Appendix A

Materials and methods

In this thesis we have reported results obtained in experiments conducted with a variety of input data sources and recommendation algorithms. In this appendix we provide additional details, not reported in previous chapters, about such datasets and recommenders. Hence, in Section A.1 we present statistics about the datasets, and in Section A.2 we describe the specific configuration setting of the recommenders. Next, in Section A.3 we detail the followed evaluation methodologies. Finally, in Sections A.4 and A.5 we present further results regarding prediction correlations and performance of dynamic recommender ensembles by means of metrics such as MAP@10 and nDCG@50. Despite the fact the thesis is mainly focused on ranking metrics, for the sake of brevity and clarity, in Chapters 6 and 7 we only reported experimental results based on the P@10 metric.

A.1 Datasets

In Section 3.4 we presented the three datasets used in this thesis, namely MovieLens, Last.fm, and CAMRa datasets. We now describe the specific partitions in which we splitted those datasets for experimentation. Specifically, we explain how the data splits were generated, and provide some statistics of the splits, such as their number of users and items, and their density, measured as the percentage of cells in the ratings matrix with known values, i.e., #ratings / (#users × #items) × 100.

A.1.1 MovieLens dataset

In addition to the well-known Netflix dataset, the MovieLens 100K and MovieLens 1M datasets – extracted from the MovieLens movie recommendation system by the GroupLens research group at University of Minnesota, USA – have been the most widely used datasets in the Recommender Systems field. The former has 943 users, 1,682 items, and 100,000 ratings, whereas the latter has 6,040 users, 3,900 items, and 1 million ratings. In this thesis, when we do not explicitly indicate which of the two datasets was used, we refer to the MovieLens 1M dataset.

In the experiments we always performed a 5-fold cross validation strategy to generate 5 random 80-20% disjoint splits of rating sets. As rating sets, in the **MovieLens 100K** dataset we used the partition provided in its public distribution, and in the **MovieLens 1M** dataset we used 5 splits with 200,000 ratings, each of them randomly selected.

Property	Overall	Average	Average in training	Average in test
No. users	943	854.60 (± 165.44)	943 (± 0.00)	766.20 (± 205.06)
No. items	1,682	1,531.20 (± 127.18)	1651.60 (± 4.77)	1410.80 (± 11.52)
No. ratings	100,000	50,000 (± 31,622.78)	80,000 (± 0.00)	20,000 (± 0.00)
Density	6.30%	3.56% (± 1.72)	5.14% (± 0.01)	1.99% (± 0.67)
Avg. no. rated items per user	106.04	56.46 (± 30.56)	84.84 (± 0.00)	28.09 (± 9.43)
Max. no. rated items per user	737	450.80 (± 213.68)	650.20 (± 35.37)	251.40 (± 45.59)
Min. no. rated items per user	20	3.70 (± 3.16)	6.40 (± 2.07)	1 (± 0.00)
Max. no. users rating an item	583	293.30 (± 182.82)	466.40 (± 14.06)	120.20 (± 9.71)
Min. no. users rating an item	1	1 (± 0.00)	1 (± 0.00)	1 (± 0.00)

Table A.1. Summary of statistics of the MovieLens 100K dataset.

Property	**Overall**	**Average**	**Average in training**	**Average in test**
No. users	6040	6038.40 (± 1.84)	6040 (± 0.00)	6036.80 (± 1.10)
No. items	3706	3,576.50 (± 109.48)	3,680.20 (± 6.26)	3,472.80 (± 6.61)
No. ratings	1,000,000	500,104.50 (± 316,293.86)	800,167.20 (± 0.45)	200,041.80 (± 0.45)
Density	4.47%	2.28% (± 1.39)	3.60 (± 0.01)	0.95 (± 0.00)
Avg. no. rated items per user	165.60	82.81 (± 52.36)	132.48 (± 0.00)	33.14 (± 0.01)
Max. no. rated items per user	2,314	1,157 (± 732.10)	1,851.20 (± 24.00)	462.80 (± 24.00)
Min. no. rated items per user	20	5.90 (± 5.22)	10.80 (± 1.10)	1 (± 0.00)
Max. no. users rating an item	3,428	1,714 (± 1,084.24)	2,742.40 (± 22.95)	685.60 (± 22.95)
Min. no. users rating an item	1	1 (± 0.00)	1 (± 0.00)	1 (± 0.00)

Table A.2. Summary of statistics of the MovieLens 1M dataset.

Table A.1 and Table A.2 show some statistics about the MovieLens datasets. It is interesting to note that the overall sparsity in both datasets is similar (between 4% and 6%), but the number of users vs. items is quite different: in MovieLens 100K there are more items than users, whereas this situation in MovieLens 1M is the opposite.

Finally, in those experiments where we evaluated content-based recommenders, we used extended versions of the MovieLens dataset with information extracted from the Internet Movie Database[9], such as the movies' directors, actors, and genres. Details about how this information was gathered and merged with the MovieLens user profiles can be found in (Cantador, 2008).

A.1.2 Last.fm dataset

Among the two Last.fm datasets distributed by Ò. Celma (Celma and Herrera, 2008), and described in Section 3.4.2, in this thesis we used the one denoted as Last.fm 1K, since it contains the music listening history (scrobbles) of each user at the track level, and includes the timestamp at which a user listenend to a track.

For our experiments we built two versions of that dataset. Table A.3 shows some statistics about them. For building the first version (refered as Last.fm dataset from now on), we applied a 5-fold cross-validation on 80-20% training-test data splits. For building the second version, we performed a single temporal splitting of the user scrobbles, maintaining an 80-20% training-test ratio. This is equivalent to divide the data at some timestamp in such a way that 80% of the scrobblings are con-

[9] The Internet Movie Database, IMDb, http://www.imdb.com

		Five-fold			Temporal		
Property	**Overall**	**Average**	**Average in training**	**Average in test**	**Average**	**Training**	**Test**
No. users	992	991 (± 1.70)	992 (± 0.00)	990 (± 2.00)	932 (± 16.97)	920	944
No. items	176,892	108,065.20 (± 48,266.14)	153,854.20 (± 155.01)	62,276.20 (± 199.80)	118,530 (± 43,371.10)	149,198	87,862
No. scrobblings	19,129,595	9,564,797.50 (± 6,049,542.74)	15,303,676 (± 56,393.95)	3,825,919 (± 56,393.95)	9,564,798.50 (± 8,115,999.81)	15,303,677	3,825,920
No. unique scrobblings	904,309	452,154.50 (± 285,967.77)	723,447.20 (± 313.46)	180,861.80 (± 313.46)	539,553 (± 268,287.63)	729,261	349,845
Density	10.90% 0.52%	8.12% (± 2.02) 0.38% (± 0.10)	10.03% (± 0.04) 0.47% (± 0.00)	6.21% (± 0.11) 0.29% (± 0.00)	7.88% (± 4.62) 0.48% (± 0.08)	11.15% 0.53%	4.61% 0.42%
Avg. no. scrobblings per user	19,283.87	9,645.83 (± 6,094.22)	15,427.09 (± 56.85)	3,864.57 (± 57.35)	10,343.66 (± 8,896.50)	16,634.43	4,052.88
Max.no. scrobblings per user	183,094	93,795.20 (± 55,859.24)	146,475.20 (± 7,036.93)	41,115.20 (± 5,753.13)	117,872.50 (± 84,792.71)	177,830	57,915
Min. no. scrobblings per user	2	1.30 (± 0.48)	1.60 (± 0.55)	1 (± 0.00)	1 (± 0.00)	1	1
Avg. no. scrobbled items per user	911.60	455.99 (± 288.08)	729.28 (± 0.32)	182.69 (± 0.12)	581.64 (± 298.45)	792.68	370.60
Max. no. scrobbled items per user	8,452	4,226.00 (± 2,672.76)	6,761.60 (± 0.55)	1,690.40 (± 0.55)	5,707 (± 1,554.22)	6,806	4,608
Min. no. scrobbled items per user	2	1.30 (± 0.48)	1.60 (± 0.55)	1 (± 0.00)	1 (± 0.00)	1	1
Max. no. users scrobbling an item	710	356.10 (± 233.52)	568.00 (± 10.39)	144.20 (± 7.36)	527.50 (± 146.37)	631	424
Min. no. users scrobbling an item	1	1 (± 0.00)	1 (± 0.00)	1 (± 0.00)	1 (± 0.00)	1	1
Time interval (in days)	3,150	1,586.85 (± 0.02)	1,586.85 (± 0.00)	1,586.84 (± 0.02)	1,574.87 (± 331.64)	1,340	1,809

Table A.3. Summary of statistics of the Last.fm datasets.

tained in the training split. In the built dataset (refered as Last.fm temporal dataset from now on), the above timestamp is 16th October, 2008.

In both datasets we aggregated the user listening data by artist (amounting to $|\mathcal{J}| = 176{,}892$ instead of $|\mathcal{J}| \approx 960{,}000$ if tracks were used) in order to overcome the sparsity at track level. Apart from that, it is also interesting to note the difference between considering separate scrobblings (when a user listens to an artist, in our setting) against considering unique scrobblings (or number of user-item pairs), which corresponds to the typical situation in movie recommendation (where, for each movie, there are not several ratings given by a particular user).

