Information Sciences 423 (2018) 128–144

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Metalearning and Recommender Systems: A literature review and empirical study on the algorithm selection problem for Collaborative Filtering

Tiago Cunha[a,*], Carlos Soares[a], André C.P.L.F. de Carvalho[b]

[a] Faculdade de Engenharia da Universidade do Porto, Portugal
[b] ICMC, Universidade de São Paulo - São Carlos, Brasil

ARTICLE INFO

Article history:
Received 15 March 2017
Revised 4 September 2017
Accepted 18 September 2017
Available online 20 September 2017

Keywords:
Metalearning
Algorithm selection
Recommendation system
Collaborative Filtering

ABSTRACT

The problem of information overload motivated the appearance of Recommender Systems. From the several open problems in this area, the decision of which is the best recommendation algorithm for a specific problem is one of the most important and less studied. The current trend to solve this problem is the experimental evaluation of several recommendation algorithms in a handful of datasets. However, these studies require an extensive amount of computational resources, particularly processing time. To avoid these drawbacks, researchers have investigated the use of Metalearning to select the best recommendation algorithms in different scopes. Such studies allow to understand the relationships between data characteristics and the relative performance of recommendation algorithms, which can be used to select the best algorithm(s) for a new problem. The contributions of this study are two-fold: 1) to identify and discuss the key concepts of algorithm selection for recommendation algorithms via a systematic literature review and 2) to perform an experimental study on the Metalearning approaches reviewed in order to identify the most promising concepts for automatic selection of recommendation algorithms.

1. Introduction

Recommender Systems (RSs) were proposed to deal with the problem of information overload [5]. The basic idea is to embody a system with the ability to recommend to a specific user the most relevant items among a set of existing options. The success of these systems is clear in both academia and industry [25]. However, the recommendation problem is not without flaws. One of the current challenges is the lack of guidance regarding which RS algorithm would be more adequate for a given task, and, more importantly, why. Current strategies to deal with this problem involve the evaluation of every available algorithm for each problem to decide which one is more adequate. This is an expensive approach, not only regarding time, but also considering human and computational resources. An automatic alternative for algorithm selection that overcome such drawbacks is Metalearning (MtL).

MtL [6] uses Machine Learning (ML) techniques to obtain predictive models, which associate characteristics of tasks to the respective performance of algorithms. The methodology initially involves extracting characteristics from a dataset, named metafeatures, and assessing the relative performance of a group of algorithms, which will be used as metalabels.

* Corresponding author.
E-mail addresses: tiagodscunha@fe.up.pt, tiagodscunha@gmail.com (T. Cunha), csoares@fe.up.pt (C. Soares), andre@icmc.usp.br (A.C.P.L.F. de Carvalho).

https://doi.org/10.1016/j.ins.2017.09.050
0020-0255/

Afterwards, this information for several datasets is used to induce a predictive model able to represent the relationship between the metafeatures and metalabels of these datasets. If this model has a high predictive performance, it would will be able to predict the most promising algorithm without the overhead of running a full-fledged empirical evaluation and to explain why RS algorithms perform best/worst [41].

This work focuses first on providing a systematic literature review on the problem of algorithm selection for RSs. The related work is reviewed on the key dimensions required to solve the algorithm selection problem. In each dimension, the extent of the research conducted so far is discussed and lines of research for future work are suggested. Afterwards, the most suitable related work strategies for algorithm selection for Collaborative Filtering (CF) are experimentally evaluated. The experiments allow to compare the effects of the different metafeatures on the same experimental setup. A large experimental study of baselevel datasets, algorithms and evaluation metrics is conducted. Afterwards, each MtL strategy is implemented and evaluated on the same metalevel setup, i.e. same metatarget, meta-algorithms and evaluation measures. Finally, conclusions are drawn from the behavior of the current state of the art approaches with regards to several aspects of the algorithm selection problem.

This document is organized as follows: Section 2 presents a brief review on RS, focusing on CF and the evaluation protocol. Section 3 presents the methodology of algorithm selection using MtL. Next, Section 4 presents the process for systematic review on the related work about algorithm selection for RSs. Section 5 is responsible to present the empirical study performed on a subset of the Metalearning approaches discussed previously. Section 6 highlights the final conclusions and points out future work directions.

2. Recommender Systems

This section presents the basic aspects of RSs, with a focus on CF recommendations. It also describes how the performance RSs are usually evaluated.

2.1. Overview

Several RSs have been proposed in the last decades. One of the main aspects that differentiates these systems is the followed approach. These strategies can be roughly divided into eight approaches: Collaborative Filtering (CF) [37], Content based Filtering [4], Social based Filtering [23], Knowledge based Filtering [4], Hybrid Filtering [7], Context-aware Filtering [1], Deep Learning-based Recommendations [20,44] and Group Recommendations (GR) [30]. Since the majority of the focus of the MtL studies lies on CF, only this strategy will be further discussed in this paper. Further information regarding the remaining strategies is available elsewhere [1,5,46].

2.2. Collaborative Filtering

CF recommendations are based on the premise that a user will like the items favored by a similar user, which is an user with similar preferences. It uses the (implicit or explicit) feedback from each individual user to recommended items among similar users [46]. Explicit feedback is a numerical value, proportional to the user's likeliness towards an item. Implicit feedback, on the other hand, derives a numerical value from the user's interactions with the item (click-through data, like/dislike actions, time spent on a task, etc.). The data structure used in CF is known as the rating matrix R. It is described as $R^{U \times I}$, representing a set of users U, where $u \in \{1, \ldots, N\}$ and a set of items I, where $i \in \{1, \ldots, M\}$. Each element R_{ui} is the user u feedback to item i. Fig. 1 presents such matrix.

The CF algorithms are organized into two major classes: memory-based and model-based [5]. Memory-based algorithms apply heuristics to a rating matrix to compute recommendations, whereas model-based algorithms induce a model from a rating matrix and use the model to recommend items. Memory-based algorithms are mostly represented by Nearest Neighbor (NN) strategies, while model-based algorithms work with Matrix Factorization (MF). A memory-based algorithm follows 3 steps: 1) calculate the similarity between users or items, 2) create a neighborhood of similar users or items and 3) generate recommendations by sorting similar items. MF assumes that the original rating matrix values can be approximated by the multiplication of matrices with latent features that capture the underlying data patterns. Several MF algorithms have been successfully used in CF, namely, SVD++, SGD and ALS. Further information regarding this topic is available elsewhere [21].

2.3. Evaluation

The evaluation of RSs can be performed by offline and online procedures. Offline evaluation splits the dataset into training and testing subsets (using sampling strategies, such as k-fold cross-validation [18]) and assesses the performance of the trained model on the testing dataset (see Fig. 2). However, in order to compare the predicted and original values, there is an important difference from conventional k-fold cross validation: the test data must be split into hidden and observable instances.

Another important issue lies in the scopes for the evaluation metrics [25]:

- for rating accuracy, error metrics such as Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE) must be used;

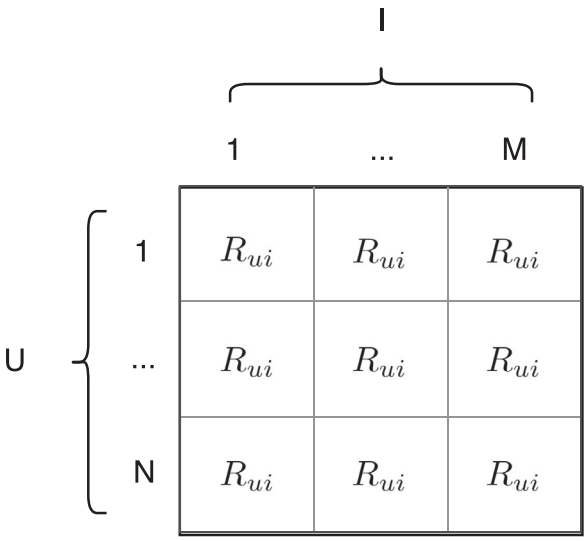

Fig. 1. Rating matrix.

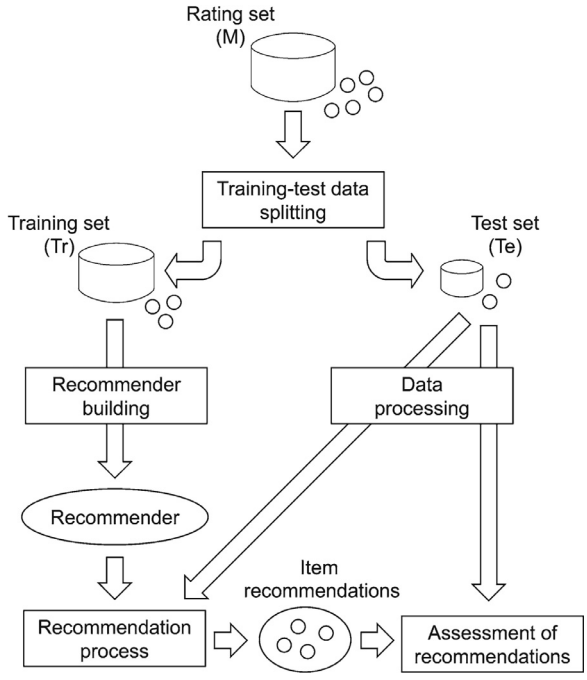

Fig. 2. Diagram of offline evaluation in RS [8].

