Chapter 7

Dynamic recommender ensembles

Hybrid recommender systems – and recommender ensembles as a particular case – have become a very popular strategy for making recommendations, since they help alleviate most of the shortcomings of the individual recommenders combined. They have, however, specific problems such as the need of deciding which information sources should be exploited, which recommenders should exploit each of these sources, and how the combination of recommenders should be configured.

In this chapter we propose a framework to decide how dynamic hybridisation should be balanced, by estimating its expected improvements on individual recommendations. Furthermore, we provide some requirements to decide when to build such hybridisation. Within the spectrum of hybrid recommendation approaches, we focus on those that linearly combine the output from several recommenders, and use different weights for generating a particular aggregation of the individual recommendations. In the standard approach, these weights are typically fixed regardless of the user for which recommendations are produced, or the recommended items. In this context we investigate the use of performance predictors to assign those weights dynamically depending on the target user or item. We evaluate our approach using the predictors proposed in the previous chapter. The results obtained show that the generated dynamic ensembles are capable of outperforming their static counterparts. Furthermore, they also show that dynamic ensembles can be improved if predictors with stronger predictive power (higher correlation values as observed in the previous chapter) are used.

In Section 7.1 we present and formulate the research problem of recommendation hybridisation. Next, in Section 7.2 we describe our proposed performance prediction framework for dynamic hybrid recommendation. Section 7.3 describes the experiments conducted and provide an overall discussion of the obtained results. Finally, in Section 7.4 some conclusions are given.

7.1 Problem statement

As described in Chapter 2, hybrid recommenders are built by the combination of different recommendation methods. In the simplest and typical case, hybrid recommendations are produced by weighting and summing the utility values output by some recommenders, forming a so called recommender ensemble where an arbitrary number of algorithms of different kinds (content-based, user-based collaborative filtering, item-based collaborative filtering, social-based, demographics-based, etc.) can be combined.

Researchers in Machine Learning have known for long that the combination of classifiers usually achieves better results than each method separately, which is also true in Recommender Systems – the Netflix prize has been a paradigmatic example of this, where all the top classified teams used large recommender ensembles. We focus on weighted hybrid approaches, as an option that begets a simple and general formulation of the dynamic balance of the combined methods R_k by just setting the weights λ_k of each method in the hybrid combination. This approach can be expressed as follows:

$$\tilde{r}(u,i) = \sum_k \lambda_k * \tilde{r}_{R_k}(u,i) \text{ s.t. } \sum_k \lambda_k = 1 \tag{7.1}$$

In this chapter we investigate whether the performance predictors proposed in the previous chapter – where we have already found degrees of correlation between the ambiguity (clarity) of the user's preferences and the accuracy of the system's recommendations – can be useful for hybridisation. Specifically, we aim to use these predictors to build **dynamic hybrid recommenders** in such a way that the weight λ_k depends not only on the recommender but also on the current user u, or potentially other variables such as the item i or other available context information. We propose to specify such weights according to the ambiguity of the user's preferences or item's patterns, that is, we aim to use the performance predictors defined in the Chapter 6 to estimate those weights.

In the next section we propose a framework to perform dynamic hybrid recommendation where we use recommendation performance predictors and we analyse different requirements related to the adaptation of such predictors to produce weights in a hybrid recommender combination. After that, three different experiments are presented, where the predictors proposed in Chapter 6 are used as dynamic weights in the combination.

7.2 A performance prediction framework for ensemble recommendation

Let us simplify Equation (7.1) to the case where only two recommenders $R1$ and $R2$ are used. In this situation, only one weighting factor λ is needed (because of the constraint for the weights to sum to one) and we would have the following formulation:

$$\tilde{r}(u,i) = \lambda * \tilde{r}_{R1}(u,i) + (1-\lambda) * \tilde{r}_{R2}(u,i) \tag{7.2}$$

In this case, since the λ weight is the same for every user u and item i we refer to such a recommender as a *static hybrid*. However, a single value of the combination parameter λ is not generally the optimal for each (user, item) pair. Therefore, instead of Equation (7.2), we may want to consider:

$$\tilde{r}(u,i) = \gamma_{R1}(u,i) * \tilde{r}_{R1}(u,i) + \gamma_{R2}(u,i) * \tilde{r}_{R2}(u,i) \tag{7.3}$$

where γ_R is the combination parameter which may depend on the current user, item, or both, and probably also depending on the recommender R. In this case we refer to such method as a *dynamic hybrid*.

A suitable assignment of the $\gamma(u,i)$ parameters is a difficult task. In our approach, however, we propose to use the performance prediction methodology developed in the previous chapter, whenever the predictors show some correlation with the performance of a recommender. In this way, since we have some evidence that the performance predictors are able to estimate in advance the performance of a user in a user or item basis, we can use such estimations to weight accordingly the ratings predicted for a given user and item pair by each recommender.

In this context, it is not granted in general to obtain improvements whenever a performance predictor is used in a dynamic ensemble. We have to devise a set of conditions in which such predictors may be used; moreover, the ensemble problem has to be well defined, which is not always true as we shall show. Hence, we define a framework for dynamic hybrid recommendation based on recommendation performance predictors, characterised by some prerequisites, a specific normalisation strategy, and a weighting distribution among recommenders. In this framework, the weights γ_R are obtained by transformations of the values obtained by a performance predictor, in a similar way as the work presented in (Yom-Tov et al., 2005b) on rank aggregation in Information Retrieval, but in the context of Recommender Systems.

7.2.1 Requirements

A first requirement to use a performance predictor for weighting the recommenders of an ensemble, is that it should correlate positively with the performance of not all but some of such recommenders, or with the performance of all the recommenders but to different degrees. If a performance predictor correlates positively with all the recommenders in an ensemble to a similar extent, it does not provide a discriminative criteria to weight the recommenders any differently.

A predictor should be used to assign weights to those recommenders of the ensemble with which it correlates for performance. These assignments also alter the weights of the uncorrelated recommenders, since the weights of all the recommenders in the ensemble need to sum to 1. However, this should not affect the overall performance contribution of these recommenders, as the resulting weight should correspond randomly with their performance (hence the unpredicted recommenders' weight can be expected to change for good as much as for bad, whereas the weight of predicted recommenders should change more often for good).

Figure 7.1 shows which correlations can be considered valid according to the statements presented above, for an ensemble with two recommenders R1 and R2. The horizontal axis depicts the correlation with respect R1 and the vertical axis with R2. Hence, the dotted area represents those situations where a predictor's correlation for R1 is higher than for R2, and thus, the predictor should weight R1. Analogously, the striped area represents the candidate situations where the predictor should weight R2. Furthermore, when correlations with R1 and R2 are too similar (diagonal) no weighting assignment is preferred, and thus, if a predictor lies in the white area it should be used for weighting neither R1 nor R2 for the reasons described above.

Another requirement is that a recommender should not have an always superior or always inferior performance to those of the rest of the ensemble's recommenders. Otherwise the problem is distorted by the fact that the best weight is the one that gets closest to 0 for the recommenders that systematically perform worse (or 1 for the best), regardless of how excellent or terribly bad is the applied strategy, or the predictive power of the approach, since a biased predictor (either towards 0 or 1, depending on which recommender (the worst or the best) such predictor is weighting) would obtain very good results. This issue is recognised in (van Setten, 2005) where the author presents the situation where all recommenders produce item suggestions that are all too low or all too high with respect to the true user's preferences, and then the recommender ensemble is less accurate than the best individual recommender. In summary, underperforming recommenders are useless in an ensemble to begin with, or equivalently, the over performing one(s) should be used alone, and thus, there is no true weighting problem to solve.

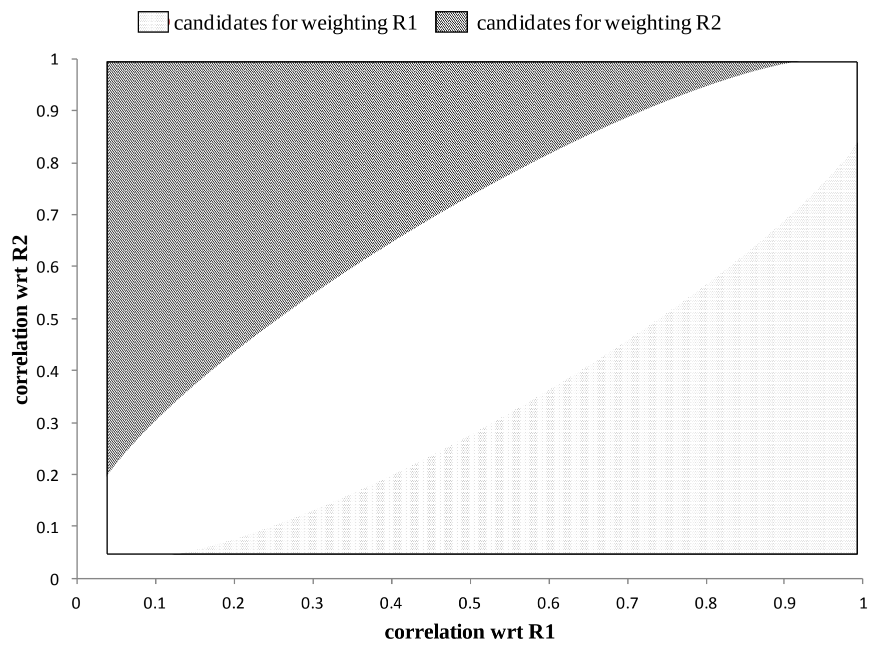

Figure 7.1. Valid predictor correlation regions for a recommender ensemble of size 2.

