Chapter 2

Recommender systems

The aim of recommender systems is to assist users in finding their way through huge databases and catalogues, by filtering and suggesting relevant items taking into account or inferring the users' preferences (i.e., tastes, interests, or priorities).

Three types of recommender systems are commonly recognised according to how recommendations are made, namely content-based filtering (CBF), collaborative filtering (CF), and social filtering (SF) systems. A CBF system suggests a user items similar to those she preferred or liked in the past, a CF system suggests a user items that people with similar preferences liked in the past, and a SF system suggests items according to the preferences of the user's social contacts in a social network. Each of these types of recommendations has its own strengths and weaknesses. In order to address and compensate particular shortcomings, combinations of different recommendation approaches are usually developed, forming the so called hybrid filtering (HF) systems.

In this chapter we provide an overview of terminology, techniques, and limitations related to the above types of recommender systems. In Section 2.1 we formalise the problem of recommendation, and introduce the different types of recommendation approaches. Next, in Section 2.2 we describe content-based recommendation approaches, rating- and log-based recommendation approaches – as special cases of collaborative filtering –, and social-based recommendation approaches. In Section 2.3 we then explain generic hybrid filtering approaches. Finally, in Section 2.4 we present particular limitations of each type of recommender systems.

2.1 Formulation of the recommendation problem

Collaborative filtering can be considered as the first proposed recommendation approach. The term was coined in the mid 90's by Goldberg and colleagues when developing an automatic filtering system for electronic mail (Goldberg et al., 1992), although sometimes the stereotypes defined in (Rich, 1979) have been considered as an earlier reference. Collaborative filtering has been classed as part of the Information Retrieval area by several authors (Belkin and Croft, 1992; Foltz and Dumais, 1992), who have considered recommender systems as a particular case of information filtering. However, only a few recent attempts have been made at bringing recommender systems and information retrieval models together, by establishing equivalences between them (Wang et al., 2008b; Wang et al., 2008a; Bellogín et al., 2011b). Instead, recommender systems have been traditionally investigated from a different perspective, such as preference prediction and Machine Learning (Breese et al., 1998), upon which the main prediction models and evaluation metrics have been developed.

In this context distinct formulations and notations have been proposed. The overview by Adomavicius and Tuzhilin (2005) are among the most cited. In that work the recommendation problem is defined as follows. Let $\mathcal{U}$ be a set of users, and let $\mathcal{I}$ be a set of items. Let $g: \mathcal{U} \times \mathcal{I} \rightarrow \mathrm{R}$, where R is a totally ordered set, be a utility function such that $g(u,i)$ measures the gain of usefulness of item i for user u. Then, for each user $u \in \mathcal{U}$, we aim to choose items $i^{\max,u} \in \mathcal{I}$, unknown to the user, which maximise the utility function g, that is:

$$\forall u \in \mathcal{U}\ ,\ i^{max,u} = \arg\max_{i \in \mathcal{I}} g(u,i)$$

Depending on the exploited source of user preference information, and the way in which the utility function g is estimated for different users, the following two main types of recommender systems are commonly distinguished: 1) content-based recommender systems, in which a user is suggested items similar to those she liked or preferred in the past, and 2) collaborative filtering systems, in which a user is suggested items that people with similar preferences liked in the past. We extend this classification by also considering social recommender systems, i.e., systems in which a user is suggested items that friends – e.g. in an online social network – liked in the past. These systems are related but significantly different from collaborative filtering systems. Moreover, we distinguish two types of collaborative filtering systems, based on the form of their input: systems that exploit explicit user ratings (rating-based systems), and systems that exploit implicit user preference information (log-based systems). The rating assigned to an item by a particular user is typically interpreted as the true utility of that item for the user. There are systems, however, where no explicit ratings are available, but where user interests can be inferred from implicit

feedback information. In order to provide item recommendations in such systems, two plausible approaches do exist: 1) directly exploiting implicit preference data (Wang et al., 2008b; Deshpande and Karypis, 2004; Das et al., 2007; Hu et al., 2008; Linden et al., 2003), and 2) transforming implicit preference data into explicit ratings to be exploited by standard CF strategies (Celma, 2010; Jawaheer et al., 2010; Adams, 2007).

Other types of recommender systems have been considered in the literature, although they will not be described in detail herein; these are knowledge-based, utility-based, and demographic-based recommender systems.They use, respectively, semantic descriptions of the user preferences and item characteristics, an utility function over the items that describes the users' preferences, and demographic information about the users. For further descriptions and examples of these techniques, see (Burke, 2002) and (Ricci et al., 2011).

For any of the above mentioned types of recommender systems, models can be combined to improve their separate performance, or other characteristic of interest, such as the capability of providing more diverse and novel recommendations, and offering better explanations of recommendations. When such a combination is performed, the recommendation approach is considered a hybrid recommender (or hybrid filtering) system (Burke, 2002).

