Available online at www.sciencedirect.com
ScienceDirect
journal homepage: www.elsevier.com/locate/cosrev

Survey

A survey of active learning in collaborative filtering recommender systems

Mehdi Elahi[a,*], *Francesco Ricci*[b], *Neil Rubens*[c]

[a] *Politecnico di Milano, Milan, Italy*
[b] *Free University of Bozen-Bolzano, Bozen-Bolzano, Italy*
[c] *University of Electro-Communications, Tokyo, Japan*

ARTICLE INFO

Article history:
Received 14 July 2015
Received in revised form
9 May 2016
Accepted 10 May 2016

Keywords:
Recommender systems
Collaborative filtering
Active learning
Rating elicitation
Preference elicitation
Cold start
New user
New item

ABSTRACT

In collaborative filtering recommender systems user's preferences are expressed as ratings for items, and each additional rating extends the knowledge of the system and affects the system's recommendation accuracy. In general, the more ratings are elicited from the users, the more effective the recommendations are. However, the usefulness of each rating may vary significantly, i.e., different ratings may bring a different amount and type of information about the user's tastes. Hence, specific techniques, which are defined as "active learning strategies", can be used to selectively choose the items to be presented to the user for rating. In fact, an active learning strategy identifies and adopts criteria for obtaining data that better reflects users' preferences and enables to generate better recommendations.

So far, a variety of active learning strategies have been proposed in the literature. In this article, we survey recent strategies by grouping them with respect to two distinct dimensions: *personalization*, i.e., whether the system selected items are different for different users or not, and, *hybridization*, i.e., whether active learning is guided by a single criterion (heuristic) or by multiple criteria. In addition, we present a comprehensive overview of the evaluation methods and metrics that have been employed by the research community in order to test active learning strategies for collaborative filtering. Finally, we compare the surveyed strategies and provide guidelines for their usage in recommender systems.

Contents

1. Introduction 2
2. Recommendation techniques 3
3. Collaborative filtering 3
   3.1. Neighbourhood-based models 4

* *Corresponding author.*
E-mail address: mehdi.elahi@polimi.it (M. Elahi).
URLs: http://www.polimi.it (M. Elahi), http://www.unibz.it (F. Ricci), http://www.uec.ac.jp (N. Rubens).

http://dx.doi.org/10.1016/j.cosrev.2016.05.002
1574-0137/

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

3.2. Latent factor models ... 4
3.3. Cold start problem in collaborative filtering ... 4
4. Active learning ... 5
5. Active learning in collaborative filtering ... 6
5.1. Non-personalized active learning ... 7
5.1.1. Non-personalized single-heuristic strategies ... 7
5.1.2. Non-personalized combined-heuristic strategies ... 9
5.2. Personalized active learning ... 9
5.2.1. Personalized single-heuristic strategies ... 10
5.2.2. Personalized combined-heuristic strategies ... 12
6. Evaluation of active learning for collaborative filtering ... 13
6.1. Offline vs. online evaluation ... 13
6.2. Single user focussed evaluation ... 14
6.3. Natural acquisition of ratings ... 14
7. Comparison of active learning techniques ... 15
7.1. Performance summary ... 15
7.2. Practical guidelines ... 17
8. Conclusions and future work ... 17
References ... 18

1. Introduction

This article surveys the state-of-the-art of active learning for collaborative filtering recommender systems. *Active Learning* in recommender systems tackles the problem of obtaining high quality data that better represents the user's preferences and improves the recommendation quality. This is done by identifying for each user a set of items contained in the system catalogue which have not been rated yet by the user, and by asking the user to rate them. The ultimate goal is to acquire additional ratings that will enable the system to generate better recommendations.

It is worth noting that users are typically not interested and are reluctant to rate items: this activity represents a cognitive cost for the user. For that reason it is important to carefully design an active learning strategy for identifying a small set of items to rate, whose ratings will convey to the system valuable information about users' preference, that is, ratings that will most improve the system generated recommendations.

Active learning is a subfield of machine learning [1–3]. While several machine learning tasks have been studied and a wide range of techniques have been already proposed and applied [4,1,5–8], the application of these techniques often requires a significant amount of high quality data that are not always easily available [9]. Active learning is precisely motivated by these situations where training data is expensive to obtain. In this case, active learning specifies how the data points that could better help the system to perform its task should be selected [10].

So far, several active learning strategies have been proposed and evaluated. We are here interested in those that have been applied to collaborative filtering recommender systems. These strategies have different features, and implement various heuristics for selecting the items to be presented to the users to rate. Instead of directly minimizing the system prediction error, by using heuristic, they try to improve other system properties that influence the system error. For instance, by acquiring ratings for popular items (as the popularity-based strategy does) the system tries to simply acquire more ratings (since many users are familiar with the popular items and can rate them). However, by adopting this heuristic the system may also acquire too many high ratings, as popular items tend to be rated high, hence the system may be also erroneously biased to predict higher ratings [11,12].

In this article, we classify all the important active learning strategies that we have found in the literature with respect to two dimensions that are descriptive and discriminative:

- **Personalization**: that describes to what extent the selection of items for the user to rate is adapted to the user's characteristics. We have *non-personalized*, and *personalized* strategies. In a non-personalized strategy the users are requested to rate the same items. Personalized strategies, on the other hand, ask each user to rate specific items. Personalization is an important aspect in active learning since the users have different tastes, preferences and experiences, hence, the usefulness of the various ratings could vary greatly from user to user. Selecting the items to rate while taking into account the preferences of each user, may provide a more pleasant experience to the user (e.g. by presenting them with items that they can actually rate), and at the same time may be more informative for the system.
- **Hybridization**: this dimension describes whether the strategy takes into account a single heuristic (criterion) for selecting the items for the users to rate or it combines several ones in order to create a more effective strategy. In this regard, the strategies can be classified into *single-heuristic* (or *individual*) and *combined-heuristic* (or *combined*) strategies. Single-heuristic strategies implement a unique item selection rule and select items only based on that. Combined-heuristic strategies hybridize single-heuristic strategies by aggregating and combining some of them, hence utilizing multiple item selection rules in order to better identify the more useful items to rate.

These dimensions where first identified in a previous short article [13]. Here, we illustrate more strategies and we provide

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

further details for all of them. Moreover, we add a new section that discusses evaluation methodologies for active learning and we provide summary comparisons of the various strategies presented here.

The main contributions of this article are: (a) novel dimensions for classification of active learning strategies in collaborative filtering, i.e., personalization and hybridization; (b) a comprehensive analysis and classification of more than 25 strategies (see Fig. 4) as well as the brief description of their potential pros and cons; (c) a sub-classification of the strategies for an even better and more informative description and discrimination, e.g., static combined-heuristic or adaptive combined-heuristic strategies; (d) a discussion of the evaluation methods and metrics employed by the research community.

The remainder of the article is structured as follows: Section 2 briefly reviews the main recommendation techniques; Section 3 focuses on collaborative filtering techniques and discusses some open issues in this area; Section 4 describes a general model of active learning; Section 5 concentrates on the application of active learning to collaborative filtering and describes in detail several active learning strategies; Section 6 illustrates and discusses evaluation methods for active learning in collaborative filtering; Section 7 provides a summary comparison of the performances of the reviewed strategies; finally, Section 8 discusses a number of new topics that may need to be studied further and concludes this survey article.

2. Recommendation techniques

Choosing the right products to consume or purchase, e.g., movies to watch, nowadays can be challenging: there are often thousand of appealing choices that we can reach just with one-click. While this increasing number of choices provides more chances to consumers to find the products that satisfy their special needs, it may also overwhelm them with an excess of choice [14]. *Recommender systems* (RSs) tackle this problem by offering to the user personalized suggestions for items (products or services) that match the user's needs and wants better than mainstream products [15–18]. Getting recommendations from others by words, reference letters, media reports, or guides, are common social practices. Recommender systems enhance this social process by helping people to explore or search for available items, such as, books, articles, webpages, movies, music, restaurants, or even jokes. Recommender systems suggest to users items that are judged to be desirable based on the analysis of their preferences [19–21]. Several approaches have been proposed and used for recommendation generation [22,23,21,17]:

- *Content-based* [24–26]: suggests items of interest based on their distinguishing features. For instance, news recommender systems consider the terms contained in news articles and recommend to user news articles with terms contained in the articles that the user have read and liked before.
- *Demographic* [27–29]: generates recommendations by identifying users that are similar to the target one with respect to demographic information. This approach tries to categorize the users by their personal attributes and makes recommendations based on their demographic classes: similar users receive similar recommendations.

	K-PAX	Life of Brian	Memento	Notorious
Alice	4	3	2	4
Bob	∅	4	5	5
Cindy	2	2	4	∅
David	3	∅	5	2

Fig. 1 – An example of a rating matrix [23].

- *Utility-based* [30,31,29]: the recommendations are those that maximize the learned utility function of the user. Items are scored according to their utility and those with the highest predicted utility are suggested to the user.
- *Knowledge-based* [32,29,33]: suggests items that are inferred – with a specific reasoning process – to match the users' needs and preferences. Knowledge-based approaches leverage explicit knowledge about how a particular item meets the user's need. Making it possible to reason about the relationship between user's needs and recommendations.
- *Collaborative Filtering* [34,35,23]: decides what items to recommend to the target user by observing the ratings of similar users. Similarity is computed by looking just at the users ratings: users are similar if they co-rate similarly the items. Collaborative filtering recommends to the target user items that are highly rated by similar users.
- *Hybrid approaches* [36,22,37]: combine two or more approaches, among those quoted above, in order to cope with the specific limitations of a single solution.

All the above mentioned recommendation techniques make use of large data sets, describing the preferences of the users, their characteristic features and the features of the item being recommended. Recommender systems need to acquire this information before computing recommendations for items to suggest. However, collecting this information poses an important issue: what information is more useful to achieve the best performance of the recommender system?

In this article we address this problem focusing on collaborative filtering, and we will present techniques that are aimed at identifying which preferences, i.e., ratings, should be elicited by the system to generate more effective recommendations.

3. Collaborative filtering

As mentioned before, a collaborative filtering (CF) recommender system uses ratings for items provided by a collection of users. It recommends items that the target user has not yet considered but will likely enjoy [34,35]. Ratings are stored in a $m \times n$ matrix where m is the number of users and n is the number of items (Fig. 1). The rows of this matrix store the users' ratings and the columns the items' ratings. If a user has rated an item, the corresponding entry will be assigned to the given rating. This is the input data to a collaborative filtering system. When a new user registers to the system a new empty row is added to this matrix. Similarly, when a new item is added to the catalogue a new empty column is added.

A collaborative filtering system computes recommendations by exploiting relationships and similarities between users and between items. These relations are derived from

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

the interactions of the users with the items managed by the system. For instance, *Amazon*[1] or *Last.fm*[2] users browse and search items to buy or to listen to. They read users' comments and browse recommendations computed by the system using the access logs, and the ratings of the users' community.

Utilizing these ratings provided by a collection of users, collaborative filtering recommends items which the target user is estimated to be interested in. In order to do that, the system predicts the target user's ratings for the items that have not been rated yet, hence the system has no direct knowledge to determine whether the user likes or dislikes them. According to the predicted ratings, the items are ranked, and those with the highest predicted ratings are recommended to the user.

While the accuracy of a collaborative filtering depends on the specific rating prediction algorithm that is adopted (described in Section 3.1), still the rating data that is used by the system can significantly influence its performance. We will discuss this aspect after having discussed two popular rating prediction algorithms: neighbour-based and latent factor models.

3.1. Neighbourhood-based models

Neighbourhood-based techniques are either user- or item-based [35]. User-based approaches compute rating predictions for a target user by using two sets of data: the ratings of the target user and the ratings of other like-minded users, i.e., users with similar patterns of ratings. An item's rating prediction for the target user is then based on how the item was rated by the users similar to the target one. There are several ways for computing rating predictions in user-based approaches. A popular one is shown in the equation below, where $r_{ui} \in \{1, \ldots, 5\}$ is the known (five stars) rating of the user u for the item i, and $\hat{r}_{ui}$ is the system prediction of the rating of u for i:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in N_i(u)} sim(u, v)(r_{vi} - \bar{r}_v)}{\sum_{v \in N_i(u)} |sim(u, v)|}. \quad (1)$$

Here $\bar{r}_u$ denotes the average rating of user u, $sim(u, v)$ is the similarity of the users u and v, and $N_i(u)$ is a set of users similar to user u (neighbours) who rated item i. The user-to-user similarity can be computed using different approaches. A popular one is *Pearson correlation* [39]:

$$sim(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2 \sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}} \quad (2)$$

where I_{uv} is the set of items co-rated by users u and v.

