalog coverage*). A similar distinction is considered in (Herlocker et al., 2004) and (Salter and Antonopoulos, 2006). In (Herlocker et al., 2004) it is acknowledged that item coverage is particularly important for the tasks of *find all good items* and *annotation in context*. Besides, a system with low coverage is expected to be less valuable to users and the authors propose to combine coverage with accuracy measures to yield an overall "practical accuracy" measure for the system, in such a way that coverage is raised only because recommenders produce bogus predictions.

Beyond coverage, two recommendation characteristics have become very popular recently: **novelty** and **diversity**. Already a large amount of work has focused on defining metrics for measuring such characteristics (Lathia et al., 2010; Shani and Gunawardana, 2011; Vargas and Castells, 2011; Zhang and Hurley, 2009), and designing algorithms to provide novel and/or diverse recommendations (Jambor and Wang, 2010b; Onuma et al., 2009; Weng et al., 2007; Zhou et al., 2010).

Novel recommendations are those that suggest the user items she did not know about prior to the recommendation (Shani and Gunawardana, 2011), referred to as non-obvious items in (Herlocker et al., 2004; Zhang et al., 2002). Novelty can be directly measured in online experiments by directly asking users whether they are familiar with the recommended item (Celma and Herrera, 2008). However, it is also interesting to measure novelty in an offline experiment, so as not to restrict its evaluation to costly and hardly reproducible online experiments.

Novelty can be introduced into recommendations by using a topic taxonomy (Weng et al., 2007), where items containing novel topics are appreciated. Typically, novel topics are obtained by clustering the previously observed topics for each user. In (Lathia et al., 2010), novelty measures the amount of new items appearing in the recommended lists over time. In (Onuma et al., 2009) a technique based on graphs is introduced to suggest nodes (items) well connected to older choices, but at the same time well connected to unrelated choices.

Metrics based on Information Theoretic properties of the items being recommended have also been proposed by several authors. In (Bellogín et al., 2010) the entropy function is used to capture the novelty of a recommendation list, in (Zhou et al., 2010) the authors use the self-information of the user's top recommended items, and in (Filippone and Sanguinetti, 2010) the Kullback-Leibler divergence is used.

In Information Retrieval, diversity is seen as an issue of avoiding redundancy and finding results that cover different aspects of an information need (Radlinski et al., 2009). In that context, most of the proposed methods and metrics make use of (explicit or inferred) query aspects (topics or interpretations) to rank higher the most likely results (Demidova et al., 2010), or diversify a prior result set (Clarke et al., 2008; Agrawal et al., 2009; Chandar and Carterette, 2010; Radlinski et al., 2008; Rafiei et al., 2010).

In recommender systems diversity has been typically defined in an ad-hoc way, often mixing concepts such as diversity, novelty and coverage. For example, in (Salter and Antonopoulos, 2006) the authors make use of the catalog coverage defined above as a measure of recommendation diversity. A similar assumption is done in (Kwon, 2008). In (Zhou et al., 2010) the authors show that by tuning appropriately a hybrid recommender it is possible to obtain simultaneous gains in both accuracy and diversity, which is measured as the inter-list distance between every pair of users in the collection. Zhang and Hurley (2008) measure the novelty of an item by the amount of diversity it brings to the recommendation list, which is computed using a distance or dissimilarity function.

More formal definitions for diversity have also been proposed. In (Lathia et al., 2010) the authors propose to analyse diversity of top-N lists over time by comparing the intersection of sequential top-N lists. A statistical measure of diversity is proposed in (Zhang and Hurley, 2009), where the authors consider a recommendation algorithm to be fully diverse if it is equally likely to recommend any item that the user likes. In (Jambor and Wang, 2010b), the introduction of the covariance matrix into the optimisation problem leads to promote items in the long tail. A similar result is obtained in (Celma and Herrera, 2008), where the items with fewer interactions within the community of users (long tail) are assumed to be more likely to be unknown. Based on item similarities and focused on content-based algorithms, the authors in (Bradley and Smyth, 2001) propose a quality metric which considers both the diversity and similarity obtained in the recommendation list. A definition based on the entropy of the probability distributions of each recommender with respect to the items is proposed in (Bellogín et al., 2010), and the Gini's index is used in (Fleder and Hosanagar, 2009).

Finally, in (Vargas and Castells, 2011) a formal framework for the definition of novelty and diversity metrics is presented, where several previous metrics are unified

by identifying three ground concepts at the roots of novelty and diversity: choice, discovery, and relevance.

Other metrics such as serendipity, privacy, adaptivity, confidence, and scalability have been less explored in the literature, but their importance and application to recommender systems have already been discussed, making clear their relation with the user's experience and satisfaction, which is the ultimate goal of a "good" recommender system (Herlocker et al., 2004; McNee et al., 2006; Shani and Gunawardana, 2011).

3.3 Experimental setup

An important decision in the experimental configuration of a recommender evaluation is the dataset partition strategy. How the datasets are partitioned into training and test sets may have a considerable impact on the final performance results, and may cause some recommenders to obtain better or worse results depending on how this partition is configured. Although an exhaustive analysis of the different possibilities to choose the ratings/items to be hidden is out of the scope of this thesis, we briefly discuss now some of the most well-known methods used.

First, we have to choose whether or not to take time into account (Gunawardana and Shani, 2009). Time-based approaches naturally require the availability of user interaction data timestamps. A simple approach is to select a time point in the available interaction data timeline to separate training data (all interaction records prior to that point) and test data (dated after the split time point). The split point can be set so as to, for instance, have a desired training/test ratio in the experiment. The ratio can be global, with a single common split point for all users, or user-specific, to ensure the same ratio per user. Time-based approaches have the advantage of more realistically matching working application scenarios, where "future" user likes (which would translate to positive response to recommendations by the system) are to be predicted based on past evidence. As an example, the well-known Netflix prize provided a dataset where the test set for each user consisted on her most recent ratings (Bennett and Lanning, 2007).

If we ignore time, there are at least the following three strategies to select the items to hide from each user: a) sample a fixed number (different) for each user; b) sample a fixed (but the same for all) number for each user, also known as *given n* or *all but n* protocols; c) sample a percentage of all the interactions using *cross-validation*. The most usual protocol is the last one (Goldberg et al., 2001; Sarwar et al., 2001), although several authors have also used the *all but n* protocol (Breese et al., 1998; Wang et al., 2008a). The MovieLens datasets provide random splits following a five-fold cross validation strategy, as we shall see in the next section.

Nonetheless, independently from the dataset partition, it is recognised that the goals for which an evaluation is performed may be different in each situation, and thus, a different setting (and partition protocol) should be developed (Herlocker et al., 2004; Gunawardana and Shani, 2009). If that is not the case, the results obtained in a particular setting would be biased and difficult to use in further experiments, for instance, in an online experimentation.

Furthermore, as mentioned earlier, in order to evaluate ranked recommendations for a target user u, it is required to select two sets of items, namely relevant and not relevant. In the next chapter, we describe different possibilities explored in the literature, along with a detailed analysis of these alternatives and the possible biases that may appear.

3.4 Evaluation datasets

In this section we present three datasets that were used in the experimental parts of this thesis. The datasets correspond to different domains: movie recommendation, in which user preferences are provided in the form of ratings, and music recommendation, where user preferences are derived from implicit (log-based) evidence. Furthermore, one of the datasets includes social information that can be exploited by social filtering algorithms.

3.4.1 MovieLens dataset

The GroupLens research lab[1] has released different datasets obtained from user interaction in the MovieLens recommender system. At the time of writing, there are three publicly available MovieLens datasets of different sizes:

- The 100K dataset, containing 100,000 ratings for 1,682 movies by 963 users.
- The 1M dataset, with one million ratings has 6,040 users and 3,900 movies.
- The 10M dataset, with 10 million ratings consists of almost 71,600 users and 10,700 movies, and 100,000 tag assignments.

Although there are larger public datasets (such as the one provided for the well-known competition organised by Netflix[2] between 2006 and 2009), the first two MovieLens datasets are currently, by far, the most used in the field.

The ratings range on a 5-star scale in all three datasets; the 100K and 1M versions only use "integer" stars, and 10M uses "half star" precision (ten discrete rating values). Every user has at least 20 ratings in any of the datasets.

[1] GroupLens research lab, http://www.grouplens.org

[2] Netflix site, http://www.netflix, and Netflix Prize webpage, http://www.netflixprize.com

3.4.2 Last.fm dataset

Last.fm is a social music website. At the moment, the site has more than 40 million users (claimed 30 million active in 2009[3]) in more than 190 countries. Several authors have analysed and used this system for research purposes; special mention deserves those who have made their datasets public, such as (Konstas et al., 2009), (Celma, 2010), and (Cantador et al., 2011).

In 2010, Òscar Celma released two datasets collected using the Last.fm API. The first one (usually referred to as 360K) contains the number of plays (called *scrobblings* in the platform) of almost 360,000 users, counted on music artists, amounting to more than 17 million of (user, artist, playcounts) tuples. The second dataset (named 1K) contains fewer users (nearly 1,000) but, in contrast to the previous one, the whole listening history of each user is collected as tuples (user, timestamp, artist, music track) for up to 19 million tuples.

3.4.3 CAMRa dataset

In 2010 the 1st Challenge on Context-aware Movie Recommendation (CAMRa 2010[4]) was held at the 4th ACM conference on Recommender Systems (RecSys 2010). The challenge organisers released four datasets that were used in three different challenge tracks (Adomavicius et al., 2010). These tracks were focused on temporal recommendation (*weekly recommendation*), recommendation based on mood (*Moviepilot track*), and social recommendation (*Filmtipset track*). Two different datasets were provided for the first track, whereas the second and third tracks were assigned a different dataset each (Said et al., 2010).