Furthermore, in some experiments we transformed log-based information to explicit ratings by using the method described in (Celma, 2010) and (Celma, 2008). This method takes into account the number of times a user listened to an artist, in such a way that the artists located in the 80-100% interquintile range of the user's listening distribution receive a rating of 5 (in a five point scale), those in the next interquintile range are mapped to a rating of 4, and so on. Additionally, the use of the coefficient of variation $CV(u) = \sigma(u)/\mu(u)$ is proposed in (Celma, 2008) to discriminate between skewed and uniform distributions. We have not considered this coefficient since it produced strange behaviours in the recommenders, such as too many ties in the recommended items and errors in the computation of some correlations (since the mean would have the same value for every rating, see Equation (2.5)).

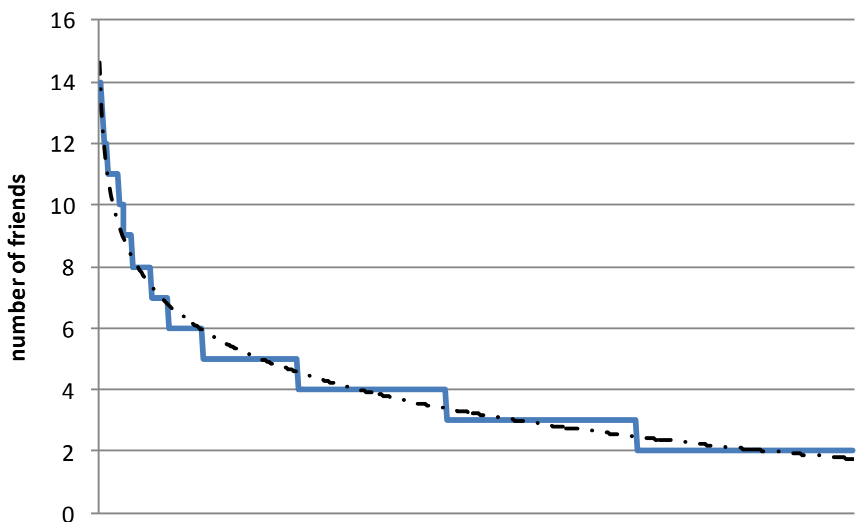

Figure A.1. Friend distribution among the users composing the original test set of the CAMRa '10 social track. Note that a logarithmic regression line fits almost perfectly with the above distribution. The total number of users is 439, and the maximum and minimum numbers of friends are 14 and 2 respectively.

On this dataset, content-based recommenders used the artists' tags. The considered tags for each artist were the most popular tags assigned to that artist according to the Last.fm API[10].

A.1.3 CAMRa dataset

Among the several datasets provided in the different CAMRa challenges (Said et al., 2010) (Said et al., 2011), in this thesis we used the dataset published at the social track of CAMRa '10. This dataset was gathered from the Filmtipset community, and contains social links between users, movie ratings, movie comments, and other attributes of users and movies. The design of this dataset and, more specifically, the selection of test users as provided in the challenge's social track is representative of online applications in which every target user has a non-empty list of contacts (see Figure A.1). This is the case of social-centric systems such as Facebook, Linkedin, and Twitter, but not of many social media applications, such as Delicious and Last.fm, where the coverage of their social network is partial – not all registerd users really use the social part of the system.

In fact, the Filmtipset dataset belongs to the latter case. Considering the set of all users, more than 10,000 out of about 16,500 users do not have any friends in the system. The number of contacts per user follows a power law distribution, where the average number of friends per user is 0.95, and the mode among users (with at least one friend) is 1. If we take this dataset as representative of social media systems, the

[10] Documentation related with the method used, http://www.lastfm.es/api/show/artist.getTopTags

		Social		Collaborative-Social	
Property	**Overall**	**Training**	**Test**	**Training**	**Test**
No. users	16,473	16,473	439	16,473	793
No. items	24,222	24,222	1,915	24,212	2670
No. ratings	3,091,075	3,075,346	15,729	3,069,888	21,187
Density	0.77%	0.77%	1.87%	0.77%	1.00%
Avg. no. rated items per user	187.64	186.69	35.83	186.36	26.72
Max. no. rated items per user	3,435	3,435	314	3,435	314
Min. no. rated items per user	1	1	1	1	1
Max. no. users rating an item	14,339	14,290	111	14,255	134
Min. no. users rating an item	1	1	1	1	1
Avg no. friends per user	0.95	0.95	3.85	0.95	2.13
Max. no. friends per user	24	24	14	24	14
Min. no. friends per user	0	0	2	0	0

Table A.4. Summary of statistics of the CAMRa datasets.

presence of contacts by itself does not guarantee the accuracy of social recommendation. Moreover, intermediate cases, where social data is available but not enough to support optimal recommendation, would rather seem to be the norm. We therefore simulate an alternative scenario by adding an equal amount of users without friends to a new test set by sampling randomly the same number of test users in the original test set (i.e., 439 users), but forcing them to have no friends. We name the original dataset as CAMRa Social, and the modified one as CAMRa Collaborative-Social or simply CAMRa Collaborative, since it is not focused on social information. Table A.4 shows some statistics about these datasets, from which the minimum number of friends per user in the test sets is the main difference between the above datasets (2 in the original, social dataset, and 0 in the collaborative test).

A.2 Configuration of recommendation algorithms

In this section we provide details about the implementation of the recommendation algorithms used in Chapters 4, 6, 7, and 8. Table A.5 shows a summary of such recommenders, along with references to the chapters where the recommenders were evaluated. In the following we describe each of these recommenders, and provide their parameter values, explicit formulations, and references. The recommenders, categorised according to the type of recommendations they provide, are the following:

Recommender	Chapter(s) where the recommender is evaluated
CB	Chapters 6 and 7
IB	Chapters 6 and 7
ItemPop	Chapters 4, 6, and 7
kNN	Chapters 4, 6, and 7
MF	Chapter 4
pLSA	Chapters 4, 6, and 7
Random	Chapters 4 and 6
Resnick	Chapter 8
TFL1	Chapters 6 and 7
TFL2	Chapters 6 and 7

Table A.5. List of the recommenders evaluated in this thesis, and the chapters where their evaluations are reported.

Non-personalised recommenders

- **ItemPop**: a non-personalised recommender based on the popularity of the item being recommended. It basically counts the number of training ratings of an item, and generates a recommendation score for the item based on such number.
- **Random**: a non-personalised recommender that generates a random recommendation score in a required value range.

Content-based recommender

- **CB**: a content-based recommender, which, similarly to the one described in (Martinez et al., 2009), computes the cosine similarity between user and item vectors whose components can represent any possible content-based feature. A different configuration of features is used depending on the dataset. For the MovieLens dataset, we used item attributes such as movie genre, director, and country, as appeared in IMDb. Other features like actors and keywords were also tested, but yielded worse performance. Specifically, we used the most popular 3 countries for each item, 3 directors, and 8 genres per movie, as suggested in (Cantador, 2008); besides, each of these features was weighted as follows: the country feature was assigned a weight of 0.26, director, 0.06, and genre, 0.66. For the Last.fm dataset, we used as features the 50 most popular tags related to each artist.

Rating-based recommenders

- **IB**: an item-based collaborative filtering recommender, in which Equation (2.8) is used along with Pearson's correlation as the similarity metric between items (analogous to Equation (2.5), but considering items instead of users). Moreover, a constraint was implemented in order to remove some noise from neighbour items: only predictions produced by at least two similar items were considered; that is, if we replaced S_i by $\mathcal{I}_u$, then those users with only one item similar to the target item i in their profiles did not receive any recommendation.

- **kNN**: a user-based (nearest neighbour) collaborative filtering recommender, in which a slight modification of Equation (2.3) is used to adapt the item-based algorithm proposed in (Koren, 2008). We used 100 neighbours and Pearson's correlation as similarity metric, as defined in Equation (2.5). Specifically, we considered the following equation:

$$g(u,i) = b(u,i) + \frac{1}{\sum_{v \in N_k(u,i)} sim_{sh}(u,v)} \sum_{v \in N_k(u,i)} sim_{sh}(u,v)\big(r(v,i) - b(v,i)\big)$$

where $sim_{sh}(u,v)$ is the shrunk similarity $sim(u,v)n(u,v)/(n(u,v)+\lambda_2)$, $n(u,v)$ being $|\mathcal{I}_u \cap \mathcal{I}_v|$, that is, the number of users who rated both items. Besides, $b(u,i) = \mu + b_u + b_i$ is the user-item bias learnt solving the following least squares problem, where μ indicates the overall average rating:

$$\min_{b_u, b_i} \sum_{(u,i)} \left[(r(u,i) - \mu - b_u - b_i)^2 + \lambda_1 \left(\sum_u b_u^2 + \sum_i b_i^2 \right) \right]$$

In our experiments no tuning of the λ_1 regularisation parameter was done (partially because this method indeed optimises RMSE, which is not our main goal), and used a fixed value of 0.02. We used a learning rate of 0.005, and 100 iterations in the optimisation process. Furthermore, we did not shrink the similarity, so an effective $\lambda_2 = 0$ was used. Additionally, at least 5 neighbours had to participate in the prediction process to consider a predicted item as valid.