2.2 Recommendation techniques

As mentioned above, the main goal of a recommender system is to provide users with the most useful items according to their preferences. For such purpose, different strategies may be used, which can be categorised based on the type of data exploited, namely content-based, rating- and log-based collaborative filtering, and social recommendation strategies. In this section we formalise these strategies. We shall use the following notation. Letters u and v will be reserved for users ($u, v \in \mathcal{U}$), whereas i and j will denote items ($i, j \in \mathcal{I}$), $\mathcal{U}$ and $\mathcal{I}$ being, respectively, the set of users and items in the system. Besides, $r \in R$ will denote a particular rating value, and R will be the set of possible rating values, either discrete (typically, $R = \{1,2,3,4,5\}$) or continuous (e.g. $R = [0,5]$). Finally, $\tilde{r}$ shall denote a rating prediction (as opposed to observed ratings denoted by r).

2.2.1 Content-based recommenders

Content-based filtering (CBF, or simply content-based) techniques recommend items similar to those previously liked by a user. An extensive survey of this type of techniques can be found in (Lops et al., 2011; Pazzani and Billsus, 2007), and

(Adomavicius and Tuzhilin, 2005). In this section we briefly discuss some of the main approaches proposed in the field.

Content-based recommendation algorithms build a user's profile based on the features of the objects rated by the user, which are assumed to reflect the user's content-based interests (Lops et al., 2011). In general, a CBF technique can be classified according to whether a model is built from underlying data, commonly based on Machine Learning techniques (Lops et al., 2011; Pazzani and Billsus, 1997; de Gemmis et al., 2008), or use a heuristic function to compute item scores, mainly inspired on Information Retrieval methods (Diederich and Iofciu, 2006; Balabanovic and Shoham, 1997; Cantador et al., 2010).

Probabilistic methods in general and the naïve Bayes approach in particular generate a probabilistic model based on previously observed data. The naïve Bayes model estimates the *a posteriori* probability $P(c|d)$ of document d belonging to class c, based on the *a priori* probability $P(c)$ for the class, the probability $P(d)$ of observing the document, and the probability $P(d|c)$ of observing the document given the class (Lops et al., 2011), as follows:

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

In recommendation the naïve Bayes method is used to estimate the probability that a document (an item) is either relevant or irrelevant (class c), based on the information available for each user, that is, documents already rated are used to build the $P(d|c)$ probabilities. This approach has been used by many different authors (Mooney and Roy, 2000; Semeraro et al., 2007; de Gemmis et al., 2008; Lops et al., 2011).

Alternative methods for classifying the items in a system as relevant or irrelevant for each user include decision trees and neural networks (Pazzani and Billsus, 1997). These techniques, similarly to the naïve Bayes method, estimate in which class each (unrated or unobserved) item fits best with the user's profile.

Techniques based on Information Retrieval methods are specified by the way users and items are represented and the similarity function used between them. Typically they use a vector space model where each feature is weighted in a particular way. For instance, instead of using the frequency of each feature in a user/item profile, more complex functions from the Information Retrieval field may be used, such as TF-IDF and BM25 (Cantador et al., 2010). Furthermore, many different feature spaces have been considered in the literature: keywords (Lieberman, 1995; Pazzani et al., 1996), tags (Diederich and Iofciu, 2006; Michlmayr and Cazer, 2007), and semantic concepts enriched by different techniques (Magnini and Strapparava, 2001; Eirinaki et al., 2003; Cantador, 2008).

Regarding the feature vector similarity, the most common measure is the cosine similarity, even though the standard dot product between two vectors has also been used (Cantador et al., 2010):

$$sim_{dot}(d_i, d_j) = \sum_k w_{ki} w_{kj} \tag{2.1}$$

$$sim_{cos}(d_i, d_j) = \frac{sim_{dot}(d_i, d_j)}{\sqrt{\sum_k w_{ki}^2} \sqrt{\sum_k w_{kj}^2}} \tag{2.2}$$

where w_{ki} is the weight assigned (by any of the techniques mentioned before) to the feature k in document i.

In recommender systems items are suggested by decreasing order of similarity with the user, whose profile is represented in the same form of the documents (that is, in the space of features under consideration). The similarities are computed as the feature vector similarity between each (unrated or unobserved) document in the collection and the user's vector.

2.2.2 Rating-based recommenders

Collaborative filtering (CF) techniques match people with similar preferences, or items with similar choice patterns from users, in order to make recommendations. Unlike CBF, CF methods aim to predict the utility of items for a particular user according to the items previously evaluated by other like minded users. These methods have the interesting property that no item descriptions are needed to provide recommendations, since the methods merely exploit information about past ratings. Compared to CBF approaches, CF also has the salient advantage that a user may benefit from other people's experience, thereby being exposed to potentially novel recommendations beyond her own experience (Adomavicius and Tuzhilin, 2005).

In this section we focus on those CF techniques based on explicit numeric ratings, which are the most common in the literature. For additional references, see (Desrosiers and Karypis, 2011; Koren and Bell, 2011) and (Adomavicius and Tuzhilin, 2005). Most of our discussion nonetheless applies to log-based recommenders alike. In fact, as we shall show in the next section, most of the rating-based techniques can be used when no ratings are available (although the equivalence introduces additional assumptions).