7.2.2 Predictor normalisation

The output of a predictor is required to correlate with the performance of a recommender, but it is not necessarily by itself a good value for weighting the recommender in an ensemble, as already pointed out in (Hauff et al., 2009). In order to generate appropriate weights, the predictor output should be transformed by a monotonic function into values on a comparable scale, such as simply $[0,1]$. We shall call this transformation "normalisation."

In this context, different transformations can be applied. Mapping the minimum value to 0 and the maximum to 1 is the simplest transformation, also known as *min-max* score normalisation (Renda and Straccia, 2003). Another common approach is to map (named *rank-sim* by Renda and Straccia, 2003) the predictor scores onto evenly distributed points in the $[0,1]$, preserving their order. Min-max preserves the original predictor score distribution, while rank-sim maps it onto a uniform distribution. There is no obvious a priori reason to decide which case is preferable, to preserve the original distribution, or to equalise it somehow, and in fact more complex normalisation techniques could be used, like the one proposed in (Fernández et al., 2006b).

7.2.3 Weight distribution among recommenders

Once the predictor output has been normalised, it still needs a final adjustment to ensure, among other things, that the sum of the weights assigned to the ensemble's recommenders is 1. How this step is done depends, mainly, on how many recommenders are weighted by predictors, more specifically on whether all or only some of the combined recommenders are treated by performance predictors. Hence, we consider two options for the distribution of the weights among the recommenders:

a) Only some of the recommenders in the ensemble are given dynamic weights. The rest of the recommenders receive the same weight, ensuring the weights of the ensemble's recommenders sum up to 1. This can be done in different ways:
    - Assigning a weight of 0.5 to the unpredicted recommenders, and dividing all weights by the total sum. This strategy is named as *fixed weight* or **FW**.
    - Assigning the dynamic weights to the corresponding recommenders, if we assume that their sum is ≤ 1, then we divide 1 minus the sum of dynamic coefficients equally among the unpredicted recommenders. We denote this strategy as *one minus* or **OM**. If the sum is greater than 1, we have to divide by the total sum and normalise it by the total number of predictors.

b) All recommenders are weighted using a specific predictor per recommender. This is not easy to grant in general, as there may not be predictors for all the recommenders combined. In case this option is taken, the weights can be simply normalised by the sum of weights.

Furthermore, if the output of each recommender has a different range, it would be necessary to apply an additional normalisation step to the recommender scores. The most usual strategies are the ones described in the previous section: score or rank normalisation (Renda and Straccia, 2003).

7.3 Experimental results

We next report experiments assessing the usefulness of the proposed predictors for adjusting the weights of a recommender ensemble, once their predictive power has been confirmed against the recommenders' actual performance, as reported in the previous chapter. We identify the combinations of recommenders that meet the conditions stated in the previous section for the dynamic combination problem to make sense and select the performance predictors to be applied based on their observed correlation with the performance of the recommenders (as reported in Section 6.5), and the requirements proposed in this chapter, i.e., that one recommender in the ensemble should have a positive correlation with the predictor, and the other should have an opposite or near neutral correlation. Then, we compare dynamic against static ensembles.

Among the different ways to set up static ensembles of two recommenders we take as baselines a) the best performing one in test, and b) the best theoretical static one without prior information, i.e., one with $\lambda = 0.5$. Intuitively, an even weighting is the optimum over the – theoretical – set of all recommender ensembles: if say $\lambda_B = 0.3$ was the best weight for the combination of two recommenders R1+R2, then $\lambda = 0.3$ should be fairly bad for the permutation R2+R1 ($\lambda = 1 - 0.3 = 0.7$ being best). If we assume that performance loss is convex with respect to $|\lambda - \lambda_B|$ – it can be seen that otherwise the hybrid may underperform its constituents –, then $\lambda = 0.5$ is the best compromise for R1+R2 and R2+R1. Since the set of all possible ensembles includes all the permutations of the combined recommenders, $\lambda = 0.5$ is the best (theoretical) overall weight.

We also take as "skylines" (upper bound baselines) an oracle performance predictor consisting of the performance of the recommender itself. We shall refer to this method as 'perfect correlation', where the true performance of both recommenders is used as a weight for hybridisation (hence, such predictor would have a correlation of 1.0 with the recommender's performance), whereas we shall refer to it as 'PC-OM' and 'PC-FW' when the performance of only one recommender is used (the same recommender being weighted by the predictors) along with the one minus or the fixed weight strategy for weight distribution (see Section 7.2.3). In all cases we apply a rank normalisation technique on the recommenders' scores.

In the subsequent sections we present three experiments conducted to evaluate the proposed performance predictors. In the first experiment we use the rating-based predictors and test both user- and item-based performance predictors presented in Section 6.2.1. We use the MovieLens dataset, and compare the results with four of the evaluation methodologies presented in Chapter 4, i.e., AR, 1R, P1R, and U1R. In the second experiment we use predictors based on log data. We evaluate the predictors presented in Section 6.2.2 on the two versions of the Last.fm dataset using the 1R methodology. Finally, in the third experiment we test the social-based predictors presented in Section 6.3 on the CAMRa dataset and the AR methodology.

7.3.1 Dynamic recommender ensembles on rating data

As a first instantiation of our framework for building dynamic recommender ensembles described in Section 7.2, we first have to identify the recommenders to combine, that is: one of the recommenders should have a positive correlation with the predictor, while the other should have an opposite or near neutral correlation; besides, they should not perform very differently.

According to the correlation results presented in Section 6.5.1, we identify the pairs of recommenders presented in Table 7.1 as combinations meeting the conditions stated above. The first three ensembles are combinations of a collaborative filtering with a content-based recommendation method. The last ensemble combines a user-based collaborative filtering method with a non-personalised method, and the rest of the ensembles are combinations of two collaborative filtering methods. Al-

	R1	R2
HRU1	TFL1	CB
HRU2	TFL2	CB
HRU3	kNN	CB
HRU4	kNN	IB
HRU5	kNN	pLSA
HRU6	kNN	ItemPop

Table 7.1. Selected recommenders for building dynamic ensemble using user performance predictors that exploit rating-based information (MovieLens dataset).

though some of these combinations have not been typical in the recommender systems literature, in our study they serve as a proof of concept to check whether the proposed dynamic recommender ensemble framework is useful in general or not. We refer the reader to Appendix A.2 for more details about the implementation of the recommenders.

The first two rows of Table 7.2, Table 7.3, Table 7.4, and Table 7.5 show the P@10 values for each of the combined recommenders obtained using the AR, 1R, U1R, and P1R methodologies, respectively. In Appendix A.5.1 we report results with other evaluation metrics. Note that, as mentioned in Chapter 4, in the AR methodology the absolute values are not meaningful since they depend on the amount of relevant information in test; on the other hand, for the 1R related methodologies (i.e., 1R, U1R, and P1R) the precision at 10 metric has an upper bound on 0.1, since there is only one relevant item in each ranking.

In these tables we may observe that among the six considered ensembles, there are cases where the first recommender (with respect to which the performance is predicted) performs better, worse, or similarly to the second recommender. This situation changes accross methodologies and provides for a comparison of the resulting effects when the stated requirements are not met. Analogously, the predictors' correlations may change depending on the evaluation methodology followed, as observed in Section 6.5.1. Specifically, the recommenders presented in Table 7.1 where chosen according to the correlation results obtained for the AR methodology, and we may observe that some of the conditions stated above do not hold for some of the selected cases, for instance, correlation between most of the predictors and kNN recommender is negligible in the 1R, U1R, and P1R methodologies, in contrast with the results found for the AR methodology.