- for classification accuracy, one must use Precision/Recall measures or Area Under the Curve (AUC);
- for ranking accuracy, the typical metrics are Normalized Discounted Cumulative Gain (NDCG), Mean Reciprocal Rank (MRR) and Expected Reciprocal Rank (ERR) [22,24]

The online evaluation arose from the necessity of assessing other aspects beyond those accessible offline, in which the actual input of a real user cannot be reproduced in a better way [18]. The most popular metric is the user acceptance ratio, which refers to the amount of items the user liked divided by the total amount of items recommended to the user. Additional online evaluation metrics are described elsewhere [43].

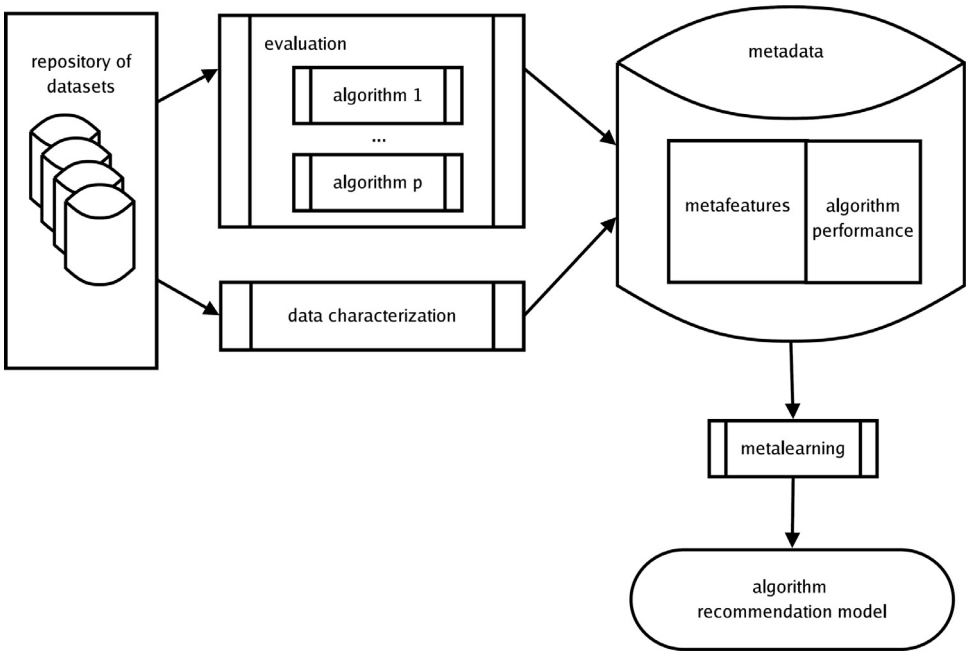

Fig. 3. Training process of a Metalearning process [6].

3. Metalearning

MtL is concerned with discovering patterns in data and understanding the effect on the behavior of algorithms [41]. It has been extensively used for algorithm selection [32,35,39]. The algorithm selection task can be viewed as a learning problem itself, by casting it as a predictive task. For such, it uses a metadataset, where each meta-example corresponds to a dataset. For each meta-example, the predictive features are characteristics (metafeatures) extracted from the corresponding dataset and the targets are the performance (metalabels) of a set of algorithms when they are applied to the dataset [6].

A MtL process addresses the algorithm selection problem similarly to a traditional learning process. It induces a metamodel, which can be seen as the metaknowledge able to explain why certain algorithms work better than others in datasets with specific characteristics. The metamodel can also be used to predict the best algorithm(s) for a new dataset/problem [38]. Fig. 3 presents the training stage of the MtL process.

One of the main challenges in MtL is to define which are the metafeatures that effectively describe how strongly a dataset matches the bias of each algorithm [6]. The MtL literature divides the metafeatures into three main groups [38,41]: statistical and/or information-theoretical, model-based and landmarkers.

Statistical and/or information-theoretical metafeatures describe the dataset characteristics using a set of measures from statistics and information theory. These metafeatures assume that there are patterns in the dataset which can be related to the most suitable algorithms for these datasets. Examples include simple measures, such as the number of examples and features in the dataset, to more advanced measures, such as entropy, skewness and kurtosis of features and even mutual information and correlation between features [6].

Model-based characteristics are properties extracted from models induced from the dataset. In a classification or regression MtL scenario, they refer, for instance, to the number of leaf nodes in a decision tree [6]. The rationale is that there is a relationship between model characteristics and algorithm performance which cannot be captured from the data directly. For instance, a decision tree with an unusually large number of leaf nodes may indicate overfitting.

Finally, landmarkers are fast estimates of the algorithm performance on the dataset. There are two different types of landmarkers: those obtained from the application of fast and simple algorithms on complete datasets (e.g. a decision stump can be regarded as a simplified version of a decision tree) and those which are achieved by using complete models for samples of datasets, also known as subsampling landmarkers [6] (e.g. applying the full decision tree on a sample).

4. Algorithm selection for Recommender Systems

The problem of algorithm selection was first conceptualized by Rice [33], which defined the following search spaces:

- the problem space P, representing the set of instances of a problem;
- the feature space F, containing measurable characteristics of the instances generated by a feature extraction process applied to an instance of P;

- the algorithm space A, as the set of all suitable algorithms for the problem;
- the performance space Y, which represents the mapping of each algorithm to a set of performance measures;

Considering this framework, the problem of algorithm selection can be stated as: "*for a given problem instance $x \in P$, with features $f(x) \in F$, find the selection mapping $S(f(x))$ into the algorithm space A, such that the selected algorithm $\alpha \in A$ maximizes the performance mapping $y(\alpha(x)) \in Y$.* " [33].

In the scope of this work, the problem space P is a set of recommendation datasets, the feature space F is composed by the metafeatures used to describe the datasets, the algorithm space A contains all the available algorithms and the performance space Y is described by a set of suitable evaluation measures.

4.1. Methodology

In order to perform a systematic review on algorithm selection for RSs, several online databases were consulted: Elsevier's Scopus, Thomson Reuters's Web of Science, IEEE Xplore Digital Library, Google Scholar and ACM Digital Library. The keywords combined the keywords: *"Recommender System", "metalearning", "algorithm selection"* and *"performance prediction"*. This document reviews only approaches that relate the performance of RS algorithms with data characteristics.

4.2. Research questions

Considering the MtL workflow, the most important dimensions of the algorithm selection problem for RS were identified. The dimensions are split into of base- and metalevels and are translated into Research Questions (RQ):

- **RQ1:** What is the coverage of recommendation strategies in MtL studies?
- **RQ2:** Are the public datasets used representative and enough?
- **RQ3:** Is the pool of recommendation algorithms suitable and complete?
- **RQ4:** How well are the RSs performance measures covered?
- **RQ5:** Are the metafeatures used diversified in nature and enough?
- **RQ6:** In the studies, what is the typical metatarget?
- **RQ7:** Which algorithms are employed in the metalevel?
- **RQ8:** Are the evaluation measures used in the metalevel suitable?

4.3. Related work

The literature provides a few approaches for the RS algorithm selection problem. The first work studies the CF algorithm selection problem by mapping the data onto a graph instead of a rating matrix [19]. Graph-dependent metafeatures are derived to select among NN algorithms. The selection process takes advantage of a domain-dependent rules-based model.

Other papers analyzed the rating matrix using statistical and/or information-theoretical metafeatures to select among NN and MF algorithms [2,13]. The problems were addressed as regression tasks, where the goal was to optimize the RMSE performance.

An alternative approach, using a decision tree regression model, was developed later [16]. It studied the underlying relationship between user ratings and neighborhood data and the expected error of the recommendations of a NN algorithm. This approach focused on describing metafeatures for users, instead of the entire dataset.

Another proposal looked to the co-ratings instead of the original ratings [26]. This extended the original data available in the rating matrix with relationships among users. It uses a regression model to predict the RMSE performance of NN and MF algorithms.

The algorithm selection problem was extended beyond CF, when a GR approach was proposed [49]. It used several classification algorithms to rank vote aggregation algorithms. For such, it derives domain-dependent metafeatures and relates them with the performance of the algorithms in terms of error.

The most comprehensive MtL study for CF algorithm selection studies the selection of MF algorithms using an extensive experimental setup [10]. It considers multiple algorithms and evaluation metrics at both the base- and metalevels. Furthermore, it proposed a set of systematically generated metafeatures which extends the union of almost all the ones available in the previous literature.

The related work for the literature review is presented in Table 1. Each work is described in terms of the RQ identified earlier. Some RQs are sub-divided, using the notation:

- The baselevel algorithms (RQ3) are organized by type - Heuristics (H), Nearest Neighbors (NN), Matrix Factorization (MF) and others (O).
- The base-evaluation measures (RQ4) are organized into error based (E), classification accuracy (CA) and ranking accuracy (RA).
- The metafeatures (RQ5) are divided according to the subject evaluated: user (U), item (I), ratings (RT), data structure (S) and others (O).
- The metatargets (RQ6) are: 1) best algorithm (BA), 2) ranking of algorithms (RA) and 3) performance estimation (PE).
- The metalevel algorithms (RQ7) use classification (C), regression (RG) or other (O) types of algorithms.