In general, CF approaches are commonly classified into two main categories: **model-based** and **memory-based**. Model-based approaches build statistical models of user/item rating patterns to provide automatic rating predictions. Some approaches learn such models by performing some form of dimensionality reduction in order to uncover latent factors between users and items, e.g. by such techniques as

Singular Value Decomposition (SVD) for matrix factorisation (Billsus and Pazzani, 1998; Koren et al., 2009), probabilistic Latent Semantic Analysis (pLSA), or Latent Dirichlet Allocation (LDA) (Hofmann, 2003; Blei et al., 2003). Other approaches use probabilistic models where the recommendation task is modelled by user and item probability distributions (Wang et al., 2006b; Wang et al., 2008a), e.g. by learning a probabilistic model with a maximum entropy estimation (Pavlov et al., 2004; Zitnick and Kanade, 2004), Bayesian networks (Breese et al., 1998), and Boltzmann machines (Salakhutdinov et al., 2007). A graph-based model that exploits positive and negative preference data is proposed in (Clements et al., 2009). Besides, other Machine Learning techniques have also been proposed, such as artificial neural networks (Billsus and Pazzani, 1998) and clustering strategies (Kohrs and Merialdo, 1999; Cantador and Castells, 2006).

Memory-based approaches, on the other hand, make rating predictions based on the entire rating collection (Adomavicius and Tuzhilin, 2005; Desrosiers and Karypis, 2011). These approaches can be user- and item-based strategies. **User-based** strategies are built on the principle that a particular user's rating records are not equally useful to all other users as input for providing personal item suggestions (Herlocker et al., 2002). Central aspects to these algorithms are thus a) how to identify which neighbours form the best basis to generate item recommendations for the target user, and b) how to properly make use of the information provided by them. Typically, neighbourhood identification is based on selecting those users who are more similar to the target user according to a similarity metric (Desrosiers and Karypis, 2011). The similarity between two users is generally computed by a) finding a set of items that both users have interacted with, and b) examining to what degree the users displayed similar behaviors (e.g. rating, browsing and purchasing patterns) on these items. This basic approach can be complemented with alternative comparisons of virtually any user feature a system has access to, such as personal demographic and social network data. It is also common practice to set a maximum number of neighbours (or a minimum similarity threshold) to restrict the neighbourhood size either for computational efficiency, or in order to avoid noisy users who are not similar enough. Once the target user's neighbours are selected, the more similar a neighbour is to the user, the more her preferences are taken into account as input to produce recommendations. For instance, a common user-based approach consists of predicting the relevance of an item for the target user by a linear combination of her neighbours' ratings, weighted by the similarity between the target user and such neighbours.

In the following equations we present two versions of a user-based CF technique; in the first one rating deviations from the user's and neighbour's rating means are considered (Resnick et al., 1994), whereas in the second one the raw scores given by each neighbour are used (Aggarwal et al., 1999; Shardanand and Maes, 1995):

$$\tilde{r}(u,i) = \bar{r}(u) + C \sum_{v \in N_k(u,i)} sim(u,v)\big(r(v,i) - \bar{r}(v)\big) \tag{2.3}$$

$$\tilde{r}(u,i) = C \sum_{v \in N_k(u,i)} sim(u,v) r(v,i) \tag{2.4}$$

where C is a normalisation factor (different in each formulation) and $N_k(u,i)$ is a neighbourhood of size k, which may use information of the target item i. As stated in (Adomavicius and Tuzhilin, 2005), these techniques simply use a different function to aggregate the ratings from the neighbourhood. Note that in this case the utility function $g(u,i)$ is assumed to be equivalent to the predicted rating $\tilde{r}(u,i)$, although alternative transformations could be applied if required. Additionally, the similarity between two users is generally computed by means of the Pearson's correlation coefficient or the cosine similarity between the vectors representing each user's preferences (Adomavicius and Tuzhilin, 2005):

$$sim_{Pearson}(u,v) = \frac{\sum_i \big(r(u,i) - \bar{r}(u)\big)\big(r(v,i) - \bar{r}(v)\big)}{\sqrt{\sum_i \big(r(u,i) - \bar{r}(u)\big)^2} \sqrt{\sum_i \big(r(v,i) - \bar{r}(v)\big)^2}} \tag{2.5}$$

$$sim_{Cosine}(u,v) = \frac{\sum_i r(u,i) r(v,i)}{\sqrt{\sum_i r(u,i)^2} \sqrt{\sum_i r(v,i)^2}} \tag{2.6}$$

Note that these similarities are equivalent when the data is centered on the mean. Nonetheless, some authors have reported that the performance of recommenders based on Pearson's similarity is superior to that of cosine's (Breese et al., 1998; Herlocker et al., 1999). Moreover, other similarity measures and modifications on how the neighbours are selected and weighted have been proposed, either by modifying the similarity measure (McLaughlin and Herlocker, 2004; Ma et al., 2007), by using clustering methods to compute a user's neighbourhood (O'Connor and Herlocker, 1999; Xue et al., 2005), or by learning the best interpolation weights for rating prediction (Bell and Koren, 2007; Koren, 2008). Additionally, in the context of trust-based recommendation, the neighbours are weighted (and selected) according to their importance from the target user's point of view (O'Donovan and Smyth, 2005; Weng et al., 2006; Kwon et al., 2009; Hwang and Chen, 2007).