In the tables we may also observe that the best static ensemble is different depending on the evaluation methodology and the combined recommenders. The performance values of the best static ensembles, on the other hand, show an interesting situation that does depend on the specific considered ensemble, namely, whether the (best) static ensembles outperform or not both recommenders. For the AR methodology (Table 7.2), in the case of HRU1, HRU3, HRU5, and HRU6, the best static

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0024	0.0696	0.0307	0.0307	0.0307	0.0307
R2 (λ=0.0)	0.0163	0.0163	0.0163	0.0001	0.1454	0.0897
Baseline (λ=0.5)	0.0106	0.0473	0.0363	0.0008	0.1142	0.0808
Best static	0.0180	0.0668	0.0392	0.0078	0.1475	0.0937
(best λ)	(0.1)	(0.9)	(0.9)	(0.9)	(0.1)	(0.1)
Perfect correlation	$\underline{0.0189}$	$\underline{0.0732}$	$\underline{0.0401}$	$\underline{0.0311}$	0.1469	$\underline{0.0980}$
PC-OM	0.0176	0.0721	0.0434	0.0091	0.1489	0.0958
PC-FW	0.0177	0.0541	0.0379	0.0025	0.1478	0.0958
Entropy-OM	$\mathbf{0.0110}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0685}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0388}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0069}^{\blacktriangledown}_{\blacktriangle}$	$0.1126^{\blacktriangledown}_{\triangledown}$	$0.0791^{\blacktriangledown}_{\blacktriangledown}$
ItemSimple-OM	$\mathbf{0.0170}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0685}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0390}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0072}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.1496}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0919}^{\blacktriangledown}_{\blacktriangle}$
ItemUser-OM	$\mathbf{0.0172}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0680}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0386}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0068}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1513}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0924}^{\blacktriangledown}_{\blacktriangle}$
RatUser-OM	$\mathbf{0.0177}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0687}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0393}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0072}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.1535}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0931}^{\triangledown}_{\blacktriangle}$
RatItem-OM	$\mathbf{0.0178}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0674}^{\triangle}_{\blacktriangle}$	$\mathbf{0.0389}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0066}^{\blacktriangledown}_{\blacktriangle}$	$\underline{\mathbf{0.1542}}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0928}^{\triangledown}_{\blacktriangle}$
IRUser-OM	$\mathbf{0.0169}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0668}_{\blacktriangle}$	$\mathbf{0.0387}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0066}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1487}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0922}^{\blacktriangledown}_{\blacktriangle}$
IRItem-OM	$\mathbf{0.0172}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0655}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0378}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0061}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1500}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0918}^{\blacktriangledown}_{\blacktriangle}$
IRUserItem-OM	$\mathbf{0.0170}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0665}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0388}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0066}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1498}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0916}^{\blacktriangledown}_{\blacktriangle}$
Entropy-FW	$\mathbf{0.0111}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0528}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0369}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0027}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1156}^{\blacktriangledown}_{\blacktriangle}$	$0.0807^{\blacktriangledown}_{\triangledown}$
ItemSimple-FW	$\mathbf{0.0156}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0529}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0369}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0027}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1433}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0908}^{\blacktriangledown}_{\blacktriangle}$
ItemUser-FW	$\mathbf{0.0166}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0529}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0368}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0028}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1468}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0915}^{\blacktriangledown}_{\blacktriangle}$
RatUser-FW	$\mathbf{0.0170}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0528}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0370}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0028}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1498}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0919}^{\blacktriangledown}_{\blacktriangle}$
RatItem-FW	$\mathbf{0.0170}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0529}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0369}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0027}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1499}^{\blacktriangle}_{\blacktriangle}$	$\mathbf{0.0918}^{\blacktriangledown}_{\blacktriangle}$
IRUser-FW	$\mathbf{0.0161}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0526}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0371}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0029}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1420}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0912}^{\blacktriangledown}_{\blacktriangle}$
IRItem-FW	$\mathbf{0.0163}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0525}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0367}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0027}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1459}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0909}^{\blacktriangledown}_{\blacktriangle}$
IRUserItem-FW	$\mathbf{0.0164}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0527}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.0372}^{\blacktriangledown}_{\triangle}$	$\mathbf{0.0028}^{\blacktriangledown}_{\blacktriangle}$	$\mathbf{0.1452}^{\triangledown}_{\blacktriangle}$	$\mathbf{0.0908}^{\blacktriangledown}_{\blacktriangle}$

Table 7.2. Dynamic ensemble performance values (P@10) using AR methodology and user predictors (MovieLens dataset). Improvements over the baseline are in bold, the best result for each column is underlined. The value a of each dynamic hybrid is marked with a_y^x, where x and y indicate, respectively, statistical difference with respect to the best static (upper, x) and with respect to the baseline (lower, y). Moreover, $\blacktriangle$ and $\triangle$ indicate, respectively, significant and non-significant improvements over the corresponding recommender. A similar convention with $\blacktriangledown$ and $\triangledown$ indicates values below the recommender performance. Statistical significance is established by paired Wilcoxon $p < 0.05$ in all cases.

outperforms both recommenders, but this is not observed for HRU2 nor for HRU4. In the latter scenarios, thus, it seems hybridisation would not be so useful for combination.

Additionally, regarding the normalisation of the predictor's output we evaluate two normalisation techniques: rank and score normalisation. Since there is no prior information about which normalisation technique would provide better results, we test both, and report the best results in each situation, which are usually achieved by the rank-sim normalisation technique. Finally, the weigh strategy is also included as a parameter of the experiments. Since we only have a predictor for one of the recommenders in the ensemble (denoted as R1), as we explained in Section 7.2.3, we may weight the unpredicted recommender as one minus the predictor value (OM), or as 0.5 and then divide the weights of the two recommenders by the sum of weights (FW).