Table 1
Summary table on the algorithm selection problem for RS. Each column refers to the RQ identified in this work (see Section 4.2): RQ1 (recommendation strategies), RQ2 (baselevel datasets), RQ3 (baselevel algorithms), RQ4 (baselevel evaluation measures), RQ5 (metafeatures), RQ6 (metatarget), RQ7 (meta-algorithms) and RQ8 (metalevel evaluation). RQ 3, 4, 5, 6 and 7 are divided in several categories (see Section 4.3).

Ref.	Baselevel									Metalevel								
	RQ1	RQ2	RQ3				RQ4			RQ5					RQ6	RQ7			RQ8
			H	NN	MF	O	E	CA	RA	U	I	RT	S	O		C	RG	O	
[19]	CF	3	2	2	–	–	–	3	1	–	–	–	2	2	BA	–	–	1	AUC
[2]	CF	4	–	2	1	–	1	–	–	1	1	1	3	–	PE	–	1	–	Correlation
[13]	CF	1	1	2	1	1	1	–	–	3	–	–	–	–	PE	–	1	–	Correlation
[16]	CF	3	–	1	–	–	1	–	–	11	–	–	–	–	PE	–	1	–	MAE
[26]	CF	4	–	1	1	–	1	–	–	–	–	–	3	–	PE	–	1	–	Correlation
[49]	GR	4	11	–	–	–	1	–	–	5	–	–	–	–	RA	4	–	–	MRR
[10]	CF	32	4	–	13	–	3	1	3	30	30	10	4	–	BA	10	–	–	Accuracy

4.4. Discussion

This section analyses the results presented in Table 1 regarding the several aspects identified in the RQ (see Section 4.2).

4.4.1. Recommendation strategies

Since this research area is still in the early stages (all works were published in the last 6 years), it is expected that only a few RS strategies would have been studied. In fact, like in the RS research area, the majority of the researches has been performed on CF. This is justified by the lack of public frameworks beyond this recommendation strategy. The exception is a recent study on the algorithm selection problem for Group Recommendation (GR) [49]. Therefore, it is essential to 1) expand the scope of RS strategies studied and 2) perform a deeper analysis of the algorithm selection problem for CF.

4.4.2. Datasets

Most related work use at most 4 datasets to investigate algorithm selection, with the exception of the most recent work [10]. While on some cases this may be acceptable if the problem is appropriately modeled (for instance, select the best algorithm for each user instead of per each dataset [16]), this small number is usually a drawback. In fact, algorithm selection strategies must use a large amount of diverse datasets to ensure a proper exploration of the problem space P. This dimension must be improved, although there are few public datasets.

The public datasets used in the related work are: Amazon Reviews [27], BookCrossing [50], Epinions [34], Flixter [48], Jester [15], LastFM [3], MovieLens [17], MovieTweetings [12], Netflix [29], Tripadvisor [42] and Yahoo! Music [45]. Only two works use private data [19,49], which are excluded from further analysis.

The results show a large variation in how frequent each dataset is used in these works. The most common dataset belongs to the MovieLens category (used 5 times out of 7). This follows the trend in the RS research area, where these datasets are considered benchmarks. The second choices fell on the Flixter, Jester and Netflix datasets. While the first two are used in about half of the works, the Netflix dataset is more frequent and should be present in more studies. It became popularity after a world-wide competition that finished in 2009 [21]. However, the main difficulty lies with the fact that the dataset is no longer available for download. The remaining datasets appear only once.

Although it is important to analyze the related work in terms of amount and source of the datasets, it is even more important to review some of their characteristics. The inspection of the datasets characteristics illustrates the following aspects: 1) the number of users and items is rather small when compared to the number of ratings, 2) the sparsity values are usually greater than 0.9 and 3) the most common rating scale ranges from 1 to 5. Overall, the datasets cover a wide range in terms of scale and structure of the rating matrix in CF. This shows that the collection of public datasets is representative, although not exhaustive. Several points need to be improved, including the analysis of different levels of sparsity and understanding the effects on the performance and to extend the types of scales used. These improvements can be obtained by creating artificial datasets. This can be achieved by applying permutations to the original datasets (maintaining some global characteristics), by creating entirely new data recurring to well-known distributions, by adding noise to the datasets or by creating artificial datasets using simulation [9].

4.4.3. Baselevel algorithms

The baselevel algorithms frequently used are distributed in the following categories: Heuristics (18), MF (16), NN (8) and Others (1). The fact that MF and NN approaches are abundant is expected, since they are the most commonly used in CF. The large amount of heuristics refers mostly to the GR approach, which studies 11 algorithms of this nature. In CF, heuristics are typically associated with naive approaches, such as random and most popular algorithms. It is also clear that newer studies change the focus from NN algorithms to MF. This is expected, since MF is now the standard in CF.

The most frequently used algorithms are user-based NN and SVD++, closely followed by item-based NN. This represents the most basic approaches for CF and are therefore available in a larger amount of recommendation platforms. Newer approaches, such as MF (besides SVD++), are usually more difficult to find in recommendation platforms. This is an important limitation, given MF's relevance. Averages and most popular algorithms are more common than a random approach. This is expected, since they are better baselines [21].

Although the algorithms used are suitable for CF, the authors do not know of any study that takes in account a complete and diverse pool of algorithms. Even the most recent and complete study fails in pursuing NN algorithms [10]. In order to properly explore the algorithm space A for CF, it is important to evaluate the new algorithms. Nowadays, there is a large number of resources to conduct experimental evaluations, specially for CF. However, none of them deals with the algorithm selection problem. A platform for model management that could integrate several RS algorithms and datasets, while providing faithful and fair evaluation would be the best solution to this problem. However, the closest systems are frameworks devoted solely to the evaluation of RSs [36].

4.4.4. Baselevel evaluation

Table 1 shows that most evaluation measures are error based (used in 6 out of 7 works). On the other hand, accuracy measures (either classification or ranking based) are only used in 2 works. The most frequently used error measures are RMSE and MAE, classification accuracy evaluated through precision, recall and AUC and ranking accuracy assessed by MRR and NDCG. The evaluation procedures usually assess only one aspect of the recommendation process, contradicting the guidelines from the RS literature [18]. In fact, only 2 works expanded the evaluation scope [10,19]. This clearly demonstrates the incompleteness of the evaluation in the related work. Further investigations are required to improve the exploration of the performance space Y, including increase the scope of offline evaluation measures and perform the same studies using online rather than offline evaluation procedures. The first can be achieved by using recent, yet not fully validated, measures. These try to assess non standard dimensions of the RS problem such as novelty, satisfaction and diversity. Also, online evaluation measures could be used to compare the actual feedback with the predictions, since they are claimed to be better.

4.4.5. Metafeatures

The metafeatures used in the related works are all statistical and information-theoretical measures. They can be organized into several categories: user (50), item (31), ratings distribution (11), data structure (12) and others (2). Most of them focuses on users. In fact, some research uses only metafeatures of this dimension of the problem [13,16,49]. Characteristics related to the data structure are also common and are available in 4 out of 7 works. One work used all previous metafeatures in an user co-occurrence matrix [26].

The number of metafeatures used usually ranges from 3 to 11, with only one exception that increases this number to 74 [10]. However, this is not necessarily an advantage. A smaller number avoids the curse of dimensionality and is more suitable when a small number of baseline algorithms are evaluated. However, assuming the existence of more recommendation algorithms, further and more diverse metafeatures should be studied to properly explore the feature space F.

The analysis of metafeatures in a deeper level allows us to understand which type of statistical and information-theoretical measures were used in the related work. The functions used are mostly ratios, averages and sums. This is expected, since they are the simplest metafeatures found in the MtL literature. Entropy, Gini index and standard deviation appear in the second position (in 3 works). All other functions appear only once.

Despite the fact that diverse metafeatures are proposed, few studies look towards different aspects of the problem. In fact, 4 studies focus their metafeatures on a single subject, which typically is the user. Plus, there are few examples of metafeatures that look towards relationships between the different subjects of the problem. This makes difficult to find complex patterns in the data, restricting the metaknowledge extracted. Recent works have successfully extended the nature and amount of metafeatures, improving dataset characterization for CF algorithm selection. These extensions include: 1) propose and adapt new metafeatures for other RS strategies; 2) propose problem-specific (and eventually domain-specific) metafeatures and 3) study new metafeatures besides statistical and/or information-theoretical, such as for instance, landmarkers. This list is by no means exhaustive, since it is difficult to anticipate the necessities to properly describe the datasets of other RS problems besides CF. It is especially difficult when it comes to guarantee their informative power.

4.4.6. Metatarget

Related work on CF algorithm selection has adopted several metatargets. The most common is the performance estimation (PE), available in 4 out of 7 works. There are also two examples of best algorithm (BA) and ranking (RA) selections. Although recent works have looked beyond PE [10,49], real applications can benefit from the use of BA or RA due to interpretability issues.

4.4.7. Metalevel algorithms

Usually, only one algorithm is used in the metalevel. This algorithm must match the required metatarget. As can be seen in Table 1, when PE is used as metatarget, regression algorithms are used (mainly linear regression algorithms). For BA and RA, popular classification algorithms, like rule based classifiers, Naive Bayes, SVM and kNN, are used. An exception happens when a custom procedure based on rules is used [19]. Although the number of algorithms used in the metalevel does not have the same impact as the number used in the baselevel, the use of a larger and more diverse set of algorithms in the

metalevel increases the chances of uncover hidden relationships in the metadataset. Only two studies using more than one algorithm in the metalevel [10,49]. A relevant future work is the application of RS on both the base- and metalevel. Although this topic has not received any attention so far for the selection of recommendation algorithms, it has been successful in other domains. Such works are important to understand whether MtL approaches are indeed the best way to tackle the algorithm selection problem.