Item-based strategies, on the other hand, recognise patterns of similarity between the items themselves, instead of between user choices like user-based approaches do. In general item-based recommenders look at each item on the target user's list of chosen/rated items, and find other items that seem to be "similar" to that item (Shardanand and Maes, 1995; Sarwar et al., 2001). The item similarity is usually defined in terms of rating correlations between users, although cosine-based or probability-based similarities have also been proposed (Deshpande and Karypis,

2004). As stated in (Sarwar et al., 2001), adjusted cosine similarity has been proved to obtain better performance than other item similarities. This similarity subtracts the user's average rating from each co-rated pair in the standard cosine formulation:

$$sim(i,j) = \frac{\sum_u \big(r(u,i) - \bar{r}(u)\big)\big(r(u,j) - \bar{r}(u)\big)}{\sqrt{\sum_u \big(r(u,i) - \bar{r}(u)\big)^2}\sqrt{\sum_u \big(r(u,j) - \bar{r}(u)\big)^2}} \tag{2.7}$$

The rating prediction computed by item-based strategies is generally estimated as follows (Sarwar et al., 2001):

$$\tilde{r}(u,i) = C \sum_{j \in S_i} sim(i,j) r(u,j) \tag{2.8}$$

We have to note that the set of more similar items S_i is generally replaced by $\mathcal{I}_u$ – the set of items rated by user u – since for any other item, the rating provided by the user is assumed to be zero, and thus, it does not contribute to the summation.

2.2.3 Log-based recommenders

Different methods have been proposed to use implicit evidence of user preferences. The work of Oard and Kim (1998) represents one of the first attempts to exploit implicit user feedback to estimate future ratings in a recommender system. In general most of recent approaches have used formal models (generally probabilistic) in order to introduce implicit data for recommendation, although some approaches using ad-hoc techniques can be found. For example, Linden et al. (2003) use a simple vector representation, where each component represents purchased items, and recommendations are obtained by ranking each item according to how many similar users purchased it. Bernhardsson (2009) proposes a graph item-based algorithm that finds the closest tracks for a given track using probabilistic LSA (pLSA), and then derives the recommendations using heuristic and model-based probabilities, by brute force.

Additionally, several formal algorithms have been proposed to use implicit user feedback from log data: namely matrix factorisation, such as SVD (Hu et al., 2008) and pLSA (Das et al., 2007), and language models and other probabilistic approaches (Wang et al., 2006a; Wang, 2009; Wang et al., 2008b; Deshpande and Karypis, 2004). These algorithms aim to capture the user's preferences by considering the consumed (purchased, listened, browsed, etc.) items as evidence of positive relevance for the user. This fact often leads to binary models in which the number of times the user has consumed each item is not taken into consideration. Nevertheless, a benefit of using binary data is that it allows to better account for the fact that ratings are not missing at random – or equivalently, that users choose deliberately which items to rate (Marlin et al., 2007). Besides this a general concern about negative preferences has arisen. For instance, in (Lee and Brusilovsky, 2009), (Wang et al., 2008c), and

(Xin and Steck, 2011) the authors attempt to incorporate negative preferences inferred from implicit data.

Other authors have proposed different transformations in order to obtain explicit ratings from implicit feedback. The most naive approach is to make a correspondence between the existence of an item in a log record and a (frequency-independent) rating. For example, the algorithm proposed in (Ali and van Stam, 2004) cannot distinguish an explicit +1 rating from the rating inferred from implicit data. This is the same procedure that can be found in (Lee et al., 2008), but with other transformations based on the time in which an item was entered in the system and consumed.

In (Baltrunas and Amatriain, 2009), based on (Celma, 2010) and (Celma, 2008), the authors use a more elaborate mapping where the number of times a user listened to an artist (or track) is taken into account, in such a way that the artists (tracks) located in the 80-100% interquintile range of the user's playcount distribution receive a rating of value 5 (in a five point scale), the next interquintile range is mapped to a rating of value 4, and so on. This technique has also been used in other works, such as (Vargas and Castells, 2011). A similar technique is presented in (Jawaheer et al., 2010), where three methods are proposed in order to calculate the preference of a user for an artist: i) absolute, where the raw count of the number of times that artist has been played is used; ii) normalised, where the preference is inferred by the ratio between the counts for an artist and the total number of artists played by the user; and iii) logarithmic, similar to the previous one but smoothing the preference values by applying a logarithmic transformation.

Finally, Adams (2007) proposes a complete ad-hoc formula that takes several parameters into consideration, such as the number of times the current track has been played and skipped, the number of seconds when it was skipped, and the number of days since it was last played.

In conclusion, there is no definitive unique method for transforming implicit into explicit data. Moreover, it is unclear to what extent the mapping is reliable (Hu et al., 2008), since it inherently represents different information gathered from the user – for instance, negative preferences can only be fetched using explicit data. However, a recent study reported a strong relation between the amount of times users listen to an album, and the rating they provide to the album (Parra and Amatriain, 2011).

2.2.4 Social-based recommenders

Recommender systems that exploit social information, such as contacts and interactions between users, have started to be developed in recent years. We shall henceforth refer to this type of recommendation approaches as Social Filtering (SF) systems. Recommendations by SF approaches have the interesting property that they

are generally easier to explain than user-based CF approaches. Recommendations through friends are indeed easy to interpret by end-users. They also help dealing with the cold start problem, where new users are more difficult to provide recommendations for as long as it is not possible to reliably compute their similarity with other users for lack of data (Golbeck, 2006; Arazy et al., 2009).