3.2. Latent factor models

Latent factor models offer a different approach to rating prediction. A popular example of latent factor models is Matrix Factorization [40,41,34,34,42]. Matrix factorization learns a vector of latent features (called factors) for each user and item. Each entry in the factor vector reflects how well an item or a user possess a particular latent aspect. For example, when the products are movies, the factors might represent genres, i.e., how much a movie is a comedy, a drama, or an action, or even other uninterpretable dimensions. User factor vectors measure the preference of the user for each factor. The goal of a factorization algorithm is to split the original rating matrix R into two matrices S and M in such a way that their product approximates the original matrix and predicts the missing ratings in the matrix:

$$R \approx SM^T \quad (3)$$

where S is $|U| \times F$ matrix, and M is $|I| \times F$ matrix. F is the number of factors and it is a parameter that must be optimized. Regularized matrix factorization is a technique for computing the above mentioned decomposition and it was proposed by Simon Funk [43]. The factor matrices are computed by minimizing the sum of the prediction errors, $e_{ui} = r_{ui} - \hat{r}_{ui}$, on the training rating data and by using regularization and early stopping in order to avoid overfitting of the training data [40,41,34,42]:

$$s_{uf} = s_{uf} + \gamma\,(e_{ui}\, m_{if} - \lambda\, s_{uf}) \quad (4)$$

$$m_{if} = m_{if} + \gamma\,(e_{ui}\, s_{uf} - \lambda\, m_{if}) \quad (5)$$

where γ and λ are meta-parameters used for controlling the speed of factor learning and the regularization respectively. When the factors are learned the computed rating predictions are trimmed to be in the rating range from 1 to 5.

3.3. Cold start problem in collaborative filtering

A major problem of collaborative filtering is the *cold start*, i.e., when the system has not yet acquired enough ratings to generate reliable recommendations. The most typical situations are when the system is unable to accurately recommend existing items to a new user (*new user* problem) or to recommend a new item to the existing users (*new item* problem) [23,44–46].

While these are the most common cold-start situations, there is a related issue, i.e., the *Sparsity* of the rating data. Sparsity measures the inverse of the density of the available ratings, and it is defined as:

$$1 - \frac{\#\text{ of available ratings}}{\#\text{ of all possible ratings}}. \quad (6)$$

Sparsity is typically very close to 1 (%100) in real recommender systems because users rate a very small percentage of the available items. For this reason it is challenging for the system to generate accurate recommendations.

There are various approaches for solving this problem and improving collaborative filtering. One of them consists of using *hybrid* recommendation techniques, e.g., to combine collaborative and content-based filtering [47,22,48,23,17]. It is also possible to address the cold-start problem with cross-domain recommendation techniques. These techniques aim at improving the recommendations in a target domain by making use of information about the user preferences in an auxiliary domain [49]. In this case the knowledge of the preferences of the user is transferred from an auxiliary domain to

[1] http://www.amazon.com [38].
[2] http://www.last.fm.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

the target domain in order to build there more complete and accurate user models and item recommendations. For example, using the knowledge of ratings and tags assigned by the users to items in an auxiliary domain it is possible to better learn the latent factors in the target domain [50].

Another possibility is to complement the given rating data with other sources of information about the items. For example, in multimedia recommender systems, it has been shown that audio–visual features, e.g., variation of colour, camera and object motion, and lighting, can be automatically extracted from movies in order to solve the cold start problem and to effectively generate relevant recommendations [45,46]. In a food recommender, one can exploit the information brought by tags assignments performed by users [51,52]. And to recommend restaurants it is possible to use information about the restaurant cuisine or location [32,23,17].

An alternative group of approaches consists of exploiting additional information about the users, such as their personality, in order to improve the estimation of the user's preferences [53]. Studies conducted on user personality characteristics support the conjecture that it may be useful to exploit this information in collaborative filtering [54–56]. As another example in this group of approaches, we may cite [57] where the authors propose a graph-based method to solve the cold start problem in news recommendation domain. This method uses referral link of the new user as well as the currently visiting page in order to generate recommendations for new users.

In addition to the above mentioned approaches, *active learning*, which is the topic of this article, has shown promising results in dealing with the cold start problem. Active learning tackles this problem at the root, by focusing on obtaining more (high quality) data that better represents a user's preferences and improves the most the performance of the system. In the next section, we will first discuss the general characteristics of active learning, and then we will focus on the specific application of active learning to collaborative filtering.

4. Active learning

Machine learning deals with computational methods that enable a system to improve its performance by exploiting data [1–3]. Several techniques have been proposed and applied in machine learning [4,1,5–8]. However, for applying these techniques often a considerable amount of high quality data is required [9]. Such data can be obtained either passively or actively. When the training data is provided in advance, or the learner makes no effort to obtain new data, learning is called *passive*. Fig. 2 (top) illustrates graphically the general data processing schema of any passive learning approach: the passive learner is given the entire data set and the model (or classifier) is built based on the given data. An example of passive learning can be described in the context of document classification where one could, passively, consider all the documents included in a collection and ask a human expert to correctly classify them into a set of predefined classes (e.g., sport, science, or politics) [58]. The data that is acquired in this process is then exploited to train a machine learning algorithm in order to correctly classify new documents when it is needed.

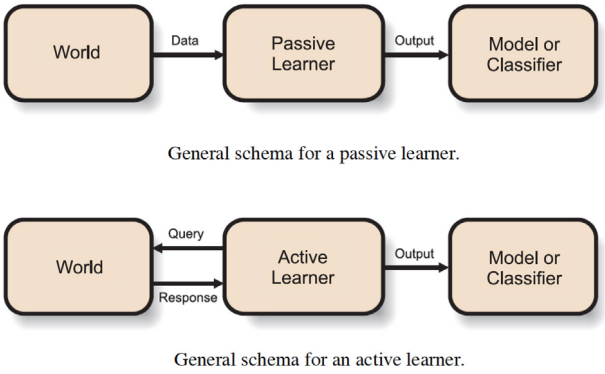


Fig. 2 – The general schema for active learning vs. passive learning [58].

However, acquiring the training data in this way may be expensive, time consuming, and data not strictly useful to improve the system performance may be processed. Moreover, in most of the cases, there is no control on the goodness of the acquired data. Consequently, the acquired data may be uninformative and therefore useless. That is where active approaches come in, trying to carefully selecting the data to acquire. "This process of guiding the sampling process by querying for certain types of instances based upon the data that the system has seen so far is called *Active Learning*" [58]. Here, by querying we mean asking (to a "teacher") the label of an instance. Fig. 2 (bottom) illustrates graphically the general schema of active learning: the active learner is not given the entire data (in contrast to the passive learner). Instead, the learner can query, receive new data, and update the classifier model accordingly. To better illustrate this idea with an example, suppose that there is a classroom with students listening to their teacher. A passive learner is a student sitting and listening to the teacher and simply collecting the information. Conversely, the active learner is a student that questions, listens to the answers, and asks some new questions based on the former answers [58].

In general, active learning is well-motivated in various machine learning tasks where data is not sufficient or expensive to acquire. Since the active learner selectively chooses the training data points, the number of data points used for training a model based on active learning is typically lower than the number used by the passive learner. For example, automatic media classification requires a training procedure that can be very difficult or expensive. It may require experts to label several resources (e.g. articles, webpages, images, audios, and videos) as relevant or irrelevant to a particular information need. In addition, sometimes not all these labels are actually useful: e.g., labelling an obscure, unpopular, and rare movie that is not similar to any other movie in the considered collection. Similarly, in speech recognition, labelling the speech utterances is costly in terms of time and required knowledge. Annotating the words in a speech can take a long time and the problem becomes harder for rare languages [10]. In these cases active learning can be used in order to reduce the number of training labels [10,19].

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Active learning can also be useful for information extraction. In this case, the system must access accurately labelled documents in order to extract information, such as relations among the entities (e.g. a person *works* in a company). That process takes time and sometimes requires high level expertise. Active learning can be applied to minimize the number of labelled documents needed for training [10].

Several specific active learning algorithms have been proposed and [58] illustrates a generic schema for describing them. This schema is not specific to collaborative filtering, and hence, in principle, it can be instantiated by most of the concrete applications. Given a training set of N input–output pairs $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where $x_i \in X$ is an instance, and $y_i \in Y$ is a label, we assume that there is a model $\mathcal{M}$ that maps input to output $\mathcal{M} : X \longrightarrow Y$, and another function $Loss(\mathcal{M})$ which measures the error of the model (the smaller the better). At every iteration j of the active learning process, the learner selects a query $q_j \in potentialQueries \subset X$, requests the label of it, and obtains the label y_j. q_j is an instance whose label is unknown and has the property that $\mathcal{M}'$, which is the model $\mathcal{M}$ re-trained by adding to the set of previously labelled instances the new pair (q_j, y_j), has the lowest loss. Since the true label of q_j is unknown when the system selects it, a natural solution is to score a query q_j with the expected loss of $\mathcal{M}'$ [58]:

$$Loss(q_j) = E(Loss(\mathcal{M}')). \tag{7}$$

for j:=1 *to totalIterations* **do**
  foreach q_j *in potentialQueries* **do**
    Evaluate $Loss(q_j)$;
  end
  Ask query q_j for which $Loss(q_j)$ is the lowest;
  Update model $\mathcal{M}$ with query q_j and response (q_j, y_j);
end
return model $\mathcal{M}$;

Algorithm 1: General algorithm for Active Learning [58]

In the generic algorithm 1, the learner has an important role, i.e., it decides which data point (q_j, y_j) should be acquired. However, since this is not possible to know in advance which points will contribute more to minimize the loss, a heuristic is normally used. Such heuristic is the gist of any *Active Learning Strategy*. In the next section we will discuss the application of active learning in collaborative filtering and we will illustrate several types of strategies.

5. Active learning in collaborative filtering

Active learning for recommender systems has been initially motivated by the need to implement more effective sign up processes [35]. In the sign up stage the system actively selects and proposes to the users individual items [59–63,19,64] or groups of items to rate [65,66]. For that, the system evaluates the entire set of items and selects the items that are estimated to be the most useful ones. The ratings given

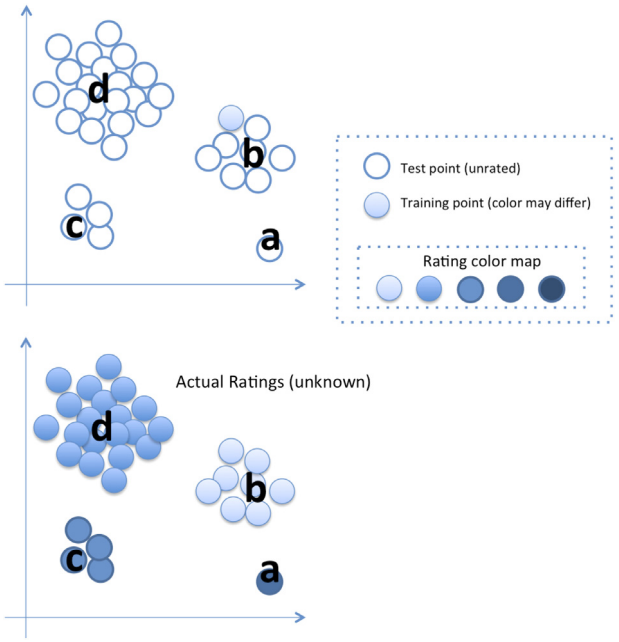


Fig. 3 – Active learning, an illustrative example [19]. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

to these items are supposed to improve the accuracy of the system the most.

In order to better explain that, let us make an example: Fig. 3 illustrates an hypothetical distribution of the rating data [19]. The upper chart in the figure depicts a starting state, in which the system has requested the user to rate just a movie from the upper right group, let us say a Sci-Fi one. The two dimensions shown in the chart represent two hypothetical features of the items. The chart below shows the actual ratings of a user (colour coded) but the system is not aware of these preferences. The charts are also showing four possibilities for selecting the next movie to be rated, i.e., movie a, b, c or d. If the system selects the movie a, say a very peculiar movie, its rating may not influence the prediction accuracy, since, no other movie is located nearby. Conversely, if the movie b is selected, it may let the system to make predictions for the movies within the same area. However, the system has already some information for predicting ratings within this area. If the movie c is selected, the system will be able to make new predictions, but only for the other three movies in this area, say Zombie movies. Ultimately, by selecting movie d, the system will be able to make rating predictions for a large number of movies that are located nearby, in the same area (say Comedy movies). Therefore, selecting movie d is likely to be the ideal choice as it allows the system to improve the accuracy of the predictions for the largest number of items [19].