4.4.8. Metalevel evaluation

The last RQ focuses on the evaluation measures used in the metalevel. Once again, these must be in conformity with the metatarget and meta-algorithm. This is noted by the usage of error based measures or correlation assessments for PE [2,13,16,26], classification accuracy measures for BA [10,19] and ranking accuracy measures [49]. One can conclude that all related work performs validation for the algorithm selection task, while using suitable measures.

4.5. Summary

After reviewing in extent each key topic of the algorithm selection problem for CF, we present the key conclusions found:

- With the exception of one study on Group Recommendation, only CF has been studied on the algorithm selection problem.
- The public datasets are representative (although not exhaustive) and enough in recent works. However, previous studies use only few datasets.
- The pool of recommendation algorithms studied in algorithm selection studies is always suitable, but never complete.
- Most approaches evaluate CF with a single measure. However, recent studies are shifting the paradigm towards combining evaluation measures.
- Although a diverse set of metafeatures is available, the related works typically use few metafeatures.
- Although the typical metatarget is performance estimation, the trend has shifted towards choosing the best algorithm.
- Despite most studies working with regression algorithms to build metamodels, the change in metatarget has fueled the shift in meta-algorithms.
- Suitable metalevel evaluation measures were used in all related works.

5. Empirical study

As seen in the literature review, most related work use small scale experimental setups, which prevent the extraction of generic conclusions for RS. Furthermore, the datasets and algorithms used are different, making it difficult to quantitatively assess the merits of each approach. The main goal of this empirical study is to replicate the MtL approaches used by related works and to compare their performances on the same experimental setup. Thus, the baselevel experiments are constant for all approaches, and so are the meta-algorithms, metatarget and metalevel evaluation measures. The only difference is the set of metafeatures employed by each independent algorithm selection approach. More formally, the search spaces P, A and Y are fixed, while space F varies.

5.1. Related work

In order to choose the strategies which will be replicated in this experimental study, certain requirements must be established to ensure a fair evaluation:

- The recommendation strategy must be the same: this study will devote its attention to CF, since it is the most popular. This filters the approach for GR [49], since the metafeatures cannot be reproduced for the CF domain.
- The metafeatures must be specific to the dataset and not to users: the goal here is to compare strategies that select the best algorithm for a whole dataset. Therefore, studies which devotes attention to the selection of algorithms per CF user should be ruled out [13,16]. However, studies with adaptations, by assuming average values for the entire dataset, are kept. For these metafeatures, samples of users are used, whose size is set as the maximum between the total number of users and 1000.
- The metafeatures must reflect the characteristics of either implicit or explicit feedback datasets: since the majority of approaches are designed for explicit feedback, the others are filtered [19]. This is required because the comparison of metafeatures for different rating scales would yield unfair and unreliable results.

In order to formalize the strategies presented, this study takes advantage of a systematic metafeature generation framework [31]. The framework requires three main elements: object o, function f and post-function pf. The process applies the function to the object and, afterwards, the post-functions to that outcome in order to generate a single metafeature value. Thus, this framework can be represented using the following notation: $\{o.f.pf\}$. Next, the different metafeatures strategies are presented, while formalized in this framework.

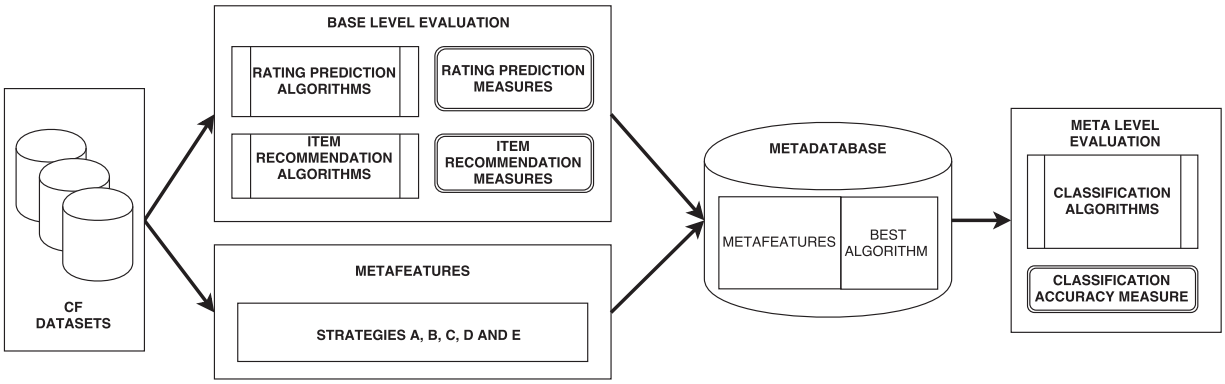

Fig. 4. Experimental procedure. This is an adaptation of Fig. 3 to the experiments carried out in this work.

5.1.1. Strategy A

This strategy generates metafeatures using the systematic framework [10]. It considers the objects *o* dataset, user and item and applies several functions *f* to characterize these objects: original ratings, sums of ratings, counts of ratings and average ratings. The post-functions *pf* used are maximum, minimum, mean, standard deviation, median, mode, entropy, Gini index, skewness and kurtosis. Details regarding the calculation of these functions are available elsewhere [28,40]. Additionally, this strategy includes 4 simple statistics: number of users, items, ratings and matrix sparsity. This results in 74 metafeatures, which were reduced by correlation feature selection, ending up with: *dataset.ratings.kurtosis, dataset.ratings.standard_deviation, dataset.nusers.∅, dataset.sparsity.∅, item.count.kurtosis, item.count.minimum, item.mean.entropy, item.sum.maximum, user.mean.minimum, user.sum.kurtosis, user.mean.skew* and *user.sum.entropy*.

5.1.2. Strategy B

This strategy [26] creates an auxiliary data structure from which the metafeatures are extracted. This structure, known as co-ratings matrix, is created by 1) random sample of users in the original ratings matrix, 2) assigning users to different equivalence classes (EC) depending on the number of ratings assigned and 3) creating a co-ratings matrix that compares each pair of equivalence classes by the average number of common co-rated items. The metafeatures are: *dataset.sparsity.∅*, *EC.co-ratings.gini* and *EC.co-ratings.entropy*.

5.1.3. Strategy C

This strategy extracts the following metafeatures directly from the dataset [2]: *dataset.density.∅, item.count.gini, item.count.skewness, user.count.gini, user.count.skewness* and *dataset.ratings.variance*.

5.1.4. Strategy D

One of the strategies that required adaptation for this experimental study looks at the problem as the selection of algorithms per user [16]. The adaptations involved averaging values of metafeatures among all users in the dataset and working with samples of users for metafeatures with high computational requirements. The complete list of metafeatures used in the strategy is: *user.mean.mean, user.count.mean, item.count.mean, user.standard_dev.mean, item.mean.mean, user.neighbours.mean, user.average_similarity.mean, user.clustering_coef.mean, user_coratings.jaccard.mean, user.TFIDF.mean* and *item.entropy.mean*.

5.1.5. Strategy E

The other strategy that approached the algorithm selection on user level was also adapted [13]. However, due the simpler nature of the metafeatures used, it was required only to average the values across users. The metafeatures produced are: *user.count.mean, user.mean.mean* and *user.variance.mean*.

5.2. Experimental procedure

We present now the experimental procedure used to compare the merits of the several metafeatures strategies presented earlier. Fig. 4 presents the base- and metalevels in terms of data, algorithm and evaluation measures. Next we present each level in detail.

5.2.1. Baselevel

The baselevel experiments refer to the CF problem, where a collection of datasets is evaluated on a pool of suitable algorithms. The 38 datasets used are split up into several domains, namely Amazon Reviews [27], BookCrossing [50], Flixter [48],

Table 2
Summary description about the datasets used in the experimental study.

Dataset	#users	#items	#ratings
Amazon Apps	132,391	24,366	264,233
Amazon Automotive	85,142	73,135	138,039
Amazon Baby	53,188	23,092	91,468
Amazon Beauty	121,027	76,253	202,719
Amazon CD	157,862	151,198	371,275
Amazon Clothes	311,726	267,503	574,029
Amazon Digital Music	47,824	47,313	83,863
Amazon Food	76,844	51,139	130,235
Amazon Games	82,676	24,600	133,726
Amazon Garden	71,480	34,004	99,111
Amazon Health	185,112	84,108	298,802
Amazon Home	251,162	123,878	425,764
Amazon Instant Video	42,692	8882	58,437
Amazon Instruments	33,922	22,964	50,394
Amazon Kindle	137,107	131,122	308,158
Amazon Movies	7278	1847	11,215
Amazon Office	90,932	39,229	124,095
Amazon Pet Supplies	74,099	33,852	123,236
Amazon Phones	226,105	91,289	345,285
Amazon Sports	199,052	127,620	326,941
Amazon Tools	121,248	73,742	192,015
Amazon Toys	134,291	94,594	225,670
Bookcrossing	7780	29,533	39,944
Flixter	14,761	22,040	812,930
Jester1	2498	100	181,560
Jester2	2350	100	169,783
Jester3	2493	96	61,770
Movielens 100k	94	1202	9759
Movielens 10m	6987	9814	1,017,159
Movielens 1m	604	3421	106,926
Movielens 20m	13,849	16,680	2,036,552
Movielens Latest	22,906	17,133	2,111,176
MovieTweetings latest	3702	7358	39,097
MovieTweetings RecSys2014	2491	4754	20,913
Tripadvisor	77,851	10,590	151,030
Yahoo! Movies	764	4078	22,135
Yahoo! Music	613	4620	30,852
Yelp	55,233	46,045	211,627

Jester [15], MovieLens [17], MovieTweetings [12], Tripadvisor [42], Yahoo! Music and Movies [45] and Yelp [47]. It should be noted that a domain may contain multiple datasets. Table 2 presents the datasets and some of their basic characteristics.