- **MF**: a matrix factorisation recommender, as implemented in the Mahout library by means of the Expectation Maximisation (EM) algorithm. We used 100 iterations and 50 features, leaving the rest of the parameters as default (i.e., 0.005 as learning rate, 0.02 as regularisation parameter, and 0.005 as random noise). We have to note that the original method proposed in (Koren et al., 2009) used Alternating Least Squares to learn the user and item factorised vectors. However, the version implemented in Mahout provided worse results for this learning method, and thus we used the EM algorithm in our experiments.

- **pLSA**: a probabilistic Latent Semantic Analysis recommender, in which we use the co-occurrence latent semantic model as defined in (Hofmann, 2004). That is, the prediction is made by computing $p(i|u) = \sum_z p(i|z)p(z|u)$. The models based on ratings produced worse results, and are not applicable with implicit (log-based) user preference data. This is why we preferred to use the co-occurrence model over the one based on ratings. In the experiments we used 50 factors and 50 iterations.

- **Resnick**: a user-based (neearest neighbour) collaborative filtering recommender, implemented as in Equation (2.3), where rating deviations from the user's and neighbour's rating means are considered (Resnick et al., 1994). As discussed in Chapter 8, different neighbourhood sizes were tested. We used Pearson's correlation as the user similarity metric, like in Equation (2.5).

- **TFL1**: an item-based collaborative filtering recommender, which uses the TF method with normalisation s_{00}, as defined in (Bellogín et al., 2011b). This is equivalent to an standard item-based CF algorithm without dividing by the similarities values, that is:

$$g(u,i) = \sum_{j \in S_i} sim(i,j) r(u,j)$$

We did not consider the neighbourhood size, replacing S_i by $\mathcal{I}_u$, and using Pearson's correlation as similarity metric $sim(i,j)$.

- **TFL2**: an item-based collaborative filtering recommender in which, similarly to the TFL1 recommender, we used the TF method with normalisation s_{01}, and L_2 norm as defined in (Bellogín et al., 2011b). This is equivalent to an standard item-based CF, but instead of normalising by the sum of similarity values, it divides by the square root of the sum of similarity values squared, that is:

$$g(u,i) = \frac{1}{\sqrt{\sum_{j \in S_i} sim(i,j)^2}} \sum_{j \in S_i} sim(i,j) r(u,j)$$

Like before, we did not consider the neighbourhood size, and used Pearson's correlation as the similarity metric.

Social-based recommenders

- **Personal**: a social filtering recommender presented in (Ben-Shimon et al., 2007), which utilises Equation (2.11) for score prediction. We set $K = 2$ and $L = 6$, along with a constraint specifying that at least two friends have to participate on the item prediction in order to be considered valid.

- **PureSocial**: a social filtering recommender, which is an adaptation of the standard user-based collaborative filtering in which friends are used as neighbours, as proposed in (Liu and Lee, 2010) and (Bellogín et al., 2012). Specifically, we used Equation (2.4) as the user-based method, but where the neighbourhood $N_k(u, i)$ is built by means of the user's social network. No neighbourhood size is used, and, like in the previous recommender, at least two friends have to participate on the item prediction to consider the suggestion as a valid one.

We implemented these recommenders on top of the Mahout library[11], and will make the developed source code publicly available at the following URL: http://ir.ii.uam.es/~alejandro/thesis.

Additionally, we implemented the static and dynamic weighted recommender ensembles evaluated in Chapter 7 using the rank fusion library provided in (Fernández et al., 2006a) and (Fernández et al., 2006b). This library contains a series of techniques for the two basic stages of any rank fusion problem: score normalisation and score combination. In this thesis we conducted the following procedure: a) taking each item ranking (in a user basis) for every recommender in an ensemble, b) normalising each ranking using either rank or score normalisation techniques (Renda and Straccia, 2003), c) combining the normalised rankings using a weighted sum to compute the score of each item (i.e., combSUM method); the weight assigned to each ranking may come from a performance predictor, as explained in Section 7.2.3, and d) ranking the items according to the scores produced in the last combination stage.

A.3 Configuration of evaluation methodologies

In Chapter 4 we presented several evaluation methodologies that have been further elaborated and used in the experiments of Chapters 6 and 7. In this section we provide specific examples regarding how these methodologies build the set of items to recommend. We also indicate the value of the parameters required by each method that have been used in the different chapters of this dissertation.

In Chapter 4 we classified the different target item selection strategies according to three design settings: the base candidate settings (AI or TI), the relevant item selection (AR or 1R), and the non-relevant item selection (AN or NN). Table A.6 shows the specific configurations of these methodologies, according to the notation introduced in Chapter 4. Note that the uniform methodologies (U1R, UAR, and uuUAR) take as additional parameters η and ξ; for the MovieLens 1M dataset, these parameters took the following values: $\eta = 118$, $\nu = 350$, and $\xi = 99$.

[11] Available at http://mahout.apache.org

Methodology	Base candidate	Relevant item selection	Non-relevant item selection	Configuration	Chapters
1R	TI	1R	NN	$\text{N}_u = 99$	4, 6, and 7
AR	TI	AR	AN		4, 6, and 7
P1R	TI	1R	NN	$\text{N}_u = 99$ $m = 10$	4, 6, and 7
U1R	TI	1R	NN	$\text{N}_u = 99$ $r_{test}(i) = \eta, \forall i$	4, 6, and 7
UAR	AI/TI	AR	AN	$r_{test}(i) = \eta, \forall i$	4
uuU1R	TI	1R	NN	$\text{N}_u = 99$ $r_{test}(i) = \nu, \forall i$ $r_{test}(u) = \xi, \forall u$	6 and 7

Table A.6. Configuration of the evaluation methodologies used in the thesis.

In the following we present several toy examples to illustrate how the different methodologies behave. For these examples we consider three users as follows:

- A user u_1 with training set $R_{train}(u_1) = \{i_1, i_2\}$, and test set $R_{test}(u_1) = \{i_3, i_4\}$.
- A user u_2 with $R_{train}(u_2) = \{i_1, i_2, i_3\}$, and test set $R_{test}(u_2) = \{i_5\}$.
- A user u_3 with $R_{train}(u_3) = \{i_3, i_4, i_5\}$, and test set $R_{test}(u_3) = \{i_1, i_2\}$.

According to these training and test data, we next compare the target items selected for user u_1 by 1R and AR methodologies. The AR methodology selects all items contained in the test set (TI-AN) as non-relevant, and all items in the active user's test (AR); hence, it produces the following set to be scored and ranked by a given recommender: $\{i_3, i_4, i_5\}$. The 1R methodology, on the other hand, produces a ranking for each relevant item as follows: $\{i_3, i_5\}$ for item i_3, and $\{i_4, i_5\}$ for item i_4, where we use $N_u = 1$ for illustration purposes.

In the example we also have that $r(u_1) = r(u_2) = 4$, and $r(u_3) = 5$, and for the items $r(i_1) = r(i_2) = r(i_3) = 3$ and $r(i_4) = r(i_5) = 2$, where $r(u)$ denotes the number of items rated by a user (and similarly for items) as denoted in Chapter 4. With the uniform methodologies we have to build a different training/test split for each user. Specifically, in the U1R and UAR methodologies we need to ensure that every item has been rated the same number of times in the test, whereas in the uuU1R methodology we also have the constraint that all users have to appear the same number of times in the test set. Therefore, the following split would be valid for the U1R or UAR methodologies:

$$R_{train}(u_1) = \{i_4\};\ R_{test}(u_1) = \{i_1, i_2, i_3\}$$
$$R_{train}(u_2) = \{i_2, i_3, i_5\};\ R_{test}(u_2) = \{i_1\}$$
$$R_{train}(u_3) = \{i_1, i_4, i_5\};\ R_{test}(u_3) = \{i_2, i_3\}$$

In this case $\eta = 2$ for every item in the test set. Then, we can apply the 1R or AR methodology to obtain the corresponding rankings for the U1R or UAR methodologies, respectively. However, since we have a different amount of ratings for each user, this configuration is not valid for the uuU1R methodology. A valid configuration for this methodology would be:

$$R_{train}(u_1) = \{i_3, i_4\};\ R_{test}(u_1) = \{i_1, i_2\}$$
$$R_{train}(u_2) = \{i_2, i_5\};\ R_{test}(u_2) = \{i_1, i_3\}$$
$$R_{train}(u_3) = \{i_1, i_4, i_5\};\ R_{test}(u_3) = \{i_2, i_3\}$$

Here, we have $\nu = 2$ and $\xi = 2$ for all the users and items in the test set. In this context, if we apply the 1R methodology to this split we would obtain results corresponding to the uuU1R methodology.