Shepitsen et al. (2008) propose a personalisation approach for recommendation in folksonomies that relies on hierarchical tag clusters. The approach suggests the most similar items to the user's closest cluster by means of the cosine similarity measure. Other approaches focus on graph based techniques for finding the most relevant items for a particular user through hybrid networks involving people, items, and tags (Konstas et al., 2009; Clements et al., 2010). In this context alternative methods have been proposed to deal with data sparsity. Besides, prediction accuracy is improved by means of factor analysis based on probabilistic matrix factorisation, employing both the users' social network information and rating records (Ma et al., 2008). Ma et al. (2009) combine the recommendations made by trusted friends with those generated by a matrix factorisation algorithm. In a similar way, Jamali and Ester (2009) propose to perform a random walk on the trust network, considering the similarity of users in the termination condition; then, the top rated items are recommended. Both approaches are competitive in cold start situations.

Complementarily, simpler algorithms (referred to as "pure" social recommenders henceforth) have also been proposed in (Liu and Lee, 2010) and (Bellogín et al., 2012). In (Liu and Lee, 2010) an adaptation of the user-based CF technique is proposed, where the set of nearest neighbours is replaced by the target user's (explicit) friends. That is:

$$N_k(u,i) = \{v \in \mathcal{U}: v \text{ is friend of } u\} \tag{2.9}$$

This lets easily incorporate social information into the CF prediction equation, building a straightforward technique that enables a direct interpretation of the suggestions, namely those items recommended by friends. Similarly, based on a recommender proposed in (Barman and Dabeer, 2010), where the items suggested to a user are the most popular among her set of similar users, in (Bellogín et al., 2012) we proposed a friends' popularity recommender that suggests the target user those items most popular for her set of friends. A score is generated by transforming the item position with the following equation, once a ranking has been generated using the score $f(u,i)$:

$$\begin{aligned} f(u,i) &= |\{v \in \mathrm{U}: v \text{ is friend of } u \text{ and } rat(v,i) \neq \phi\}| \\ g(u,i;N) &= 1 - \frac{pos(u,i)}{N} \end{aligned} \tag{2.10}$$

where $pos(u,i)$ represents the position of item i in the top-N recommended list for user u. We may trim the returned list at some level N, or assume N to be exactly the

length of the generated recommendation list. Obviously, the computed scores cannot be interpreted as ratings, but as a utility or ranking score. In (Bourke et al., 2011) Bourke and colleagues also make use of the social graph of a user to build the neighbourhood, analysing the perceived trust, which is found to be higher when the users are given the opportunity to manually select the neighbourhood to be used for computing recommendations.

Ben-Shimon et al. (2007) propose a recommendation approach based on the distances between users in the social graph. The approach uses Breadth-First Search to build a social tree for each user u denoted as $X(u, L)$, where L is the maximum number of levels taken into consideration in the algorithm, and K is an attenuation coefficient of the social network that determines the extent of the effect of $d(u, v)$, that is, the impact of the distance between two users in the social graph (e.g. by using an algorithm that computes the distance between two nodes in a graph, such as the Dijkstra's algorithm (Dijkstra, 1959). Hence, when $K = 1$ the impact is constant, and the resulting ranking is sorted by the popularity of the items. Furthermore, for that value of K, no expansion is applied and only directly connected users are involved in the score computation. Once the value of K is chosen, a rating score is generated according to the following equation:

$$g(u, i) = \sum_{v \in X(u,L)} K^{-d(u,v)} r(v, i) \tag{2.11}$$

An alternative way of introducing social information into a recommender system is by the so called trust-based recommendation approaches, even though social relationships and trust relationships do not model exacly the same concept (Ma et al., 2011). Trust-aware recommenders, in contrast with those defined in Section 2.2.2, make use of trust networks, where users express a level of trust on other users (Massa and Avesani, 2007a). These recommenders need a trust network and a trust metric, so that trustworthiness of every user can be computed. Depending on the available data, we would have to infer a plausible trust network, from the information we already know about users, such as social interactions among users or explicit trust relations. Typically, uniform trust values from each user are assumed, since no distinction can be made among a user's contacts. For example, a user with 4 friends would have a trust level of 0.25 for each friend, whereas a user with 2 friends would have such trust level of 0.5.

Once the trust network is defined, either explicitly or implicitly, we can set different definitions for the trust metrics depending on whether they are global (a global reputation value is calculated for each user) or local (a trust score is computed between a source user on a target user). Social-based trust metrics make use of explicit trust networks of users, built upon friendship relationships (Massa and Bhattacharjee, 2004) and explicit trust scores between individuals in a system (Ma et al., 2009; Wal-

ter et al., 2009). These metrics and, to some extent, their inherent meanings, are different with respect to rating-based metrics. Nonetheless, Ziegler and Lausen (2004) conduct a thorough analysis that shows empirical correlations between trust and user similarity, suggesting that users tend to create social connections with people who have similar preferences. Once such a correlation is proved, techniques based on social-based trust are applicable.