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0221	0.0690	0.0437	0.0437	0.0437	0.0437
R2 (λ=0.0)	0.0221	0.0221	0.0221	0.0074	0.0836	0.0649
Baseline (λ=0.5)	0.0338	0.0536	0.0469	0.0327	0.0749	0.0658
Best static	0.0338	$\underline{0.0720}$	0.0514	0.0455	$\underline{0.0856}$	0.0696
(best λ)	(0.4)	(0.9)	(0.8)	(0.9)	(0.1)	(0.2)
Perfect correlation	$\underline{0.0370}$	0.0715	$\underline{0.0553}$	$\underline{0.0458}$	0.0840	$\underline{0.0723}$
PC-OM	0.0358	0.0683	0.0507	0.0353	0.0811	0.0709
PC-FW	0.0343	0.0592	0.0482	0.0344	0.0803	0.0699
Entropy-OM	0.0332$^{\blacktriangledown}_{\blacktriangledown}$	**0.0662**$^{\blacktriangledown}_{\blacktriangle}$	**0.0472**$^{\blacktriangledown}_{\vartriangle}$	**0.0382**$^{\blacktriangledown}_{\blacktriangle}$	0.0709$^{\blacktriangledown}_{\blacktriangledown}$	0.0626$^{\blacktriangledown}_{\blacktriangledown}$
ItemSimple-OM	0.0304$^{\blacktriangledown}_{\blacktriangledown}$	**0.0666**$^{\blacktriangledown}_{\blacktriangle}$	**0.0473**$^{\blacktriangledown}_{\blacktriangle}$	**0.0384**$^{\blacktriangledown}_{\blacktriangle}$	**0.0844**$^{\blacktriangledown}_{\blacktriangle}$	**0.0681**$^{\blacktriangledown}_{\blacktriangle}$
ItemUser-OM	0.0305$^{\blacktriangledown}_{\blacktriangledown}$	**0.0660**$^{\blacktriangledown}_{\blacktriangle}$	**0.0471**$^{\blacktriangledown}_{\vartriangle}$	**0.0381**$^{\blacktriangledown}_{\blacktriangle}$	**0.0847**$^{\blacktriangledown}_{\blacktriangle}$	**0.0680**$^{\blacktriangledown}_{\blacktriangle}$
RatUser-OM	0.0307$^{\blacktriangledown}_{\blacktriangledown}$	**0.0666**$^{\blacktriangledown}_{\blacktriangle}$	**0.0478**$^{\blacktriangledown}_{\blacktriangle}$	**0.0386**$^{\blacktriangledown}_{\blacktriangle}$	**0.0850**$^{\blacktriangledown}_{\blacktriangle}$	**0.0680**$^{\blacktriangledown}_{\blacktriangle}$
RatItem-OM	0.0305$^{\blacktriangledown}_{\blacktriangledown}$	**0.0663**$^{\blacktriangledown}_{\blacktriangle}$	**0.0475**$^{\blacktriangledown}_{\blacktriangle}$	**0.0385**$^{\blacktriangledown}_{\blacktriangle}$	**0.0849**$^{\blacktriangledown}_{\blacktriangle}$	**0.0678**$^{\blacktriangledown}_{\blacktriangle}$
IRUser-OM	0.0304$^{\blacktriangledown}_{\blacktriangledown}$	**0.0655**$^{\blacktriangledown}_{\blacktriangle}$	**0.0470**$^{\blacktriangledown}_{\vartriangle}$	**0.0381**$^{\blacktriangledown}_{\blacktriangle}$	**0.0839**$^{\blacktriangledown}_{\blacktriangle}$	**0.0675**$^{\blacktriangledown}_{\blacktriangle}$
IRItem-OM	0.0298$^{\blacktriangledown}_{\blacktriangledown}$	**0.0644**$^{\blacktriangledown}_{\blacktriangle}$	0.0457$^{\blacktriangledown}_{\blacktriangledown}$	**0.0370**$^{\blacktriangledown}_{\blacktriangle}$	**0.0839**$^{\blacktriangledown}_{\blacktriangle}$	**0.0671**$^{\blacktriangledown}_{\blacktriangle}$
IRUserItem-OM	0.0305$^{\blacktriangledown}_{\blacktriangledown}$	**0.0655**$^{\blacktriangledown}_{\blacktriangle}$	**0.0471**$^{\blacktriangledown}_{\vartriangle}$	**0.0381**$^{\blacktriangledown}_{\blacktriangle}$	**0.0841**$^{\blacktriangledown}_{\blacktriangle}$	**0.0674**$^{\blacktriangledown}_{\blacktriangle}$
Entropy-FW	**0.0339**$^{\vartriangle}_{\vartriangle}$	**0.0594**$^{\blacktriangledown}_{\blacktriangle}$	**0.0472**$^{\blacktriangledown}_{\vartriangle}$	**0.0356**$^{\blacktriangledown}_{\blacktriangle}$	0.0686$^{\blacktriangledown}_{\blacktriangledown}$	0.0650$^{\blacktriangledown}_{\blacktriangledown}$
ItemSimple-FW	0.0321$^{\blacktriangledown}_{\blacktriangledown}$	**0.0596**$^{\blacktriangledown}_{\blacktriangle}$	**0.0473**$^{\blacktriangledown}_{\blacktriangle}$	**0.0358**$^{\blacktriangledown}_{\blacktriangle}$	**0.0837**$^{\blacktriangledown}_{\blacktriangle}$	**0.0684**$^{\blacktriangledown}_{\blacktriangle}$
ItemUser-FW	0.0320$^{\blacktriangledown}_{\blacktriangledown}$	**0.0594**$^{\blacktriangledown}_{\blacktriangle}$	**0.0471**$^{\blacktriangledown}_{\vartriangle}$	**0.0356**$^{\blacktriangledown}_{\blacktriangle}$	**0.0843**$^{\blacktriangledown}_{\blacktriangle}$	**0.0683**$^{\blacktriangledown}_{\blacktriangle}$
RatUser-FW	0.0321$^{\blacktriangledown}_{\blacktriangledown}$	**0.0596**$^{\blacktriangledown}_{\blacktriangle}$	**0.0475**$^{\blacktriangledown}_{\blacktriangle}$	**0.0359**$^{\blacktriangledown}_{\blacktriangle}$	**0.0848**$^{\blacktriangledown}_{\blacktriangle}$	**0.0684**$^{\blacktriangledown}_{\blacktriangle}$
RatItem-FW	0.0321$^{\blacktriangledown}_{\blacktriangledown}$	**0.0595**$^{\blacktriangledown}_{\blacktriangle}$	**0.0473**$^{\blacktriangledown}_{\blacktriangle}$	**0.0358**$^{\blacktriangledown}_{\blacktriangle}$	**0.0847**$^{\blacktriangledown}_{\blacktriangle}$	**0.0684**$^{\blacktriangledown}_{\blacktriangle}$
IRUser-FW	0.0320$^{\blacktriangledown}_{\blacktriangledown}$	**0.0592**$^{\blacktriangledown}_{\blacktriangle}$	**0.0471**$^{\blacktriangledown}_{\vartriangle}$	**0.0356**$^{\blacktriangledown}_{\blacktriangle}$	**0.0834**$^{\blacktriangledown}_{\blacktriangle}$	**0.0680**$^{\blacktriangledown}_{\blacktriangle}$
IRItem-FW	0.0318$^{\blacktriangledown}_{\blacktriangledown}$	**0.0588**$^{\blacktriangledown}_{\blacktriangle}$	0.0465$^{\blacktriangledown}_{\blacktriangledown}$	**0.0349**$^{\blacktriangledown}_{\blacktriangle}$	**0.0835**$^{\blacktriangledown}_{\blacktriangle}$	**0.0674**$^{\blacktriangledown}_{\blacktriangle}$
IRUserItem-FW	0.0320$^{\blacktriangledown}_{\blacktriangledown}$	**0.0592**$^{\blacktriangledown}_{\blacktriangle}$	**0.0471**$^{\blacktriangledown}_{\vartriangle}$	**0.0356**$^{\blacktriangledown}_{\blacktriangle}$	**0.0837**$^{\blacktriangledown}_{\blacktriangle}$	**0.0678**$^{\blacktriangledown}_{\blacktriangle}$

Table 7.3. Dynamic ensemble performance values (P@10) using 1R methodology and user predictors (MovieLens dataset).

Table 7.2 shows the results obtained following the AR methodology. We may observe how, except in three cases, dynamic ensembles outperform the baseline. Interestingly, for HRU5, the best performing method is not the one obtained with the 'perfect correlation' approach, as we may expect, but with our dynamic ensembles based on the user clarity performance predictors. This is due to the fact that the corresponding predictor for the first recommender (P@10 values for kNN) also has a strong correlation with the performance of the second recommender (pLSA), and thus, it does not satisfy the requirement that the correlation values should not be too similar for both recommenders.

Table 7.3 shows the results obtained with the 1R methodology. Note that in this case the correlations were consistently lower than those obtained with the AR methodology. In particular, this is emphasised in the results of the dynamic ensemble HRU1, which do not outperform the baseline for almost any predictor. This can be explained with the results reported in Table 6.9, where the TFL1 recommender obtains a near-zero correlation, and thus, the correlation requirement of our framework is not satisfied. Specifically, this fact highlights the importance of the strength in the correlation between the predictor and the recommender performance, as stated in Section 7.2.1. Furthermore, we may observe in the table that for two combinations