Alternatively, for uuU1R we can also derive a valid configuration directly from a given uniform split, i.e., the one used for U1R and UAR. To do this, we would exploit the degree of freedom we have to select the set of not-relevant items for each user. Hence, we have to guarantee that all the non-relevant items selected for each user have to appear the same number of times in every target set to be recommended. This constraint would be satisfied depending on the user and item distributions, where a greedy algorithm (by brute force) could be applied if enough ratings are available. For instance, in our previous example we cannot derive a valid configuration from the split presented for U1R and UAR. In such a case, an alternative split should be made as explained above.

For simplicity purposes, in the examples we have assumed that all items contained in the test set are relevant. However, this is not the general case, and a threshold on the minimum rating value should be set. To be consistent across methodologies, only items rated as 5 by a user are considered relevant. Thus, for the AR methodology, although every item in a user's test set is considered, in the evaluation only those items with 5 stars are considered relevant. On the other hand, the 1R methodology builds one ranking for each relevant item. That is, in the example above, if $r(u_1, i_3) = 5$ and $r(u_1, i_4) = 3$, then a unique ranking would be generated and evaluated, the one corresponding to $\{i_3, i_5\}$.

As additional material, we will make publicly available an implementation of the above evaluation methodologies at http://ir.ii.uam.es/~alejandro/thesis.

A.4 Additional results about correlations

Next we report additional experiments to those presented in Chapter 6 regarding the computation of correlation coefficients between the performance predictors and recommenders. Specifically, we provide experimental results obtained with metrics different to P@10, and alternative correlation coefficients such as Spearman's and Kendall's.

A.4.1 Correlations of user predictors using rating data

In this section we provide results obtained by using different correlation coefficients and metrics in the evaluation of user predictors based on ratings. Specifically, we provide results obtained with Spearman's ρ and Kendall's τ correlation coefficients, and metrics such as MAP@10 (because it is considered as more stable than precision (Manning et al., 2008; Baeza-Yates and Ribeiro-Neto, 2011)), recall@10 (as an inversely related metric to precision), and nDCG@50 (because it includes graded relevance and a different cutoff).

Table A.7, Table A.8, and Table A.9 show the results obtained when the above three metrics are used along with the Pearson's correlation coefficient and the AR methodology. Comparing these results with those shown in Table 6.7, we can observe that most of the correlation trends are similar to the ones presented in Chapter 6. Specifically, the results with nDCG@50 are almost equivalent to those obtained with P@10, proving that our predictors are also consistent with other metrics apart from precision. On the other hand, the correlation values with MAP@10 and recall@10 metrics have some differences with respect to the P@10 metric, being lower in general. In fact, the correlation with CB and pLSA recommenders is negative, probably due to the (inverse) relation between precision and recall, which makes very difficult to optimise both metrics at the same time.

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	0.005	-0.031	0.009	0.047	0.094	-0.049	0.024	0.281
Count (test)	0.004	-0.029	0.009	0.047	0.092	-0.052	0.022	0.276
Mean	0.009	0.010	-0.002	-0.065	-0.036	-0.002	0.007	-0.103
Standard deviation	0.004	0.013	0.011	-0.018	-0.029	-0.023	0.015	-0.069
ItemSimple Clarity	0.007	-0.031	0.010	0.040	0.089	-0.054	0.026	0.274
ItemUser Clarity	0.005	-0.029	0.011	0.039	0.089	-0.054	0.028	0.272
RatUser Clarity	0.006	-0.031	0.009	0.048	0.093	-0.049	0.024	0.273
RatItem Clarity	0.005	-0.029	0.008	0.041	0.087	-0.053	0.022	0.267
IRUser Clarity	0.006	-0.026	0.005	0.051	0.083	-0.046	0.021	0.265
IRItem Clarity	0.003	-0.021	0.009	0.038	0.076	-0.046	0.023	0.234
IRUserItem Clarity	0.005	-0.025	0.007	0.049	0.082	-0.047	0.024	0.261
Entropy	0.000	-0.032	0.007	0.040	0.103	-0.037	0.021	0.296

Table A.7. Pearson's correlation values between rating-based user predictors and MAP@10 for different recommenders, using the AR methodology on the MovieLens 1M dataset.

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	-0.001	-0.048	0.009	0.012	0.053	-0.120	0.020	0.213
Count (test)	-0.002	-0.047	0.008	0.009	0.051	-0.123	0.018	0.208
Mean	0.009	0.009	-0.002	-0.067	-0.024	-0.017	0.005	-0.104
Standard deviation	0.004	0.014	0.011	-0.038	-0.034	-0.035	0.019	-0.076
ItemSimple Clarity	0.000	-0.049	0.009	0.000	0.047	-0.131	0.023	0.203
ItemUser Clarity	-0.001	-0.046	0.011	0.003	0.049	-0.121	0.025	0.205
RatUser Clarity	0.000	-0.049	0.009	0.011	0.055	-0.116	0.020	0.201
RatItem Clarity	-0.002	-0.046	0.007	0.008	0.050	-0.115	0.019	0.199
IRUser Clarity	0.001	-0.040	0.004	0.018	0.047	-0.108	0.018	0.201
IRItem Clarity	-0.002	-0.036	0.006	0.006	0.039	-0.106	0.020	0.176
IRUserItem Clarity	0.000	-0.040	0.006	0.016	0.046	-0.109	0.021	0.197
Entropy	-0.004	-0.048	0.006	0.010	0.058	-0.124	0.018	0.252

Table A.8. Pearson's correlation values between rating-based user predictors and recall@10 for different recommenders, on the MovieLens 1M dataset and using the AR methodology

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	0.085	0.012	0.010	0.221	0.228	0.144	0.152	0.528
Count (test)	0.085	0.015	0.010	0.225	0.231	0.148	0.153	0.525
Mean	0.023	0.037	-0.014	-0.035	0.011	0.050	0.040	-0.084
Standard deviation	0.003	0.019	0.010	-0.046	-0.069	-0.048	0.021	-0.096
ItemSimple Clarity	0.094	0.020	0.011	0.226	0.235	0.156	0.173	0.540
ItemUser Clarity	0.084	0.014	0.013	0.205	0.220	0.134	0.167	0.513
RatUser Clarity	0.087	0.005	0.010	0.221	0.240	0.139	0.160	0.517
RatItem Clarity	0.081	0.008	0.008	0.201	0.218	0.123	0.158	0.493
IRUser Clarity	0.079	0.022	-0.001	0.216	0.201	0.136	0.138	0.492
IRItem Clarity	0.074	0.032	0.007	0.184	0.173	0.119	0.145	0.440
IRUserItem Clarity	0.079	0.023	0.003	0.212	0.198	0.133	0.146	0.485
Entropy	0.078	0.025	0.004	0.224	0.235	0.192	0.127	0.561

Table A.9. Pearson's correlation values between rating-based predictors and nDCG@50 for different recommenders, on the MovieLens 1M dataset and using the AR methodology.

As pointed out by other authors, Pearson's correlation values by themselves may not be enough to completely understand the relation between analysed variables (Hauff et al., 2009; Carmel and Yom-Tov, 2010). Because of that, in Table A.10 and Table A.11 we provide results found when Spearman's and Kendall's correlations are used, along with the P@10 metric and the AR methodology. Comparing these results with those shown in Table 6.7, we can observe that strong correlations are also obtained using non parametric coefficients such as Kendall's τ. More specifically, our results show that $r \geq \rho \geq \tau$, with respect to the Pearson's r, Spearman's ρ, and Kendall's τ correlation coefficients.

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	0.112	0.165	0.020	0.457	0.367	0.465	0.124	0.585
Count (test)	0.114	0.174	0.020	0.473	0.375	0.484	0.124	0.589
Mean	0.015	0.064	0.000	0.010	0.025	0.106	0.030	-0.024
Standard deviation	0.007	0.005	0.009	-0.032	-0.038	-0.037	0.009	-0.064
ItemSimple Clarity	0.117	0.176	0.021	0.474	0.382	0.492	0.130	0.602
ItemUser Clarity	0.112	0.168	0.021	0.442	0.362	0.457	0.131	0.583
RatUser Clarity	0.113	0.157	0.021	0.469	0.388	0.482	0.122	0.598
RatItem Clarity	0.113	0.169	0.019	0.460	0.378	0.477	0.130	0.592
IRUser Clarity	0.110	0.164	0.019	0.449	0.364	0.454	0.123	0.571
IRItem Clarity	0.107	0.174	0.017	0.422	0.318	0.414	0.123	0.532
IRUserItem Clarity	0.110	0.164	0.020	0.447	0.362	0.453	0.125	0.571
Entropy	0.112	0.166	0.020	0.460	0.369	0.468	0.124	0.588

Table A.10. Spearman's correlation values between rating-based user predictors and P@10 for different recommenders, on the MovieLens 1M dataset and using the AR methodology.