Golbeck and Hendler (2006) propose a metric called TidalTrust to infer trust relationships by recursive search. Inferred trust values are used for every user who has rated a particular item in order to select only those users with high trust values. Then, a weighted average between ratings and trust provides the predicted ratings. A similar algorithm is used in (Walter et al., 2009), where the prediction is based on the ratings of the trusted neighbours. Different integrations of the trust metric into the recommendation process are proposed in (Massa and Avesani, 2007a), along with two metrics: PageRank and MoleTrust. The former is considered as a global metric based on the well-known PageRank algorithm (Brin and Page, 1998); the latter is a local metric based on a Depth-First graph traversal algorithm with an adjustable trust propagation horizon (Massa and Avesani, 2007a).

Finally, as proposed in (Massa and Avesani, 2007a), two ways to incorporate these trust metrics into the recommendation models can be considered. The first one makes use of the trust metric instead of the similarity metric in the standard user-based CF formula. The second one, on the other hand, computes the average between Pearson's similarity and the trust metric when both values are available; otherwise it uses the only available value, thus overcoming the natural data sparsity. Recently, Guo et al. (2012) propose to merge the ratings from the trusted neighbours in order to decrease sparsity prior to the computation of the predicted rating.

2.3 Combining recommender systems

The proliferation of new recommendation strategies is giving rise to an increasing variety of available options for the development of recommender systems. Research in Machine Learning has long shown that the combination of methods usually achieves better results than each method separately, which is also true in Recommender Systems – the Netflix prize has been a paradigmatic example of this, where all the top classified teams used large recommender ensembles, which can be considered as a case of hybrid filtering approaches.

In such a hybrid approach the most important decision is how to combine the information. First, however, it has to be decided what kind of information is going to be used in the ensemble. The standard approach in the literature is to combine CBF and CF recommenders, overcoming the sparsity and restricted feature problems of individual recommenders, as we shall see in the next section. However, other types

and sources of information, such as social contacts and timestamps, have been recently integrated into the classical formulation of standard recommendation techniques.

In (Burke, 2002) a detailed taxonomy of hybrid recommender systems is presented, classifying existing approaches into the following types:

- **Cascade**: the recommendation is performed as a sequential process in such a way that one recommender refines the recommendations given by the other.
- **Feature augmentation**: the output from one recommender is used as an additional input feature for other recommender.
- **Feature combination**: the features used by different recommenders are integrated and combined into a single data source, which is exploited by a single recommender.
- **Meta-level**: the model generated by one of the recommenders is used as the input for other recommender. As stated in (Burke, 2002): "this differs from feature augmentation: in an augmentation hybrid, we use a learned model to generate features for input to a second algorithm; in a meta-level hybrid, the entire model becomes the input."
- **Mixed**: recommendations from several recommenders are available, and are presented together at the same time by means of certain ranking or combination strategy.
- **Weighted**: the scores provided by the recommenders are aggregated using a linear combination or a voting scheme.
- **Switching**: a special case of the previous type considering binary weights, in such a way that one recommender is turned on and the others are turned off.

The use of a specific type of hybrid recommendation method depends on the final application, but, more importantly, on the type of recommenders being combined. Indeed, Burke (2002) presents an analysis of the possible hybrids, their limits and incompatibilities, based on a representative subset of the recommendation techniques available nowadays. Moreover, the author notes that some combinations turn out to be redundant because of the symmetry in the hybridisation process for some of the techniques listed above: weighted, mixed, switching, and feature combination. Incompatible combinations arise for the feature combination and meta-level techniques, where in some situations one of the recommenders is not able to use the model or the features generated by the other recommender.

Burke (2002) focuses on hybrid techniques where the information being combined consists of ratings (to be used by CF recommenders), content features (to be used by CBF, knowledge-based, and utility-based recommenders), and demographic

information. In the following, we survey hybrid recommenders where inputs in the form of social information, collaborative (either ratings or logs), and content features have been used. In the next section we analyse the limitations of these types of techniques, together with the benefits that hybridisation may bring.

Among these possibilities, the most popular combination (probably due to its inherent interest) consists of blending content-based and collaborative filtering recommenders. In fact, one of the first proposed hybrid techniques (Balabanovic and Shoham, 1997) makes use of these two recommendation approaches by suggesting items similar to the user's profile (using content-based profiles) and those items highly rated by a user with a similar profile, by means of a collaborative formulation where neighbours are determined using a content-based similarity. In a similar way, Pazzani (1999) combines content-based, demographic, and collaborative information using two techniques: by plugging content-based similarity functions into collaborative methods and by combining the final rankings produced by each recommender seeking a consensus, that is, how many systems recommend each item, and in what ranking position are both considered to build the final ranking.

In (Rojsattarat and Soonthornphisaj, 2003) a technique to derive a less sparse pseudo rating matrix is proposed. More specifically, a pseudo user-ratings vector for every user is built with the item ratings provided by user u when available, or the ratings predicted by a content-based recommender otherwise. Gunawardana and Meek (2009) propose to combine content and collaborative information in a coherent manner by using a specific type of probabilistic models, Boltzmann machines. These models let encoding the above sources of information as features, and then, weights are learned to reflect how each feature helps predict the user ratings. Other probabilistic models for combining these sources of information have been proposed in (Yu et al., 2003), where a hierarchical Bayesian model learns a prior distribution by using probabilistic Support Vector Machines (SVMs).