Experiments were carried out with MyMediaLite [14]. Two types of CF problems were addressed: Rating Prediction (RP) and Item Recommendation (IR). While in RP the goal is to predict the missing rating an user would assign to a new instance, in IR the goal is to recommend a list of ranked items matching the user's preferences. Since the problems are different, so are the algorithms and evaluation measures. The following CF algorithms were used in this work for the RP problem: Matrix Factorization (MF), Biased Matrix Factorization (BMF), Latent Feature Log Linear Model (LFLLM)), SVD++, 3 variants of Sigmoid Asymmetric Factor Model (SIAFM, SUAFM and SCAFM), User Item Baseline (UIB) and Global Average (GA). In the case of IR, the algorithms used are BPRMF, Weighted BPRMF (WBPRMF), Soft Margin Ranking MF (SMRMF), WRMF and Most Popular (MP). Nearest neighbor algorithms are excluded due to incompatibility with the size of the datasets.

In IR, the algorithms are evaluated using the ranking accuracy metric Normalized Discounted Cumulative Gain (NDCG) and the classification accuracy metric Area Under the Curve (AUC). In the case of RP, the algorithms are evaluated using the error based measures Root Mean Squared Error (RMSE) and Normalized Mean Absolute Error (NMAE). All performances were assessed using 10-fold cross-validation. Following the standard approach in MtL research, the algorithms were trained using the default parameters suggested in the literature or the implementation used. By not tuning the parameters of the baselevel algorithms, the optimal performance of each algorithm is not obtained. However, it prevents us from biasing the performance results in favor of any of the algorithms, thus ensuring a fair assessment.

5.2.2. Metalevel

The metalevel setup includes the metafeatures, the metatarget and the measures used to evaluate the metamodels. The metafeature extraction process involves applying all strategies discussed in Section 5.1 to all CF datasets listed in Table 2. The outcome is 5 different sets of metafeatures.

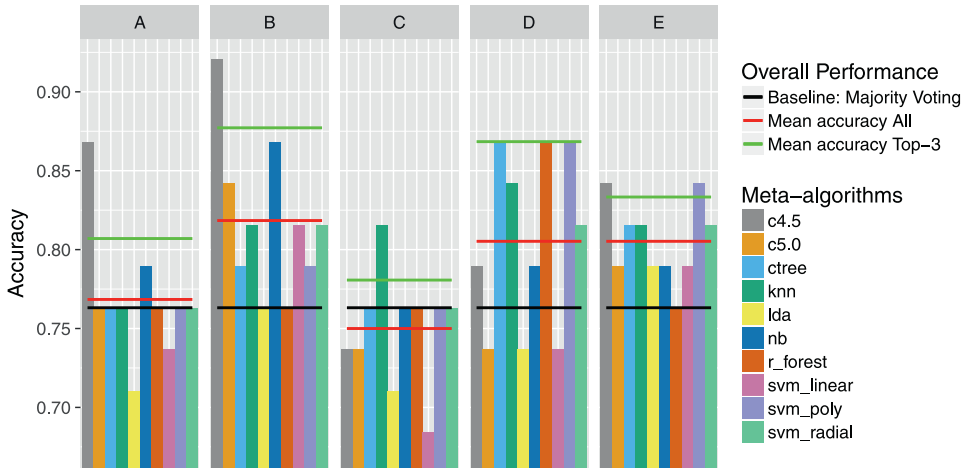

Fig. 5. Performance of the metafeature strategies on the AUC metatarget.

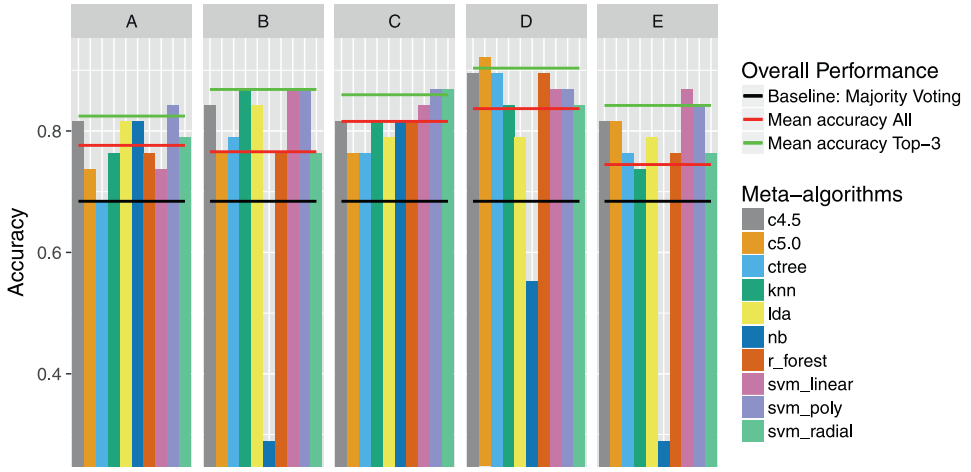

Fig. 6. Performance of the metafeature strategies on the NDCG metatarget.

The metatarget is built by identifying the best algorithm for each specific dataset. Since we use different baselevel evaluation measures, then it is expected that different algorithms are selected as the best choice for a specific dataset for different measures. Thus, NDCG and AUC are used to select the best algorithms in IR tasks, while in RP tasks, RMSE and NMAE are used. This creates 4 different metatargets, which when combined with the 5 different sets of metafeatures, leads to the total amount of 20 metadatasets to be used in the algorithm selection problem.

These problems are addressed as classification tasks. We tested 11 algorithms on each of those metadatasets, with different biases: ctree, C4.5, C5.0, kNN, LDA, Naive Bayes, SVM (linear, polynomial and radial kernels), random forest and a baseline algorithm: majority vote. The majority vote does not take into account any metafeatures and always predicts the class which appears more often. Since the metadatasets have a reduced number of examples, the accuracy of the metalevel algorithms was estimated using a leave one out strategy.

5.3. Results

5.3.1. Metalevel accuracy

The metalevel performance of the strategies regarding the accuracy of the metamodels in each metatarget is presented in Figs. 5–8. In each figure, all strategies are presented alongside each other, to facilitate their comparison. Within each figure, the accuracy of the best tuned meta-algorithms are presented. There are also indications regarding the baseline, i.e. majority voting. Thus, a model whose performance is lower than the baseline, is not considered informative. Two other reference values are also presented: the mean accuracy of all algorithms and the mean accuracy of the top 3 best meta-algorithms. Their purpose is to understand the robustness of the strategies on the average and best case scenarios, respectively.

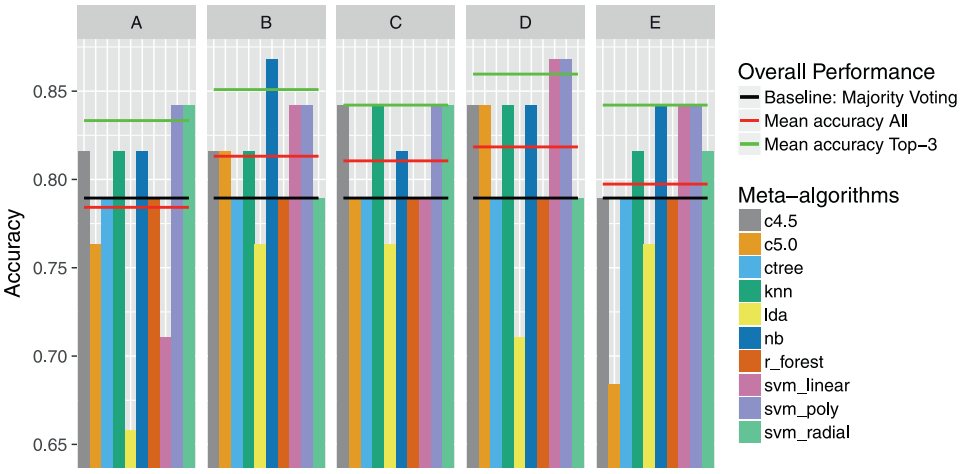

Fig. 7. Performance of the metafeature strategies on the RMSE metatarget.