These datasets were gathered from the Filmtipset[5] and Moviepilot[6] communities, and, depending on the track, contained social links between users, movie ratings, movie reviews, review ratings, comments about actors and movies, movie directors and writers, lists of favourite movies, moods, and links between similar movies. Filmtipset is the largest online social community in the movie domain in Sweden, with more than 90,000 registered users and 20 million ratings in its database. Moviepilot, on the other hand, is the leading online movie and TV recommendation community in Germany; it has over 100,000 registered users and a database of over 40,000 movies with roughly 7.5 million ratings (Said et al., 2010).

Further editions of this challenge have also released datasets related to recommendation tasks (focused on group recommendation in 2011 (Said et al., 2011) and

[3] Announcement, http://blog.last.fm/2009/03/24/lastfm-radio-announcement

[4] CAMRa site, http://2010.camrachallenge.com/

[5] Filmtipset site, http://www.filmtipset.se

[6] Moviepilot site, http://www.moviepilot.de

on additional context information in 2012). However, they were not used in this thesis, and thus, they are not described in detail here.

3.5 Summary

In the Recommender Systems literature several evaluation metrics, protocols, and methodologies have been defined. It remains unclear the equivalence between them and the extent to which they would provide comparable results.

The problem of evaluating recommender systems has been a major object of study and methodological research in the field since its earliest days. Error-based metrics have widely dominated the field, and precision-based metrics have started to be adopted more recently. Other metrics from the Machine Learning field have been proposed but they are not widely used in the community yet. Moreover, metrics for additional dimensions such as novelty or diversity have also started to be researched in the last few years.

There are, still, important characteristics of the evaluation methodologies and metrics that remain unexplored. In contrast to the Information Retrieval community, where statistical analysis and eventual biases in the evaluation as a whole have been studied (Buckley et al., 2006; Aslam et al., 2006; Soboroff, 2004), there is a lack of such an analysis for recommender systems. This raises a key issue for our research which shall be analysed in depth in the next chapter, where we propose some alternative methodologies to overcome some of the possible biases that may arise.

Chapter 4

Ranking-based evaluation of recommender systems: experimental designs and biases

There is an increasing consensus in the Recommender Systems community that the dominant error-based evaluation metrics are insufficient, and to some extent inadequate, to properly assess the practical effectiveness of recommendations. Seeking to evaluate recommendation rankings – which largely determine the effective accuracy in matching user needs – rather than predicted rating values, Information Retrieval metrics have started to be applied to evaluate recommender systems.

In this chapter we analyse the main issues and potential divergences in the application of Information Retrieval methodologies on recommender system evaluation, and provide a systematic characterisation of experimental design alternatives for this adaptation. We lay out an experimental configuration framework upon which we identify and analyse specific statistical biases arising in the adaptation of Information Retrieval metrics to recommendation tasks, which considerably distort the empirical measurements, hindering the interpretation and comparison of results across experiments. We propose two experimental design approaches that effectively neutralise such biases to a large extent. We support our findings and proposals through both analytical and empirical evidence.

We start the chapter by introducing the problem of (un)biased evaluation in recommender systems. The reminder of the chapter follows by revisiting the principles and assumptions underlying the Information Retrieval evaluation methodology: the Cranfield paradigm (Section 4.2). After that, in Section 4.3 we elaborate a formal synthesis of the main approaches to the application of Information Retrieval metrics to recommendation. In Sections 4.4 and 4.5 we analyse, respectively, the sparsity and popularity biases of Information Retrieval metrics on recommendation tasks. We present and evaluate two approaches to avoid these biases in Section 4.6,and end with some conclusions in Section 4.7.

4.1 Introduction

There seems to be a raising awareness in the Recommender Systems (RS) community that important – or even central – open questions remain to be addressed concerning the evaluation of recommender systems. As we mentioned in the previous chapter, the error in predicting held-out user ratings has been by far the dominant offline evaluation methodology in the RS literature (Breese et al., 1998; Herlocker et al., 2004). The limitations of this approach are increasingly evident, and have been extensively pointed out (Cremonesi et al., 2010). The prediction error has been found to be far from enough or even adequate to assess the practical effectiveness of a recommender system in matching user needs. The end users of recommendations receive lists of items rather than rating values, whereby recommendation accuracy metrics – as surrogates of the evaluated task – should target the quality of the item selection and ranking, rather than the numeric system scores that determine this selection.

For this reason, researchers are turning towards metrics and methodologies from the Information Retrieval (IR) field (Barbieri et al., 2011; Cremonesi et al., 2010; Herlocker et al., 2004), where ranking evaluation has been studied and standardised for decades. Yet, gaps remain between the methodological formalisation of tasks in both fields, which result in divergences in the adoption of IR methodologies, hindering the interpretation and comparability of empirical observations by different authors. The use of IR evaluation techniques involves the adoption of the Cranfield paradigm (Voorhees and Harman, 2005), and common metrics such as precision, mean average precision (MAP), and normalised Discounted Cumulative Gain (nDCG) (Baeza-Yates and Ribeiro-Neto, 2011). Given the natural fit of top-n recommendation in an IR task scheme, the adoption of IR methodologies would seem straightforward. However, recommendation tasks, settings, and available datasets for offline evaluation involve subtle differences with respect to the common IR settings and experimental assumptions, which result in substantial biases to the effectiveness measurements that may distort the empiric observations and hinder comparison across systems and experiments.

Furthermore, how to measure the performance of a recommender system is a key issue in our research. The variability in the experimental configurations, and the observed statistical biases of the evaluation methodologies should be well understood, since we aim to predict the performance of a system. We should avoid the situation where a metric shows some source of noise together with the recommender's quality, since then a predictor capturing only that noise would appear as an (equivocal) effective performance predictor.

Taking up from prior studies on the matter (Cremonesi et al., 2010; Herlocker et al., 2004; Shani and Gunawardana, 2011; Steck, 2011), we revisit the methodological assumptions underlying IR metrics, and analyse the differences between Recom-

mender Systems and Information Retrieval evaluation settings and their implications. Upon this, we identify two sources of bias in IR metrics on recommender systems: data sparsity and item popularity. We characterise and study the effect of these two factors both analytically and empirically. We show that the value range of common IR metrics is determined by the density of the available user preference information, to such an extent that the measured values per se are not meaningful, except for the purpose of comparison within a specific experiment. Furthermore, we show that the distribution of ratings among items has a drastic effect on how different algorithms compare to each other. Finally, we propose and analyse two approaches to mitigate popularity biases on the measured ranking quality, providing theoretical and empirical evidence of their effectiveness.

4.2 Cranfield paradigm for recommendation

Information Retrieval evaluation methodologies have been designed, studied, and refined over the years under the so-called *Cranfield paradigm* (van Rijsbergen, 1989; Voorhees, 2002b). In the Cranfield paradigm, as e.g. typically applied in the TREC campaigns (Voorhees and Harman, 2005), information retrieval systems are evaluated on a dataset comprising a set of documents, a set of queries – referred to as *topics* and consisting of a description or representation of user information needs –, and a set of relevance judgments by human assessors – referred to as *ground truth*. The assessors manually inspect queries and documents, and decide whether each document is relevant or not for a query. Theoretically, each query-document pair should be assessed for relevance, which, for thousands or millions of documents, is obviously unfeasible. Therefore, a so-called *pooling* approximation is applied, in which the assessors actually inspect and judge just a subset of the document collection, consisting of the union of the top-n documents returned by a set of systems for each query. These systems for pooling are commonly the ones to be evaluated and compared, and n is called the *pooling depth*, typically ranging around 100 documents. While this procedure obviously misses some relevant documents, it has been observed that the degree of incompleteness is reasonably small, and the missing relevance does not alter the empiric observations significantly, at least up to some ratio between the pooling depth and the collection size (Buckley et al., 2007).

Whereas in a search system users may enter multiple queries, the recommendation task – in its classic formulation – typically considers a single "user need" per user, that is, a user has a set of cohesive preferences which defines her main interests. In this view a natural fit of recommendation in the Cranfield paradigm would take users – as an abstract construct – as the equivalent of queries in ad-hoc retrieval (the user need to be satisfied), and items as equivalent to documents (the objects to be retrieved and ranked), summarised in Table 4.1. A first obvious difference is that

Task element	TREC ad-hoc retrieval task	Recommendation task
Information need expression	Topic (query and description)	User profile
Candidate answers	All documents in the collection	Target item set
	Same for all queries	One or more per user, commonly different
Document data available as system input	Document content	Training ratings, item features
Relevance	Topical, objective	Personalised, subjective
Ground truth	Relevance judgments	Test ratings
Relevance assessment	Editorial assessors	End users
Relevance knowledge coverage	Reasonably complete (pooling)	Highly incomplete (inherently to task)

Table 4.1. Fitting the recommendation task in the Cranfield IR evaluation paradigm

queries are explicit representations of specific information needs, whereas in the recommendation setting, user profile records are a global and implicit representation of what the user may need or like. Still, the query-user mapping is valid, inasmuch as user profiles may rightfully fit in the IR scheme as "vague queries."

The definition of ground truth is less straightforward. User ratings for items, as available in common recommendation datasets, are indeed relevance judgments of items for user needs. However, many recommendation algorithms (chiefly, collaborative filtering methods) require these "relevance judgments" as input to compute recommendations. The rating data withholding evaluation approach, pervasive in RS research, naturally fits here: some test ratings can be held out as ground truth and the rest be left as training input for the systems. Differently from TREC, here the "queries" and the relevance assessments are both entered by the same people: the end-users. Furthermore, how much data are taken for training and for ground truth is left open to the experiment designers, thus adding a free variable to be watched over as it significantly impacts the measurements.

On the other hand, whereas in the IR setting all the documents in the collection are candidate answers for all queries, the set of target items on which recommender systems are tested for each user need not be necessarily the same. As already described in the previous chapter, in general, the items with a test rating are included in the candidate set for the raters, though not necessarily in a single run (Cremonesi et al., 2010). Moreover, it is common to select further non-rated target items, but not necessarily all the items (Bellogín et al., 2011a). Furthermore, the items rated by a user in the training set are generally excluded from the recommendation to this user. The way these options are configured has a drastic effect on the resulting measurements, with variations in orders of magnitude (Bellogín et al., 2011a).