Also from a machine learning perspective, an ensemble technique known as stacking is used in (Bao et al., 2009), which learns multiple classifiers for different prediction levels: at the first level, the recommendation techniques (a user-based CF, an item-based CF, and a CBF algorithm) output a rating prediction, which may be combined at the second level by a meta-learning algorithm that uses the predictions as meta-features.

Alternatively, the same model can also be combined with itself using different parameter values. For instance, in (Gantner et al., 2010) different factor models are combined, where each model may have different regularisation parameters, stop conditions and dimensionality values. Jahrer et al. (2010) combine a set of diverse CF recommenders by using different machine learning techniques such as linear regression, neural networks, and a combination of bagging and gradient boosting trained with decision trees.

Furthermore, hybrid models have been proposed combining social and content or collaborative information. In (Konstas et al., 2009) a Random Walk algorithm is applied to a graph comprising of tags, social information, and implicit feedback from users. In this way, more elaborate patterns and rules than the standard correlation measure between users are provided. A similar approach can be found in (Liu et al., 2010) for tag recommendation. The approach defined in (Clements et al., 2010) improves search and recommendation by combining tags and ratings, and integrating them into the user's social network also using a Random Walk algorithm. Hotho et al. (2006) exploit social information along with tag content by converting a folksonomy into a graph and then applying a weight-spreading algorithm for folksonomies called FolkRank (similar to the well-known PageRank algorithm (Brin and Page, 1998). Finally, Jamali and Ester (2009) combine information from the social network (in terms of trust between users) and ratings (collaborative) in order to alleviate the cold-start problem. In that work, the authors make use of the collaborative information as a termination condition of a random walk performed over the trust network by considering the similarity of users; additionally, the authors also combine those two sources for computing two sets of neighbours and, then, merging the items produced from those similar users.

2.4 General limitations of recommender systems

Each type of recommendation technique has strengths and weaknesses, well known in the field. We have already noted the main characteristics of each technique, which are largely dependent on the source of information being used. In this section we analyse the main limitations of each technique. Furthermore, although ideally hybrid recommendation techniques would overcome the problems of the combined techniques, there are certain limitations that are inherent to the recommendation problem, and thus, have to be addressed indepedently. Besides, by combining different methods, additional problems, along with more limitations, arise.

2.4.1 Limitations of single recommendation algorithms

In this section we describe the different limitations identified in the literature for the main types of recommenders described in the previous sections.

The main limitations of CBF approaches are the following (Adomavicius and Tuzhilin, 2005; Pazzani and Billsus, 2007; Cantador, 2008):

- **Restricted content analysis**. Content-based recommendations depend on the available features explicitly associated with the items. These features should be in a form that can be automatically parsed by a computer, or manually ex-

tracted somehow, which, depending on the domain, could be unfeasible or very difficult to maintain.

- **New user**. A user has to show some preference (ratings) for a sufficient number of items before a recommender can build a reliable content-based user profile.
- **Overspecialisation**. Since content-based recommenders only retrieve items similar to what the user has already rated, recommendations are very similar and, probably, well known to the user, providing little (or none) novelty from the user perspective.
- **Portfolio effect**. Related to the previous limitation, sometimes the recommended items are very similar among them, leading to a set of insufficiently diverse or too redundant item suggestions.

CF approaches have the following general weaknesses:

- **Rating data sparsity**. The number of observed user-item interactions (e.g. ratings) is generally very small compared to the number of all user-item pairs. This fact may cause CF algorithms to produce unreliable recommendations, since they have been inferred from insufficient data.
- **Grey sheep**. Since collaborative recommendations rely on the tastes of similar people to suggest new items, when a user has very specific or unusual preferences, it will be more difficult for the system to find good neighbours, and thus, to recommend interesting items.
- **New item**. Until a new item has been rated by a substantial number of users, a recommender system may not be able to recommend it; hence, popular items tend to have advantage in this kind of systems.
- **New user**. Like in the content-based approaches, until a user has not provided with enough ratings, the system is unable to recommend her interesting unknown items.

In addition to these weaknesses, log-based CF techniques have other limitations. Specifically, they are not able to capture negative preferences from the user since unobserved items cannot be inferred as unliked items (they may represent items unknown for the user). In contrast, it is easier to capture this type of information because it is less expensive for the user than providing a rating. Furthermore, although the problem of ratings missing not at random is ubiquitous and inherent to any recommender system – since users typically rate only a small fraction of the available items – log-based recommenders, and more specifically, the binary data inferred from these implicit interactions, have the theoretical advantage that they are able to exploit implicit preferences since the items observed by the users are deliberately

Problem	Description	CBF	CF	SF
Restricted content analysis	Items to be recommended must have available data related to their features. This data is often unavailable or incomplete.	Yes	No	No
Overspecialisation	CBF recommenders are trained with the content features of the items. All the recommended items are similar to those already rated.	Yes	No	No
Portfolio effect	CBF recommenders suggest items based on the item features. An item is recommended even if it is too similar to a previously rated item.	Yes	No	No
New user	A user has to rate enough items in order to infer their preferences. When a new user enters into the system she has no ratings.	Yes	Yes	No
New item	Items have to be rated by a substantial number of users for being recommended. Recently incorporated items have none or insufficient ratings.	No	Yes	No
Grey sheep	A user has to be similar to others in the community to receive recommendations. Users whose tastes are unusual may not receive useful suggestions.	No	Yes	No
Rating data sparsity	Ratings are used to train user and item models. The number of available ratings is usually small.	No	Yes	No
Social sparsity	Social connections are used to build social models. The number of connections per user may be small.	No	No	Yes
New social connection	A user has to be connected with someone else to receive recommendations. When the user is new, she may not have any social connections.	No	No	Yes
Social similarity	Similarity based on social connections is used in SF recommenders. Two users socially connected may or may not have interests in common.	No	No	Yes

Table 2.1. List of common problems in CBF, CF, and SF systems.

selected by them. Thus, potentially more useful information about the user can be gathered (Koren and Bell, 2011).