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	0.092	0.134	0.017	0.358	0.296	0.353	0.102	0.471
Count (test)	0.094	0.143	0.017	0.373	0.305	0.371	0.102	0.477
Mean	0.013	0.052	0.000	0.008	0.020	0.079	0.025	-0.018
Standard deviation	0.006	0.004	0.008	-0.024	-0.030	-0.028	0.007	-0.050
ItemSimple Clarity	0.096	0.143	0.017	0.371	0.308	0.374	0.106	0.484
ItemUser Clarity	0.091	0.136	0.018	0.344	0.292	0.346	0.107	0.467
RatUser Clarity	0.093	0.127	0.017	0.367	0.313	0.366	0.100	0.481
RatItem Clarity	0.092	0.137	0.016	0.359	0.304	0.362	0.106	0.476
IRUser Clarity	0.090	0.133	0.015	0.350	0.293	0.344	0.100	0.457
IRItem Clarity	0.087	0.141	0.014	0.328	0.255	0.312	0.101	0.423
IRUserItem Clarity	0.090	0.133	0.017	0.348	0.292	0.343	0.102	0.456
Entropy	0.092	0.134	0.017	0.359	0.297	0.355	0.101	0.472

Table A.11. Kendall's correlation values between rating-based user predictors and P@10 for different recommenders, on the MovieLens 1M dataset and using the AR methodology.

Predictor	Random	CB	IB	kNN	TFL1	TFL2
Count (training)	0.288	0.255	0.170	0.488	0.529	0.545
Count (test)	0.384	0.384	0.242	0.592	0.602	0.623
Mean	-0.063	-0.009	-0.060	-0.018	-0.103	0.012
Standard deviation	-0.011	-0.033	0.045	-0.016	0.031	-0.135
ItemSimple Clarity	0.282	0.257	0.146	0.491	0.521	0.564
ItemUser Clarity	0.289	0.255	0.189	0.479	0.540	0.530
RatUser Clarity	0.282	0.234	0.182	0.469	0.507	0.516
RatItem Clarity	0.239	0.191	0.184	0.395	0.442	0.426
IRUser Clarity	0.149	0.171	-0.092	0.257	0.253	0.399
IRItem Clarity	0.232	0.218	0.152	0.372	0.453	0.416
IRUserItem Clarity	0.279	0.265	0.105	0.444	0.523	0.545
Entropy	0.263	0.256	0.110	0.497	0.499	0.574

Table A.12. Pearson's correlation values between rating-based user predictors and nDCG@50 for different recommenders, on the MovieLens 100K dataset and using the AR methodology.

We also compare the results shown in Table A.9 with those obtained on different datasets. Table A.12 shows the correlation values for the MovieLens 100K dataset, as published in (Bellogín et al., 2011b). We observe that the correlation does not

Predictor	Random	CB	IB	ItemPop	kNN	pLSA	TFL1	TFL2
Count (training)	0.389	0.464	0.100	0.718	0.553	0.697	0.490	0.793
Count (test)	0.392	0.475	0.100	0.728	0.562	0.711	0.492	0.797
Mean	-0.093	-0.151	-0.022	-0.117	-0.041	-0.152	-0.123	-0.104
Standard deviation	0.017	0.036	0.009	-0.029	-0.045	-0.025	0.018	-0.069
ItemSimple Clarity	0.383	0.453	0.100	0.721	0.563	0.700	0.478	0.802
ItemUser Clarity	0.391	0.465	0.112	0.696	0.544	0.678	0.514	0.783
RatUser Clarity	0.387	0.442	0.116	0.702	0.557	0.665	0.498	0.788
RatItem Clarity	0.366	0.426	0.096	0.663	0.524	0.633	0.500	0.765
IRUser Clarity	0.376	0.469	0.068	0.671	0.498	0.664	0.499	0.742
IRItem Clarity	0.358	0.443	0.082	0.622	0.469	0.609	0.482	0.684
IRUserItem Clarity	0.378	0.470	0.083	0.664	0.494	0.658	0.513	0.736
Entropy	0.313	0.442	0.056	0.687	0.508	0.747	0.341	0.710

Table A.13. Pearson's correlation values between rating-based user predictors and P@10 for different recommenders, on the MovieLens 1M dataset and using the AR methodology, but considering all the items in test set as relevant.

change significantly; only the correlation values for the CB, IB and TFL1 recommenders increase in the MovieLens 100K dataset.

Additionally, as shown in Chapter 6, when different experimental designs are tested, the selection of relevant and not relevant items is very important. In the AR methodology, as described in Section A.3, we consider as relevant those items whose ratings are 5 (to some extent in order to be consistent with the rest of the methodologies). Table A.13, on the other hand, shows that the correlation changes when all the items in the test set are considered relevant. The first thing to note when we compare these results with those shown in Table 6.7 is that the absolute correlation values are now much higher. However, if we take the correlations with the Random recommender as a reference, these relative correlations are very similar in both tables. Moreover, the trend in predictive power of the predictors (that is, which predictors have more or less predictive power) is consistent across this dimension, which evidences the stability of the evaluation methodology used to measure the predictive power of the recommendation performance predictors.

A.4.2 Correlations of user predictors using log data

In this section we focus on complementing our results with correlations between the predictor values and the MAP@10 metric, since we have observed in the previous sections that other correlation coefficients are consistent with Pearson's.

Hence, in Table A.14 we report the results with the temporal split, whereas in Table A.15 we present the results for the random split on the Last.fm dataset. Respectively, we have to compare these tables against Table 6.14 and Table 6.15. This comparison shows that the results obtained using the precision and MAP metrics are equivalent, in contrast to what happened in the previous sections. This is because in this experiment we use the 1R methodology where there is only one relevant item, in

Predictor	Random	CB	ItemPop	kNN	pLSA
Average Count	0.001	0.173	0.027	-0.054	0.163
Count	0.028	0.188	-0.044	0.140	0.115
Mean	-0.059	-0.407	0.037	-0.089	-0.220
Standard deviation	-0.034	-0.191	-0.035	-0.119	-0.094
Autocorrelation	0.049	0.156	-0.079	-0.105	0.001
TimeSimple Clarity	-0.044	-0.435	0.053	-0.257	-0.212
ItemTime Clarity	0.024	0.142	0.085	0.316	0.084
ItemPriorTime Clarity	0.069	0.219	0.240	0.345	0.204
Frequency Clarity	-0.039	-0.421	-0.248	-0.307	-0.365
ItemSimple Clarity	0.008	0.118	-0.052	0.275	0.014

Table A.14. Pearson's correlation values between log-based user predictors and MAP@10 for different recommenders, on the Last.fm temporal dataset and using the 1R methodology.

Predictor	Random	CB	ItemPop	kNN	pLSA
Average Count	-0.043	-0.065	-0.139	0.020	-0.136
Count	-0.031	-0.212	-0.194	-0.040	-0.260
Mean	0.004	0.173	0.111	0.008	0.117
Standard deviation	-0.010	0.116	0.116	0.021	0.099
Autocorrelation	0.010	-0.032	-0.078	-0.007	-0.063
TimeSimple Clarity	0.041	0.304	0.277	0.082	0.344
ItemTime Clarity	0.037	-0.147	0.020	0.052	-0.084
ItemPriorTime Clarity	0.036	-0.028	0.210	0.149	0.072
Frequency Clarity	0.001	-0.038	-0.286	-0.158	-0.211
ItemSimple Clarity	0.028	-0.240	-0.131	-0.021	-0.213

Table A.15. Pearson's correlation values between log-based user predictors and MAP@10 for different recommenders, on the Last.fm five-fold dataset and using the 1R methodology.

contrast to the AR methodology used in that experiment, and thus, these two metrics are equivalent. Similar results are obtained for recall and nDCG metrics for the same reasons.

A.4.3 Correlations of user predictors using social data

In this section we present additional results regarding the correlations between social-based performance predictors and evaluation metrics. Like in the previous section, here we only show correlations with respect to the MAP@10 metric, which, in this case, do not provide results equal to those of the precision metric since we use the AR methodology instead of the 1R methodology.

Table A.16 shows Pearson's correlation values on the social version of the CAMRa dataset. We observe that the correlations are much lower than those presented in Table 6.16; in some situations even the sign of the correlation changes, like for most of the values of kNN. A more interesting situation is observed on the CAMRa Collaborative dataset (Table A.17), where strong but negative correlations arise, in particular for the ItemPop and pLSA recommenders. This result may have a direct impact on the performance of the dynamic ensembles, since the correlation of

Predictor	Random	ItemPop	kNN	pLSA	Personal	PureSocial
Count (training)	0.016	0.104	0.022	0.024	0.013	0.066
Count (test)	0.086	-0.032	0.021	-0.167	-0.215	-0.294
Mean	-0.047	0.074	0.075	0.009	0.003	0.025
Standard deviation	-0.030	-0.065	-0.144	-0.061	-0.063	-0.004
Avg neighbour degree	-0.025	-0.057	-0.031	-0.124	-0.120	-0.240
Betweenness centrality	-0.015	0.061	-0.010	-0.015	0.011	-0.076
Clustering coefficient	0.028	-0.024	0.035	-0.106	0.001	-0.133
Degree	-0.026	-0.070	-0.065	-0.130	-0.184	-0.185
Ego components size	-0.044	0.019	-0.051	0.029	-0.036	0.021
HITS	-0.010	-0.018	0.056	-0.002	0.082	0.040
PageRank	-0.020	-0.022	-0.047	0.041	-0.059	0.041
Two-hop neighbourhood	-0.039	-0.040	-0.062	-0.118	-0.148	-0.227
ItemSimple Clarity	0.012	0.132	0.031	0.037	0.023	0.080

Table A.16. Pearson's correlation values between social-based user predictors and MAP@10 for different recommenders, on the CAMRa social dataset and using the AR methodology.