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0294	0.0524	0.0381	0.0381	0.0381	0.0381
R2 (λ=0.0)	0.0223	0.0223	0.0223	0.0068	0.0718	0.0406
Baseline (λ=0.5)	0.0345	0.0440	0.0396	0.0283	0.0639	0.0493
Best static	0.0351	0.0536	0.0424	0.0384	0.0732	0.0493
(best λ)	(0.6)	(0.9)	(0.7)	(0.9)	(0.1)	(0.5)
Perfect correlation	<u>0.0389</u>	<u>0.0552</u>	<u>0.0493</u>	<u>0.0396</u>	<u>0.0742</u>	<u>0.0559</u>
PC-OM	0.0373	0.0485	0.0471	0.0332	0.0732	0.0548
PC-FW	0.0355	0.0459	0.0429	0.0307	0.0722	0.0535
Entropy-OM	0.0345$^{\blacktriangledown}$	**0.0518**$^{\blacktriangledown}_{\blacktriangle}$	**0.0404**$^{\blacktriangledown}_{\blacktriangle}$	**0.0337**$^{\blacktriangledown}_{\blacktriangle}$	0.0615$^{\blacktriangledown}_{\blacktriangledown}$	0.0471$^{\blacktriangledown}_{\blacktriangledown}$
ItemSimple-OM	0.0333$^{\blacktriangledown}_{\blacktriangledown}$	**0.0519**$^{\blacktriangledown}_{\blacktriangle}$	**0.0403**$^{\blacktriangledown}_{\blacktriangle}$	**0.0339**$^{\blacktriangledown}_{\blacktriangle}$	**0.0723**$^{\blacktriangledown}_{\blacktriangle}$	0.0444$^{\blacktriangledown}_{\blacktriangledown}$
ItemUser-OM	0.0334$^{\blacktriangledown}_{\blacktriangledown}$	**0.0517**$^{\blacktriangledown}_{\blacktriangle}$	**0.0403**$^{\blacktriangledown}_{\blacktriangle}$	**0.0336**$^{\blacktriangledown}_{\blacktriangle}$	**0.0726**$^{\blacktriangledown}_{\blacktriangle}$	0.0438$^{\blacktriangledown}_{\blacktriangledown}$
RatUser-OM	0.0335$^{\blacktriangledown}_{\blacktriangledown}$	**0.0521**$^{\blacktriangledown}_{\blacktriangle}$	**0.0410**$^{\blacktriangledown}_{\blacktriangle}$	**0.0341**$^{\blacktriangledown}_{\blacktriangle}$	**0.0728**$^{\blacktriangledown}_{\blacktriangle}$	0.0435$^{\blacktriangledown}_{\blacktriangledown}$
RatItem-OM	0.0334$^{\blacktriangledown}_{\blacktriangledown}$	**0.0516**$^{\blacktriangledown}_{\blacktriangle}$	**0.0406**$^{\blacktriangledown}_{\blacktriangle}$	**0.0341**$^{\blacktriangledown}_{\blacktriangle}$	**0.0726**$^{\blacktriangledown}_{\blacktriangle}$	0.0434$^{\blacktriangledown}_{\blacktriangledown}$
IRUser-OM	0.0333$^{\blacktriangledown}_{\blacktriangledown}$	**0.0511**$^{\blacktriangledown}_{\blacktriangle}$	**0.0401**$^{\blacktriangledown}_{\blacktriangle}$	**0.0336**$^{\blacktriangledown}_{\blacktriangle}$	**0.0718**$^{\blacktriangledown}_{\blacktriangle}$	0.0440$^{\blacktriangledown}_{\blacktriangledown}$
IRItem-OM	0.0326$^{\blacktriangledown}_{\blacktriangledown}$	**0.0504**$^{\blacktriangledown}_{\blacktriangle}$	0.0388$^{\blacktriangledown}_{\blacktriangledown}$	**0.0325**$^{\blacktriangledown}_{\blacktriangle}$	**0.0714**$^{\blacktriangledown}_{\blacktriangle}$	0.0430$^{\blacktriangledown}_{\blacktriangledown}$
IRUserItem-OM	0.0334$^{\blacktriangledown}_{\blacktriangledown}$	**0.0511**$^{\blacktriangledown}_{\blacktriangle}$	**0.0401**$^{\blacktriangledown}_{\blacktriangle}$	**0.0336**$^{\blacktriangledown}_{\blacktriangle}$	**0.0719**$^{\blacktriangledown}_{\blacktriangle}$	0.0437$^{\blacktriangledown}_{\blacktriangledown}$
Entropy-FW	**0.0347**$^{\blacktriangledown}_{\blacktriangle}$	**0.0472**$^{\blacktriangledown}_{\blacktriangle}$	**0.0402**$^{\blacktriangledown}_{\blacktriangle}$	**0.0308**$^{\blacktriangledown}_{\blacktriangle}$	0.0636$^{\blacktriangledown}_{\blacktriangledown}$	0.0486$^{\blacktriangledown}_{\blacktriangledown}$
ItemSimple-FW	0.0342$^{\blacktriangledown}_{\blacktriangledown}$	**0.0473**$^{\blacktriangledown}_{\blacktriangle}$	**0.0402**$^{\blacktriangledown}_{\blacktriangle}$	**0.0309**$^{\blacktriangledown}_{\blacktriangle}$	**0.0720**$^{\blacktriangledown}_{\blacktriangle}$	0.0467$^{\blacktriangledown}_{\blacktriangledown}$
ItemUser-FW	0.0342$^{\blacktriangledown}_{\blacktriangledown}$	**0.0471**$^{\blacktriangledown}_{\blacktriangle}$	**0.0401**$^{\blacktriangledown}_{\blacktriangle}$	**0.0308**$^{\blacktriangledown}_{\blacktriangle}$	**0.0724**$^{\blacktriangledown}_{\blacktriangle}$	0.0467$^{\blacktriangledown}_{\blacktriangledown}$
RatUser-FW	0.0343$^{\blacktriangledown}_{\triangledown}$	**0.0474**$^{\blacktriangledown}_{\blacktriangle}$	**0.0405**$^{\blacktriangledown}_{\blacktriangle}$	**0.0310**$^{\blacktriangledown}_{\blacktriangle}$	**0.0727**$^{\blacktriangledown}_{\blacktriangle}$	0.0469$^{\blacktriangledown}_{\blacktriangledown}$
RatItem-FW	0.0342$^{\blacktriangledown}_{\blacktriangledown}$	**0.0472**$^{\blacktriangledown}_{\blacktriangle}$	**0.0403**$^{\blacktriangledown}_{\blacktriangle}$	**0.0309**$^{\blacktriangledown}_{\blacktriangle}$	**0.0725**$^{\blacktriangledown}_{\blacktriangle}$	0.0469$^{\blacktriangledown}_{\blacktriangledown}$
IRUser-FW	0.0341$^{\blacktriangledown}_{\blacktriangledown}$	**0.0470**$^{\blacktriangledown}_{\blacktriangle}$	**0.0401**$^{\blacktriangledown}_{\blacktriangle}$	**0.0308**$^{\blacktriangledown}_{\blacktriangle}$	**0.0714**$^{\blacktriangledown}_{\blacktriangle}$	0.0469$^{\blacktriangledown}_{\blacktriangledown}$
IRItem-FW	0.0338$^{\blacktriangledown}_{\blacktriangledown}$	**0.0467**$^{\blacktriangledown}_{\blacktriangle}$	0.0393$^{\blacktriangledown}_{\triangledown}$	**0.0302**$^{\blacktriangledown}_{\blacktriangle}$	**0.0712**$^{\blacktriangledown}_{\blacktriangle}$	0.0464$^{\blacktriangledown}_{\blacktriangledown}$
IRUserItem-FW	0.0341$^{\blacktriangledown}_{\blacktriangledown}$	**0.0471**$^{\blacktriangledown}_{\blacktriangle}$	**0.0401**$^{\blacktriangledown}_{\blacktriangle}$	**0.0308**$^{\blacktriangledown}_{\blacktriangle}$	**0.0716**$^{\blacktriangledown}_{\blacktriangle}$	0.0469$^{\blacktriangledown}_{\blacktriangledown}$

Table 7.4. Dynamic ensemble performance values (P@10) using the <u>U1R methodology</u> and user predictors (MovieLens dataset)

(HRU2 and HRU5) the best performance results are not obtained by dynamic approaches, but by the best static approaches in contrast with what we found for the AR methodology. This situation is different to the one obtained when we evaluate using MAP@10 (see Appendix A.4.1), where the best results are always obtained by dynamic ensembles.

Table 7.4 and Table 7.5 show the performance values obtained with the unbiased methodologies proposed in Chapter 4, that is, U1R and P1R. Following the U1R methodology (Table 7.4) we obtain similar results to those obtained in the 1R methodology except for HRU6. In contrast, with the P1R methodology (Table 7.5) our framework does not show improvements over any baseline. We may see that the 'perfect correlation' methods are able to obtain better, although very close, values than those of the best static ensemble. This means that there is room for improvement in this methodology, and that the performance of the dynamic recommender ensembles could be improved if better performance predictors were found.

	HRU1	HRU2	HRU3	HRU4	HRU5	HRU6
R1 (λ=1.0)	0.0203	0.0348	0.0265	0.0265	0.0265	0.0265
R2 (λ=0.0)	0.0197	0.0197	0.0197	0.0208	0.0604	0.0282
Baseline (λ=0.5)	0.0470	0.0579	0.0539	0.0269	0.0763	0.0560
Best static	0.0470	0.0593	0.0541	0.0278	0.0796	0.0560
(best λ)	(0.5)	(0.6)	(0.6)	(0.7)	(0.4)	(0.5)
Perfect correlation	0.0464	0.0579	0.0546	0.0314	0.0767	0.0564
PC-OM	0.0425	0.0554	0.0528	0.0296	0.0746	0.0537
PC-FW	0.0429	0.0542	0.0504	0.0282	0.0764	0.0522
Entropy-OM	0.0431▼	0.0564▼	0.0502▼	0.0261▼	0.0698▼	0.0521▼
ItemSimple-OM	0.0358▼	0.0509▼	0.0429▼	0.0261▼	0.0689▼	0.0441▼
ItemUser-OM	0.0361▼	0.0512▼	0.0431▼	0.0261▼	0.0675▼	0.0444▼
RatUser-OM	0.0362▼	0.0514▼	0.0436▼	0.0263▼	0.0663▼	0.0446▼
RatItem-OM	0.0361▼	0.0511▼	0.0432▼	0.0262▼	0.0661▼	0.0444▼
IRUser-OM	0.0365▼	0.0513▼	0.0435▼	0.0263▼	0.0687▼	0.0447▼
IRItem-OM	0.0357▼	0.0504▼	0.0421▼	0.0257▼	0.0669▼	0.0439▼
IRUserItem-OM	0.0365▼	0.0513▼	0.0434▼	0.0263▼	0.0675▼	0.0447▼
Entropy-FW	0.0457▼	0.0577▼	0.0524▼	0.0265▼	0.0745▼	0.0546▼
ItemSimple-FW	0.0410▼	0.0540▼	0.0475▼	0.0266▼	0.0720▼	0.0498▼
ItemUser-FW	0.0409▼	0.0538▼	0.0473▼	0.0265▼	0.0706▼	0.0497▼
RatUser-FW	0.0410▼	0.0540▼	0.0477▼	0.0267▼	0.0691▼	0.0499▼
RatItem-FW	0.0411▼	0.0541▼	0.0476▼	0.0266▼	0.0688▼	0.0499▼
IRUser-FW	0.0410▼	0.0538▼	0.0474▼	0.0266▼	0.0721▼	0.0496▼
IRItem-FW	0.0406▼	0.0534▼	0.0467▼	0.0263▼	0.0699▼	0.0491▼
IRUserItem-FW	0.0409▼	0.0538▼	0.0474▼	0.0266▼	0.0706▼	0.0496▼

Table 7.5. Dynamic ensemble performance values (P@10) using the P1R methodology and user predictors (MovieLens dataset).