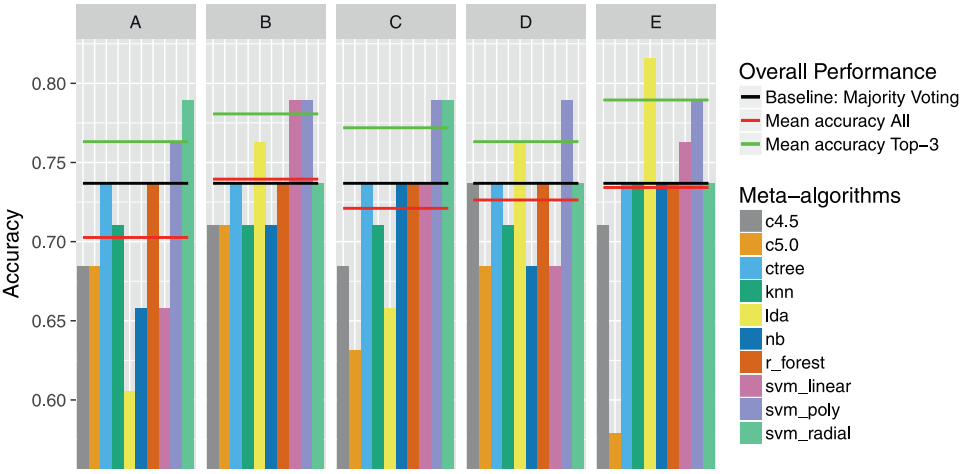

Fig. 8. Performance of the metafeature strategies on the NMAE metatarget.

Regarding Fig. 5, one can observe that all strategies except C have average performances above the baseline. Strategy C has a poor performance, since only one of the meta-algorithms (knn) beats the baseline. This strategy is also responsible by the worst performance, obtained with SVM with linear kernel. Strategy A barely beats the baseline, since two algorithms score below and six have the same performance of the baseline. This strategy only works with the models built with c4.5 and Naive Bayes. All remaining strategies have comparable performance, although strategy B seems slightly better. One important aspect is the performance of the c4.5 algorithm, which obtains the highest performance across all strategies. In terms of the average performance of the top 3 models, all strategies outperform the baseline. However, strategies B and D have higher performance, but with different algorithms. While the best algorithms in B are c4.5, c5.0 and Naive Bayes, in D they are ctree, random forest and SVM with polynomial kernel.

The performance results for the NDCG metatarget are presented in Fig. 6. Here, all strategies beat the baseline both in terms of average of all models and average of the best 3 models. In fact, there are only three occasions in which the baseline is not beaten: for the algorithm Naive Bayes in strategies B, D and E. The strategy with the best performance in both types of averages is D. It has the 4 best models found for this metatarget (c4.5, c5.0, ctree and random forest). The worst strategy by average of all algorithms is E, although this is mainly due to the low performance of the Naive Bayes algorithm. In terms of the average of the top 3 algorithms, strategy A performs slightly worst than the others. However, the worst algorithm in this strategy is the ctree, whose performance is similar to the baseline.

Fig. 7 presents the results for the RMSE metatarget. In terms of the average performance of all algorithms, A is the worst, since it is the only to score below the baseline. This is fueled by the performance of three algorithms (c5.0, LDA and SVM with linear kernel). The best strategies in this case are B, C and D, with low variation on the performance results, although all have one algorithm that was unable to beat the baseline: LDA. In terms of the performance of the top 3 algorithms, all

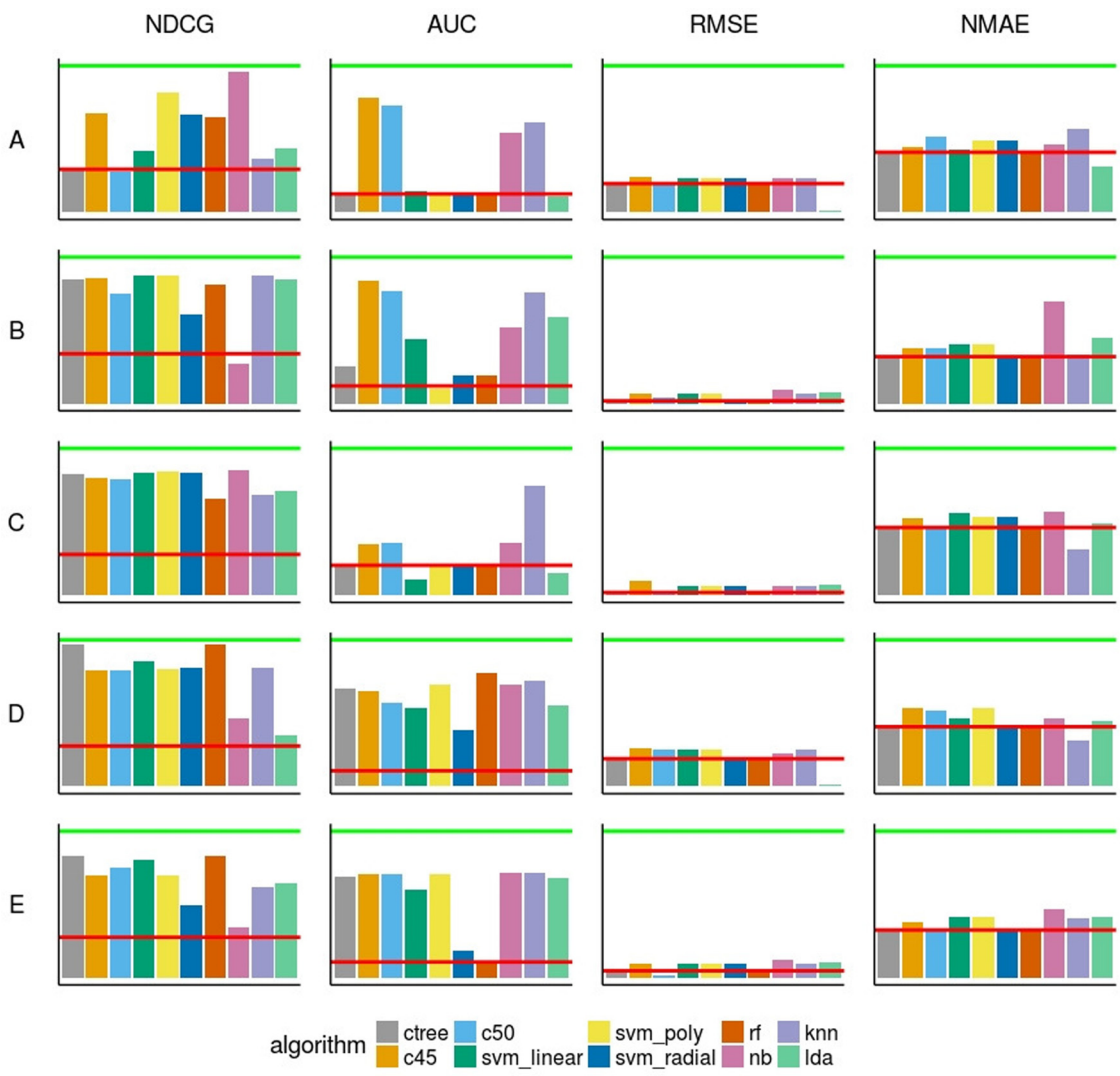

Fig. 9. Impact of meta-algorithms on the baselevel performance.

strategies yield good results, well above the baseline. The best and second best strategies in this case are D (with algorithms SVM with linear and polynomial kernels) and B (achieving highest performance with Naive Bayes), respectively.

Finally, the accuracy results for the NMAE metatarget are presented in Fig. 8. This metatarget has the worst results when considering the average accuracy of all models per strategy: four strategies present values below the baseline and only strategy B has a slightly higher accuracy, which may not be significant. In this metatarget, several algorithms perform poorly, such as the examples of the c5.0 for strategy E and LDA in strategy A. On top of this, at most 3 algorithms per strategy are able to beat the baseline. The only algorithm to beat the baseline in all strategies is the polynomial SVM. When the comparison considers the average performance of the top 3 algorithms, the scenario changes: all strategies yield good results. The best results are achieved with strategy E and algorithm LDA.

5.3.2. Baselevel impact

An important aspect to evaluate in MtL problems is the impact of the metamodels trained on the baselevel performance. For such, the metamodels are evaluated by the performance of the baselevel algorithm predicted as the best in each dataset. The values are then averaged to obtain a single value that depicts the overall performance of the metamodel across all datasets. In these comparisons, two elements are crucial: the performance provided by the baseline metamodel (i.e. majority voting), the lower bound, and the so called oracle, which contains the performance of the actual best algorithm, the upper bound. This analysis is important to understand the real value of the metamodels and especially since it is missing in the original works.

Table 3
Computational time required for the extraction of each type of metafeature strategy.

Strategy	Total time (seconds)	Average time (seconds)
A	47.75	1.26
B	11957.95	314.68
C	4.64	0.12
D	35167.35	925.46
E	259.73	6.83

Fig. 9 presents the results from all strategies across all metatargets. In each pair strategy-metatarget, the baselevel performance from all meta-algorithms is shown. The baseline and oracle results are presented as the red and green horizontal lines, respectively. The results were presented in a format to facilitate their readability. All values are independently scaled and the results for RMSE and NMAE were inverted to uniform the analysis process. Thus, although the goal is to minimize these measures, the inverted scale allows the following analysis: an algorithm is better than another if its performance is higher.