Predictor	Random	ItemPop	kNN	pLSA	Personal	PureSocial
Count (training)	0.004	-0.072	0.054	-0.096	0.012	0.067
Count (test)	0.031	-0.119	0.056	-0.163	-0.215	-0.294
Mean	-0.045	0.119	0.042	0.057	0.005	0.026
Standard deviation	0.041	-0.043	-0.139	-0.041	-0.063	-0.004
Avg neighbour degree	0.086	-0.168	0.044	-0.148	-0.120	-0.240
Betweenness centrality	-0.005	0.010	-0.008	-0.039	0.012	-0.075
Clustering coefficient	-0.002	-0.133	0.099	-0.149	-0.001	-0.135
Degree	0.045	-0.174	0.039	-0.198	-0.183	-0.185
Ego components size	0.030	-0.126	0.016	-0.143	-0.035	0.022
HITS	-0.009	-0.023	0.027	0.085	0.082	0.040
PageRank	0.000	-0.136	0.024	-0.134	-0.058	0.041
Two-hop neighbourhood	0.080	-0.148	0.016	-0.151	-0.146	-0.227
ItemSimple Clarity	0.002	-0.068	0.057	-0.092	0.022	0.081

Table A.17. Pearson's correlation values between social-based user predictors and MAP@10 for different recommenders, on the CAMRa collaborative dataset and using the AR methodology.

the predictors is now very different. We refer the reader to Section A.5.3 where we show how the dynamic ensembles perform when this metric is used.

A.5 Additional results about dynamic ensembles

Next we present additional results obtained in the experiments aimed to compare static and dynamic hybrid recommendations. We report values of metrics different to P@10, which has been extensively used in the thesis. In particular, we focus on MAP@10 in order to provide a full overview of the predictors' behaviour, since correlations with respect to this metric have been presented in the previous sections.

A.5.1 Performance results from dynamic ensembles on rating data

In this section we report experiments where dynamic hybrid recommenders are built by means of rating-based performance predictors. Table A.18 and Table A.19 show performance values (in terms of the MAP metric) of the hybrid recommenders by using the AR and 1R methodologies respectively. Additionally, Table A.20 shows performance values using item predictors along with the uuU1R methodology.

We can observe that the results from Table A.18 are quite similar to those presented in Table 7.2: in most cases the dynamic hybrid recommenders outperform the baseline static recommender, and the best result for each ensemble is obtained either by using the perfect correlation predictor or one of the clarity-based performance predictors. The only difference of these results with those of the P@10 metric (Table 7.2) is that when we evaluate with MAP@10 the baseline outperforms the dynamic ensembles for the combination HRU3. We have to note that the best static recommender is very different for this combination: whereas for P@10 the best result is obtained when $\lambda = 0.9$, for MAP@10 the best result is obtained for $\lambda = 0.5$, where, as we observed in Chapter 7, the rating-based user predictors seem to perform worse when the best static recommender is close to that value.

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0005	0.0298	0.0116	0.0116	0.0116	0.0116
R2 (λ=0.0)	0.0086	0.0086	0.0086	0.0001	0.1047	0.0551
Baseline (λ=0.5)	0.0043	0.0268	0.0271	0.0003	0.0794	0.0499
Best static	0.0087	0.0310	0.0271	0.0022	0.1108	0.0584
(best λ)	(0.1)	(0.9)	(0.5)	(0.9)	(0.1)	(0.1)
Perfect correlation	<u>0.0099</u>	0.0373	0.0297	<u>0.0119</u>	0.1166	<u>0.0663</u>
PC-OM	0.0097	<u>0.0399</u>	<u>0.0305</u>	0.0045	0.1125	0.0602
PC-FW	0.0097	0.0296	0.0278	0.0008	0.1136	0.0615
Entropy-OM	**0.0057**$^{▼}_{▲}$	**0.0333**$^{△}_{▲}$	0.0260$^{▽}_{▽}$	**0.0009**$^{▼}_{▲}$	**0.0863**$^{▼}_{▲}$	**0.0509**$^{▼}_{△}$
ItemSimple-OM	**0.0090**$^{△}_{▲}$	**0.0330**$^{△}_{▲}$	0.0256$^{▽}_{▽}$	**0.0009**$^{▼}_{▲}$	**0.1149**$^{▲}_{▲}$	**0.0572**$^{▼}_{▲}$
ItemUser-OM	**0.0091**$^{△}_{▲}$	**0.0332**$^{△}_{▲}$	0.0255$^{▽}_{▽}$	**0.0008**$^{▼}_{▲}$	**0.1161**$^{▲}_{▲}$	**0.0576**$^{▼}_{▲}$
RatUser-OM	**0.0093**$^{△}_{▲}$	**0.0335**$^{△}_{▲}$	0.0262$^{▽}_{▽}$	**0.0009**$^{▼}_{▲}$	**0.1178**$^{▲}_{▲}$	**0.0575**$^{▼}_{▲}$
RatItem-OM	**0.0093**$^{△}_{▲}$	**0.0329**$^{△}_{▲}$	0.0259$^{▽}_{▽}$	**0.0008**$^{▼}_{▲}$	<u>**0.1185**</u>$^{▲}_{▲}$	**0.0576**$^{▼}_{▲}$
IRUser-OM	**0.0087**$^{△}_{▲}$	**0.0326**$^{△}_{▲}$	0.0257$^{▽}_{▽}$	**0.0008**$^{▼}_{▲}$	**0.1146**$^{▲}_{▲}$	**0.0580**$^{▼}_{▲}$
IRItem-OM	**0.0091**$^{△}_{▲}$	**0.0321**$^{△}_{▲}$	0.0250$^{▽}_{▼}$	**0.0008**$^{▼}_{▲}$	**0.1160**$^{▲}_{▲}$	**0.0577**$^{▼}_{▲}$
IRUserItem-OM	**0.0090**$^{△}_{▲}$	**0.0325**$^{△}_{▲}$	0.0256$^{▽}_{▽}$	**0.0008**$^{▼}_{▲}$	**0.1154**$^{▲}_{▲}$	**0.0578**$^{▼}_{▲}$
Entropy-FW	**0.0054**$^{▼}_{▲}$	**0.0282**$^{▼}_{▲}$	0.0264$^{▽}_{▼}$	**0.0006**$^{▼}_{▲}$	**0.0849**$^{▼}_{▲}$	**0.0508**$^{▼}_{△}$
ItemSimple-FW	**0.0085**$^{▼}_{▲}$	**0.0281**$^{▼}_{▲}$	0.0263$^{▽}_{▼}$	**0.0006**$^{▼}_{▲}$	**0.1111**$^{▲}_{▲}$	**0.0573**$^{▼}_{▲}$
ItemUser-FW	**0.0088**$^{▲}_{▲}$	**0.0282**$^{▼}_{▲}$	0.0262$^{▽}_{▼}$	**0.0006**$^{▼}_{▲}$	**0.1131**$^{△}_{▲}$	**0.0571**$^{▼}_{▲}$
RatUser-FW	**0.0090**$^{△}_{▲}$	**0.0281**$^{▼}_{▲}$	0.0265$^{▼}_{▽}$	**0.0006**$^{▼}_{▲}$	**0.1147**$^{▲}_{▲}$	**0.0569**$^{▼}_{▲}$
RatItem-FW	**0.0089**$^{△}_{▲}$	**0.0278**$^{▼}_{▲}$	0.0264$^{▽}_{▽}$	**0.0006**$^{▼}_{▲}$	**0.1154**$^{▲}_{▲}$	**0.0573**$^{▼}_{▲}$
IRUser-FW	**0.0085**$^{▼}_{▲}$	**0.0280**$^{▼}_{▲}$	0.0266$^{▽}_{▽}$	**0.0007**$^{▼}_{▲}$	**0.1094**$^{▼}_{▲}$	**0.0568**$^{▼}_{▲}$
IRItem-FW	**0.0089**$^{▲}_{▲}$	**0.0278**$^{▼}_{▲}$	0.0256$^{▼}_{▼}$	**0.0006**$^{▼}_{▲}$	**0.1123**$^{△}_{▲}$	**0.0570**$^{▼}_{▲}$
IRUserItem-FW	**0.0085**$^{▼}_{▲}$	**0.0281**$^{▼}_{▲}$	0.0265$^{▽}_{▽}$	**0.0007**$^{▼}_{▲}$	**0.1116**$^{△}_{▲}$	**0.0573**$^{▼}_{▲}$

Table A.18. Dynamic ensemble performance values (MAP@10) using the rating-based user predictors, on the MovieLens 1M and using the <u>AR methodology</u>.