In summary, the **results show that our methods significantly outperform static ensembles for different recommender combinations in most of the evaluation methodologies**. Moreover, in most cases our methods also achieve the best results for each ensemble, let aside the performance of the oracle performance prediction (perfect correlation) and best static approaches, which use groundtruth (test) information, differently to the clarity- and entropy-based performance predictors.

Nevertheless, we observe that in those cases where the dynamic ensembles do not perform better than the static ensembles, the best static approaches use values of λ close to 0.5. We hypothesise that our framework may be biased towards favouring those ensembles whose recommender combination is highly unbalanced. Interestingly, although the predictors only weight one of the recommenders (not always the better performing one) a dynamic ensemble is usually able to find the optimal combination in the unbalanced cases. In particular, this could help to answer why our dynamic ensembles underperform static approaches for the U1R and P1R methodologies, since the best static in these cases seem to be often very close to 0.5.

	R1	R2
HRI1	pLSA	CB
HRI2	pLSA	kNN
HRI3	ItemPop	CB
HRI4	ItemPop	kNN

Table 7.6. Selected recommenders for building dynamic ensembles using item predictors that exploit rating data (MovieLens dataset).

Using item-based predictors

As we noted in Section 6.5.2, item-based predictors could also be valuable since they also obtain high correlations with respect to item perfomance. Table 7.6 shows the selected recommenders that satisfy the correlation requirements with item predictors. Table 7.7, Table 7.8, and Table 7.9 show the results obtained when these recommender combinations are evaluated and compared against dynamic versions (using our proposed item predictors), and using the 1R, U1R, and uuU1R methodologies. In this case, ensemble predictions are computed by means of Equation (7.3) with values $\gamma(u, i)$ only depending on the current item, that is, $\gamma(i)$.

When measuring the performance of dynamic ensembles that use item-based performance predictors, we do not compute the perfect correlation predictors because we do not have a standard metric for item performance. Apart from that, the

	HRI1	HRI2	HRI3	HRI4
R1 (λ=1.0)	0.0836	0.0836	0.0649	0.0649
R2 (λ=0.0)	0.0221	0.0437	0.0221	0.0437
Baseline (λ=0.5)	0.0909	0.0924	0.0886	0.0907
Best static (best λ)	0.0909 (0.5)	0.0924 (0.5)	0.0886 (0.5)	0.0907 (0.5)
Entropy-OM	0.0708▼▼	0.0858▼▼	0.0684▼▼	0.0831▼▼
UserSimple-OM	0.0761▼▼	0.0905▼▼	0.0723▼▼	0.0837▼▼
UserItem-OM	0.0776▼▼	0.0903▼▼	0.0749▼▼	0.0843▼▼
RatItem-OM	0.0751▼▼	0.0893▼▼	0.0712▼▼	0.0824▼▼
RatUser-OM	0.0759▼▼	0.0892▼▼	0.0674▼▼	0.0789▼▼
URItem-OM	0.0776▼▼	0.0911▼▼	0.0797▼▼	0.0885▼▼
URUser-OM	0.0781▼▼	0.0906▼▼	0.0721▼▼	0.0820▼▼
URItemUser-OM	0.0777▼▼	0.0909▼▼	0.0777▼▼	0.0869▼▼
Entropy-FW	0.0798▼▼	0.0923▼▼	0.0771▼▼	0.0895▼▼
UserSimple-FW	**0.0946**▲▲	**0.0979**▲▲	**0.0916**▲▲	**0.0949**▲▲
UserItem-FW	**0.0949**▲▲	**0.0980**▲▲	**0.0920**▲▲	**0.0950**▲▲
RatItem-FW	**0.0944**▲▲	**0.0979**▲▲	**0.0913**▲▲	**0.0948**▲▲
RatUser-FW	**0.0946**▲▲	**0.0978**▲▲	**0.0908**▲▲	**0.0942**▲▲
URItem-FW	**0.0940**▲▲	**0.0981**▲▲	**0.0923**▲▲	**0.0958**▲▲
URUser-FW	**0.0946**▲▲	**0.0978**▲▲	**0.0912**▲▲	**0.0945**▲▲
URItemUser-FW	**0.0944**▲▲	**0.0980**▲▲	**0.0921**▲▲	**0.0954**▲▲

Table 7.7. Dynamic ensemble performance values (P@10) using 1R methodology with item predictors (MovieLens dataset).

rest of the experimental settings is the same as those described above for dynamic hybrids with user-based performance predictors.

Table 7.7 shows the results obtained by using item-based predictors and the 1R methodology. We may observe that if the predictors are weighted using the FW strategy, dynamic ensembles outperform static combinations in every situation, except for the Entropy predictor. It is interesting to note that, differently to user-based predictors, the dynamic ensembles are able to outperform the best static ensemble even when they are close to the baseline with $\lambda = 0.5$. The reader may compare Table 7.4 and Table 7.7 to observe these differences.

In Table 7.8, where the methodology U1R is used, a very similar situation occurs, although not all dynamic ensembles outperform the static approach with the FW strategy. Specifically, the dynamic hybrid weighted by the URItem clarity predictor clearly obtains better performance than the rest of the dynamic and static ensembles, in particular the HRI3 and HRI4 combinations.

Finally, the performance results found for the uuU1R methodology are presented in Table 7.9, in which the test ratings – i.e., the users – are uniformly distributed over the items, items previously uniformly distributed in the test (like in the U1R methodology). In this experiment, the performance of the dynamic ensemble is much better than in the previous experiments, since **all the rating-based item predictors (except for the Entropy predictor) outperform the static baseline no matter the weighting strategy in three out of four recommender combinations**.

	HRI1	HRI2	HRI3	HRI4
R1 (λ=1.0)	0.0718	0.0718	0.0406	0.0406
R2 (λ=0.0)	0.0223	0.0381	0.0223	0.0381
Baseline (λ=0.5)	0.0764	0.0812	0.0630	0.0689
Best static (best λ)	0.0764 (0.5)	0.0812 (0.5)	0.0630 (0.5)	0.0689 (0.5)
Entropy-OM	0.0571▼▼	0.0652▼▼	0.0435▼▼	0.0508▼▼
UserSimple-OM	0.0657▼▼	0.0716▼▼	0.0399▼▼	0.0450▼▼
UserItem-OM	0.0671▼▼	0.0721▼▼	0.0425▼▼	0.0462▼▼
RatItem-OM	0.0645▼▼	0.0699▼▼	0.0392▼▼	0.0435▼▼
RatUser-OM	0.0620▼▼	0.0671▼▼	0.0335▼▼	0.0382▼▼
URItem-OM	0.0705▼▼	0.0757▼▼	0.0496▼▼	0.0532▼▼
URUser-OM	0.0650▼▼	0.0699▼▼	0.0372▼▼	0.0414▼▼
URItemUser-OM	0.0690▼▼	0.0741▼▼	0.0462▼▼	0.0500▼▼
Entropy-FW	0.0668▼▼	0.0757▼▼	0.0518▼▼	0.0595▼▼
UserSimple-FW	**0.0840**▲▲	**0.0886**▲▲	0.0601▼▼	0.0658▼▼
UserItem-FW	**0.0844**▲▲	**0.0887**▲▲	0.0609▼▼	0.0663▼▼
RatItem-FW	**0.0839**▲▲	**0.0883**▲▲	0.0598▼▼	0.0653▼▼
RatUser-FW	**0.0831**▲▲	**0.0876**▲▲	0.0573▼▼	0.0630▼▼
URItem-FW	**0.0851**▲▲	**0.0897**▲▲	**0.0642**▲▲	**0.0698**▲▲
URUser-FW	**0.0836**▲▲	**0.0881**▲▲	0.0585▼▼	0.0642▼▼
URItemUser-FW	**0.0848**▲▲	**0.0893**▲▲	0.0625▼▼	0.0680▼▼

Table 7.8. Dynamic ensemble performance values (P@10) using U1R methodology with item predictors (MovieLens dataset).