In the next sections we will describe in detail how the items are evaluated, scored, and selected by different active learning strategies. We classify these strategies into two main categories, i.e., *non-personalized*, and *personalized*, and each of

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

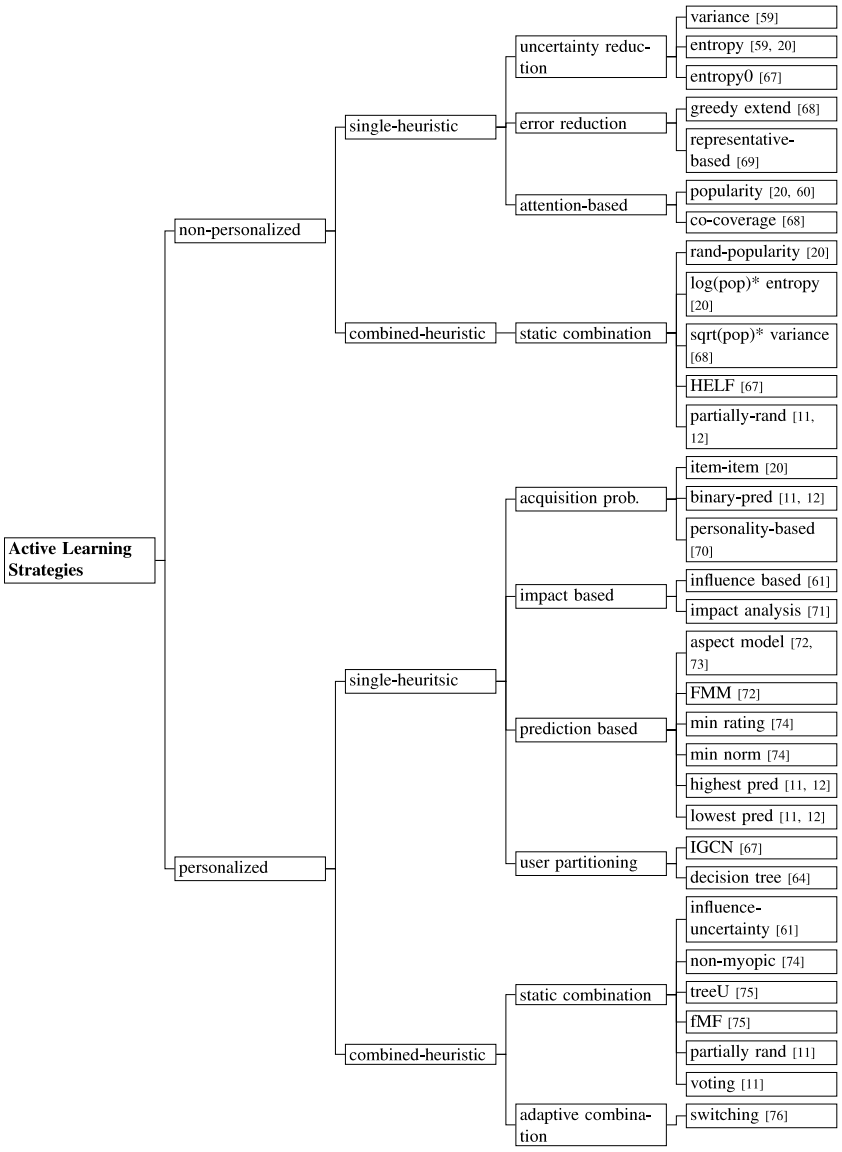


Fig. 4 – Classification of active learning strategies in collaborative filtering.

them is further partitioned into two subcategories, i.e., *single-heuristic* and *combined-heuristic*, that will be explained later.

5.1. *Non-personalized active learning*

Simpler active learning strategies do not take into account users' previously expressed ratings and request all the users to rate the same items. We refer to these strategies as non-personalized. In this case, the heuristic used for item selection does not depend on the profile of the individual users. For example, a non-personalized strategy, such as "popularity", presents to all the users the same popular items to rate.

We have identified two sub-categories of non-personalized active learning strategies: *single-heuristic* (or *individual*) and *combined-heuristic* (or *combined*). Single-heuristic strategies implement a unique item selection rule while combined-heuristic strategies hybridize together more single-heuristic strategies in order to better estimate items' usefulness.

5.1.1. *Non-personalized single-heuristic strategies*

Single-heuristic strategies are of three types: uncertainty reduction, error reduction and attention based.

Uncertainty-reduction. Uncertainty-reduction are popular strategies that select items with controversial or diverse ratings [59,60,67]. The system is supposed to be more uncertain about the users' opinion about them and asking a user to rate such items can bring useful (discriminative) information about the user's preferences. Indeed, if a similar number of users rated a movie high and low, it is not easy to decide whether the movie is good or not for a particular user. Conversely, movies with low ratings from almost all users can be considered as generally bad ones and not worth to be recommended to anybody. When the system has to decide which of the two types of movies to ask the user to rate, the first one will probably give more information about the user preferences than the second one.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

There are three strategies in this group: *variance*, *entropy*, and *entropy0*.

- *Variance* [59,60]: this strategy selects the items with the highest variance, hence, it favours the items that have been rated diversely by the users on the assumption that the variance gives an indication of the uncertainty of the system about that item's ratings [59]:

$$Variance(i) = \frac{1}{|U_i|} \sum_{u \in U_i} (r_{ui} - \bar{r}_i)^2 \quad (8)$$

  where U_i is the set of users who rated item i, and $\bar{r}_i$ is the average rating for i.
- *Entropy* [59,20]: measures the dispersion of the ratings for an item [68]. Entropy is computed by using the relative frequency of each of the five possible ratings, e.g., from 1 to 5 [59,20]:

$$Entropy(i) = -\sum_{r=1}^{5} p(r_i = r) \log(p(r_i = r)) \quad (9)$$

  where $p(r_i = r)$ is the probability that a user rate the item i as r. The drawback of this strategy is that it tends to select unpopular obscure items, since in general there are many more items with few ratings, for which the entropy may result to be high, but since it is computed on just a few ratings it tends to be unreliable. Hence, while this strategy can identify informative items, it may also pick up rarely-rated ones [20].
- *Entropy0* [68]: was proposed for solving the above mentioned limitation of entropy: the tendency to select unpopular obscure items [68]. Entropy0 addresses this problem by assigning to all the missing ratings a new value equal to 0, hence modifying the rating scale. In this way items heavily rated as 0, i.e., with few ratings, have a small entropy0 score and therefore are not selected [68].

Error-reduction. While items with diverse ratings may seem informative, they may not necessarily be the items whose ratings better help to reduce the system error. For instance, in the Netflix dataset, *Napoleon Dynamite* is a highly controversial movie, i.e., with very diverse ratings, but it is also a very unique movie (its ratings are weakly correlated to ratings of other items). Hence, while this item could be selected by the strategies in the previous group, it is not very helpful for estimating the ratings of other items [69]. In order to solve this problem, the active learning strategies mentioned in this section directly attempt to improve the predictive accuracy of the recommender, since this is the ultimate goal of any active learning strategy and it is known to be strictly correlated to the user satisfaction [69,70].

- *Greedy Extend* [69] tries to identify the items whose ratings (if elicited by the users and added to the training set) yield the lowest system RMSE [69,64]. Let us denote with A a prediction algorithm (for instance a Factor Model one), and with $F(A(L))$ the performance of A (e.g., RMSE), when it is trained on a preexistent training set extended with the ratings for the items in the list L. Then greedy extend requests to the user u to rate the items in L_u, which is defined as follows:

$$L_u = argmin_{L \subset I_u} F(A(L)) \quad (10)$$

  where I_u is the set of all the items that can be rated by u. For each candidate item in I_u, this strategy computes the reduction of RMSE, before and after adding the rating of the candidate item to the training set. The RMSE metric over the training set can be computed by adopting leave-one-out methodology [71]. Then the item with the largest RMSE reduction is added to the set L. This process is repeated and items are iteratively added until the size of L reaches the maximum number N.
- *Representative-based* [70] tries to identify a set of items that best represent the entire catalogue. These are selected based on their rating profiles, so that a linear combinations of their profiles would accurately approximate the rating profiles of the other items. Hence, the sub-matrix of the rating matrix, containing the ratings of the representative items, can approximately reconstruct the entire rating matrix:

$$R \approx CX \quad (11)$$

  where R is the $m \times n$ observed rating matrix, C is a $m \times k$ matrix built by selecting k columns (items) of the original rating matrix R, and X is a $k \times n$ matrix of free parameters. In order to find a set of k representative items and the free parameters in X the following criterion has to be satisfied:

$$minimize \ \|R - CX\|^2 \quad (12)$$

  where $\|\cdot\|$ is the Frobenius norm [72]. The active learning strategy here asks a new user to rate the k items that are identified by the above mentioned minimization problem.

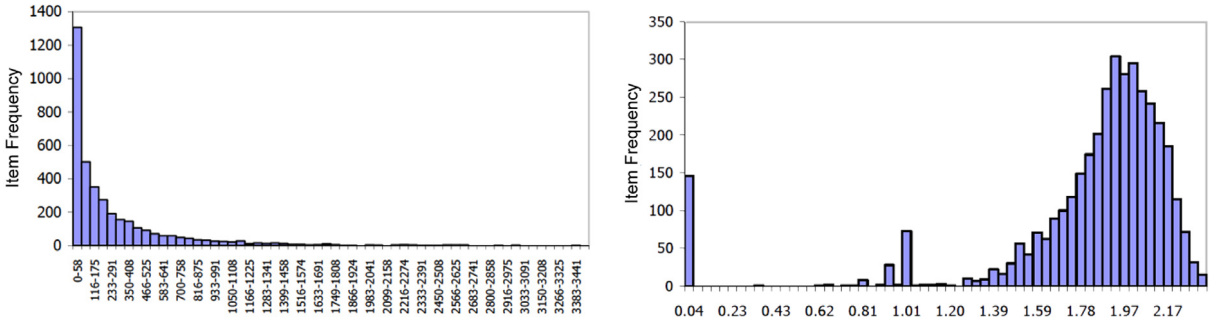
Attention-based. This group of strategies focus on selecting the items that have received the highest "attention" among the users. Such items are likely to be known by the users, and therefore they can be rated by them. Hence, these strategies usually add a lot of ratings [60,73,74]. These strategies are simple and easy to implement and they were introduced in the initial attempts to solve the cold start problem in collaborative filtering [20,73]. They are considered as baseline strategies [75,74,69,76].

- *Popularity* [20,60]: selects the most popular items, i.e., those with the highest number of ratings. It is very likely that the users are able to rate these items and consequently the size of the rating dataset can be increased [73]. However, popular items are typically widely liked by the users. Therefore, their ratings usually bring little information to the system. Moreover, this strategy may cause the *prefix bias* [20], i.e., the system trained with ratings for popular items tend also to recommend these popular items, making them even more popular.
- *Co-coverage* [69]: aims at selecting the items that are highly co-rated by the users [69]. Co-coverage is defined as follows:

$$Co\text{-}coverage(i) = \sum_{j=1}^{n} m_{ij} \quad (13)$$

  where m_{ij} is the number of users who rated both item i and item j. By selecting the items with high co-coverage, this strategy tries to identify items that are useful for collaborative filtering based predictions, which are based on patterns of co-rated items. It is important to stress that co-coverage is influenced by the items' popularity, and hence, it can request many popular items [69].

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002



(a) Popularity. **(b) Entropy.**

Fig. 5 – (a) Distribution of the popularity of the items, and (b) distribution of the entropy of the items (Movielens dataset) [68].

5.1.2. *Non-personalized combined-heuristic strategies*

So far we have presented single-heuristic non-personalized strategies. Although such strategies have shown promising results, a single criteria may fail to correctly identify the items that are worth to be rated. Therefore, it has been proposed to combine them and to identify items using multiple heuristics, e.g., improve accuracy, increase coverage or user satisfaction. For instance, as discussed before, many strategies try to find the most informative items to reduce the uncertainty of the system, but tend to ignore the prediction coverage, which is the proportion of the full set of items over which the system can make predictions. Hence an hybridization of these two strategies may be more effective than any of them when taken individually. Such hybridization can be obtained in many ways. For instance, the strategies can vote for items and the most highly voted items are selected. The combined strategies can also assign a score to each item and the total score is the multiplication (or the sum) of the scores. Or finally, the combined strategies can select few items and the union of these selected items is chosen.

De facto, there are three non-personalized combined-heuristic strategies in the literature: *Combined Random-Popularity*, *Log(popularity)*Entropy*, and *HELF*.

- *Random-Popularity* [20]: is probably the simplest example of a combined-heuristic strategy. It was used in the well known movie recommender system, called MovieLens, in order to obtain ratings in the sign-up process. This strategy extends the list of randomly selected movies for rating elicitation by inserting a popular item that comes from a manually selected list of popular movies. Hence, this strategy tries to improve the probability that users have experienced and can rate at least one of the presented movies. This strategy is also called *Classique* in [20].
- *Log(popularity)*Entropy* [20]: is a "balanced" strategy [20] as it tries to collect many ratings but also taking into account their relative informativeness. This is achieved by scoring an item with the logarithm of the popularity multiplied by the entropy of the item's ratings. The usage of the logarithm in this strategy tries to overcome the problems caused by the differences between the entropy and popularity distributions. In fact, observing Fig. 5 we can see that the distribution of the items' popularity is exponential while the distribution of the entropy is more similar to a normal distribution. Indeed, there is a huge difference among the popularity scores of few popular and many unpopular items. Conversely, this is not seen for entropy, hence popularity may dominate entropy, when the two are multiplied. Hence, computing the logarithm of the popularity transforms the exponential curve into a linear one and reduces the weight of popularity [68]. There is also a variation of this strategy that uses $\sqrt{popularity}$ instead of $\log(popularity)$ and *variance* instead of *entropy* [69].

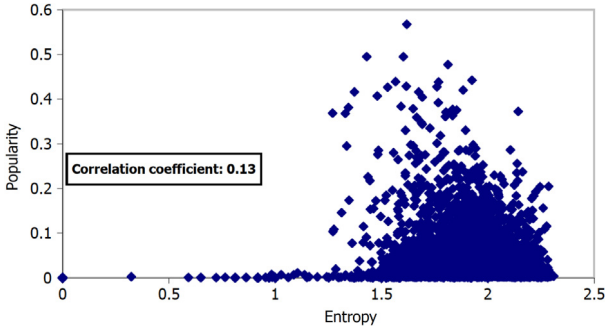


Fig. 6 – (a) Correlation between the entropy and popularity of the items (Movielens dataset) [68].