The results show that metatarget NDCG has the largest number of metamodels whose performance is better than the baseline. In fact, in all strategies, there is only one example below the baseline. On the other extreme, RMSE and NMAE present the worst results. Here, most algorithms perform similar to or worse than the baseline. In the case of RMSE, the gap between the best metamodels and the oracle is the highest. While the results for NDCG and NMAE are expected, the RMSE case is a surprise. It shows that although the metalevel accuracy for this problem is acceptable for a subset of models, the impact on the performance of the recommendation problem is low.

Regarding strategies, the results are overall balanced across metatargets. However, there are some exceptions, such as strategies C, D and E for metatarget NDCG and strategy D for metatarget AUC. In all these cases, the results from all algorithms score above the baseline. There is no pair strategy-metatarget whose baselevel performance of all algorithms is always equal to or lower than the baseline, although the performance of the best metamodels on the RMSE metatarget is just slightly above the baseline.

5.3.3. Computational cost

One important aspect to be evaluated in the comparison of metafeature strategies is the amount of time required for metafeature extraction. Table 3 presents the recorded values for the total and average amount of time required. While the first refers to the time required to extract all metafeatures for all datasets, the second indicates the average time for one dataset. This last value is an indicator of the time required for the application on a future new problem.

According to the results, strategies C, A and E are the fastest, respectively, and the only that seem feasible in terms of computational resources. These results are expected due to fact that their metafeatures are the simplesy and require only the rating matrix to be produced. On the other hand, strategies D and B are the most time consuming. The reason behind it lies with the usage of alternative data structures and the extraction of more detailed metafeatures.

5.4. Discussion

The main observations from this empirical study are the following:

- All strategies outperform the baseline with regards to the average of the top 3 algorithms. However, in terms of average performance, the strategies only seem to be effective for the NDCG metatarget, while failing for the NMAE. Regarding AUC and RMSE, there is no clear pattern. These observations indicate that solving the CF algorithm selection problem using MtL is feasible and an efficient alternative to the standard empirical study. However, it also sheds light on an important problem: the practitioner must decide what is the criteria to select the best algorithm, since the metatargets will reflect different behaviors. And if the metatargets are different, then different metafeatures are required to solve the problem.
- When comparing the average performances across metatargets, strategy D yields the best results, closely followed by B. Strategies A, C and E have the worst performances, although in different metatargets. The best strategies have in common the fact that they explore the datasets in more detail and focus the metafeatures (directly or indirectly) on the user interactions. This may be an indication that future work should look towards more complex metafeatures such as those used in these works.
- The best and worst meta-algorithms across metatargets appear to be the polynomial SVM and LDA, respectively. This can be seen by inspecting the top positions of each algorithm in each combination of strategy and metatarget. Other algorithms do not have such an evident pattern. It is hard to attempt to explain why does this phenomenon occur. Clearly the bias in the algorithms is the responsible factor, although the actual reasons are still unknown.
- In terms of baselevel impact, the strategies work well for the NDCG metatarget and poorly for RMSE. The performance in terms of strategies is comparable, although there are some special cases. This points to the importance of this type of

Table 4
Tukey's test p-values for the comparison among baseline, average of all algorithms and average of top 3 algorithms. The p-values are highlighted if the differences are significant, i.e. p-value 0.05.

Pairwise	AUC	NDCG	NMAE	RMSE	All
top3-avg	0.078571	**0.003901**	**0.000024**	**0.000074**	**0.000474**
base-avg	0.350451	**0.000191**	0.206408	0.0734732	**0.030089**
base-top3	**0.006020**	**0.000001**	**0.000357**	**0.000003**	**0.000000**

Table 5
Tukey's test p-values for the comparison among strategies. The p-values are highlighted if the differences are significant, i.e. p-value 0.05.

Pairwise	AUC	NDCG	NMAE	RMSE	All
B-A	0.060590	0.999934	0.443748	0.615853	0.593524
C-A	0.894940	0.965678	0.938393	0.705771	0.915770
D-A	0.292990	0.819273	0.840159	0.433962	0.170064
E-A	0.292990	0.987142	0.611546	0.978983	0.973180
base-A	0.999639	0.428473	0.526624	0.999722	0.949847
C-B	**0.003110**	0.910009	0.938393	0.999991	0.990722
D-B	0.973442	0.699014	0.985458	0.999722	0.973180
E-B	0.973442	0.998061	0.999815	0.954044	0.958747
base-B	**0.027991**	0.563215	0.999994	0.788278	0.130146
D-C	**0.027991**	0.998061	0.999815	0.998017	0.746388
E-C	**0.027991**	0.699014	0.985458	0.978983	0.999887
base-C	0.973442	0.096515	0.967604	0.858851	0.410626
E-D	1.000000	0.428473	0.998667	0.858851	0.593524
base-D	0.166905	**0.033901**	0.994775	0.615853	**0.016084**
base-E	0.166905	0.819273	0.999994	0.998017	0.566884

analysis: although one can rank the strategies accordingly to the metalevel accuracy, the results show that their impact on the baselevel is lower than expected.

- There is a trade-off between accuracy and computational time. The most accurate strategies has a processing time up to one thousand times lower than the time required for the fastest strategy. However, the performance is these cases is not optimal. It will be up to the practitioner to decide which one to use.

In order to validate the observations, statistical significance tests were performed. Most validations are calculated with ANOVA (to understand if there are differences in the populations studied in each observation) and Tukey's honest significant difference tests (to understand whether the differences are statistical significant or not). Table 4 presents the results of the Tukey's test for the following null hypothesis: there is no difference in performance among baseline (base), average performance of all algorithms (avg) and average performance of the top 3 algorithms (top3) for each metatarget and on a general point of view. The p-values are highlighted if the differences are significant, i.e. p-value < 0.05. One can assert that the average of the top 3 algorithms are indeed better than the baseline. The results also show that the only metatarget for which the average of all algorithms is statistical significantly better than the baseline is NDCG. The variance in the results for the other metatargets make it impossible to make such assertion for those cases. However, on the analysis for all metatargets, the difference remains statistically significant. It is also possible to conclude that the average top 3 is better than average of all algorithms for the NDCG, NMAE and RMSE metatargets and also on a global point of view.

In order to validate the statistical significance in the ranking of strategies, the previous tests, but with the results aggregated by strategy, can be performed. The comparisons are made for each metatarget and also for the combination of all metatargets. The p-values for the pairwise comparison using Tukey's test are presented in Table 5. The results show that the observations regarding the ranking of strategies do not hold statistical significance, except for few special cases. In fact, when considering all metatargets, the only statistically significant difference is that strategy D is better than the baseline. All other comparisons in the set of all metatargets do not hold any statistical significant value. This trend extends when the comparison is made for each metatarget, with the exception of AUC. Here, there is evidence to support that B is better than the baseline and that C is worse than B, D and E. In the NDCG metatarget, D is also shown to be better than the baseline.

The statistical significance of multiple algorithms on multiple datasets can be verified using Critical Difference (CD) diagrams [11]. This consists of plotting the average ranking position for each element being compared and calculate the CD interval that states that there is no evidence in the data to show that the elements within that interval can be considered different. In CD plots, two elements not connected by a line can be considered different, i.e., one algorithm is ranked higher than the other. In this work, the datasets are provided by the combinations of metafeature strategies and metatargets, which refer to the independent and dependent variables, respectively.

The CD diagram in Fig. 10 allows to confirm that SVM polynomial is the highest ranked algorithm. However, there are no statistically significant differences among SVM polynomial, SVM radial, C4.5, KNN and SVM linear. Despite this, one can

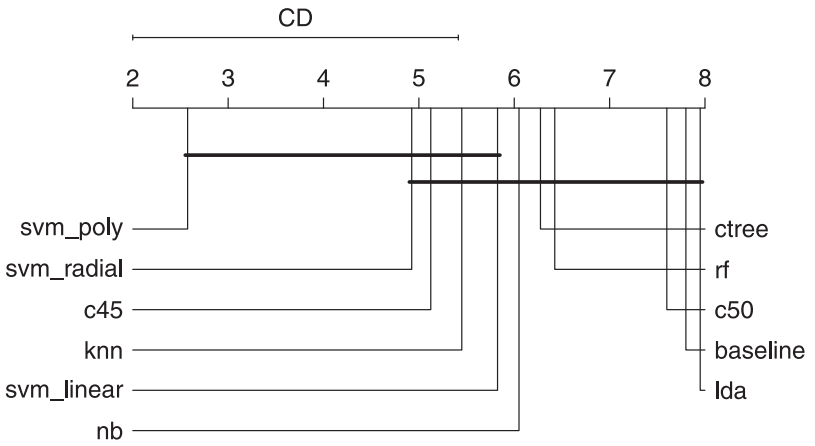

Fig. 10. CD diagrams for meta-algorithms across strategies and metatargets.

assert that SVM polynomial is better than Naive Bayes, ctree, random forest, C5.0, LDA and the baseline majority voting. In terms of statistical significance, no other conclusions can be drawn from this diagram.

6. Conclusions

This work analyzes in depth the problem of algorithm selection for Recommender Systems, with focus on Collaborative Filtering. It starts with a systematic literature review regarding related work. In this review, the problem is framed within the classical Metalearning conceptual framework and each of the available works is presented and discussed on the several dimensions of the problem. Afterwards, an empirical study is performed to assess the quality of a subset of the approaches discussed on the selection of algorithms. Experimental results regarding the metalevel accuracy, baselevel impact and computational resources are presented, discussed and statistically validated.