In addition to this, the coverage of user ratings is inherently much smaller in recommender systems' datasets compared to TREC collections. The amount of un-

known relevance – which in TREC is assumed to be negligible – is pervasive in recommendation settings (it is in fact intrinsic for the task to make sense), to a point where some assumptions of the IR methodology may not hold, and the gap between measured and real metric values becomes so significant that a metric's absolute magnitude may just lose any meaning. Still, such measurements may support comparative assessments between systems, as far as the bias is system-independent.

Finally, the distribution of relevance in the retrieval space displays popularity patterns that are absent in IR datasets. The number of users who like each item is very variable (typically long-tailed) in recommendation datasets, whereas in TREC collections very few documents are relevant for more than one query. We shall show that this phenomenon has a very strong effect not only on metric values, but more importantly on how systems compare to each other.

In order to provide a formal basis for our study we start by elaborating a systematic characterisation of design alternatives for the adaptation of IR metrics to recommender systems, taking into account prior approaches described in the literature, such as those presented in the previous chapter. This formal framework will help us to analyse and describe the measurement biases in the application of IR metrics to recommender systems, and study new approaches to mitigate them.

4.3 Experimental design alternatives

The application of Information Retrieval metrics to recommender systems evaluation has been studied by several authors in the field (Barbieri et al., 2011; Breese et al., 1998; Cremonesi et al., 2010; Herlocker et al., 2004; Shani and Gunawardana, 2011). We elaborate here an experimental design framework that aims to synthesise commonalities and differences between studies, encompassing prior approaches and supporting new variants upon a common methodological grounding. We formalise the different methodologies presented in the previous chapter, and provide an equivalence between both formulations.

In the following, given a rating set split into training and test rating sets, we say an item $i \in \mathcal{I}$ is relevant for a user $u \in \mathcal{U}$ if u rated i positively, and its corresponding rating falls in the test set. By positive rating we mean a value above some design-dependent threshold. All other items (non-positively rated or non-rated) are considered as non-relevant. Like in the previous chapter, recommender systems are requested to rank a set of target items T_u for each user. Such sets do not need to be the same for each user, and can be formed in different ways. In all configurations T_u contains a combination of relevant and non-relevant items, and the different approaches are characterised by how these are selected, as we describe next.

Design settings		Alternatives
Base candidate items		**AI** $\mathcal{C} = \mathcal{I}$
		TI $\mathcal{C} = \bigcup_{u \in \mathcal{U}} \mathrm{R}_{\mathrm{test}}(u)$
Item selection	Relevant	**AR** $\mathrm{T}_u \supset \mathrm{PR}_{\mathrm{test}}(u)$
		1R $\lvert \mathrm{T}_u^{\mathrm{r}} \cap \mathrm{PR}_{\mathrm{test}}(u) \rvert = 1$
	Non-relevant	**AN** $\mathrm{N}_u = \mathcal{C} - \mathrm{PR}_{\mathrm{test}}(u) - \mathrm{R}_{\mathrm{train}}(u)$
		NN Fixed $\lvert \mathrm{N}_u \rvert$, random sampling

Table 4.2. Design alternatives in target item set formation.

4.3.1 Target Item Sampling

We identify three significant design axes in the formation of the target item sets: candidate item selection, relevant item selection, and irrelevant item sampling. We consider two relevant alternatives for each of these axes, summarised in Table 4.2, which we describe next.

We shall use $\mathrm{R}(u)$ and $\mathrm{PR}(u)$ to denote the set of all and positively rated items by user u, respectively, and $\mathrm{r}(u)$, $\mathrm{pr}(u)$ to denote the respective size of those sets. With the subscripts "test" and "train" we shall denote the part of such sets (or their sizes) contained on the corresponding side of a data split. An equivalent notation $\mathrm{r}(i)$, $pr(i)$, and so on, will be used for the ratings of an item, and when no user or item is indicated, the total number of ratings is denoted. This notation and the rest to be used along the chapter are summarised in Table 4.3.

Let $\mathrm{N}_u = \mathrm{T}_u - \mathrm{PR}_{\mathrm{test}}(u)$ be the non-relevant target items for u. As a general rule, we assume non-relevant items are randomly sampled from a subset of candidate items $\mathcal{C} \subset \mathcal{I}$, the choice of which is a design option. We mainly find two significant alternatives for this choice: $\mathcal{C} = \mathcal{I}$ (e.g. (Shani and Gunawardana, 2011)) and $\mathcal{C} = \bigcup_{u \in \mathcal{U}} \mathrm{R}_{\mathrm{test}}(u)$ (e.g. (Bellogín et al., 2011a; Vargas and Castells, 2011)). The first one, which we denote as **AI** for "all items", matches the typical IR evaluation setting, where the evaluated systems take the whole collection as the candidate answers. The second, to which we shall refer as **TI** ("test items") is an advisable condition to avoid certain biases in the evaluation of RS, as we shall see.

Once $\mathcal{C}$ is set, for each user we select a set $\mathrm{N}_u \subset \mathcal{C} - \mathrm{PR}_{\mathrm{test}}(u) - \mathrm{R}_{\mathrm{train}}(u)$. N_u can be sampled randomly for a fixed size $\lvert \mathrm{N}_u \rvert$ (we call this option **NN** for "N non-relevant"), or all candidate items can be included in the target set, $\mathrm{N}_u = \mathcal{C} - \mathrm{PR}_{\mathrm{test}}(u) - \mathrm{R}_{\mathrm{train}}(u)$ (we refer to this as **AN** for "all non-relevant"). Some authors have even used $\mathrm{T}_u = \mathrm{R}_{\mathrm{test}}(u)$ (Basu et al., 1998; Jambor and Wang, 2010a; Jambor and Wang, 2010b), but we discard this option as it results in a highly overestimated precision (Bellogín et al., 2011a). The size of N_u is thus a configuration parameter of

the experimental design. For instance, in (Cremonesi et al., 2010) the authors propose $|\mathrm{N}_u| = 1{,}000$, whereas in (Bellogín et al., 2011a) the authors consider $\mathrm{N}_u = \bigcup_{v \in \mathcal{U}} \mathrm{R}_{\mathrm{test}}(v) - \mathrm{R}_{\mathrm{train}}(u) - \mathrm{PR}_{\mathrm{test}}(u)$, among other alternatives. To the best of our knowledge, the criteria for setting this parameter have not been analysed in detail in the literature, leaving it to common sense and/or trial and error. It is worth noting nonetheless that in general $|\mathrm{N}_u|$ determines the number of calls to the recommendation algorithms, whereby this parameter provides a handle for adjustment of the cost of the experiments.

Regarding the relevant item selection, two main options are reported in the literature, to which we shall refer as **AR** for "all relevant", and **1R** for "one relevant." In the AR approach all relevant items are included in the target set, i.e., $T_u \supset PR_{test}(u)$ (Bellogín et al., 2011a). In the 1R approach, for user u, several target item sets T_u^r are formed, each including a single relevant item (Cremonesi et al., 2010). This approach may be more sensitive to the lack of recommendation coverage, as we shall observe later on. The choice between an AR or a 1R design involves a difference in the way the ranking quality metrics are computed, as we shall discuss in the next section.

Symbol			Meaning
$\mathcal{U}$	$\mathcal{I}$	$\mathcal{C}$	Set of all users | all items | candidate items
r	r_{test}	r_{train}	Nr. of all | test | training ratings
pr	pr_{test}	pr_{train}	Nr. of all | test | training positive ratings
$R(u)$	$R_{test}(u)$	$R_{train}(u)$	Set of all | test | training items rated by u
$PR(u)$	$PR_{test}(u)$	$PR_{train}(u)$	Set of all | test | training items liked by u
$r(u)$	$pr(u)$	…	Nr. of items rated | liked | … by u
$r(i)$	$pr(i)$	…	Nr. of users who rated | like | … item i
T_u	T_u^r		Set of target items for u in AR | 1R
N_u	N_u^r		Non-relevant items added to build T_u | $\mathrm{T}_u^{\mathrm{r}}$
$\mathrm{P}_s^u@n(T_u)$			P@n of item set T_u as ranked by s for u
$top_s^u(T_u, n)$			Top n items in T_u as ranked by s for u
$\tau_s^u(i, S)$			Position of i in S ∋ i as ranked by s for u
$i_k^{u,s}$	$i_k^{u,r,s}$		The item ranked k-th in T_u | $\mathrm{T}_u^{\mathrm{r}}$ by s for u
σ			Split ratio: $\mathrm{r}_{\mathrm{test}}/\mathrm{r}$
ρ_u	ρ		$\rho_u = \lvert \mathrm{T}_u \cap \mathrm{PR}_{\mathrm{test}}(u) \rvert / \lvert \mathrm{T}_u \rvert$, $\rho = \mathrm{avg}_u\, \rho_u$
t			"Average" target set size: $1/\mathrm{avg}_{\mathrm{u}}(1/\lvert \mathrm{T}_u \rvert)$
δ			Relevance density in target sets: $\mathrm{pr}/(\mathrm{t}\lvert\mathcal{U}\rvert)$

Table 4.3. Notation summary.

4.3.2 AR vs. 1R Precision

Essentially, the way metrics are defined in AR and 1R differs in how they are averaged. In AR the metrics are computed on each target set T_u in the standard way as in IR, and then averaged over users (as if they were queries). As a representative and simple to analyse metric, we shall use $\mathrm{P}@n$ henceforth, but similar properties to all the ones discussed here are observed for other metrics such as MAP and nDCG. The mean AR precision of a recommender system s can be expressed as:

$$P_s@n = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{n}|top_s^u(T_u,n)\cap PR_{test}(u)|$$

where $top_s^u(T_u,n)$ denotes the top n items in T_u ranked by s for u.