- *HELF* [68]: stands for Harmonic mean of Entropy and Logarithm of Frequency (HELF). This strategy aims at combining popularity with informativeness, as in the previous strategy, but by using a property of the harmonic mean, being high only if both operands are high. As mentioned before the entropy strategy tends to select obscure items that are rarely rated. In fact, popularity and entropy are not highly correlated (see Fig. 6). Hence, HELF selects informative items that are also rated frequently by the users. The selection score computed by this strategy is given by the following formula:

$$HELF(i) = \frac{2 \times LF(i) \times H(i)}{LF(i) + H(i)} \tag{14}$$

where, $LF(i) = \log(|U_i|)/\log(|U|)$ is the normalized logarithm of the rating frequency of the item i, and $H(i)$ is the normalized entropy of i.

5.2. *Personalized active learning*

The second major category of active learning strategies for collaborative filtering comprises personalized strategies,

which can also be either single-heuristic or combined. There are four sub-categories of personalized single-heuristic strategies that we have identified in the literature: *acquisition probability based, influence based, prediction based*, and *decision tree based*.

5.2.1. *Personalized single-heuristic strategies*

Acquisition probability based. The strategies in this group focus on improving the performance of the system by maximizing the probability that the selected items are familiar to the user, hence, are rateable. In doing that these strategies personalize the rating requests by proposing different items to different users.

- *Item–Item* [20]: selects the items with the highest similarity to the user's previously rated items. Hence, after acquiring at least one rating from a user the similarities between the user rated item(s) and the other unrated items are computed, and the items most similar to the rated ones are presented to the user to rate. Item-to-item similarity is computed using Pearson Correlation, as for user-to-user similarity (see Section 3.1):

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)(r_{uj} - \bar{r}_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \bar{r}_i)^2 \sum_{u \in U_{ij}} (r_{uj} - \bar{r}_j)^2}} \quad (15)$$

  where U_{ij} denotes the subset of users who co-rated items i and j, $\bar{r}_i$ denotes the average rating of item i. Since this strategy does not consider the informativeness of the items, it can acquire ratings that do not improve the system in terms of prediction accuracy [20].
- *Binary Prediction* [11,12]: it first transforms the rating matrix into a new matrix with the same number of rows and columns, by mapping null entries to 0, and not null entries to 1. Hence, this new matrix models only whether a user rated an item or not, regardless of its value [40]. Then, using a factor model, a prediction for all the ratings initially set to 0 is computed and for each user, and the items with the highest predicted score are selected. This strategy tries to identify the items that the user is more likely to have experienced, in order to maximize the chances that the user can provide the requested rating. In that sense it is analogous to the popularity strategy, but it tries to make a better (personalized) prediction of what items the user can rate by exploiting the rating knowledge of each user.
- *Personality-Based Binary Prediction* [77]: it first transforms the rating matrix to the same binary matrix built by binary prediction (as described above). Then, the new matrix is used to train an extended version of the matrix factorization algorithm that profiles users not only in terms of their binary ratings (as in the previous case), but also using their known attributes: gender, age group and the scores for the Big Five personality traits on a scale from 1 to 5 [78,79]. Given a user u, an item i and the set of user attributes $A(u)$, it predicts which items have been experienced by user, using the following rule:

$$\hat{s}_{ui} = \bar{i} + b_u + q_i^\top \left(p_u + \sum_{a \in A(u)} y_a \right), \quad (16)$$

  where $\hat{s}_{ui}$, the item selection score of the strategy, is an estimation of the probability that the user u has experienced the item i, and, p_u, q_i and y_a are the latent factor vectors associated with u, i and the user attribute a, respectively. The model parameters are learned (see Section 3.2) by minimizing the associated regularized squared error function through stochastic gradient descent.
  In order to use such a strategy the system must assess the personality of the users. A questionnaire such as Ten-Item Personality Inventory (10-items TIPI), or even a shorter version, such as the Five-Item Personality Inventory (5-items FIPI), can be used [80,81,77].

Impact based. Eliciting the ratings for the items that the system is more uncertain about can be beneficial for the system to better predict the ratings of these particular items. However, that will not necessarily end up with the reduction of the uncertainty of the rating prediction for other items. Impact based strategies select items in the attempt to minimize the rating prediction uncertainty for all the items.

- *Influence Based* [61]. This strategy estimates the influence of item ratings on the rating prediction of other items and selects the items with the largest influence. First the rating prediction $\hat{r}_{ui}$ for user u and an unrated item i, is computed. Then, the rating prediction $\hat{r}_{ui}$ is decreased by 1 unit: $\hat{r}'_{ui} = \hat{r}_{ui} - 1$ (e.g., rating 5 is changed to 4). Finally two prediction models are generated, one adding $\hat{r}_{ui}$ and another adding $\hat{r}'_{ui}$ to the training set, and the absolute value of the differences of theirs predictions for the ratings of all the items different from i are computed. The influence of i is estimated by summing up all these differences. Finally, the items with the highest influence are selected for active learning [61].
- *Impact Analysis* [75]: selects items whose ratings have the highest impact on the prediction of the other ratings. In order to explain this strategy let us consider a graph-based representation of the rating dataset where users and items are nodes of the graph (see Fig. 7—left) [82]. A rating can be seen as a link between a user and an item (product). In order to compute a rating prediction, and consequently identify a possible recommendation for a user using user-based neighbour-based models (see Section 3.1), at least a four-node path should be created (see Fig. 7—right). A four-node path between user u_1, u_2, item p_1 and p_2 can be created when the user u_1 has rated item p_1 and p_2 and user u_2 has co-rated p_2. In this case, a neighbour-based can predict the rating of u_2 for p_1. Hence, the more four-node paths are created the better the prediction may become. For that reason this strategy attempts to discover which ratings produce new four-node paths.

Prediction based. These strategies use a rating prediction model in order to predict which items should be presented to the users for rating [83,84,11,12,85]. The top predicted items are selected for rating elicitation. The prediction models used by these strategies may differ. Hence, we can sub-categorize these strategies according to the prediction model they use. One of the advantages of these strategies is that they request users to rate items that are predicted as relevant for them. Hence, the user may even enjoy seeing and rating them. Moreover, such items are also likely to have been experienced by the user since are estimated to be interesting for her.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Fig. 7 – Simple Bipartite Graph Model for CF [75].

- *Aspect Model* [83,84]. This strategy models a user as a mixture of multiple interests (aspects). Every user $u \in U$ has a probabilistic membership to multiple aspects $z \in Z$, and users in the same group are assumed to have similar rating patterns. The probability of the user u giving rating $r \in R$ to an item $i \in I$ is estimated by:

$$p(r|u,i) = \sum_{z \in Z} p(r|z,i)p(z|u) \tag{17}$$

  where $p(z|u)$ models how likely the user u will belong to the group z and $p(r|z,i)$ models how likely the users in group z will assign the rating r to the item i. An extension of this strategy uses *Flexible Mixture Model (FMM)* [83,84] and determines two sets of latent aspects $\{z_u, z_i\}$: every user $u \in U$ is considered to be a mixture of multiple interests (user aspects) but also every item $i \in I$ is considered to be a mixture of patterns (item aspects).
- *Highest Predicted* [11,12]: scores items according to their predicted ratings and selects the top ones according to these scores. The items with the highest predicted ratings, are estimated to be the items that the user likes the most. Hence, it could also be more likely that the user has experienced these items. Moreover, their ratings could also reveal important information about what the user likes. We also note that this strategy is widely applied in real recommender systems, as the users are solicited to rate the recommendations.
- *Lowest Predicted* [11,12]: uses the opposite heuristic of highest predicted: it scores the items as $Maxr - \hat{r}$, where $Maxr$ is the maximum rating value (e.g., 5) and the $\hat{r}$ is the predicted rating. Lowest predicted items are likely to reveal what the user dislikes, but are also likely to elicit fewer ratings since normally there are many more items that the user does not know among those that the system would not recommend, compared to the items that are predicted to be liked by the user. Moreover, [76] argued that when a new user is registered, since the system has no or few ratings of that user, the model parameters computed for this user can be inaccurate and considerably different from the optimal ones. Therefore, it could be better to choose the items whose ratings are not correctly predicted by the recommender. On the other hand, since most of the actual ratings given by users are large (users rate often items they like) the prediction error for the items with low predicted ratings is expected to be high. Hence, eliciting low ratings should reveal large prediction errors and may impose significant changes into the model parameters. This strategy is also called *MinRating* [76].
- *MinNorm* [76]: this strategy uses matrix factorization to compute the latent factors vector of each item and selects the items whose representative vectors have the minimum Euclidean norm. The rationale of this strategy is that after active learning has acquired a number of ratings, and the prediction accuracy has achieved an acceptable level, it could be better to stabilize the prediction model and to avoid large changes of the latent factors. In order to do that, the changes of the latent factors (gradient) should be minimized. While the system has less control on the prediction error, minimizing the item factors may result in a more stable prediction model.

User partitioning. The strategies within this group first partition the existing users into a number of clusters of users of similar tastes and affinities, and then select the items whose rating will better reveal to which cluster a target user belongs [86–89].

- *Information Gain through Clustered Neighbours (IGCN)* [68]: it constructs a decision tree where each leaf node represents a users' cluster and each internal node represents a test, i.e., a specific item that is proposed to user to rate (to like, dislike, or to provide no opinion, if she has not experienced the item). Users are clustered according to their similarity values that are measured using Pearson Correlation (see Section 3.1). Starting from the root node, a new user is proposed to rate a number of items and based on the ratings she provides, she reaches one of the leaf nodes. Hence, the most informative items are those that rating them will enable to better classify the user in her representative cluster. In order to construct the tree, for each item i, information gain $IG(i)$ is computed, and the item with maximum IG is selected to further expand the tree:

$$IG(i) = H(C) - \sum_{r} \frac{|C_i^r|}{|C|} H(C_i^r) \tag{18}$$

  where C denotes the distribution of users in the clusters (i.e., the proportion of users in different clusters), $H(C)$ denotes the entropy of C, C_i^r denotes the distribution of users within each cluster who rated r the item i. For instance, C_i^5 indicates the proportion of users whose rating for item i is 5.
- *DecisionTree* [64]: this strategy uses a decision tree whose nodes, either internal or leaf, represents groups of users. When building the tree, a single or multiple candidate partitioning movie(s) for an internal node divides the users into three groups based on the ratings of the users: *Lovers*

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

(who rated the item high), *Haters* (who rated the item low), and *Unknowns* (who did not rate the item). Then, for each of these groups, the rating predictions of their unrated items are estimated. The estimated RMSE is computed as the squared root of the deviation of the predicted ratings from the true ratings. Finally, the total prediction error is computed by summing up the RMSE in the three groups and the movie(s) with minimal total error is selected for active learning. This process is iterated within each of the built 3 groups, to select the next movies.

5.2.2. *Personalized combined-heuristic strategies*

Personalized combined-heuristic is a group of strategies that hybridize personalized single-heuristic strategies by combining them in order to exploit all their advantages. The combination can be either static, which means that the combined strategies do not change over time, or adaptive, i.e., it changes over time according to certain criteria.

Static combination. The strategies in this group are: *Influence—Uncertainty based, Non-Myopic, Decision tree—Matrix Factorization (TreeU), Functional Matrix Factorization (fMF)*, and *Combined with Voting*.