The literature review shows that most work is performed on Collaborative Filtering, with a reduced number of datasets, algorithms and evaluation measures in the base-level. However, recent works have improved such dimensions, effectively contributing to advances in the field. In terms of meta-level, there are several approaches which look at the algorithm selection problem in different and valid perspectives. Most works used regression approaches with a wide range of metafeatures. However, there was no comparison among them, which prevents the understanding of their merits.

To solve this problem, we conducted an empirical study. It has shown that, in the best case scenarios, all strategies are able to create effective models to solve the algorithm selection problem. However, in a deeper look, all strategies have different behaviors depending on the metatargets and meta-algorithms used, which indicates that the research problem is not yet solved. The results have also shown that strategy D is statistical significantly better than the baseline, although the same conclusions cannot be drawn regarding the other strategies. The main conclusion of this study lies in the fact that there is no single best strategy that always outperforms the others, but rather aspects on which the performance is better. The authors highlighted the advantages and weaknesses of each work, but leave the choice of metafeature strategy to the practitioner. Nevertheless, the knowledge gathered in this document allows to positively direct future work in this research topic.

Acknowledgments

This work is financed by the ERDF Fund through the Operational Program for Competitiveness and Internationalization - COMPETE 2020 of Portugal 2020 through National Innovation Agency (ANI) as part of project 3506. The authors also wish to acknowledge the Portuguese funding institution FCT - Fundação para a Ciência e a Tecnologia for supporting their research through grant SFRH/BD/117531/2016 and the Brazilian funding agencies CNPq and FAPESP.

References

[1] 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.
[2] G. Adomavicius, J. Zhang, Impact of data characteristics on recommender systems performance, ACM Trans. Manag. Inf. Syst. 3 (1) (2012) 1–17.
[3] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, P. Lamere, The million song dataset, in: International Conference on Music Information Retrieval, 2011.
[4] Y. Blanco-Fernández, J. Pazos-Arias, A. Gil-solla, M. Ramos-Cabrer, Providing entertainment by content-based filtering and semantic reasoning in intelligent recommender systems, IEEE Trans. Consum. Electron. 54 (2) (2008) 727–735.
[5] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl. Based Syst. 46 (2013) 109–132.
[6] P. Brazdil, C. Giraud-Carrier, C. Soares, R. Vilalta, Metalearning: Applications to Data Mining, 1, Springer, 2009.
[7] R. Burke, Hybrid web recommender systems, The adaptive web (2007) 377–408.

[8] P.G. Campos, F. Díez, I. Cantador, Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols, User Model User-Adapt. Interact. 24 (1–2) (2013) 67–119.
[9] T. Cunha, R.J.F. Rossetti, C. Soares, Analysing collaborative filtering algorithms in a multi-agent environment, in: European Simulation and Modelling Conference, 2014, pp. 135–139.
[10] T. Cunha, C. Soares, A.C. de Carvalho, Selecting collaborative filtering algorithms using metalearning, in: European Conference on Machine Learning and Knowledge Discovery in Databases, 2016, pp. 393–409.
[11] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[12] S. Dooms, T. De Pessemier, L. Martens, MovieTweetings: a Movie Rating Dataset Collected From Twitter, CrowdRec Workshop, 2013.
[13] M. Ekstrand, J. Riedl, When recommenders fail: predicting recommender failure for algorithm selection and combination, ACM Conf. Recommend. Syst. (2012) 233–236.
[14] Z. Gantner, S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, MyMediaLite: A Free Recommender System Library, in: ACM Conference on Recommender Systems, 2011, pp. 305–308.
[15] K. Goldberg, T. Roeder, D. Gupta, C. Perkins, Eigentaste: a constant time collaborative filtering algorithm, Inf. Retr. 4 (2) (2001) 133–151.
[16] J. Griffith, C. O'Riordan, H. Sorensen, Investigations into user rating information and predictive accuracy in a collaborative filtering domain, in: ACM Symposium on Applied Computing, 2012, pp. 937–942. URL: http://grouplens.org/datasets/movielens/.
[17] GroupLens, MovieLens datasets, 2016.
[18] J.L. Herlocker, J.a. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 5–53.
[19] Z. Huang, D.D. Zeng, Why does collaborative filtering work? transaction-based recommendation model validation and selection by analyzing bipartite random graphs, INFORMS 23 (1) (2011) 138–152.
[20] D. Kim, C. Park, J. Oh, H. Yu, Deep hybrid recommender systems via exploiting document context and statistics of items, Inf. Sci. 417 (2017) 72–87.
[21] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[22] J. Liu, C. Sui, D. Deng, J. Wang, B. Feng, W. Liu, C. Wu, Representing conditional preference by boosted regression trees for recommendation, Inf. Sci. 327 (2016) 1–20.
[23] J. Liu, C. Wu, W. Liu, Bayesian probabilistic matrix factorization with social relations and item contents for recommendation, Decis. Support Syst. 55 (3) (2013) 838–850.
[24] W. Liu, C. Wu, B. Feng, J. Liu, Conditional preference in recommender systems, Expert Syst. Appl. 42 (2) (2015) 774–788.
[25] L. Lü, M. Medo, C.H. Yeung, Y.-C. Zhang, Z.-K. Zhang, T. Zhou, Recommender systems, Phys. Rep. 519 (1) (2012) 1–49.
[26] P. Matuszyk, M. Spiliopoulou, Predicting the performance of collaborative filtering algorithms, in: International Conference on Web Intelligence, Mining and Semantics, 2014, p. 38:1—38:6.
[27] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: ACM Conference on Recommender Systems, 2013, pp. 165–172.
[28] E.D. Michie, D.J. Spiegelhalter, C.C. Taylor, Machine learning, neural and statistical classification, 1994.
[29] Netflix, Netflix prize data set (2009). URL:http://archive.ics.uci.edu/ml/datasets/Netflix+Prize.
[30] F. Ortega, A. Hernando, J. Bobadilla, J.H. Kang, Recommending items to group of users using matrix factorization based collaborative filtering, Inf. Sci. 345 (2016) 313–324.
[31] F. Pinto, C. Soares, J. Mendes-Moreira, Towards automatic generation of Metafeatures, in: Pacific Asia Conference on Knowledge Discovery and Data Mining, 2016, pp. 215–226.
[32] R.B. Prudêncio, T.B. Ludermir, Meta-learning approaches to selecting time series models, Neurocomputing 61 (2004) 121–137.
[33] J. Rice, The algorithm selection problem, Adv. Comput. 15 (1976) 65–118.
[34] M. Richardson, R. Agrawal, P. Domingos, Trust Management for the Semantic Web, pp. 351–368.
[35] A.L.D. Rossi, A.C.P.D.L.F. de Carvalho, C. Soares, B.F. de Souza, MetaStream: a meta-learning based method for periodic algorithm selection in time-changing data, Neurocomputing 127 (2014) 52–64.
[36] A. Said, A. Bellogín, Comparative recommender system evaluation: benchmarking recommendation frameworks, in: ACM Conference on Recommender Systems, 2014, pp. 129–136.
[37] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Analysis of recommendation algorithms for e-commerce, in: ACM Conference on Electronic commerce, 2000, pp. 158–167.
[38] F. Serban, J. Vanschoren, A. Bernstein, A survey of intelligent assistants for data analysis, ACM Comput. Surv. V (212) (2013) 1–35.
[39] C. Soares, P.B. Brazdil, P. Kuba, A meta-learning method to select the kernel width in support vector regression, Mach. Learn. 54 (3) (2004) 195–209.
[40] H.E.A. Tinsley, S.D. Brown, Handbook of Applied Multivariate Statistics and Mathematical Modeling, Elsevier Science, 2000.
[41] J. Vanschoren, Understanding machine learning performance with experiment databases, Katholieke Universiteit Leuven, 2010 Ph.D. thesis.
[42] H. Wang, Y. Lu, C. Zhai, Latent aspect rating analysis without aspect keyword supervision, in: ACM SIGKDD, 2011, pp. 618–626.
[43] R. Wang, A. Liu, F. Yuan, The followee recommendation algorithm based on microblog user interest and characteristic, J. Inf. Comput. Sci. 11 (5) (2014) 1585–1596.
[44] C. Wu, J. Wang, J. Liu, W. Liu, Recurrent neural network based recommendation for time heterogeneous feedback, Knowl. Based Syst. 109 (2016) 90–103.
[45] Yahoo!, Webscope datasets, 2016. URL:https://webscope.sandbox.yahoo.com/.
[46] X. Yang, Y. Guo, Y. Liu, H. Steck, A survey of collaborative filtering based social recommender systems, Comput. Commun. 41 (2014) 1–10.
[47] Yelp, Yelp Dataset Challenge, 2016. URL:https://www.yelp.com/dataset_challenge.
[48] R. Zafarani, H. Liu, Social computing data repository at ASU, 2009. URL:http://socialcomputing.asu.edu.
[49] A. Zapata, V.H. Menéndez, M.E. Prieto, C. Romero, Evaluation and selection of group recommendation strategies for collaborative searching of learning objects, Int. J. Hum. Comput. Stud. 76 (2015) 22–39.
[50] C.-N.C. Ziegler, S.M.S. McNee, J.a.J. Konstan, G. Lausen, Improving recommendation lists through topic diversification, in: International Conference on World Wide Web, 2005, p. 22.