In the 1R design, drawing from (Cremonesi et al., 2010), we compute and average the metrics over the T_u^r sets, as follows:

$$1RP_s@n = precision(n) = \frac{1}{pr_{test}}\sum_{u\in\mathcal{U}}\sum_{r=1}^{pr_{test}(u)} P_s^u@n(T_u^r) \qquad (4.1)$$

where $\mathrm{P}_s^u@n(T_u^r)$ is the standard precision of T_u^r for u. This form to express the metric is equivalent to the original formulation in (Cremonesi et al., 2010), but lets a straightforward generalisation to any other IR metric such as MAP and nDCG, by just using them in place of $\mathrm{P}_s^u@n$ in Equation (4.1). We shall intentionally use the same symbol P to refer both to 1R and AR precisions when there is no ambiguity. Whenever there may be confusion, or we wish to stress the distinction, we shall use 1RP to explicitly denote 1R precision.

AR precision basically corresponds to the standard precision as defined in IR, whereas 1R precision, while following essentially the same principle, departs from it in the formation of runs, and the way to average values. Additionally, note that the maximum value of 1RP@n is 1/n as we shall see in the next section, mainly since each run has only one relevant item. Besides, in Section 4.4 we shall establish a formal relation between both ways to compute precision.

4.3.3 Preliminary Test

In order to illustrate the effects of the different described alternatives, we show their results on three common collaborative filtering algorithms, based respectively on probabilistic Latent Semantic Analisys (pLSA) (Hofmann, 2004), matrix factorisation (MF) (Koren et al., 2009), and user-based nearest-neighbours (kNN) (Cremonesi et al., 2010). As additional baselines, we include recommendation by popularity and random recommendation. We use two datasets: the 1M version of MovieLens, and

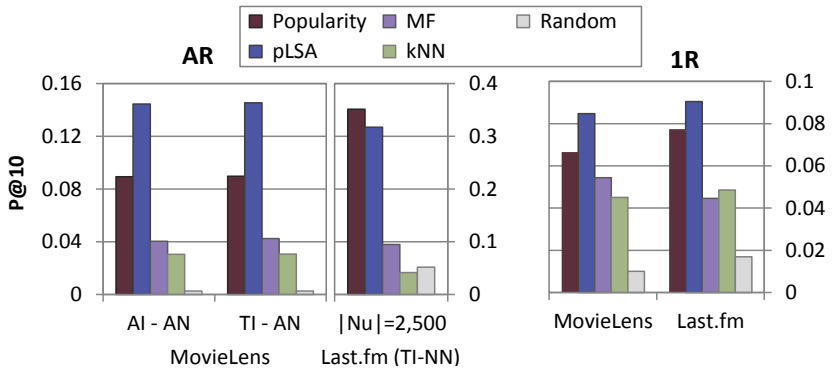

Figure 4.1. Precision of different recommendation algorithms on MovieLens 1M and Last.fm using AR and 1R configurations.

an extract from Last.fm published by Ò. Celma (Celma and Herrera, 2008). Details about the implementation and datasets partition are provided in Appendix A.

Figure 4.1 shows the P@10 results with AR and 1R configurations. For 1R we shall always use TI-NN, with $|T_u|$ = 100. This is a significantly lower value than $|T_u|$ = 1,001 reported in (Cremonesi et al., 2010), but we have found it sufficient to ensure statistical significance (e.g. Wilcoxon $p \ll 0.001$ for all pairwise differences between the recommenders in Figure 4.1), at a considerably reduced execution cost. We adopt the TI policy in 1R to avoid biases that we shall describe later. In the AR configuration we show TI-AN and AI-AN for MovieLens, though we shall generally stick to TI-AN in the rest of the chapter. In Last.fm we use only TI-NN and a temporal split, with $|N_u|$ = 2,500 for efficiency reasons, since $|\mathcal{I}|$ = 176,948 is considerably large in this dataset. We set the positive relevance rating threshold to 5 in MovieLens, as in (Cremonesi et al., 2010), whereas in Last.fm, we take any number above 2 playcounts as a sign of positive preference. We have experimented with other thresholds for positive ratings, obtaining equivalent results to all the ones that are reported here – the only difference is discussed in Section 4.6.

It can be seen that pLSA consistently performs best in most experimental configurations, closely followed by popularity, which is the best approach in Last.fm with AR, and that MF is generally superior to kNN. Some aspects strike our attention. First, even though P@10 is supposed to measure the same thing in all cases, the range of the metric varies considerably across configurations and datasets, and even the comparison is not always consistent. For instance, in AR popularity ranges from 0.08 on MovieLens to 0.35 on Last.fm; and AR vs. 1R produces some disagreeing comparisons on Last.fm. It may also be surprising that popularity, a non-personalised method, fares so well compared to other algorithms. This effect was already found recently in (Cremonesi et al., 2010) and (Steck, 2011). We also see that TI and AI produce almost the same results. This is because $\bigcup_{u \in \mathcal{U}} R_{test}(u) \sim \mathcal{I}$ in MovieLens; differences become noticeable in configurations where $\bigcup_{u \in \mathcal{U}} R_{test}(u)$ is significantly

smaller than $\mathcal{I}$, as we shall see in Section 4.6.2. As mentioned before, note that in this case, the upper bound of P@10 for the 1R methodology is 0.10.

Some of this variability may reflect actual strengths and weaknesses of the algorithms for different datasets, but we shall show that a significant part of the observed variations is due to statistical biases arising in the adaptation of the Cranfield methodology to recommendation data, and are therefore meaningless with respect to the assessment of the recommenders' accuracy. Specifically, we have found that the metrics are strongly biased to test data sparsity and item popularity. We shall analyse this in detail in Sections 4.4 and 4.5, but before that we establish a relation between AR and 1R precision that will help in this analysis.

4.3.4 Relation between AR and 1R

We have seen that AR and 1R precisions produce in general quite different values, and we shall show they display different dependencies over certain factors. We find nonetheless a direct relation between the two metrics. Specifically, 1R precision is bound linearly by NN-AR precision, that is, $1\mathrm{RP}_s@n = \Theta(\mathrm{P}_s@n)$, as we show next.

Lemma. Let us assume the irrelevant item sampling in 1R is done only once for all the test ratings of a user, that is, we select the same set of non-relevant items $N_u^r = N_u$ in the T_u^r target sets. If we denote $T_u = N_u \cup PR_{test}(u)$ – in other words, $T_u = \bigcup_r T_u^r$ –, we have:

$$\frac{|\mathcal{U}|P_s@n}{pr_{test}} \leq 1RP_s@n \leq \frac{\sum_{u\in\mathcal{U}} m_u P_s^u@m_u(T_u)}{n \cdot pr_{test}} \tag{4.2}$$

with $m_u = n + pr_{test}(u) - 1$, where $\mathrm{P}_s@n$ is the NN-AR precision computed with the target sets $\{T_u\}$.

Proof. Let i_u^r be the relevant item included in T_u^r, and let $\tau_s^u(i, S)$ denote the ranking position assigned to i by s for u within a set S, where $i \in S$. Since $T_u^r \subset T_u$, we have that $\tau_s^u(i_u^r, T_u^r) \leq \tau_s^u(i_u^r, T_u)$. This means that if i_u^r is ranked above n in T_u, then it is also above n in its target set T_u^r. Hence $\sum_{r=1}^{np_{test}(u)} |top_s^u(T_u^r, n) \cap PR_{test}(u)| \geq |top_s^u(T_u, n) \cap PR_{test}(u)|$. Summing on u, and dividing by n and pr_{test} we prove the first inequality of Equation (4.2).

On the other hand, it is easy to see that $\tau_s^u(i_u^r, T_u^k) \geq \tau_s^u(i_u^r, T_u) + pr_{test}(u) - 1$. Thus, if i_u^r is ranked above n in T_u^k, then it is above $m_u = n + pr_{test}(u) - 1$ in T_u. Thus $\sum_{r=1}^{np_{test}(u)} |top_s^u(T_u^r, n) \cap PR_{test}(u)| \leq |top_s^u(T_u, m_u) \cap PR_{test}(u)| = m_u \mathrm{P}_s@m_u(T_u)$. And the second inequality of Equation (4.2) follows again by summing on u, and dividing by n and pr_{test}. □

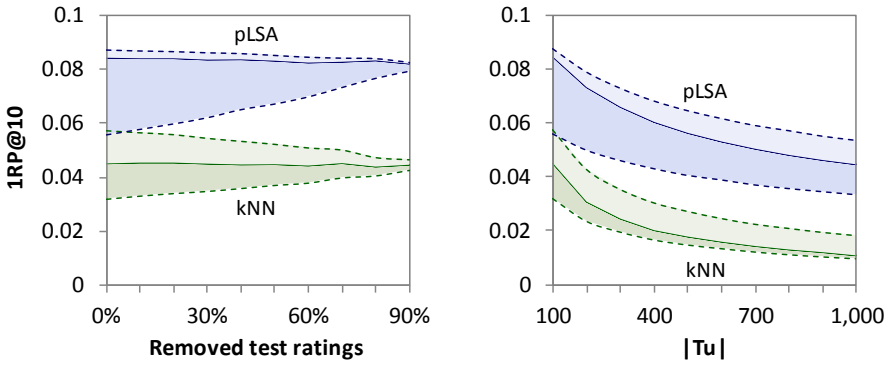

Figure 4.2. Empiric illustration of Equation (4.2). The curves show 1RP@10 and its bounds, for pLSA and kNN over MovieLens 1M. The light and dark shades mark the distance to the upper and lower bounds, respectively. The left side shows the evolution when progressively removing test ratings, and the right side displays the variation with $|T_u|$ ranging from 100 to 1,000.