- *Influence—Uncertainty based* [61]: combines two strategies, influence-based (see Section 5.2.1) and variance (see Section 5.1.1). It selects items that have both a large influence on the rating predictions of other items, and variance:

$$argmax_i \mathrm{Var}(i) I(i) \tag{19}$$

  where $I(i)$ is the influence of item i (see Section 5.2.1).
- *Non-Myopic* [76]: this strategy combines the prediction based strategies *MinRating* and *MinNorm*. Both of them use a prediction model (e.g., Matrix Factorization) to compute the items' latent factors and exploit them to identify the best items for rating elicitation. For every item i, two ranking lists are computed using these two strategies. Then considering the positions of item i in each ranking list the aggregated score is computed and the items with the minimum aggregate scores are selected for active learning:

$$\begin{aligned} score(i) &= (1-w)\ minRating\_rank(i) \\ &\quad + w\ minNorm\_rank(i) \end{aligned} \tag{20}$$

$$w = \frac{\#current\_request - 1}{\#total\_request} \tag{21}$$

  where *#current_request* is the number of times the system has already requested the user to rate some items, and *#total_request* is the total number of times the system will request the user to rate. At the beginning, when the first requests are made by the system, the items are selected mainly by MinRating, which is supposed to work better in the early stages of the system usage, i.e., when the users have not rated many items (see the description of the MinRating strategy). As more ratings are elicited the system tends to consider as more important MinNorm, since MinNorm is supposed to work better in the later stage of the system evolution, when the users have already rated many items (see the description of MinNorm) [76].
- *Decision tree—Matrix Factorization (TreeU)* [90]: combines decision tree and matrix factorization prediction based strategies in two steps. In the first step, for each user u and item i the associated latent factor vectors p_u and q_i are computed by a plain matrix factorization model (see Section 3.2):

$$\hat{r}_{ui} = p_u^T q_i \tag{22}$$

  where $p_u \in \mathbb{R}^F$ and $q_i \in \mathbb{R}^F$ are F dimensional vectors and F is the number of latent factors. Then, a decision tree with depth k, is constructed using the approach explained before (see the *Decision Tree* strategy described in Section 5.2.1) to fit the latent factors based user profiles. The prediction model then predicts ratings using the user profiles from the decision tree and the item profiles from the matrix factorization model.
- *Functional Matrix Factorization (fMF)* [90]: In this method, similarly to other decision tree methods, a decision tree is constructed for managing the interviews with the users and elicit their ratings. It builds at each node of the tree, which represents the users that have replied in a certain way to the interview questions in the nodes connecting the root to the current node, a latent user profile that functionally depends on the replies to these interview questions.
  The item profiles are instead learned by assuming that the user profiles are as those described above and minimizing the regularized prediction error of the model constructed in this way. The authors have developed an iterative optimization algorithm that alternates between decision tree construction and latent profiles extraction. It is shown that this approach improves both TreeU and original decision Tree method [64].
- *Partially Randomized Strategies:* [11] a partially randomized strategy modifies or extends the list of items returned by a strategy introducing some random items. For instance, if only few rating predictions can be computed for a given user u (e.g. it is a new user) the *highest predicted* strategy is not able to score enough items. In this case the randomized version of this strategy can add some random items for the user to rate [11]. The behaviour of these strategies depends on the amount of chosen randomness.

The next group of strategies use more complex mechanisms to combine single-heuristic strategies. The first one uses a voting mechanism while the second uses a switching mechanism, i.e., adaptively selects the best suited active learning strategy.

Combined with voting. It computes a score for an item as the number of votes given by a committee of n strategies. Each of these strategies produces its top candidate items for rating elicitation, and the items appearing more often in these lists are selected. Obviously, this strategy strongly depends on the selected voting strategies. For example, including the random strategy may impose an exploratory behaviour that could improve the system coverage [11]. In [12] the authors considered two versions of this strategy. In the first version they combined: *popularity*, *variance*, *entropy*, *highest–lowest predicted*, *binary prediction*, and *random*. In the second one they combined only 3 strategies, which were

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

selected as representative of the original six: *log(pop)*entropy*, *highest predicted*, and *random*.

Adaptive combination. Although having a static combination of single-heuristic strategies could improve system performance, this is not always the best solution. In fact, in [11] the authors compare several state-of-the-art strategies and show that there are active learning strategies that have better performance in different phases of the rating elicitation process. For instance, the static combination of many strategies may not fit the specific needs of the early phase (true cold start) or later phases of the rating elicitation process. Hence, as an alternative to the static combination of active learning strategies, it is possible to create strategies that adapt to the different situations.

In order to implement an adaptive combination of strategies one must choose a metric that evaluates the system's performance. For example, if MAE reduction is the target metric, the strategy selected in a phase is the one that is estimated to reduce more MAE in that phase. In fact, it may not be possible to design an adaptive strategy that will optimize all the possible measures of performance.

As an example of this adaptive behaviour we quote *Adaptive Combination with Switching* [91]. Every time this strategy is applied a certain percentage of the users (called exploration group) is randomly selected for choosing the best performing strategy, among a pool of strategies. Then this winning strategy is applied to the remaining users. Each strategy in the pool is tested on an equal number of random users in the exploration group: it selects items to be rated by these users, and acquires their ratings (if available). Based on the ratings acquired by the system, a factor model is trained and its MAE of the newly acquired ratings is computed. Moreover, for each individual strategy, the observed probability to acquire ratings for the selected items is also computed by estimating the ratio of the number of acquired ratings over the number of items requested to be rated. Finally, the score of each individual strategy is calculated by multiplying this probability by the rating prediction error (MAE) on the acquired ratings. The strategy with the highest score is then selected. Hence, this adaptive strategy is selecting the individual strategy that is able to acquire from the exploration group the largest number of ratings for items whose system rating prediction is currently most erroneous [91].

6. Evaluation of active learning for collaborative filtering

In the previous sections we have illustrated several active learning strategies for collaborative filtering and we discussed their benefits and drawbacks. In this section we review the methods that have been employed to assess the effect of active learning on the system performance. We discuss a number of issues that we have identified as important when designing an evaluation method for AL.

6.1. *Offline vs. online evaluation*

Active learning strategies can be evaluated *online* or *offline*. In the first case the active learning system interacts with real users by acquiring their preferences through a graphical user interface. This requires building or accessing a fully developed recommender system, with a large user community, which is expensive and time consuming. Moreover, in this way it is hard and practically impossible to compare several strategies in alternative system configurations. Conversely, in offline experiments, a pre-collected rating dataset is used to simulate the behaviour of real users interacting with the system. It is implicitly assumed that the simulated user behaviour in offline experiment is similar to that actually shown by them while interacting with a real deployed system [92].

The majority of the works in this research area [59,20,68,83,60,84,61,93,63,94,69,95] have focused on offline evaluations and only a few have been conducted online [81,77,20,96,68]. The core problem is that offline settings cannot completely simulate a natural interaction with a user. The most striking difference is that in an online interaction with a real system a candidate item, which the active learning strategy identifies for the user to rate, can be any item of the catalogue, while in offline experiments only ratings that have been already expressed by users can be requested: the simulation just adds these ratings to the training set used to build the prediction model [59]. Moreover, in many cases the simulated user is requested to provide only the ratings that she gave when the ratings' data set was acquired. Hence, the simulation implicitly and erroneously assumes that all the items selected and proposed by the system to rate are known to the user. However, it is obvious, that in a more realistic scenario, the system may select items that are not known to the user and thus cannot be rated, and it can also request ratings that are not included in the available ratings' data.

Hence, in off-line evaluation procedures, one cannot exactly measure how many items, among those selected by the active learning component, can be actually rated by the user. But, this is an important factor, since focussing only the estimation of the usefulness and informativeness of the ratings may result in requesting the user to rate obscure items that the she cannot actually rate. If this happens, the system may fail to obtain ratings and may not be able to generate recommendations, in particular for new user. However, such strategy may perform excellent in offline evaluations since it may improve a lot the accuracy of the system with minimum number of ratings.

Notwithstanding this limitation, offline evaluations can be beneficial, and there is large agreement that a sound evaluation methodology should comprise the following three steps [20,73,68,92,95]:

1. Developing the candidate active learning strategies and conducting preliminary studies (e.g. reviewing the literature, forming a hypothesis).
2. Conducting offline experiments on the existing rating dataset to discover and select the most promising strategies.
3. Conducting online experiments with real users to remove any possible bias introduced by the nature of offline experiments, and ultimately to verify the effectiveness of the selected strategies.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

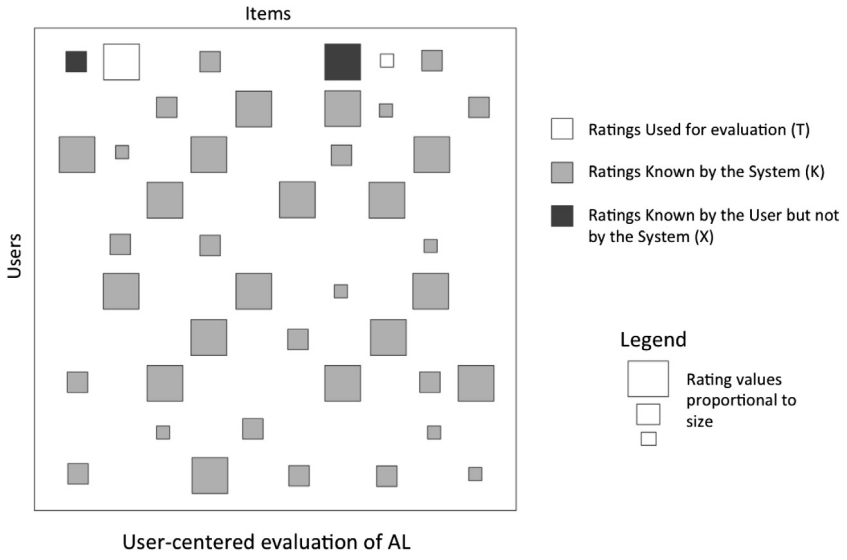


Fig. 8 – User-centred evaluation of active learning strategies [11].

In addition, it is also critical to correctly design the evaluation procedure that is used at each step. In fact an ill-designed procedure may result in a wrong or misleading outcome. In the next section we will present and discuss a popular offline evaluation procedure that was applied in the past and we will then illustrate an improvement that we have recently proposed.

6.2. *Single user focussed evaluation*

The evaluation of active learning strategies, either offline or online, has been commonly done under a single user setting. This means that the usefulness of the ratings elicited by a user was judged based on the improvement of the system performance (recommendation accuracy) for the same user [59,20,68,83,84,69,94,63,95]. This is illustrated in Fig. 8. In this setting the system is supposed to know a large number of ratings, i.e., the ratings given by several users and known by the system (grey boxes in 8), and it is focusing on a new user (top row of the matrix). The system elicits some ratings of this new user from the unknown set, i.e., ratings known by the user but unknown to the system (black boxes). Finally, the prediction of the system on test ratings (white boxes) is evaluated. This evaluation setting focusses on the new user problem and measures how the ratings elicited from a new user may help the system to generate good recommendations for this particular user.

As an example of this evaluation setting, we refer to the online experiment described in [68]: the new user enters the Movielens recommender system and completes the sign-up process by rating a number of items proposed by one of their active learning strategies. These ratings, together with the ratings of the other users, are considered for training the system. Then the user receives recommendations from the system while she can still provide more ratings anytime. These ratings, which are elicited from the user after the sign-up process, are used for testing the prediction accuracy of the system. But the accuracy is measured only on the ratings of that new user, while the elicited rating from her may improve also the predictions for other users.

In [11] it introduced a novel evaluation method that can better predict the future overall performance of the system, and it is therefore called "system-wide" evaluation of active learning (see Fig. 9). In this approach the performance of the recommender system is measured taking into account that the ratings from a user influence also the recommendation quality for other users. Hence, in the evaluation setup, the test set consists not only of ratings of the new user, but also of ratings of the other users. Indeed, for every new user, the recommendation model is trained and tested on ratings randomly selected from the ratings of all the users.

6.3. *Natural acquisition of ratings*

In a recommender system the addition of ratings to the system's database can occur either because the system actively requests the user to rate some items or because the user voluntary decides to access the rating interface and enter some ratings [96,73,68].

- *Natural Acquisition of Ratings* is the scenario where the user provides herself ratings, without being requested by the system to enter them. This can happen, for instance, when the user explores the list of available items and provides ratings for some of them. This scenario is also called *User Controlled* rating elicitation since it is up to the user to provide or not the ratings. A possible limitation of this scenario is that the users may not make a special effort for identifying which items are more useful for the system [68]. Hence, the ratings added in this scenario may be either not very informative, or not useful to improve the performance of the system's rating prediction model.
- In *Active Learning without Natural Acquisition of Ratings* the system selects and proposes a list of items to the user to rate and only these system-requested ratings are added to the database. This is not a fully realistic scenario since

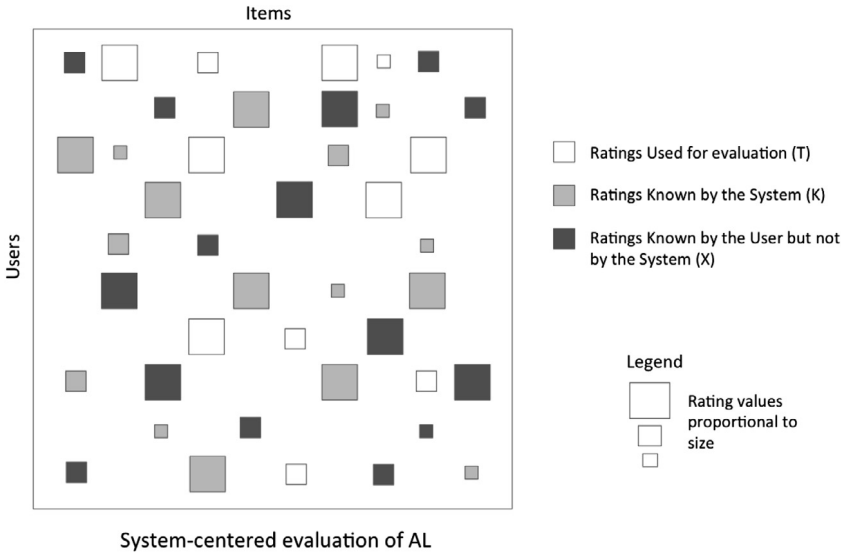


Fig. 9 – System-wide evaluation of active learning strategies [11].

active learning rating elicitation is never the unique source of new ratings. This simulation scenario fails to take into account that the user can freely add ratings for items that she has experienced even if not requested by the system. This scenario is also called *System Controlled* rating elicitation [68].