	HRI1	HRI2	HRI3	HRI4
R1 (λ=1.0)	0.0536	0.0536	0.0225	0.0225
R2 (λ=0.0)	0.0198	0.0275	0.0198	0.0275
Baseline (λ=0.5)	0.0374	0.0440	0.0239	0.0256
Best static	0.0491	0.0502	0.0239	0.0271
(best λ)	(0.9)	(0.9)	(0.6)	(0.2)
Entropy-OM	0.0324▼▼	0.0385▼▼	0.0236▽▽	**0.0280**▲▲
UserSimple-OM	**0.0510**△△	**0.0548**▲▲	0.0237▽▽	**0.0282**▲▲
UserItem-OM	**0.0514**▲▲	**0.0547**▲▲	0.0236▽▽	**0.0280**▲▲
RatItem-OM	**0.0516**▲▲	**0.0547**▲▲	0.0237▽▽	**0.0281**▲▲
RatUser-OM	**0.0523**▲▲	**0.0551**▲▲	0.0237▽▽	**0.0282**▲▲
URItem-OM	**0.0498**▲▲	**0.0536**▲▲	0.0234▼▼	**0.0280**▲▲
URUser-OM	**0.0518**▲▲	**0.0551**▲▲	0.0234▼▼	**0.0279**▲▲
URItemUser-OM	**0.0505**▲▲	**0.0542**▲▲	0.0235▽▼	**0.0280**▲▲
Entropy-FW	0.0344▼▼	0.0410▼▼	**0.0241**△△	**0.0275**▲▲
UserSimple-FW	**0.0435**△△	**0.0503**△▲	**0.0244**▲▲	**0.0276**▲▲
UserItem-FW	**0.0435**▼▲	**0.0501**△▲	**0.0245**▲▲	**0.0275**▲▲
RatItem-FW	**0.0436**▼▲	**0.0504**△▲	**0.0244**▲▲	**0.0275**▲▲
RatUser-FW	**0.0440**▼▲	**0.0509**▲▲	**0.0245**▲▲	**0.0276**▲▲
URItem-FW	**0.0429**▼▲	**0.0494**▲▲	**0.0244**▲▲	**0.0273**△▲
URUser-FW	**0.0438**▼▲	**0.0506**△▲	**0.0245**▲▲	**0.0274**△▲
URItemUser-FW	**0.0432**▼▲	**0.0498**△▲	**0.0245**▲▲	**0.0274**△▲

Table 7.9. Dynamic ensemble performance values (P@10) using uuU1R methodology with item predictors (MovieLens dataset).

In the other combination (HRI3) the best strategy is FW, the same as with the other evaluation methodologies.

7.3.2 Dynamic recommender ensembles on log data

In this section we present experiments in which log-based predictors are used to dynamically weight an ensemble's recommenders. As with rating-based information, in this case we first have to select suitable recommenders to combine according to the requirements established in our framework. Hence, we choose the combinations HL1, HL2 and HL3 presented in Table 7.10, where, as before, the performance predictors weight the recommender denoted as R1.

The Last.fm dataset contains timestamped log-based information. As noted in Chapter 4, for efficiency reasons, we only use the 1R methodology in this dataset. Table 7.11 shows the results obtained with a temporal split of the data, and Table 7.12 shows the results obtained with a random split (five-fold) of the data.

	R1	R2
HL1	kNN	CB
HL2	kNN	ItemPop
HL3	pLSA	kNN

Table 7.10. Selected recommenders for building dynamic ensembles using performance predictors that exploit log-based information (Last.fm dataset).

	HL1	HL2	HL3
R1 (λ=1.0)	0.0603	0.0603	$\underline{0.0926}$
R2 (λ=0.0)	$\underline{0.0916}$	0.0797	0.0603
Baseline (λ=0.5)	0.0852	0.0755	0.0820
Best static	0.0914	$\underline{0.0812}$	0.0925
(best λ)	(0.2)	(0.1)	(0.9)
Perfect correlation	0.0890	0.0783	0.0863
PC-OM	0.0869	0.0771	0.0851
PC-FW	0.0849	0.0751	0.0826
ItemSimple-OM	**0.0904**$^{\blacktriangledown}_{\blacktriangle}$	**0.0804**$^{\blacktriangledown}_{\blacktriangle}$	**0.0901**$^{\blacktriangledown}_{\blacktriangle}$
Autocorrelation-OM	0.0815$^{\blacktriangledown}_{\blacktriangledown}$	0.0722$^{\blacktriangledown}_{\blacktriangledown}$	0.0781$^{\blacktriangledown}_{\blacktriangledown}$
TimeSimple-OM	**0.0905**$^{\blacktriangledown}_{\blacktriangle}$	**0.0789**$^{\blacktriangledown}_{\blacktriangle}$	**0.0898**$^{\blacktriangledown}_{\blacktriangle}$
ItemTime-OM	**0.0906**$^{\blacktriangledown}_{\blacktriangle}$	**0.0804**$^{\blacktriangledown}_{\blacktriangle}$	**0.0902**$^{\blacktriangledown}_{\blacktriangle}$
ItemPriorTime-OM	**0.0885**$^{\blacktriangledown}_{\blacktriangle}$	**0.0778**$^{\blacktriangledown}_{\blacktriangle}$	**0.0863**$^{\blacktriangledown}_{\blacktriangle}$
ItemSimple-FW	**0.0903**$^{\blacktriangledown}_{\blacktriangle}$	**0.0802**$^{\blacktriangledown}_{\blacktriangle}$	**0.0891**$^{\blacktriangledown}_{\blacktriangle}$
Autocorrelation-FW	0.0842$^{\blacktriangledown}_{\blacktriangle}$	0.0746$^{\blacktriangledown}_{\blacktriangledown}$	0.0809$^{\blacktriangledown}_{\blacktriangledown}$
TimeSimple-FW	**0.0901**$^{\blacktriangledown}_{\blacktriangle}$	**0.0785**$^{\blacktriangledown}_{\blacktriangle}$	**0.0884**$^{\blacktriangledown}_{\blacktriangle}$
ItemTime-FW	**0.0904**$^{\blacktriangledown}_{\blacktriangle}$	**0.0800**$^{\blacktriangledown}_{\blacktriangle}$	**0.0891**$^{\blacktriangledown}_{\blacktriangle}$
ItemPriorTime-FW	**0.0883**$^{\blacktriangledown}_{\blacktriangle}$	**0.0775**$^{\blacktriangledown}_{\blacktriangle}$	**0.0855**$^{\blacktriangledown}_{\blacktriangle}$

Table 7.11. Dynamic ensemble performance values (P@10) using the 1R methodology with the log-based user predictors (Last.fm, temporal split).

	HL1	HL2	HL3
R1 (λ=1.0)	0.0204	0.0204	0.0836
R2 (λ=0.0)	$\underline{0.0828}$	$\underline{0.0767}$	0.0204
Baseline (λ=0.5)	0.0764	0.0643	0.0704
Best static	0.0818	$\underline{0.0767}$	$\underline{0.0837}$
(best λ)	(0.2)	(0.1)	(0.9)
Perfect correlation	0.0818	0.0760	0.0829
PC-OM	0.0816	0.0755	0.0823
PC-FW	0.0815	0.0745	0.0811
ItemSimple-OM	**0.0799**$^{\blacktriangledown}_{\blacktriangle}$	**0.0730**$^{\blacktriangledown}_{\blacktriangle}$	**0.0771**$^{\blacktriangledown}_{\blacktriangle}$
Autocorrelation-OM	0.0717$^{\blacktriangledown}_{\blacktriangledown}$	0.0596$^{\blacktriangledown}_{\blacktriangledown}$	0.0686$^{\blacktriangledown}_{\blacktriangledown}$
TimeSimple-OM	**0.0814**$^{\blacktriangledown}_{\blacktriangle}$	**0.0762**$^{\blacktriangledown}_{\blacktriangle}$	0.0518$^{\blacktriangledown}_{\blacktriangledown}$
ItemTime-OM	**0.0806**$^{\blacktriangledown}_{\blacktriangle}$	**0.0734**$^{\blacktriangledown}_{\blacktriangle}$	**0.0761**$^{\blacktriangledown}_{\blacktriangle}$
ItemPriorTime-OM	**0.0770**$^{\blacktriangledown}_{\blacktriangle}$	**0.0658**$^{\blacktriangledown}_{\blacktriangle}$	**0.0743**$^{\blacktriangledown}_{\blacktriangle}$
ItemSimple-FW	**0.0804**$^{\blacktriangledown}_{\blacktriangle}$	**0.0726**$^{\blacktriangledown}_{\blacktriangle}$	**0.0739**$^{\blacktriangledown}_{\blacktriangle}$
Autocorrelation-FW	0.0756$^{\blacktriangledown}_{\blacktriangledown}$	0.0631$^{\blacktriangledown}_{\blacktriangledown}$	0.0697$^{\blacktriangledown}_{\blacktriangledown}$
TimeSimple-FW	**0.0814**$^{\blacktriangledown}_{\blacktriangle}$	**0.0753**$^{\blacktriangledown}_{\blacktriangle}$	0.0579$^{\blacktriangledown}_{\blacktriangledown}$
ItemTime-FW	**0.0808**$^{\blacktriangledown}_{\blacktriangle}$	**0.0728**$^{\blacktriangledown}_{\blacktriangle}$	**0.0732**$^{\blacktriangledown}_{\blacktriangle}$
ItemPriorTime-FW	**0.0783**$^{\blacktriangledown}_{\blacktriangle}$	**0.0671**$^{\blacktriangledown}_{\blacktriangle}$	**0.0719**$^{\blacktriangledown}_{\blacktriangle}$