Note that the assumption $N_u^r = N_u$ in the lemma is mild, inasmuch as the statistical advantage in taking different N_u^r for each r is unclear. Even in that case, $\mathrm{P}_s@n$ and $\mathrm{avg}_{u \in \mathcal{U}}\left(m_u \mathrm{P}_s^u @ m_u(T_u)\right)$ should be reasonably stable with respect to the random sampling of N_u^r, and thus Equation (4.2) tends to hold. Figure 4.2 illustrates the relation between the AR bounds and the 1R values. The empiric observation suggests they provide similar while not fully redundant assessments. We also see that the bounding interval reduces progressively as $|T_u|$ is increased (right), and even faster with test data sparsity (left) – in sum, the metric converges to its bounds as $|T_u| \gg \mathrm{avg}_u\, pr_{test}(u) = pr_{test}/|\mathcal{U}|$.

4.3.5 Limitations of error-based metrics

The analysis presented in (Bellogín et al., 2011a) leads to question again the suitability of error metrics. As in (McLaughlin and Herlocker, 2004), we found that there is no direct equivalence between results with error- and precision-based metrics. Common sense suggests that putting more relevant items in the top-N is more important for real recommendation effectiveness than being accurate with predicted rating values, which are usually not even shown to real users. Our study confirms that measured results differ between these two perspectives. An online experiment, where real users' feedback is contrasted to the theoretic measurements, may shed further light for an objective assessment and finer analysis of which methodology better captures user satisfaction.

Furthermore, the use of error-based metrics may not be applicable depending on the dataset or the recommender evaluated. For instance, log-based datasets and probabilistic (e.g. pLSA) or popularity-based recommenders cannot be evaluated using error-based metrics because no real ratings are available in the first case, and in

the second case because such recommenders do not necessarily predict a rating, not even a score in the range of ratings (Cremonesi et al., 2010).

The application of ranking-based metrics to recommendation, nonetheless, is far from being trivial. Firstly, there are obvious differences between the Cranfield paradigm and a standard recommendation context, as described in Section 4.2. Secondly, the evaluation methodology may be sensitive to any statistical bias which may appear in the process. In the next sections we shall analyse two of these sources of bias: sparsity and popularity.

4.4 Sparsity bias

As mentioned earlier, we identify two strong biases in precision metrics when applied to recommendation. The first one is a sensitivity to the ratio of the test ratings vs. the added non-relevant items. We study this effect by an analysis of the expected precision for non-personalised and random recommendations in the AR and 1R settings.

4.4.1 Expected Precision

Let $i_k^{u,s} \in T_u$ be the item ranked at position k in the recommendation output for u by a recommender system s, and let σ be the ratio of test data in the training-test data split. In an AR setup the expected precision at n (over the sampling space of data splits with ratio σ, the sampling of N_u, and any potential non-determinstic aspect of the recommender system – as e.g. in a random recommender) is:

$$E[P_s@n] = \underset{u \in \mathcal{U}}{avg}\left(\frac{1}{n}\sum_{k=1}^{n} p\left(rel \middle| i_k^{u,s}, u, T_u\right)\right)$$

where $p(rel|i,u)$ denotes the probability that item i is relevant for user u, i.e., the probability that $i \in PR_{test}(u)$. Now we may write $p\left(rel \middle| i_k^{u,s}, u, T_u\right) \overset{\text{def}}{=} p\left(rel, T_u \middle| i_k^{s,u}, u\right) / p\left(T_u \middle| i_k^{s,u}, u\right)$, where we have $p\left(T_u \middle| i_k^{s,u}, u\right) = p\left(i_k^{s,u} \in T_u\right) = |T_u|/|\mathcal{C}|$. On the other hand, $p\left(rel, T_u \middle| i_k^{s,u}, u\right) \overset{\text{def}}{=} p\left(i_k^{s,u} \in PR_{test}(u) \cap T_u\right) = p\left(i_k^{s,u} \in PR_{test}(u)\right) = p\left(rel \middle| i_k^{s,u}, u\right)$, since $PR_{test}(u) \subset T_u$ in the AR methodology. If s is a non-personalised recommender then $i_k^{u,s}$ and u are mutually independent, and it can be seen that $\text{avg}_{u \in \mathcal{U}}\, p\left(rel \middle| i_k^{u,s}, u\right) = \text{avg}_{u \in \mathcal{U}}\, p\left(rel \middle| i_k^{u,s}\right)$. All this gives:

$$E[P_s@n] = \frac{|\mathcal{C}|}{n \cdot t} \sum_{k=1}^{n} \underset{u \in \mathcal{U}}{avg}\, p\left(rel \middle| i_k^{u,s}\right)$$

where $1/t = \text{avg}_u(1/|T_u|)$ – if T_u have all the same size, then $t = |T_u|$. As all relevant items for each user are included in her target set, we have $p\left(rel\middle|i_k^{u,s}\right) = \text{E}\left[pr_{test}\left(i_k^{u,s}\right)\right]/|\mathcal{U}|$. If ratings are split at random into test and training, this is equal to $\sigma \cdot pr\left(i_k^{u,s}\right)/|\mathcal{U}|$. Hence, we have:

$$E[P_s@n] = \frac{\sigma|\mathcal{C}|}{n \cdot t\ |\mathcal{U}|}\sum_{k=1}^{n} \underset{u\in\mathcal{U}}{avg}\ pr\left(i_k^{u,s}\right) \tag{4.3}$$

Now, if items were recommended at random, we would have $\text{E}\left[pr\left(i_k^{u,RND}\right)\right] = pr/|\mathcal{I}|$, and therefore:

$$E[P_{RND}@n] = E[P_{RND}] \sim \frac{\sigma \cdot pr}{t\ |\mathcal{U}|} = \sigma \cdot \delta \tag{4.4}$$

where δ is the average density of known relevance – which depends on how many preferences for items the users have conveyed, and the size of the target test item sets.

On the other hand, in a 1R evaluation setup, we have:

$$E[1RP_s@n] = \frac{1}{n \cdot pr_{test}}\sum_{u\in\mathcal{U}}\sum_{r=1}^{pr_{test}(u)}\sum_{k=1}^{n} p\left(rel\middle|i_k^{u,r,s}, u, T_u^r\right)$$

where $i_k^{u,r,s} \in T_u^r$ denotes the item ranked at position k in T_u^r. For random recommendation, we have $p\left(rel\middle|i_k^{u,r,RND}, u, T_u^r\right) = 1/|T_u^r| = 1/t$ since all target sets have the same size, whereby we have:

$$E[1RP_{RND}@n] = E[1RP_{RND}] = 1/t \tag{4.5}$$

4.4.2 Test Sparsity Bias

The above results for the expected random precision provide a formal insight on strong metric biases to characteristics of the data and the experimental configuration. In both Equations (4.4) for AR and (4.5) for 1R, we may express the expected random precision as $\text{E}[\text{P}_{RND}@n] = \text{avg}_u\,\rho_u = \rho$, where ρ_u is the ratio of positively rated items by u in T_u (or T_u^r, for that matter), and $\rho \sim \sigma \cdot \delta$, or $\rho = 1/t$, depending on the experimental approach. In the AR approach the density δ, and thus the ρ ratio, are also inversely proportional to t. Precision in this methodology is therefore sensitive to (grows linearly with) σ and pr, and is inversely proportional to t, whereas 1R is only sensitive (inversely proportional) to t. The expected precision of random recommendation naturally provides a lower bound for any acceptable recommender. Note that in any configuration of AR and 1R, the total precision of any system is $\text{P}_s = \text{P}_{RND} = \rho = \text{E}[\text{P}_{RND}@n]$, since as all systems are required to return

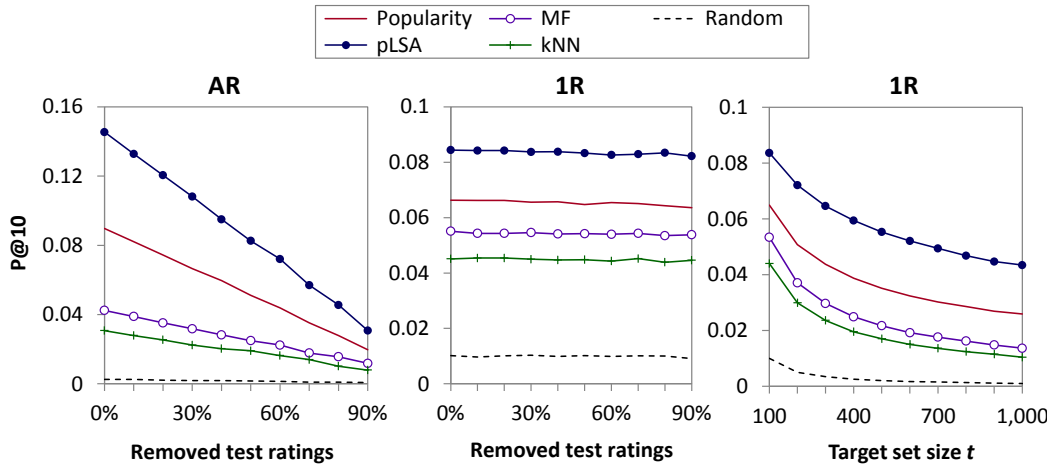

Figure 4.3. Evolution of the precision of different recommendation algorithms on MovieLens 1M, for different degrees of test sparsity. The x axis of the left and center graphics shows different amounts of removed test ratings. The x axis in the right graphic is the size of the target item sets.

(recommend) all items in the target sets T_u (or T_u^r), that is, the total precision does not depend on the ranking. At lower cutoffs, we expect to have $\mathrm{P}_s@n > \mathrm{E}[\mathrm{P}_{RND}@n] = \rho$. In other words, the lower bound – and so the expected range – for the $\mathrm{P}@n$ of recommender algorithms grows with the average ratio of relevant items per target item set.