- *Active Learning with Natural Acquisition of Ratings* denotes a more realistic scenario where the ratings are provided both by the users on a voluntary basis, and by some active learning strategy. Since in this scenario both the user and the system control the rating elicitation process, it is also called *Mixed Initiative* rating elicitation [68,53]. This more realistic evaluation scenario is introduced in [11, 91]. In particular, in this setting the authors simulate the temporal evolution of the quality of the recommendations generated by the system when, in addition to exploiting an active learning strategy for requesting the user to rate some items, a particular simulation procedure models the unsolicited addition of ratings by the users. Their results have shown that mixing active learning strategies with the natural acquisition of ratings greatly influences the performance of the evaluated strategies, hence they stress the importance of the design of more realistic simulations of users' rating behaviour.

In summary, when evaluating active learning strategies, it is important to take into account the natural acquisition of the ratings. This is a concern in both online and offline evaluation setting, because the natural acquisition of ratings may alter the results of the evaluation and ignoring natural acquisition may lead to wrong conclusions about the quality of the evaluated strategies. Although this is an important concern, we should note here that almost all of the current works on active learning for collaborative filtering have ignored it (perhaps for the sake of simplicity) and evaluated their proposed active learning strategies only under the second scenario, i.e., where the natural acquisition of ratings is not taken into account.

7. Comparison of active learning techniques

7.1. *Performance summary*

Table 1 summarizes the performance of different active learning strategies. Some observations are in order: first of all, as noted before, the majority of the surveyed works in active learning have adopted an offline evaluation approach and measured the improvement of the system rating prediction accuracy in terms of system Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) reduction. Hence, we cannot compare many of the surveyed strategies also with respect to other important performance metrics, such as those measuring the ranking quality: Normalized Discounted Cumulative Gain (NDCG) or Mean Average Precision (MAP). Moreover, most of the experiments have been conducted in the Movie domain using the MovieLens and Netflix datasets. More importantly, while the adopted evaluation methods share some similarities, still they differ significantly and hence it is difficult to draw a decisive conclusion and properly generalize the results reported in individual works. Moreover, here we report only the results obtained in the papers which originally proposed the considered strategy, as the reference benchmarks in this comparison. It is worth noting that the majority of the works in this research area focused on evaluating the active learning component when eliciting the first few ratings. Indeed, it has been argued that requesting users to rate repeatedly many items is not realistic, and the users will tend to ignore these requests or, even worse, quit the system.

It is clearly shown in the table that different strategies can improve different aspects of the recommendation quality. In terms of rating prediction accuracy (MAE/RMSE), there are various strategies that have shown excellent performance. While, some of these strategies are easy to implement (e.g., Entropy0 and Log(popularity)*Entropy), others are more complex and use more sophisticated Machine Learning algorithms (e.g., DecisionTree, and Personality-based FM).

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Table 1 – Performance comparison of active learning strategies ("✓✓" Very Good, "✓" Good, "×" Poor, "–" Not Available) ML: Movielens, NF: Netflix, EM: EachMovie, AWM: Active Web Museum, MP: MyPersonality, STS: South Tyrol Suggests, LF: Last.fm.

	Type		Strategy	Metric				Eval.		Compar. Strategies	Datasets
				MAE/ RMSE	NDCG/ MAP	Precision	# Rating	Online	Offline		
Non-personalized	Single	Uncertainty based	1. Variance [59,61]	✓	–	–	–	–	y	2, 4, 6, 9, 24	AWM, EM
			2. Entropy [20,68]	×	–	–	–	–	y	3, 6, 8, 9, 11,13, 22	EM
			3. Entropy0 [68]	✓✓	–	–	✓✓	y	y	2, 6, 8, 11, 13, 22	ML
		Error reduction	4. Greedy extend [69]	✓	–	–	–	–	y	2, 3, 6, 7, 10, 11	NF
			5. Representative [70]	–	✓✓	✓✓	–	–	y	6	NF, ML, LF
		Attention based	6. Popularity [20,68]	✓	–	–	✓✓	y	y	2,8, 9, 11, 13, 22	ML
			7. Co-coverage [69]	×	–	–	–	–	y	2, 3, 4, 6, 10, 11	NF
	Combined	Static combin.	8. Rand-pop [20,68]	×	–	–	×	y	y	2, 3, 6, 11, 13, 22	ML
			9. Log(pop)*entropy [20]	✓✓	–	–	✓	y	y	3,6, 8, 13	ML
			10. sqrt(pop)*var [69]	✓	–	–	–	–	y	2, 3, 4, 6, 7, 11	NF
			11. HELF [68]	✓✓	–	–	×	y	y	2, 3,6, 8, 13, 22	ML
			12. Non-pers-part rand. [11]	✓	✓✓	✓	×	–	y	1, 6, 9, 12, 14, 20, 21, 28,29	ML, NF
Personalized	Single	Acquisition prob.	13. Item–item [20,68]	×	–	–	✓✓	y	y	2, 3, 6, 8, 9, 11, 22	ML
			14. Binary-pred [11,12]	✓	×	✓✓	✓	–	y	1, 6, 9, 12, 20, 21, 28, 29	ML, NF
			15. Personality-based [77,97]	✓✓	✓✓	–	✓✓	y	y	3, 9, 14	STS, MP
			16. Impact analysis [75]	✓✓	–	–	–	–	y	9	ML
		Prediction based	17. Aspect model [83,84]	✓	–	–	–	–	y	2	EM, ML
			18. Min rating [76]	✓	–	–	–	–	y	19,25	ML
			19. Min norm [76]	×	–	–	–	–	y	18,25	ML
			20. Highest-pred [11,12]	✓	×	✓✓	✓	–	y	1, 6, 9, 12, 14, 21, 28,29	ML, NF
			21. Lowest-pred [11,12]	✓	×	✓	×	–	y	1, 6, 9, 12, 14, 20, 28, 29	ML, NF
	Combined	User partitioning	22. IGCN [68]	✓✓	–	–	✓	y	y	2, 3,6, 8, 11, 13	ML
			23. Decision tree [64]	✓✓	–	–	–	–	y	3, 4, 10, 11	NF
		Static combin.	24. Influence based [61]	✓✓	–	–	–	–	y	1, 4, 6, 9	ML
			25. Non-myopic [76]	✓	–	–	–	–	y	18, 19	ML
			26. TreeU [90]	✓	–	–	–	–	y	23, 27	ML, EM, NF
			27. fMF [90]	✓✓	–	–	–	–	y	23,26	ML, EM, NF
			28. Pers-partially rand. [11]	✓	✓✓	✓	×	–	y	1, 6, 9, 12, 14, 20, 21, 28,29	ML, NF
			29. Voting [11,12]	✓✓	✓✓	×	×	–	y	1, 6, 9, 12,14, 20, 21, 28	ML, NF
		Adaptive combin.	30. Switching [91]	✓✓	✓✓	–	✓✓	–	y	9, 20, 29	ML

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Strategies that have shown excellent performance in terms of ranking quality (NDCG/MAP), are Representative-based and Voting strategies. In terms of precision, prediction-based strategies (Highest-predicted, and Binary-predicted) have shown excellent performance. In terms of number of ratings acquired (# Ratings), as expected, strategies that consider the popularity of items (Popularity and Entropy0) can acquire the largest number of ratings. But, other strategies that maximize the chance that the selected items are familiar to the user (Item–item and Personality-based) can also elicit a considerable number of ratings. For these strategies the success ratio (*#acquired_ratings/#requested_items*) is the largest. This is an important factor, since strategies that only focus on the informativeness of the items may fail to actually acquire ratings, by selecting obscure items that users do not know and cannot rate.

7.2. Practical guidelines

Here we offer some practical guidelines that can be used for the design and development of effective active learning strategies in operational systems:

- *Eliciting only high-ratings can cause system bias*: proposing the users to rate items with the highest predicted ratings (recommendations) is typical in recommender systems. This is due to the fact that users usually check and rate interesting items which they like. However, it has been shown that ratings given for such items populates the dataset disproportionately with high ratings, and this ultimately biases the rating prediction algorithm towards high predicted ratings. Hence, it is not always convenient to ask to rate what users like, but it is also important to understand what they dislike.
- *Partially randomizing can be beneficial*: including few random items and asking new users to rate them was shown to be useful to improve effectiveness of the active learning strategies, particularly in the early stage of the elicitation process. Indeed, it is sufficient to add just a small portion of randomly selected ratings to the elicitation list to reduce potential biases, especially those introduced by prediction-based strategies. This is particularly useful in the early stage of the system evolution where most of the strategies can easily introduce wrong biases.
- *Personalization increases the chance of rating elicitation:* users differ very much in decision making strategies, in the knowledge of items, and in their taste and preferences. Hence, it is clearly improper to ask the users to rate the same items. Indeed, many users may have limited knowledge and they may ignore many items and hence may not properly provide ratings for these items. Personalized strategies tackle this issue by maximizing the probability that the selected items are familiar to the user, and hence are ratable. They achieve that goal by proposing to rate different items to different users. This has been shown to be very beneficial and to effectively increase the chance of acquiring ratings.
- *Additional information is beneficial*: exploiting additional source of information about the users, if available, has shown to be effective in improving the active learning process. For example, the user personality, which strongly correlates with her rating behaviour, can be exploited to adapt the active learning strategy to the user. Another valuable information about the user, which can be leveraged in active learning, is social network data which is easily obtained from the user (e.g., by means of social network login). Exploiting the social characteristics of users may help the system to better identify whether or not certain type of items may be selected and proposed to the user to rate.

8. Conclusions and future work

In this article we surveyed the state-of-the-art of active learning in collaborative filtering recommender systems. We have performed a comprehensive analysis and classified a wide range of active learning strategies, along the two descriptive and discriminative dimensions: whether they are personalized or not, and how many different item selection criteria (heuristic) are considered. In fact active learning strategies do have very different features, and implement various heuristic for selecting the items that they present to the users to rate. We believe that the classification and systematization of these strategies can help to better understand and use them, hence providing a practical resource for the practitioners and researchers in the field of recommender systems.

In addition to that, we have described alternative evaluation methods for active learning in recommender systems, pointing out some important issues that have to be considered when designing a sound evaluation procedure.

We have also identified a number of open issues that would benefit from further studies.

Sequential vs. batch active learning. In a recommender system the users are interested to see that their ratings are immediately reflected in the recommendations generated by the system. However, in many cases the recommender system is retrained only after the user submits a whole batch of ratings. This is called *Batch* active learning. While this approach may be easy and efficient to implement, and it is in fact used by many real-world recommender systems, it is also possible to utilize an alternative approach called *Sequential* active learning. In sequential active learning the items to be rated are selected incrementally, by choosing each item to be rated on the base of an analysis that takes into account the user's ratings provided previously. In fact, by applying this approach, the system can immediately exploit any rating given by the user to better decide which item to select next for the user to rate. For example, the prediction based strategies may use even a single additional rating to rebuild the rating prediction model, which may result in an improved item selection. Hence, an extension of the current batch active learning strategies can be offered by sequential selection of items where at every iteration of the rating elicitation process the item to be rated is selected after the immediate addition to the model of the rating obtained in the previous step [19].

Adaptive active learning with reinforcement learning. It would be also interesting to investigate the application of *Reinforcement Learning* for generating an adaptive active learning system.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Reinforcement learning computes a policy function that decides for each state of the system which action must be executed. In our scenario, actions correspond to selecting items to present to a user for rating. The ultimate goal of the learned policy is to maximize the accumulated reward (benefit achieved making an action) for all the possible system states (e.g., users, or general state of the data set). The key feature of this type of learning is that the learner is not "taught" what to do in the various situations, but instead it adopts a procedure for discovering which action (in each situation) results in the highest reward by trying them. To do so, the learner must sense the state of the system and take actions that change this state and yield a reward [98].

In the context of active learning for collaborative filtering, one may also define a reinforcement learning problem by considering the *actions* as the alternative active learning strategies and the *states* as the situation the system faces over the rating elicitation process. By selecting an active learning strategy, and applying it to the current situation, a *reward* is earned (e.g., reduction of the error), and using any of the standard reinforcement learning procedures, the active learner can discover which strategy is the best for each specific situation (state). We conjecture that reinforcement learning techniques could be used to learn how to select and switch among different active learning strategies according to the current system situation.

User-controlled interface. It has been shown that a proper design and development of the user interface for rating elicitation, namely, by giving to the users more process control, could result in a higher motivation for rating and ultimately can lead to more accurate recommendations [53]. It has also been shown that although users may spend more time interacting with such in-control user interfaces, they actually do not perceive any additional effort [96].

Continuous active learning. Most of the current active learning approaches for collaborative filtering implement the standard user-system interaction model, i.e., selecting and proposing a set of items to users to be rated only during the sign up process, until the user rates a sufficient number of items. An alternative interaction model is the conversational and collaborative one, i.e., not only allowing the user to rate items during the sign-up process, but also, proposing her some additional items afterwards [73]. We believe that it is important to extend the existing active learning strategies to become continuous. The system must identify the most useful items for rating elicitation not only during the sign-up process, but also later on, when the user has already rated a number of items and system can exploit these ratings to better interact with the user and better identifying additional items to rate. However, it is challenging to persuade the user to provide such ratings when she is not as motivated (e.g., since she can already obtain recommendations). In fact the system has to properly justify why the user should rate more items.