Table A.19 shows performance values of the hybrid recommenders using the 1R methodology. The outcome of this experiment is identical to that presented in Table 7.3, except that now the best performing ensemble is a dynamic hybrid recommenders, either the perfect correlation or the PC-OM, instead of the best static ensemble, further validating our framework.

Additionally, in Table A.20 we show the performance of the dynamic hybrid recommenders using item predictors with the MAP metric. We may observe that these results are very similar to those presented for P@10 in Table 7.9, which emphasises the flexibility of our approach, in terms of being able to obtain performance improvements when using different evaluation metrics.

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0559	0.3335	0.1815	0.1815	0.1815	0.1815
R2 (λ=0.0)	0.0847	0.0847	0.0847	0.0125	0.5163	0.3430
Baseline (λ=0.5)	0.1147	0.2384	0.1974	0.0989	0.4004	0.3211
Best static	0.1186	0.3369	0.2248	0.1789	0.5189	0.3618
(best λ)	(0.3)	(0.9)	(0.8)	(0.9)	(0.1)	(0.1)
Perfect correlation	<u>0.1409</u>	<u>0.3625</u>	<u>0.2584</u>	<u>0.1919</u>	0.5139	<u>0.3822</u>
PC-OM	0.1176	0.3014	0.2380	0.1375	<u>0.5213</u>	0.3785
PC-FW	0.1155	0.2678	0.2141	0.1189	0.5012	0.3715
Entropy-OM	0.1138▼▼	**0.3187**▼▲	**0.2103**▼▲	**0.1383**▼▲	0.3802▼▼	0.3087▼▼
ItemSimple-OM	0.1107▼▼	**0.3210**▼▲	**0.2111**▼▲	**0.1398**▼▲	**0.5022**▼▲	**0.3517**▼▲
ItemUser-OM	0.1084▼▼	**0.3174**▼▲	**0.2097**▼▲	**0.1379**▼▲	**0.5093**▼▲	**0.3534**▼▲
RatUser-OM	0.1051▼▼	**0.3216**▼▲	**0.2130**▼▲	**0.1405**▼▲	**0.5162**▼▲	**0.3556**▼▲
RatItem-OM	0.1046▼▼	**0.3187**▼▲	**0.2120**▼▲	**0.1400**▼▲	**0.5168**▼▲	**0.3560**▼▲
IRUser-OM	0.1109▼▼	**0.3131**▼▲	**0.2092**▼▲	**0.1382**▼▲	**0.4991**▼▲	**0.3495**▼▲
IRItem-OM	0.1073▼▼	**0.3062**▼▲	**0.2025**▼▲	**0.1328**▼▲	**0.5030**▼▲	**0.3484**▼▲
IRUserItem-OM	0.1082▼▼	**0.3127**▼▲	**0.2091**▼▲	**0.1381**▼▲	**0.5035**▼▲	**0.3497**▼▲
Entropy-FW	**0.1160**▼▲	**0.2703**▼▲	**0.2042**▼▲	**0.1184**▼▲	0.3965▼▼	0.3196▼▼
ItemSimple-FW	0.1142▼▼	**0.2712**▼▲	**0.2049**▼▲	**0.1192**▼▲	**0.4864**▼▲	**0.3491**▼▲
ItemUser-FW	0.1120▼▼	**0.2700**▼▲	**0.2037**▼▲	**0.1181**▼▲	**0.4951**▼▲	**0.3509**▼▲
RatUser-FW	0.1097▼▼	**0.2713**▼▲	**0.2058**▼▲	**0.1194**▼▲	**0.5049**▼▲	**0.3535**▼▲
RatItem-FW	0.1095▼▼	**0.2707**▼▲	**0.2046**▼▲	**0.1193**▼▲	**0.5066**▼▲	**0.3542**▼▲
IRUser-FW	0.1146▼▼	**0.2699**▼▲	**0.2044**▼▲	**0.1183**▼▲	**0.4828**▼▲	**0.3474**▼▲
IRItem-FW	0.1109▼▼	**0.2672**▼▲	**0.2010**▼▲	**0.1158**▼▲	**0.4905**▼▲	**0.3461**▼▲
IRUserItem-FW	0.1123▼▼	**0.2697**▼▲	**0.2043**▼▲	**0.1181**▼▲	**0.4894**▼▲	**0.3481**▼▲

Table A.19. Dynamic ensemble performance values (MAP@10) using the rating-based user predictors, on the MovieLens 1M dataset and using the <u>1R methodology</u>.

	HRI1	HRI2	HRI3	HRI4
R1 (λ=1.0)	0.2498	0.2498	0.0835	0.0835
R2 (λ=0.0)	0.0717	0.0914	0.0717	0.0914
Baseline (λ=0.5)	0.1550	0.1746	0.0878	0.0932
Best static	0.2146	0.2233	0.0892	0.0954
(best λ)	(0.9)	(0.9)	(0.7)	(0.2)
Entropy-OM	0.1283▼▼	0.1412▼▼	0.0872▼▼	**0.0979**▲▲
UserSimple-OM	**0.2116**▼▲	**0.2273**▲▲	**0.0879**▼▲	**0.0975**▲▲
UserItem-OM	**0.2129**▼▲	**0.2267**▲▲	0.0866▼▼	**0.0973**▲▲
RatItem-OM	**0.2166**▲▲	**0.2305**▲▲	0.0872▼▼	**0.0975**▲▲
RatUser-OM	**0.2231**▲▲	**0.2385**▲▲	0.0870▼▼	**0.0977**▲▲
URItem-OM	**0.2021**▼▲	**0.2136**▼▲	0.0869▼▼	**0.0973**▲▲
URUser-OM	**0.2176**▲▲	**0.2312**▲▲	0.0856▼▼	**0.0974**▲▲
URItemUser-OM	**0.2071**▼▲	**0.2200**▼▲	0.0870▼▼	**0.0973**▲▲
Entropy-FW	0.1382▼▼	0.1517▼▼	0.0873▼▼	**0.0981**▲▲
UserSimple-FW	**0.1770**▼▲	**0.1950**▼▲	**0.0900**▲▲	**0.0975**▲▲
UserItem-FW	**0.1771**▼▲	**0.1953**▼▲	**0.0902**▲▲	**0.0973**▲▲
RatItem-FW	**0.1776**▼▲	**0.1957**▼▲	**0.0902**▲▲	**0.0975**▲▲
RatUser-FW	**0.1793**▼▲	**0.1976**▼▲	**0.0904**▲▲	**0.0974**▲▲
URItem-FW	**0.1737**▼▲	**0.1907**▼▲	**0.0893**▲▲	**0.0969**▲▲
URUser-FW	**0.1782**▼▲	**0.1964**▼▲	**0.0903**▲▲	**0.0971**▲▲
URItemUser-FW	**0.1752**▼▲	**0.1931**▼▲	**0.0899**▲▲	**0.0974**▲▲

Table A.20. Dynamic ensemble performance values (MAP) using the rating-based item predictors, on the MovieLens 1M dataset and using the uuU1R methodology.

A.5.2 Performance results from dynamic ensembles on log data

In this section we compare the performance values of hybrid recommenders using the 1R methodology with log-based predictors, and P@10 and MAP@10 metrics. From Table A.21 and Table A.22 we can observe that the performance values are very similar to those shown in Table 7.11 and Table 7.12, respectively. This may be due to the fact that correlations for MAP@10 presented in Section A.4.2 were also analogous, since precision and MAP are almost equivalent under the 1R methodology as there is only one relevant item for each evaluated ranking. From Table A.21 we have to note, nonetheless, that the combination HL2 obtains worse performance values for the OM weighting strategy. Another difference is that the best performing ensemble for the combination HL2 now is achieved by the perfect correlation method, not by the best static recommender, as shown in Table 7.11 and Table 7.12. This shows that there would be room for improvement in the HL2 combination if we are able to define predictors with stronger correlation values. A similar situation is found for the combination HL3 in the five-fold split.

	HL1	HL2	HL3
R1 (λ=1.0)	0.4094	0.4094	0.7229
R2 (λ=0.0)	0.8255	0.5601	0.4094
Baseline (λ=0.5)	0.6922	0.6244	0.6429
Best static	0.7913	0.6326	0.7256
(best λ)	(0.1)	(0.3)	(0.1)
Perfect correlation	0.7485	0.6470	0.6940
PC-OM	0.7272	0.6239	0.6763
PC-FW	0.7188	0.6240	0.6669
ItemSimple-OM	**0.7742**▼▲	0.6235▼▽	**0.7165**▼▲
Autocorrelation-OM	0.6592▼▼	0.5962▼▼	0.6163▼▼
TimeSimple-OM	**0.7676**▼▲	0.5913▼▼	**0.6955**▼▲
ItemTime-OM	**0.7762**▼▲	0.6200▼▽	**0.7149**▼▲
ItemPriorTime-OM	**0.7354**▼▲	0.6223▼▼	**0.6777**▼▲
ItemSimple-FW	**0.7597**▼▲	**0.6329**▲▲	**0.7106**▼▲
Autocorrelation-FW	0.6827▼▼	0.6177▼▼	0.6353▼▼
TimeSimple-FW	**0.7514**▼▲	0.6048▼▼	**0.6893**▼▲
ItemTime-FW	**0.7608**▼▲	**0.6292**▼▲	**0.7061**▼▲
ItemPriorTime-FW	**0.7276**▼▲	**0.6278**▼▲	**0.6714**▼▲

Table A.21. Dynamic ensemble performance values (MAP@10) using the log-based user predictors, on the Last.fm temporal split and using the 1R methodology.