The ρ ratio – hence the random precision – thus depends on several aspects of the experimental setup (the experimental approach, the split ratio σ, the number of non-relevant items in the target sets), and the test collection (the number of ratings, the number of users). Therefore, since ρ and the random precision can be adjusted arbitrarily by how the test sets are split, constructed, etc., we may conclude that **the specific value of the metric has a use for comparative purposes, but has no particular meaning by itself, unless accompanied by the corresponding average relevance ratio** ρ of the target test sets. This is naturally in high contrast to common IR datasets, where both the document collection and the relevance information are fixed and not split or broken down into subsets. In fact, the metric values reported in the TREC campaigns have stayed within a roughly stable range over the years (Armstrong et al., 2009a; Armstrong et al., 2009b). Note also that the sparsity bias we analyse here is different from the impact of training data sparsity in the performance of collaborative filtering systems. What we describe is a statistical bias caused by the sparsity of test data (as a function of overall data sparsity and/or test data sampling), and its effect does not reflect any actual variation whatsoever in the true recommendation accuracy.

The sparsity bias explains the precision range variations observed earlier in Figure 4.1. The empirically obtained values of random precision match quite exactly the

theoretically expected ones. To what extent the random recommendation analysis generalises to other algorithms can be further analysed empirically. Figure 4.3 illustrates the bias trends over rating density and target set size, using the experimental setup of Section 4.3.3 (with TI-AN in AR, and TI-NN in 1R). We show only the results in MovieLens – they display a similar effect on Last.fm. In the left and center graphics, we simulate test sparsity by removing test ratings. In the right graphic we vary $t = |T_u|$ in a 1R configuration. We observe that the empirical trends confirm the theoretical analysis: precision decreases linearly with density in the AR methodology (left graphic, confirming a linear dependence on δ), whereas precision is independent from the amount of test ratings in the 1R approach (center), and shows inverse proportionality to t (right). It can furthermore be seen that the biased behavior analytically described for random recommendation is very similarly displayed by the other recommenders (only differing in linear constants). This would confirm the explanatory power of the statistical trend analysis of random recommendation, as a good reference for similar biases in other recommenders. On the other hand, even though the precision values change drastically in magnitude, it would seem that the comparison between recommenders is not distorted by test sparsity. We find other biases in precision measurements, however, which do affect the comparison of recommenders, as we study in the next section.

4.5 Popularity bias

Sparsity is not the only bias the metric measurements are affected by. The high observed values for a non-personalised method such as recommendation by popularity raise the question of whether this really reflects a virtue of the recommender, or some other bias in the metric. We seek to shed some light on the question by a closer study.

4.5.1 Popularity-Driven Recommendation

Even though they contradict the personalisation principle, the good results of popularity recommendation can be given an intuitive explanation. By averaging over all users, precision metrics measure the overall satisfaction of the user population. A method that gets to satisfy a majority of users is very likely to perform well under such metrics. In other words, average precision metrics tend to favour the satisfaction of majorities, regardless of the dissatisfaction of minorities, whereby algorithms that target majority tastes will expectably yield good results on such metrics. This implicitly relies on the fact that on a random item split, the number of test ratings for an item correlates with its number of training ratings, and its number of positive ratings correlates with the total number of ratings. More formally, the advantage of

popularity-oriented recommendation comes from the fact that in a random rating split, $\mathrm{E}[pr_{test}(i)] \propto pr(i) \propto \mathrm{E}[pr_{train}(i)] \propto r_{train}(i)$, which means that the items with many training ratings will tend to have many positive test ratings, that is, they will be liked by many users according to the test data. We analyse this next, more formally and in more detail.

In a popularity recommender $i_k^{u,POP}$ is the k-th item in the target set with most ratings in the training set – i.e., the system ranks items by decreasing order of $r_{train}(i_k^{u,POP})$. This ranking is almost user-independent (except for those, statistically negligible, user items already in training which are excluded from the ranking) and therefore, for an AR experimental design, Equation (4.3) applies. Since we have $\sum_{k=1}^{n} pr(i_k^{u,POP}) = \max_s \sum_{k=1}^{n} pr(i_k^{u,s})$ (as far as $\mathrm{E}[pr_{test}(i)] \propto r_{train}(i)$ for a random training-test split), the popularity recommendation is the best possible non-personalised system, maximising $\mathrm{E}[\mathrm{P}_s@n]$. Popularity thus achieves a considerably high precision value, just for statistical reasons.

For a 1R experimental design, using Equation (4.2) (lemma) we have:

$$\frac{|\mathcal{U}|E[P_s@n]}{pr_{test}} \leq E[1RP_s@n] \leq \frac{\sum_u E[m_u P_s^u@m_u(T_u)]}{n \cdot pr_{test}}$$

Now, since $\mathrm{P}_s@n$ and $\mathrm{P}_s^u@m_u$ above are computed by AR, we may elaborate from Equation (4.3) for a non-personalised recommender, and we get:

$$\frac{|\mathcal{I}|}{n \cdot t \cdot pr} \underset{u \in \mathcal{U}}{avg} \sum_{k=1}^{n} pr(i_k^{u,s}) \leq E[1RP_s@n] \leq \frac{|\mathcal{C}|}{n \cdot t \cdot pr} \underset{u \in \mathcal{U}}{avg} \sum_{k=1}^{m_u} pr(i_k^{u,s})$$

This experimental approach is thus equally biased to popular items, since the latter optimise $\sum_{k=1}^{n} pr(i_k^{u,s})$.

Note that the advantage of popularity over other recommenders is highly dependent on the skewness in the distribution of ratings over items: if all items were equally popular, the popularity recommender would degrade to random recommendation – in fact slightly worse, as $pr_{test}(i) \propto r_{test}(i) = r/|\mathcal{I}| - r_{train}(i)$, so popular items would have fewer positive test ratings. On the other extreme, if a few items (less than n) are liked by most users, and the rest are liked by very few, then popularity approaches the maximum precision possible.

4.5.2 Popularity Distributions

In order to illustrate how the dependence between the popularity precision and the background popularity distribution evolves, we simulate different degrees of skewness in rating distributions. As a simulated distribution pattern we use a shifted power law $r(i_k) = c_1 + \beta(c_2 + k)^{-\alpha}$, where α determines the skewness (e.g. $\alpha \sim 1.4$ for MovieLens 1M). Figure 4.4 (left) shows the shape of generated distributions ranging

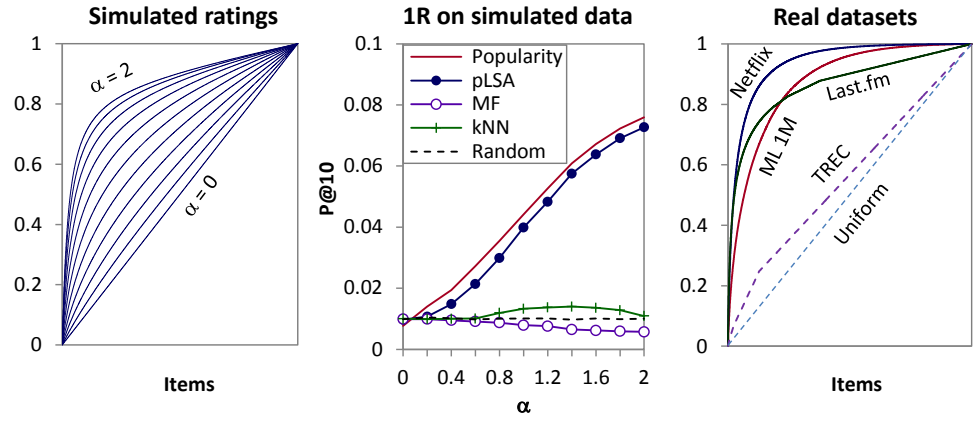

Figure 4.4. Effect of popularity distribution skewness on the popularity bias. The left graphic shows the cumulated popularity distribution of artificial datasets with simulated ratings, with skewness ranging from α = 0 to 2. The x axis represents items by popularity rank, and the y axis displays the cumulative ratio of ratings. The central graphic shows the precision of different recommendation algorithms on each of these simulated datasets. The right graphic shows the cumulative distribution of positive ratings in real datasets.

from uniform ($\alpha = 0$) to a very steep long-tailed popularity distribution ($\alpha = 2$), and (center) how the measured precision evolves in this range. The artificial data are created with the same number of users, items, and ratings (therefore the same rating density) as in MovieLens 1M, setting c_1 and c_2 by a fit to this dataset, and enforcing these constraints by adjusting β. The rating values are assigned randomly on a 1-5 scale, also based on the prior distribution of rating values in Movie-Lens.

The results in Figure 4.4 (center) evidence the fact that the precision of popularity-based recommendation is heavily determined by the skewness of the distribution. It benefits from steep distributions, and degrades to slightly below random (0.0077 vs. 0.0100) when popularity is uniform. This slightly below-random performance of popularity recommendation at $\alpha = 0$ is explained by the fact that $E[pr_{test}(i)] \propto E[r_{test}(i)] = r(i) - E[r_{train}(i)]$ is inverse to the popularity ranking by $r_{train}(i)$ when $r(i)$ is uniform, as predicted at the end of the previous section. kNN and MF stay essentially around random recommendation. This is because the data are devoid of any consistent preference pattern (as collaborative filtering techniques would assume) in this experiment, since the ratings are artificially assigned at random, and the results just show the "pure" statistical dependency to the popularity distribution. pLSA does seem to take advantage of item popularity, as it closely matches the effectiveness of popularity recommendation. We show only the 1R design, but the effect is the same in AR.