Active learning for context information acquisition. In context aware recommender systems (CARS), the context of the user plays significant role in recommendation accuracy [99–104]. While there are several types of contextual factors that can be automatically obtained from sensors (e.g., weather, temperature, location, daytime, season, and weekday), there are other factors that are hard to infer automatically (e.g., budget, companion, mood, and transport mean) [105,80,106]. Moreover, not all the contextual factors are equally useful for the system to improve the recommendation accuracy. Some factors may not influence the users' preferences for items (e.g., the day of the week when the user visited a museum). Hence, actively selecting the contextual factors that are truly informative and relevant is in order to improve the system accuracy and to make easier for the user to enter ratings. We believe that investigating the application of active learning in context-aware recommender systems could lead towards more effective recommender systems [107].

Finally, it is worth noting that active learning for collaborative filtering is a multi disciplinary field overlapping with a broad range of topics: machine learning, data mining, information retrieval, recommender systems, human computer interaction and cognitive science. Hence, our survey is by no means comprehensive, and focused on a selection of aspects that we deem as of practical importance for the design and development of recommender systems.

While this article focused on active learning applications to collaborative filtering, it would be interesting to extend our analysis to other types of recommender systems, such as content-based and context-aware. To the best of our knowledge, such an analysis has not been done so far and could lead to identifying novel areas for the application of active learning in the context of recommender systems.

REFERENCES

[1] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
[2] B. Settles, Active Learning, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers, 2012.
[3] F. Olsson, A Literature Survey of Active Machine Learning in the Context of Natural Language Processing, Tech. Rep. 06, Swedish Institute of Computer Science, Box 1263, SE-164 29 Kista, Sweden, 2009, URL: http://soda.swedish-ict.se/3600/1/SICS-T-2009-06-SE.pdf.
[4] T.M. Mitchell, Machine Learning, first ed., McGraw-Hill, Inc., New York, NY, USA, 1997.
[5] M. Sipser, Introduction to the Theory of Computation, Vol. 27, ACM, New York, NY, USA, 1996, http://dx.doi.org/10.1145/230514.571645.
[6] E. Alpaydin, Introduction to Machine Learning, second ed., The MIT Press, 2010.
[7] Y.S. Abu-Mostafa, M. Magdon-Ismail, H.-T. Lin, Learning From Data, AMLBook, 2012.
[8] P. Flach, Machine Learning: The Art and Science of Algorithms that Make Sense of Data, Cambridge University Press, New York, NY, USA, 2012.
[9] M. Ge, M. Helfert, A review of information quality research—develop a research agenda, in: ICIQ, 2007, pp. 76–91.
[10] B. Settles, Active Learning Literature Survey, Computer Sciences, Technical Report 1648, University of Wisconsin–Madison, 2009.
[11] M. Elahi, F. Ricci, N. Rubens, Active learning strategies for rating elicitation in collaborative filtering: A system-wide perspective, ACM Trans. Intell. Syst. Technol. 5 (1) (2014) 13:1–13:33. http://dx.doi.org/10.1145/2542182.2542195.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

[12] M. Elahi, V. Repsys, F. Ricci, Rating elicitation strategies for collaborative filtering, in: C. Huemer, T. Setzer (Eds.), EC-Web, in: Lecture Notes in Business Information Processing, vol. 85, Springer, 2011, pp. 160–171. http://dx.doi.org/10.1007/978-3-642-23014-1_14.
[13] M. Elahi, F. Ricci, N. Rubens, Active learning in collaborative filtering recommender systems, in: M. Hepp, Y. Hoffner (Eds.), E-Commerce and Web Technologies, in: Lecture Notes in Business Information Processing, vol. 188, Springer International Publishing, 2014, pp. 113–124. http://dx.doi.org/10.1007/978-3-319-10491-1_12.
[14] C. Anderson, The Long Tail, Random House Business, 2006.
[15] P. Resnick, H.R. Varian, Recommender systems, Commun. ACM 40 (3) (1997) 56–58. http://dx.doi.org/10.1145/245108.245121.
[16] U. Shardanand, P. Maes, Social information filtering: Algorithms for automating “word of mouth&rdquo, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'95, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1995, pp. 210–217. http://dx.doi.org/10.1145/223904.223931.
[17] F. Ricci, L. Rokach, B. Shapira, Recommender systems: Introduction and challenges, in: Recommender Systems Handbook, Springer, US, 2015, pp. 1–34. http://dx.doi.org/10.1007/978-1-4899-7637-6_1.
[18] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender Systems: An Introduction, first ed., Cambridge University Press, New York, NY, USA, 2010.
[19] N. Rubens, M. Elahi, M. Sugiyama, D. Kaplan, Active learning in recommender systems, in: Recommender Systems Handbook—Chapter 24: Recommending Active Learning, Springer, US, 2015, pp. 809–846. http://dx.doi.org/10.1007/978-1-4899-7637-6_24.
[20] A.M. Rashid, I. Albert, D. Cosley, S.K. Lam, S.M. McNee, J.A. Konstan, J. Riedl, Getting to know you: Learning new user preferences in recommender systems, in: Proceedings of the 7th International Conference on Intelligent User Interfaces, IUI'02, ACM, New York, NY, USA, 2002, pp. 127–134. http://dx.doi.org/10.1145/502716.502737.
[21] X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2009 (2009) p 4. http://dx.doi.org/10.1155/2009/421425.
[22] R. Burke, Hybrid recommender systems: Survey and experiments, User Model. User-Adapt. Interact. 12 (4) (2002) 331–370. http://dx.doi.org/10.1023/A:1021240730564.
[23] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749. http://dx.doi.org/10.1109/TKDE.2005.99.
[24] M. Balabanović, Y. Shoham, Fab: Content-based, collaborative recommendation, Commun. ACM 40 (3) (1997) 66–72. http://dx.doi.org/10.1145/245108.245124.
[25] P. Lops, M. De Gemmis, G. Semeraro, Content-based recommender systems: State of the art and trends, in: Recommender Systems Handbook, Springer, 2011, pp. 73–105. http://dx.doi.org/10.1007/978-0-387-85820-3_3.
[26] M. de Gemmis, P. Lops, C. Musto, F. Narducci, G. Semeraro, Semantics-aware content-based recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 119–159. http://dx.doi.org/10.1007/978-1-4899-7637-6_4.
[27] M.J. Pazzani, A framework for collaborative, content-based and demographic filtering, Artif. Intell. Rev. 13 (5–6) (1999) 393–408. http://dx.doi.org/10.1023/A:1006544522159.
[28] Y. Wang, S.C.-F. Chan, G. Ngai, Applicability of demographic recommender system to tourist attractions: A case study on trip advisor, in: Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology—Vol. 03, WI-IAT'12, IEEE Computer Society, Washington, DC, USA, 2012, pp. 97–101. http://dx.doi.org/10.1109/WI-IAT.2012.133.
[29] G. Beliakov, T. Calvo, S. James, Aggregation functions for recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 777–808. http://dx.doi.org/10.1007/978-1-4899-7637-6_23.
[30] R.H. Guttman, A.G. Moukas, P. Maes, Agent-mediated electronic commerce: A survey, Knowl. Eng. Rev. 13 (2) (1998) 147–159. http://dx.doi.org/10.1017/S0269888998002082.
[31] S.-L. Huang, Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods, Electron. Commer. Res. Appl. 10 (4) (2011) 398–407. http://dx.doi.org/10.1016/j.elerap.2010.11.003.
[32] R. Burke, Knowledge-based recommender systems, 2000.
[33] A. Felfernig, G. Friedrich, D. Jannach, M. Zanker, Constraint-based recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 161–190. http://dx.doi.org/10.1007/978-1-4899-7637-6_5.
[34] Y. Koren, R. Bell, Advances in collaborative filtering, in: Recommender Systems Handbook, Springer, 2015, pp. 77–118. http://dx.doi.org/10.1007/978-1-4899-7637-6_3.
[35] C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, in: F. Ricci, L. Rokach, B. Shapira, P.B. Kantor (Eds.), Recommender Systems Handbook, Springer, 2011, pp. 107–144. http://dx.doi.org/10.1007/978-0-387-85820-3_4.
[36] M. Degemmis, P. Lops, G. Semeraro, A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation, User Model. User-Adapt. Interact. 17 (3) (2007) 217–255. http://dx.doi.org/10.1007/s11257-006-9023-4.
[37] G. Adomavicius, Y. Kwon, Multi-criteria recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 847–880. http://dx.doi.org/10.1007/978-1-4899-7637-6_25.
[38] G. Linden, B. Smith, J. York, Amazon.com recommendations: Item-to-item collaborative filtering, IEEE Internet Comput. 7 (1) (2003) 76–80. http://dx.doi.org/10.1109/MIC.2003.1167344.
[39] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, CSCW'94, ACM, New York, NY, USA, 1994, pp. 175–186. http://dx.doi.org/10.1145/192844.192905.
[40] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: KDD'08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2008, pp. 426–434. http://dx.doi.org/10.1145/1401890.1401944.
[41] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[42] C.-C. Ma, A guide to singular value decomposition for collaborative filtering, Computer 42 (2009) 30–37.
[43] S. Funk, Netflix update: Try this at home, 2006. http://sifter.org/~simon/journal/20061211.html.
[44] A.I. Schein, A. Popescul, L.H. Ungar, D.M. Pennock, Methods and metrics for cold-start recommendations, in: SIGIR'02: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 2002, pp. 253–260. http://dx.doi.org/10.1145/564376.564421.
[45] Y. Deldjoo, M. Elahi, P. Cremonesi, F. Garzotto, P. Piazzolla, M. Quadrana, Content-based video recommendation system based on stylistic visual features, J. Data Semantics (2016) 1–15. http://dx.doi.org/10.1007/s13740-016-0060-9.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

[46] Y. Deldjoo, M. Elahi, M. Quadrana, P. Cremonesi, Toward building a content-based video recommendation system based on low-level features, in: E-Commerce and Web Technologies—16th International Conference on Electronic Commerce and Web Technologies, EC-Web 2015, Valencia, Spain, September 2015, Revised Selected Papers, 2015, pp. 45–56. http://dx.doi.org/10.1007/978-3-319-27729-5_4.

[47] I.S.C. Nicholas, C.K. Nicholas, Combining content and collaboration in text filtering, in: Proceedings of the IJCAI'99 Workshop on Machine Learning for Information Filtering, 1999, pp. 86–91.

[48] M. Ge, M. Elahi, I. Fernaández-Tobías, F. Ricci, D. Massimo, Using tags and latent factors in a food recommender system, in: Proceedings of the 5th International Conference on Digital Health 2015, DH'15, ACM, New York, NY, USA, 2015, pp. 105–112. http://dx.doi.org/10.1145/2750511.2750528.

[49] I. Cantador, P. Cremonesi, Tutorial on cross-domain recommender systems, in: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys'14, ACM, New York, NY, USA, 2014, pp. 401–402. http://dx.doi.org/10.1145/2645710.2645777.

[50] M. Enrich, M. Braunhofer, F. Ricci, Cold-start management with cross-domain collaborative filtering and tags, in: E-Commerce and Web Technologies—14th International Conference, EC-Web 2013, Prague, Czech Republic, August 27–28, 2013. Proceedings, 2013, pp. 101–112.

[51] M. Elahi, M. Ge, F. Ricci, D. Massimo, S. Berkovsky, Interactive food recommendation for groups, in: RecSysâ14, October 6–10, 2014, Foster City, Silicon Valley, CA, USA, 2014, p. 1.

[52] M. Elahi, M. Ge, F. Ricci, I. Fernández-Tobías, S. Berkovsky, D. Massimo, Interaction design in a mobile food recommender system, in: Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems, IntRS 2015, Co-located with ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 19, 2015, 2015, pp. 49–52.

[53] P. Pu, L. Chen, R. Hu, Evaluating recommender systems from the user's perspective: survey of the state of the art, User Model. User-Adapt. Interact. 22 (4–5) (2012) 317–355. http://dx.doi.org/10.1007/s11257-011-9115-7.

[54] R. Hu, P. Pu, Enhancing collaborative filtering systems with personality information, in: Proceedings of the Fifth ACM Conference on Recommender Systems, RecSys'11, ACM, New York, NY, USA, 2011, pp. 197–204. http://dx.doi.org/10.1145/2043932.2043969.

[55] R. Hu, P. Pu, A comparative user study on rating vs. personality quiz based preference elicitation methods, in: Proceedings of the 14th International Conference on Intelligent User Interfaces, IUI'09, ACM, New York, NY, USA, 2009, pp. 367–372. http://dx.doi.org/10.1145/1502650.1502702.

[56] M. Tkalcic, A. Kosir, J. Tasic, The ldos-peraff-1 corpus of facial-expression video clips with affective, personality and user-interaction metadata, J. Multimodal User Interfaces 7 (1–2) (2013) 143–155. http://dx.doi.org/10.1007/s12193-012-0107-7.