	HL1	HL2	HL3
R1 (λ=1.0)	0.0453	0.0453	0.5824
R2 (λ=0.0)	0.5901	0.5387	0.0453
Baseline (λ=0.5)	0.4233	0.3961	0.3308
Best static	0.5728	0.5317	0.5463
(best λ)	(0.1)	(0.1)	(0.1)
Perfect correlation	0.5820	0.5396	0.5728
PC-OM	0.5805	0.5403	0.5768
PC-FW	0.5795	0.5357	0.5611
ItemSimple-OM	**0.5395**▼▲	**0.4894**▼▲	**0.4397**▼▲
Autocorrelation-OM	0.3972▼▼	0.3659▼▼	**0.3642**▼▲
TimeSimple-OM	**0.5561**▼▲	**0.5206**▲▲	0.2405▼▼
ItemTime-OM	**0.5407**▼▲	**0.4881**▼▲	**0.4375**▼▲
ItemPriorTime-OM	**0.4577**▼▲	**0.4108**▼▲	**0.4276**▼▲
ItemSimple-FW	**0.5240**▼▲	**0.4791**▼▲	**0.3826**▼▲
Autocorrelation-FW	0.4191▼▼	0.3896▼▼	**0.3523**▼▲
TimeSimple-FW	**0.5372**▼▲	**0.5033**▼▲	0.2813▼▼
ItemTime-FW	**0.5243**▼▲	**0.4779**▼▲	**0.3819**▼▲
ItemPriorTime-FW	**0.4582**▼▲	**0.4201**▼▲	**0.3783**▼▲

Table A.22. Dynamic ensemble performance values (MAP@10) using the log-based user predictors, on the Last.fm five-fold split and using the 1R methodology.

A.5.3 Performance results from dynamic ensembles on social data

In this section we extend the results presented in Section 7.3.3, where P@10 and methodology AR were used in the two versions of CAMRa dataset. Here we use the same methodology, but the MAP@10 metric, as in the previous sections. In Table A.23 we can observe that there are some differences in the social version of the dataset when compared against the results shown in Table 7.14. We may attribute such differences to the different optimal static hybrid recommenders obtained since the correlations have not changed significantly. For instance, for HS1, the best lambda for static hybrids was 0.3 with P@10 and now it is 0.7 with MAP@10.

It is worth noting that even when the best static is so different from one metric to the other, the best performance is still obtained with the perfect correlation ensemble, which again reinforces the idea that finding predictors with higher correlations would be able to dynamically select the best weights in a user basis.

Finally, in Table A.24 we present the results regarding the collaborative version of the CAMRa dataset. In this case, we have a strong evidence towards the benefits of using performance predictors. First, we have to recall the absolute values of the pLSA correlations are stronger for the MAP metric than for precision. Then, now we

	HS1	HS2	HS3	HS4
R1 (λ=1.0)	0.2002	0.2002	0.2192	0.2192
R2 (λ=0.0)	0.1203	0.0364	0.1203	0.0364
Baseline (λ=0.5)	0.2175	0.2287	0.2555	0.2398
Best static	0.2222	0.2349	0.2630	0.2408
(best λ)	(0.3)	(0.9)	(0.3)	(0.9)
Perfect correlation	$\underline{0.2632}$	0.2562	0.2652	$\underline{0.2511}$
PC-OM	0.2451	$\underline{0.2576}$	$\underline{0.2668}$	0.2497
PC-FW	0.2355	0.2378	0.2661	0.2446
AvgNeighDeg-OM	**0.2176**▽△	**0.2403**▽△	**0.2570**▽△	0.2229▽▼
BetCentrality-OM	0.2031▼▽	0.2232▽▽	0.2146▼▼	0.2236▽▼
ClustCoeff-OM	0.2082▼▽	0.2201▼▽	0.2261▼▼	0.2155▽▼
Degree-OM	0.2147▼▽	**0.2303**▼△	**0.2615**▽△	0.2166▽▼
EgoCompSize-OM	0.2078▼▽	0.2276▽▽	**0.2577**▽△	0.2214▽▼
HITS-OM	0.2127▼▽	**0.2428**▽△	0.2187▼▼	0.2231▽▼
PageRank-OM	0.2108▼▽	**0.2289**▽△	**0.2573**▽△	0.2238▽▼
TwoHopNeigh-OM	0.2121▼▽	**0.2377**▽△	0.2545▽▽	0.2202▽▼
AvgNeighDeg-FW	**0.2218**▽▲	**0.2370**▽△	**0.2574**▽△	0.2300▼▼
BetCentrality-FW	0.2171▼▽	**0.2351**▽△	0.2460▼▼	0.2344▼▼
ClustCoeff-FW	0.2129▼▽	**0.2348**▽△	0.2434▼▽	0.2344▼▼
Degree-FW	**0.2176**▽▲	**0.2373**▽▲	**0.2627**▽△	0.2332▼▽
EgoCompSize-FW	0.2124▼▽	**0.2373**▽▲	**0.2616**▽△	0.2352▼▽
HITS-FW	0.2164▼▽	**0.2380**▽▲	0.2405▼▼	0.2331▼▼
PageRank-FW	0.2169▼▽	**0.2363**▽△	**0.2574**▽△	0.2336▼▼
TwoHopNeigh-FW	**0.2177**▽▲	**0.2373**▽▲	**0.2587**▽△	0.2323▼▼

Table A.23. Dynamic ensemble performance values (MAP@10) using the log-based user predictors, on the CAMRa social dataset and using the AR methodology.

	HS1	HS2	HS3	HS4
R1 (λ=1.0)	0.1234	0.1234	0.1334	0.1334
R2 (λ=0.0)	0.1474	0.0195	0.1474	0.0195
Baseline (λ=0.5)	0.2190	0.1441	0.2359	0.1520
Best static	0.2228	0.1494	0.2440	0.1520
(best λ)	(0.3)	(0.9)	(0.2)	(0.5)
Perfect correlation	0.2379	0.1597	0.2396	0.1590
PC-OM	0.1494	0.1577	0.1657	0.1535
PC-FW	0.1492	0.1462	0.1660	0.1498
AvgNeighDeg-OM	**0.2226**▽△	**0.1511**△△	**0.2426**▽▲	0.1409▼▼
BetCentrality-OM	0.2069▼▼	0.1412▼▽	0.2119▼▼	0.1410▼▼
ClustCoeff-OM	0.2071▼▼	0.1390▼▽	0.2199▼▼	0.1386▼▼
Degree-OM	0.2175▼▽	0.1457▼▽	**0.2454**△▲	0.1378▼▼
EgoCompSize-OM	0.2142▼▽	0.1441▽	**0.2417**▽△	0.1395▼▼
HITS-OM	0.2100▼▼	**0.1519**△△	0.2136▼▼	0.1419▼▼
PageRank-OM	0.2110▼▼	**0.1447**▽△	**0.2416**▽△	0.1405▼▼
TwoHopNeigh-OM	0.2156▼▽	**0.1514**△△	**0.2469**△▲	0.1392▼▼
AvgNeighDeg-FW	**0.2214**▽△	**0.1501**△▲	**0.2410**▽▲	0.1464▼▼
BetCentrality-FW	0.2145▽△	**0.1479**▽△	0.2254▼▼	0.1475▼▼
ClustCoeff-FW	0.2167▽△	**0.1474**▽△	0.2266▼▼	0.1474▼▼
Degree-FW	**0.2197**▽△	**0.1497**△▲	**0.2437**▽▲	0.1464▼▼
EgoCompSize-FW	0.2189▼▽	**0.1490**▽▲	**0.2405**▽▲	0.1474▼▼
HITS-FW	0.2157▽△	**0.1502**△▲	0.2241▼▼	0.1461▼▼
PageRank-FW	0.2170▽△	**0.1476**▽△	**0.2440**▲	0.1465▼▼
TwoHopNeigh-FW	**0.2216**▽△	**0.1502**△▲	**0.2423**▽▲	0.1469▼▼

Table A.24. Dynamic ensemble performance values (MAP@10) using the log-based user predictors, on the CAMRa collaborative dataset and using the AR methodology.

have to note that the performance values are now better fo MAP than for precision, in particular for HS1 and HS3, where dynamic hybrid recommenders outperform the baselines in a larger number of cases.