Regarding CF in general, memory-based approaches achieve lower performance than model-based approaches. However, as stated in (Desrosiers and Karypis, 2011) and (Koren and Bell, 2011), good prediction accuracy does not guarantee an effective and satisfying user experience. Hence, the main advantages of memory-based recommenders are simplicity, justifiability, efficiency, and stability.

Finally, SF approaches have other limitations, as we describe next:

- **Social sparsity**. Social filtering methods need that every user has to be connected through at least one contact in the social network to be able to produce recommendations, which is not a typical situation for most of the users in a system.

- **New social connection**. Recommendations may get biased if a user has a very small social network, up to the point that if she has only one connection, every social recommendation would be generated based on the activity of just one user.
- **Social similarity**. The fact that two users share some kind of connection in a social network does not necessarily mean that these users have similar interests. Although some studies have shown some correlation between both (Ziegler and Lausen, 2004), the misuse of this similarity may lead to bad recommendations, even though the user's experience may be improved in terms of diversity and serendipity.

As a summary, Table 2.1 shows a comparison of the main limitations for the three types of recommendation algorithms described.

2.4.2 Limitations of recommender ensembles

As we have explained in the previous section, each type of recommendation – CBF, CF, and SF – has its own limitations. Hybrid filtering systems are normally out of this analysis since they compensate the shortcomings of one approach by the strengths of the other, unless both suffer from the same problem, as in the case of a new user when we combine CBF and CF approaches.

In general, hybrid recommenders are useful for alleviating the individual limitations of the combined recommenders. However, recommender ensembles do not always outperform individual recommenders. Van Setten (2005) describes the situation where all recommenders produce predictions that are "on the same side of the rating the user would give, all too low or all too high." In this situation the ensemble would be less accurate than the best individual recommender. Additionally, when a particular recommender always obtains superior/inferior performance than the rest of recommenders in the ensemble, the corresponding recommender ensemble may not be useful. In that case the underperforming recommenders are useless from the beginning, whereas the over performing one should be used alone, and there is no point in combining them.

The above issues assume that a particular metric is aimed to be optimised. Needless to say that the use of multiple recommenders may provide better results with respect to other evaluation properties, such as diversity, novelty, and serendipity, probably at the expense of a lower quality or accuracy of the recommendations (Shani and Gunawardana, 2011).

Additionally, the recommender ensemble problem is similar to that of combining classifiers in the Machine Learning field, a well studied research problem in that community (Kuncheva, 2004). In such context, the diversity in the classifier outputs is known to be a requirement for the combination to be effective. Thus, whenever

some classifiers in an ensemble fail, these errors should be made on different objects, in order to let a final performance improvement with the ensemble. In (Kuncheva, 2004) and (Kuncheva and Whitaker, 2003) Kuncheva and Whitaker present a number of diversity metrics, and analyse the relation of such metrics with respect to the accuracy of a recommender ensemble, although they do not provide a systematically formulation of such relation. As stated by the authors, the problem of classifier combination and its relation with diversity may rise from the underlying meaning of diversity: whether it is a characteristic of the set of classifiers, or it is more complex and a mixture of the characteristics of the set of classifiers, the combiner, and the errors.

Finally, although many different hybrid filtering approaches have been proposed for recommender systems, there is a lack of a similar analysis to the one performed in Machine Learning, where the different characteristics of the datasets and individual recommenders have been investigated and assessed. A preliminary analysis was performed in (Bellogín et al., 2010), but an in-depth and larger-scale study would benefit the community, considering different evaluation perspectives and, probably, borrowing from the Machine Learning research on this topic.

2.5 Summary

Along over two decades of research and commercial development, recommender systems have proved to be a successful technology to overcome the information overload that burdens users in modern online media. The inherent possibility of dealing with diverse sources of information, such as the content of the items and the collaborative and social interactions among users and between users and a system, has enabled the development of rich strategies based on each of these evidences, deriving content-based, collaborative, and social filtering recommendation approaches. Furthermore, as each particular type of recommendation technique has its own limitations and weaknesses, hybrid strategies have been proposed that combine the suggestions generated by different techniques in different ways. The success of ensemble approaches has been recently evidenced in the Netflix prize, where the top classified teams used different forms of recommender ensembles.

There are, however, general limitations remain unsolved, and are still considered as open research problems in the field. We have mentioned the sparsity of the information (either in the forms of content-based attributes, collaborative ratings, and social connections), and the new user problem, but other problems, not related to an specific recommendation technique, have been identified in the literature, and deserve special attention by themselves, such as the need of contextualisation, the explanation of the recommendations, and the efficiency in computing recommendations (Adomavicius and Tuzhilin, 2005).