[57] M. Trevisiol, L.M. Aiello, R. Schifanella, A. Jaimes, Cold-start news recommendation with domain-dependent browse graph, in: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys'14, ACM, New York, NY, USA, 2014, pp. 81–88. http://dx.doi.org/10.1145/2645710.2645726.

[58] S. Tong, Active learning: Theory and applications (Ph.D. thesis), The Department of Computer Science, 2001.

[59] A. Kohrs, B. Merialdo, Improving collaborative filtering for new users by smart object selection, in: Proceedings of International Conference on Media Features, ICMF, 24.

[60] I.R. Teixeira, F.d.A.T.d. Carvalho, G. Ramalho, V. Corruble, Activecp: A method for speeding up user preferences acquisition in collaborative filtering systems, in: Proceedings of the 16th Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence, SBIA'02, Springer-Verlag, London, UK, 2002, pp. 237–247. URL: http://dl.acm.org/citation.cfm?id=645853.669613.

[61] N. Rubens, M. Sugiyama, Influence-based collaborative active learning, in: Proceedings of the 2007 ACM Conference on Recommender Systems, RecSys'07, ACM, New York, NY, USA, 2007, pp. 145–148. http://dx.doi.org/10.1145/1297231.1297257.

[62] M. Elahi, F. Ricci, V. Repsys, System-wide effectiveness of active learning in collaborative filtering, in: F. Bonchi, W. Buntine, R. Gavald, S. Gu (Eds.), International Workshop on Social Web Mining, Co-located with IJCAI, Universitat de Barcelona, Spain, 2011, p. 1.

[63] S. Kutty, F. Yu, Recommendations and Predictions with ET-Greedy Active Learning, Tech. Rep., Electrical Engineering and Computer Science Department of University of Michigan, 2009.

[64] N. Golbandi, Y. Koren, R. Lempel, Adaptive bootstrapping of recommender systems using decision trees, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM'11, ACM, New York, NY, USA, 2011, pp. 595–604. http://dx.doi.org/10.1145/1935826.1935910.

[65] S. Chang, F.M. Harper, L. Terveen, Using groups of items for preference elicitation in recommender systems, in: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, CSCW'15, ACM, New York, NY, USA, 2015, pp. 1258–1269. http://dx.doi.org/10.1145/2675133.2675210.

[66] B. Loepp, T. Hussein, J. Ziegler, Choice-based preference elicitation for collaborative filtering recommender systems, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'14, ACM, New York, NY, USA, 2014, pp. 3085–3094. http://dx.doi.org/10.1145/2556288.2557069.

[67] C. Boutilier, R.S. Zemel, B. Marlin, Active collaborative filtering, in: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 2002, pp. 98–106.

[68] A.M. Rashid, G. Karypis, J. Riedl, Learning preferences of new users in recommender systems: an information theoretic approach, SIGKDD Explor. Newsl. 10 (2008) 90–100. http://dx.doi.org/10.1145/1540276.1540302.

[69] N. Golbandi, Y. Koren, R. Lempel, On bootstrapping recommender systems, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM'10, ACM, New York, NY, USA, 2010, pp. 1805–1808. http://dx.doi.org/10.1145/1871437.1871734.

[70] N.N. Liu, X. Meng, C. Liu, Q. Yang, Wisdom of the better few: cold start recommendation via representative based rating elicitation, in: Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23–27, 2011, 2011, pp. 37–44. http://dx.doi.org/10.1145/2043932.2043943.

[71] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys'10, ACM, New York, NY, USA, 2010, pp. 39–46. http://dx.doi.org/10.1145/1864708.1864721.

[72] R.A. Horn, C.R. Johnson (Eds.), Matrix Analysis, Cambridge University Press, New York, NY, USA, 1986.

[73] G. Carenini, J. Smith, D. Poole, Towards more conversational and collaborative recommender systems, in: Proceedings

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

of the 8th International Conference on Intelligent User Interfaces, IUI'03, ACM, New York, NY, USA, 2003, pp. 12–18. http://dx.doi.org/10.1145/604045.604052.

[74] L. He, N.N. Liu, Q. Yang, Active dual collaborative filtering with both item and attribute feedback, in: W. Burgard, D. Roth (Eds.), AAAI, AAAI Press, 2011, pp. 1186–1191. URL: http://dblp.uni-trier.de/db/conf/aaai/aaai2011.html#HeLY11.

[75] C.E. Mello, M.-A. Aufaure, G. Zimbrao, Active learning driven by rating impact analysis, in: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys'10, ACM, New York, NY, USA, 2010, pp. 341–344. http://dx.doi.org/10.1145/1864708.1864782.

[76] R. Karimi, C. Freudenthaler, A. Nanopoulos, L. Schmidt-Thieme, Non-myopic active learning for recommender systems based on matrix factorization, in: IRI, IEEE Systems, Man, and Cybernetics Society, 2011, pp. 299–303. http://dx.doi.org/10.1109/IRI.2011.6009563.

[77] M. Elahi, M. Braunhofer, F. Ricci, M. Tkalcic, Personality-based active learning for collaborative filtering recommender systems, in: AI* IA 2013: Advances in Artificial Intelligence, Springer International Publishing, 2013, pp. 360–371. http://dx.doi.org/10.1007/978-3-319-03524-6_31.

[78] R.R. McCrae, O.P. John, An introduction to the five-factor model and its applications, J. Pers. 60 (2) (1992) 175–215. http://dx.doi.org/10.1111/j.1467-6494.1992.tb00970.x.

[79] S.D. Gosling, P.J. Rentfrow, W.B. Swann Jr., A very brief measure of the big-five personality domains, J. Res. Pers. 37 (6) (2003) 504–528.

[80] M. Braunhofer, M. Elahi, F. Ricci, Techniques for cold-starting context-aware mobile recommender systems for tourism, Intell. Artif. 8 (2) (2014) 129–143. http://dx.doi.org/10.3233/IA-140069.

[81] M. Braunhofer, M. Elahi, M. Ge, F. Ricci, Context dependent preference acquisition with personality-based active learning in mobile recommender systems, in: P. Zaphiris, A. Ioannou (Eds.), Learning and Collaboration Technologies. Technology-Rich Environments for Learning and Collaboration, in: Lecture Notes in Computer Science, vol. 8524, Springer International Publishing, 2014, pp. 105–116. http://dx.doi.org/10.1007/978-3-319-07485-6_11.

[82] Z. Huang, D.D. Zeng, Why does collaborative filtering work? Transaction-based recommendation model validation and selection by analyzing bipartite random graphs, INFORMS J. Comput. 23 (1) (2011) 138–152. http://dx.doi.org/10.1287/ijoc.1100.0385.

[83] R. Jin, L. Si, A Bayesian approach toward active learning for collaborative filtering, in: UAI'04, Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence, July 7–11 2004, Banff, Canada, 2004, pp. 278–285.

[84] A.S. Harpale, Y. Yang, Personalized active learning for collaborative filtering, in: SIGIR'08: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 2008, pp. 91–98. http://dx.doi.org/10.1145/1390334.1390352.

[85] R. Karimi, C. Freudenthaler, A. Nanopoulos, L. Schmidt-Thieme, Active learning for aspect model in recommender systems, in: CIDM, IEEE, 2011, pp. 162–167. http://dx.doi.org/10.1109/CIDM.2011.5949431.

[86] M. Sun, F. Li, J. Lee, K. Zhou, G. Lebanon, H. Zha, Learning multiple-question decision trees for cold-start recommendation, in: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, WSDM'13, ACM, New York, NY, USA, 2013, pp. 445–454. http://dx.doi.org/10.1145/2433396.2433451.

[87] R. Karimi, A. Nanopoulos, L. Schmidt-Thieme, A supervised active learning framework for recommender systems based on decision trees, User Model. User-Adapt. Interact. 25 (1) (2015) 39–64. http://dx.doi.org/10.1007/s11257-014-9153-z.

[88] F. Hu, Y. Yu, Interview process learning for top-n recommendation, in: Proceedings of the 7th ACM Conference on Recommender Systems, RecSys'13, ACM, New York, NY, USA, 2013, pp. 331–334. http://dx.doi.org/10.1145/2507157.2507205.

[89] S.-L. Lee, Commodity recommendations of retail business based on decisiontree induction, Expert Syst. Appl. 37 (5) (2010) 3685–3694. http://dx.doi.org/10.1016/j.eswa.2009.10.022.

[90] K. Zhou, S.-H. Yang, H. Zha, Functional matrix factorizations for cold-start recommendation, in: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'11, ACM, New York, NY, USA, 2011, pp. 315–324. http://dx.doi.org/10.1145/2009916.2009961.

[91] M. Elahi, F. Ricci, N. Rubens, Adapting to natural rating acquisition with combined active learning strategies, in: ISMIS'12: Proceedings of the 20th International Conference on Foundations of Intelligent Systems, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 254–263. http://dx.doi.org/10.1007/978-3-642-34624-8_30.

[92] G. Shani, A. Gunawardana, Evaluating recommendation systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer Verlag, 2010, pp. 257–298. http://dx.doi.org/10.1007/978-0-387-85820-3_8.

[93] S.-T. Park, W. Chu, Pairwise preference regression for cold-start recommendation, in: Proceedings of the Third ACM Conference on Recommender Systems, RecSys'09, ACM, New York, NY, USA, 2009, pp. 21–28. http://dx.doi.org/10.1145/1639714.1639720.

[94] W. Zeng, M.-S. Shang, T.-Y. Qian, Useful acquiring ratings for collaborative filtering, in: IEEE Youth Conference on Information, Computing and Telecommunication, 2009, YC-ICT'09, 2009, pp. 483–486. http://dx.doi.org/10.1109/YCICT.2009.5382452.

[95] D. Kluwer, J.A. Konstan, Evaluating recommender behavior for new users, in: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys'14, ACM, New York, NY, USA, 2014, pp. 121–128. http://dx.doi.org/10.1145/2645710.2645742.

[96] S.M. McNee, S.K. Lam, J.A. Konstan, J. Riedl, Interfaces for eliciting new user preferences in recommender systems, in: Proceedings of the 9th International Conference on User Modeling, UM'03, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 178–187. URL: http://dl.acm.org/citation.cfm?id=1759957.1759988.

[97] I. Fernandez Tobias, M. Braunhofer, M. Elahi, F. Ricci, C. Ivan, Alleviating the new user problem in collaborative filtering by exploiting personality information, User Model. User-Adapt. Interact. (UMUAI) 26 (2016) (Personality in Personalized Systems) http://dx.doi.org/10.1007/s11257-016-9172-z.

[98] R.S. Sutton, A.G. Barto, Introduction to Reinforcement Learning, first ed., MIT Press, Cambridge, MA, USA, 1998.

[99] G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 191–226. http://dx.doi.org/10.1007/978-1-4899-7637-6_6.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002

[100] L. Baltrunas, B. Ludwig, F. Ricci, Matrix factorization techniques for context aware recommendation, in: Proceedings of the Fifth ACM Conference on Recommender Systems, RecSys'11, ACM, New York, NY, USA, 2011, pp. 301–304. http://dx.doi.org/10.1145/2043932.2043988.

[101] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional approach, ACM Trans. Inf. Syst. 23 (1) (2005) 103–145. http://dx.doi.org/10.1145/1055709.1055714.

[102] M. Braunhofer, M. Elahi, F. Ricci, User personality and the new user problem in a context-aware point of interest recommender system, in: Information and Communication Technologies in Tourism 2015, Springer International Publishing, 2015, pp. 537–549. http://dx.doi.org/10.1007/978-3-319-14343-9_39.

[103] G. Adomavicius, B. Mobasher, F. Ricci, A. Tuzhilin, Context-aware recommender systems, AI Mag. 32 (3) (2011) 67–80. http://dx.doi.org/10.1609/aimag.v32i3.2364.

[104] S. Anand, B. Mobasher, Contextual recommendation, in: B. Berendt, A. Hotho, D. Mladenic, G. Semeraro (Eds.), From Web to Social Web: Discovering and Deploying User and Content Profiles, in: Lecture Notes in Computer Science, vol. 4737, Springer, Berlin, Heidelberg, 2007, pp. 142–160. http://dx.doi.org/10.1007/978-3-540-74951-6_8.

[105] M. Braunhofer, M. Elahi, M. Ge, F. Ricci, Context dependent preference acquisition with personality-based active learning in mobile recommender systems, in: Learning and Collaboration Technologies. Technology-Rich Environments for Learning and Collaboration, Springer International Publishing, 2014, pp. 105–116. http://dx.doi.org/10.1007/978-3-319-07485-6_11.

[106] M. Braunhofer, I. Fernández-Tobìas, F. Ricci, Parsimonious and adaptive contextual information acquisition in recommender systems, in: Proceedings of IntRs'15.

[107] L. Baltrunas, Context-aware collaborative filtering recommender systems (Ph.D. thesis), The Department of Computer Science, Free University of Bozen–Bolzano, 2011.

Please cite this article in press as: M. Elahi, et al., A survey of active learning in collaborative filtering recommender systems, Computer Science Review (2016), http://dx.doi.org/10.1016/j.cosrev.2016.05.002