This observation also explains the difference between datasets from IR and those from recommendation with regards to the popularity bias. Figure 4.4 (right) shows the cumulative distribution of positive user interaction data per item in three

datasets: Netflix, MovieLens, and Last.fm (the dataset in Section 4.3.3). The shapes of the curves are typical of long-tailed distributions, where a few popular items accumulate most of the preference data (Celma, 2010; Celma and Cano, 2008). This contrasts with the distribution of positive relevance judgments over documents in TREC data (same figure) – where we have aggregated 30 individual tracks, filtering out the documents that are not relevant to any query, and obtaining a set of 703 queries, 129,277 documents, and 149,811 positive judgments. The TREC distribution is considerably flatter, not far from uniform: 87.2% of documents are relevant to just one query, and the maximum number of positive assessments per document is 25 (3.6% of queries), whereas the top popular item in Netflix, MovieLens, and Last.fm, is liked by 20.1%, 32.7% and 73% of users, respectively.

Several reasons account for this difference between retrieval and recommender datasets. First, in IR queries are selected by design, intending to provide a somewhat varied testbed to compare retrieval systems. Hence, including similar queries with overlapping relevance would not make much sense. Second, queries in natural search scenarios are generally more specific and narrower than global user tastes for recommendation, whereby the corresponding relevant sets have much less intersection. Furthermore, the TREC statistics we report are obtained by aggregating the data of many tracks, in order to seek any perceptible popularity slant. The typical TREC experiments are actually run on separate tracks comprising typically 50 queries, where very few documents, if any, are relevant to more than one query. Note also that even though we have filtered out over 0.7 million non-relevant plus nearly 5 million unlabeled documents in the TREC statistics, the non-relevant documents actually remain as input to the systems, contrarily to experiments in the recommender domain, thus making up an even flatter relevance distribution. Moreover, in the usual IR evaluation setting, the systems have no access to the relevance data – thus, they have no means to take a direct bias towards documents with many judgments –, whereas in recommendation, this is the primary input the systems (particularly collaborative filtering recommenders) build upon. The popularity phenomenon has therefore never been an issue in IR evaluation, and neither the metrics nor the methodologies have had to even consider this problem, which arises now when bringing them to the recommendation setting – where the overlap between user preferences is not only common, but actually needed by collaborative filtering algorithms.

4.6 Overcoming the popularity bias

After analysing the effects of popularity in precision metrics, the issue remains: to what extent do the good results of popularity recommendation reflect only a statistical bias in a metric, or any degree of actual recommendation quality? The same question should be raised for pLSA, which seems to follow the popularity trends quite

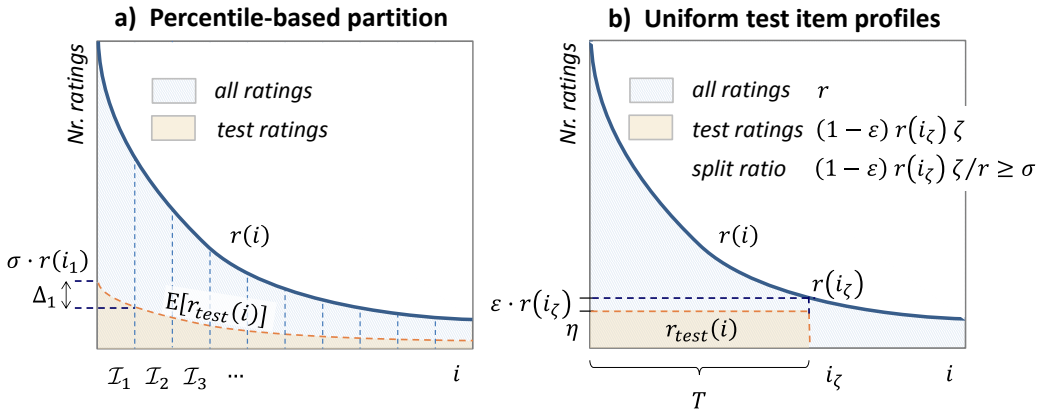

Figure 4.5. Rating splits by a) a popularity percentile partition (left), and b) a uniform number of test ratings per item (right). On the left, the red dashed split curve represents $E[pr_{test}(i)]$ – i.e., the random split ratio needs not be applied on a per-item basis – whereas on the right it does represent $pr_{test}(i)$.

closely. We address the question by proposing and examining alternative experimental configurations, where the statistical role of popularity gets reduced, as we propose next.

4.6.1 Percentile-Based Approach (P1R)

We propose a first approach to neutralise the popularity bias, which consists in partitioning the set of items into m popularity percentiles $\mathcal{I}_k \subset \mathcal{I}$, breaking down the computation of accuracy by such percentiles, and averaging the m obtained values. By doing so, in a common long-tailed popularity distribution, the margin for the popularity bias is considerably reduced, as the difference Δ_k in the number of positive test ratings per item between the most and least popular items of each percentile is not that high. The popularity recommender is forced to recommend as many unpopular as popular items, thus leveling the statistical advantage to a significant extent. It remains the optimal non-personalised algorithm, but the difference – and thus the bias – is considerably reduced. The technique is illustrated in Figure 4.5a.

A limitation of this approach is that it restricts the size of the target sets by $|T_u| \leq |\mathcal{I}|/m$. For instance, for $m = 10$ in MovieLens 1M, this imposes a limit of $|T_u| \leq \sim 370$, which seems acceptable for 1R. The restriction can be more limiting in the AR approach, e.g. the TI and AI options cannot be applied (except within the percentiles). For this reason, we will only apply the percentile technique in the 1R design, a configuration to which we shall refer as **P1R**.

4.6.2 Uniform Test Item Profiles (UAR, U1R)

We now propose a second technique consisting of the formation of data splits where all items have the same amount of test ratings. The assumption is that the items with a high number of training ratings will no longer have a statistical advantage by having more positive test ratings. That is, the relation $\mathrm{E}[pr_{test}(i)] \propto r_{train}(i)$ described in Section 4.5.1 breaks up. The approach consists of splitting the data by picking a set T of candidate items, and a number η of test ratings per item so that $|T|\eta/r = \sigma$. For this to be possible, it is necessary that $(1-\varepsilon)\, r(i) \geq \eta, \forall i \in T$, where ε is a minimum ratio of training ratings per item we consider appropriate. In particular, in order to allow for n-fold cross-validation, we should have $\varepsilon \geq 1/n$. The selection of T can be done in several ways. We propose to do so in a way that it maximises $|T|$, i.e., to use as many different target test items as possible, avoiding a biased selection towards popular items. If we sort $i_k \in \mathcal{I}$ by popularity rank, it can be seen that this is achieved by picking $T = \{i_k \in \mathcal{I} | k \leq \zeta\}$ with $\zeta = \max\{k|\,(1-\varepsilon)\, r(i_k)\, k/r \geq \sigma\}$, so that $\eta = (1-\varepsilon)\, r(i_\zeta)$. Figure 4.5b illustrates this procedure.

The expected effect of this approach is that the statistical relation $\mathrm{E}[pr_{test}(i)] \propto pr(i)$ no longer holds, and neither should hold now, as a consequence, the rationale described in Section 4.5.1 for popularity being the optimum non-personalised recommender. In fact, since $\mathrm{E}[pr_{test}(i)] = \eta \cdot pr(i)/r(i)$ for any $i \in T$, and $\eta = \sigma \cdot r/|T|$, it can be seen that if $\mathcal{C} = T$ (TI policy) Equation (4.3) for AR yields:

$$E[P_s@n] = \frac{\sigma \cdot r}{n \cdot t|\mathcal{U}|} \sum_{k=1}^{n} \underset{u \in \mathcal{U}}{avg} \frac{pr\left(i_k^{u,s}\right)}{r\left(i_k^{u,s}\right)}$$

for any non-personalised recommender. If the ratio $pr\left(i_k^{u,s}\right)/r\left(i_k^{u,s}\right)$ of positive ratings does not depend on k, we have $\mathrm{E}[\mathrm{P}_s@n] = \mathrm{E}[\mathrm{P}_{RND}@n] = \sigma \cdot \delta$. This means that popularity recommendation may get some advantage over other recommenders only if – and to the extent that – popular items have a higher ratio of positive ratings than unpopular items, and popularity recommendation will degrade to random precision otherwise. On the other hand, it can be seen that if $\mathcal{C} \supsetneq T$ (i.e., the TI policy is not adhered to), then $\mathrm{E}[\mathrm{P}_{RND}@n]$ would get reduced by a factor of $|T|/|\mathcal{C}|$.

For a non-personalised recommender in a 1R design, elaborating from Equations (4.2) and (4.3) we get:

$$\frac{r}{n \cdot t \cdot pr} \underset{u \in \mathcal{U}}{avg} \sum_{k=1}^{n} \frac{pr\left(i_k^{u,s}\right)}{r\left(i_k^{u,s}\right)} \leq E[1RP_s@n] \leq \frac{r}{n \cdot t \cdot pr} \underset{u \in \mathcal{U}}{avg} \sum_{k=1}^{m_u} \frac{pr\left(i_k^{u,s}\right)}{r\left(i_k^{u,s}\right)},$$

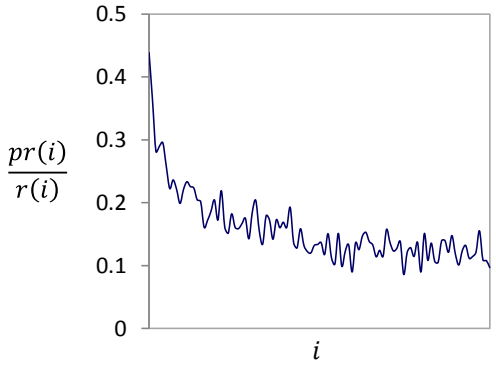

Figure 4.6. Positive ratings ratio vs. popularity rank in MovieLens 1M. The graphic plots $pr(i)/r(i)$, where items are ordered by decreasing popularity. We display averaged values for 100 popularity segments, for a smoothed trend view.

an equivalent situation where the measured precision of popularity recommendation is bound by the potential dependence between the ratio of positive ratings and popularity.

