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):

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

where $\mathrm{rank}(u,i)$ 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):

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

where the discount function $f_{\mathrm{dis}}\big(rel(u,i_p),p\big)$ is usually defined as $f_{\mathrm{dis}}(x,y) = (2^x-1)/\log(1+y)$ or simply $f_{\mathrm{dis}}(x,y) = x/\log y$ if $y>1$, $f_{\mathrm{dis}}(x,y)=x$ 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 $\mathrm{nDCG@}k$.

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:

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

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:

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

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.