Figure 4.6 shows this ratio as $pr(i)/r(i)$ with respect to the item popularity rank in MovieLens 1M. It can be seen that indeed the ratio grows with popularity in this dataset, which does lend an advantage for popularity recommendation. Even so, we may expect the bias to be moderate – but this has to be tested empirically, as it depends on the dataset. Note also that in applications where all ratings are positive (as e.g. in our Last.fm setup), popularity – and any non-personalised recommender – would drop exactly to random precision ($\mathrm{E}[\mathrm{P}_s@n] = \sigma \cdot \delta$ in AR and $1/t$ in 1R).

A limitation of this approach is that the formation of T may impose limits on the value of σ, and/or the size of T. If the popularity distribution is very steep, T may turn out small and therefore biased to a few popular items. Moreover, there is in general a solution for T only up to some value of σ – it is easy to see (formally, or just visually in Figure 4.5) that as $\sigma \to 1$ there is no item for which $(1-\varepsilon)\, r(i_k)\, k/r \geq \sigma$, unless the popularity distribution was uniform, which is never the case in practice. We have however not found these limitations to be problematic in practice, and common configurations turn out to be feasible without particular difficulty. For instance, in MovieLens 1M we get $|T|$ = 1,703 for $\sigma = 0.2$ with ε = 0.2 (allowing for a 5-fold cross-validation), resulting in η = 118 test ratings per item.

This method can be used, as noted, in both the AR and 1R approaches. We shall refer to these combinations as **UAR** and **U1R** respectively, where 'U' stands for the "uniform" number of item test ratings. In U1R it is important to set $\mathcal{C} = T$ in order to sample non-relevant items within T (i.e., $N_u \subset T$, for the TI policy). Otherwise, popularity would have a statistical advantage over other recommenders, as it would systematically rank irrelevant items in $N_u - T$ below any relevant item in T, whereas

other algorithms might not. The same can be considered in UAR, unless the experimental setup requires $|T_u| > |T|$, as e.g. in the AI design. In that case a slight popularity bias would arise, as we shall see next.

4.6.3 Experimental Results

Figure 4.7 compares the results measured by 1R, AR and their corresponding popularity-neutralising variants. The setup is the same as in previous sections, except that for AR, we take TI-NN with $|N_u|$ = 1,700, to level with UAR in random precision. All the results correspond to MovieLens 1M except Last.fm where indicated. It can be seen that P1R, U1R and UAR effectively limit the popularity bias. The techniques seem to be more effective on 1R than AR: U1R and (even more) P1R actually place the popularity algorithm by the level of random recommendation, whereas the measured popularity precision decreases in UAR, but remains above kNN. The advantage of popularity over randomness in U1R and P1R is explained by the bias in the ratio of positive ratings in popular items (Figure 4.6). This ratio is constant in Last.fm, whereby popularity drops to random in U1R, as predicted by our analysis in the previous section, proving that the popularity bias remaining in the uniform-test approach is caused by this factor. This residual bias is higher in U1R than P1R, because in the former, N_u is sampled over a larger popularity interval ($|T|$ = 1,703 vs. $|\mathcal{I}|$ / 10 = 370 items), giving a higher range for advantage by popularity, which also explains why the latter still overcomes kNN in UAR. We may observe the importance of using the TI policy in UAR, without which (in AI-UAR) a higher bias remains. We also show the effect of removing the 10% most popular head items from the test data (and also from $\mathcal{C}$, i.e., they are excluded from N_u sampling) in 1R, as a simple strategy to reduce the popularity bias (Cremonesi et al., 2010). We see that this technique reduces the measured precision of popularity, but it is not quite as effective as the proposed approaches.

It is finally worth emphasising how **the percentile and uniform-test approaches discriminate between pure popularity-based recommendation and an algorithm like pLSA**, which does seem to take popularity as one of its signals, but not the only one. The proposed approaches allow uncovering the difference, neutralising popularity but not pLSA, which remains the best algorithm in all configurations.

As we mentioned in Section 4.3, we have taken precision as a simple and common metric for our study, but all the presented analysis and proposed alternatives straightforwardly generalise to other standard IR metrics, such as MAP, nDCG, and Mean Reciprocal Rank (MRR). Their application is direct in the AR setting; and they can be applied in 1R by simply introducing them in place of precision in the internal summation of Equation (4.1). Figure 4.7 shows results for nDCG, where we see that the analysed patterns hold just the same. The AR approach provides room for a

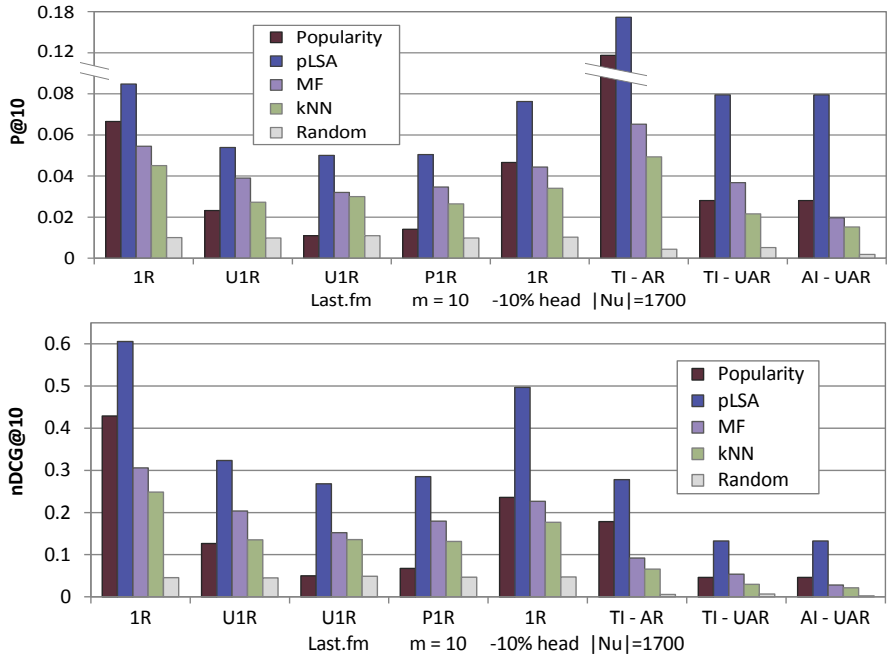

Figure 4.7. Precision and nDCG of recommendation algorithms on MovieLens 1M (and Last.fm only where indicated) using the 1R, U1R, P1R (m = 10 percentiles), AR, and UAR methodologies. The "-10% head" bars show the effect of removing the 10% most popular items from the test data (Cremonesi et al., 2010).

slightly wider metric variety than 1R, in the sense that some metrics reduce to each other in 1R. For instance, for a single relevant item, MAP is equivalent to Mean Reciprocal Rank ($\mathrm{MRR} = 1/k$ where k is the rank of the first relevant item). And nDCG is insensitive to relevance grades in 1R (the grade of the single relevant item cancels out), whereas grades do make a difference in AR.

4.7 Conclusions

The application of Information Retrieval methodologies to the evaluation of recommender systems is not necessarily as straightforward as it may seem. Hence, it deserves close analysis and attention to the differences in the experimental conditions, and their implications on the explicit and implicit principles and assumptions on which the metrics build. We have proposed a systematic characterisation of design alternatives in the adaptation of the Cranfield paradigm to recommendation tasks, aiming to contribute to the convergence of evaluation approaches. We have identified assumptions and conditions underlying the Cranfield paradigm which are not granted in usual recommendation experiments. We have detected and examined resulting statistical biases, namely test sparsity and item popularity, which do not arise in common test collections from IR, but do interfere in recommendation experi-

ments. Sparsity is clearly a noisy variable that is meaningless with respect to the value of a recommendation. Whether popularity is in the same case is less obvious; we propose experimental approaches that neutralise this bias, leaving way to an unbiased observation of recommendation accuracy, isolated from this factor. With a view to their practical application, we have identified and described the pros and cons of the array of configuration alternatives and variants analysed in this study.

In general, we have found that evaluation metrics computed in AR and 1R approaches differ in how they are averaged. This means, more specifically, that precision obtained by approaches following a 1R design is bound linearly by precision of AR approaches. Moreover, we have observed that a percentile-based evaluation considerably reduces the margin for the popularity bias, although the main limitation of this approach is that it specifies a constraint on the size of the possible target sets. Additionally, a uniform-test approach removes any statistical advantage provided by having more positive test ratings. Furthermore, we have found that both approaches discriminate between pure popularity-based recommendation and an algorithm like pLSA.

The main goal of our research addresses a second-order problem: we aim to predict the accuracy of the predictions of recommendation algorithms. As we shall see, the (second-order) evaluation of our researched methods relies on the (first-order) evaluation metrics and methodologies by which the recommendation algorithms' accuracy is measured. In order to consistently evaluate our methods, the primary recommendation evaluation has to be reliable and well-understood. Any bias in the process would lead to inconclusive or misleading results about the predictive power of our methods. For this reason, the results presented in this chapter are a necessity for the main goal of this thesis, but the outcome can be of more general use. Specifically, in the following chapters we shall compare how the different methodologies (with and without neutralised biases) may impact the observations on the predictive power of our predictors.

The popularity effects in recommender systems have started to be reported in recent work (Cremonesi et al., 2011; Cremonesi et al., 2010; Steck, 2011). Our research complements such findings by seeking principled theoretical and empirical explanations for the biases, and providing solutions within the frame of IR evaluation metrics and methodology – complementarily to the potential definition of new special-purpose metrics (Steck, 2011). The extent to which popularity is a noisy signal may be further analysed by developing more complete metric schemes incorporating gain and cost dimensions, where popular items would expectably score lower. Such metrics may e.g. account for the benefits (to both recommendation consumers and providers) drawn from novel items in typical situations (Vargas and Castells, 2011), as a complement to plain accuracy. Online tests with real users should also be valuable for a comparative assessment of offline observations, and the validation of experimental alternatives.