Table 7.12. Dynamic ensemble performance values (P@10) using the 1R methodology with log-based user predictors (Last.fm, five-fold random split).

We can see that the results of both tables are analogous. **The dynamic ensembles weighted by the log-based performance predictors outperform the baseline static ensemble in all cases, except with the Autocorrelation predictor**. This result is consistent with the correlations presented in Table 6.14 and Table 6.15, where autocorrelation obtained the lowest (absolute) correlation value for the kNN recommender on both versions of the dataset. Regarding the pLSA recommender (in the combination HL3), the Autocorrelation and TimeSimple predictors obtain com-

	R1	R2
HS1	Personal	pLSA
HS2	Personal	kNN
HS3	PureSocial	pLSA
HS4	PureSocial	kNN

Table 7.13. Selected recommenders for building dynamic ensembles using social-based user predictors (CAMRa dataset).

parable correlations with the combined recommenders, yet the performance of the corresponding dynamic ensembles is very different, thus suggesting that, although we have found a dependence between the predictors' power in terms of correlation, and their effectiveness in weighting hybrids, this is not a strict necessary condition to obtain improvements over the static ensembles.

The best performance values were achieved either by single recommenders or by the best static ensembles. When the best results are obtained by single recommenders emphasises the fact that no hybridisation is required for that combination (like in HL1 and HL3 for the temporal split, and HL1 and HL2 for the random split). In the other case, when the best results are achieved by the best static ensembles, it may restrict the usefulness of our approach, although our proposed dynamic ensembles significantly outperform the baseline static ensembles for some predictors such as TimeSimple and ItemSimple. We have to recall that the best static ensembles are in fact optimised using the test set, which is clearly not a fair comparison. The results of the perfect correlation ensembles in the random split are always better than those obtained by the performance predictors, confirming that predictors with stronger correlations should obtain better performance results when used for dynamic ensembles.

7.3.3 Dynamic recommender ensembles on social data

In the third experiment we exploit the social information available in the CAMRa dataset to combine collaborative and social filtering recommenders using social-based performance predictors. Table 7.13 shows the recommender combinations selected based on the correlations obtained in Section 6.5.4. Here, we present 4 ensembles where the two social filtering recommenders, Personal and PureSocial, are combined with two collaborative filtering recommenders, pLSA and kNN. We saw in Section 6.5.4 that most of the social-based predictors obtained higher correlations with the social filtering recommenders, and lower or negligible correlations with the collaborative filtering recommenders, at least for the social version of the dataset (Table 6.16). The situation for the collaborative-social version was not so clear, but for the sake of coherence, we use the same set of ensembles in both versions of the dataset.

	HS1	HS2	HS3	HS4
R1 (λ=1.0)	0.1732	0.1732	0.1760	0.1760
R2 (λ=0.0)	0.1110	0.0473	0.1110	0.0473
Baseline (λ=0.5)	0.1813	0.1821	0.2006	0.1929
Best static (best λ)	0.1842 (0.7)	0.1899 (0.8)	0.2012 (0.4)	0.1952 (0.6)
Perfect correlation	0.2018	0.1929	0.2089	0.1979
PC-OM	0.1872	0.1875	0.2048	0.1946
PC-FW	0.1863	0.1869	0.2042	0.1994
AvgNeighDeg-OM	0.1795▽▽	**0.1896**▽△	0.1973▽▽	0.1804▼▼
BetCentrality-OM	0.1744▼▽	0.1804▼▽	0.1833▼▼	0.1777▼▼
ClustCoeff-OM	0.1786▽▽	0.1786▼▽	0.1836▼▼	0.1753▼▼
Degree-OM	0.1738▼▽	**0.1839**▽△	0.1976▽▽	0.1765▼▼
EgoCompSize-OM	0.1756▼▽	**0.1833**▽△	0.1967▽▽	0.1827▼▼
HITS-OM	0.1774▽▽	**0.1911**△▲	0.1813▼▼	0.1798▼▼
PageRank-OM	0.1762▼▽	**0.1842**▽△	0.1917▼▽	0.1801▼▼
TwoHopNeigh-OM	0.1756▼▽	**0.1851**▽△	0.1964▽▽	0.1777▼▼
AvgNeighDeg-FW	0.1807▽▽	**0.1896**▽▲	0.2003▽▽	0.1914▽▽
BetCentrality-FW	0.1801▽▽	**0.1872**▽△	**0.2024**△△	**0.1929**▽△
ClustCoeff-FW	0.1804▽▽	**0.1875**▽▲	0.2003▽▽	0.1890▼▽
Degree-FW	0.1798▽▽	**0.1887**▽▲	0.2000▽▽	**0.1929**▽△
EgoCompSize-FW	0.1789▼▽	**0.1896**▽▲	**0.2009**▽△	**0.1938**▽△
HITS-FW	0.1801▽▽	**0.1902**△▲	0.1997▽▽	0.1926▽▽
PageRank-FW	0.1810▽▽	**0.1875**▽△	0.2003▽▽	0.1923▽▽
TwoHopNeigh-FW	0.1801▽▽	**0.1905**△▲	0.2000▽▽	0.1926▽▽

Table 7.14. Dynamic ensemble performance values (P@10) using the AR methodology with social-based user predictors (CAMRa, social dataset).

As we mentioned in Section 6.5.4, due to the lack of coverage of the social filtering recommenders, the only methodology that provides sensible results is the AR methodology. In this section we present the results obtained using this methodology on the two available versions of the CAMRa dataset: social and collaborative-social.

Table 7.14 shows the results obtained on the social version of the CAMRa dataset. We see that only for one out of the four recommender combinations, the dynamic ensembles consistently outperform the baseline static ensemble. However, it is interesting to note that the best value is always achieved by the perfect correlation ensemble, which means that further improvements could be possible if we were able to find predictors with stronger correlations.

In the collaborative-social version of the dataset (Table 7.15) the results are similar, except that now for HS2, the best result is obtained by the best static ensemble. Moreover, a larger number of dynamic ensembles outperform the baseline static ensemble HS3, whereas at least one dynamic ensemble outperforms the baseline HS1, which is a better result than the one shown in the previous Table 7.14. We hypothesise this is because on this version of the dataset the individual recommenders display a more similar performance to each other (compare the differences between R1 and R2 in Table 7.14 and Table 7.15).

Furthermore, some of the correlations obtained for the CAMRa collaborative dataset are more discriminative between the combined recommenders, in the sense that, for instance, the correlations between the two-hop neighbourhood predictor and the Personal recommender were -0.123 and -0.121 in the social and collaborative-social datasets, respectively. However, the correlations between the two-hop neighbourhood predictor and kNN were 0.004 and 0.130, that is, in the second dataset the relative distance in correlation between these two recommenders is larger, according to the correlation with respect to the predictor. This change in the correlations may explain the fact that in Table 7.15 some of the dynamic ensembles outperform the perfect correlation ensemble, which does not take the relative correlation into account with respect to each individual recommender, as noted in 7.3.1.

In general, **the HITS predictor obtains the best results among the dynamic ensembles for some of the tested combinations. Other predictors such as the betweenness centrality and the ego components size produce more competitive ensembles in the social version of the dataset**, whereas the degree and the average neighbour degree preditors provide better results for more than one combination in the CAMRa collaborative dataset.

	HS1	HS2	HS3	HS4
R1 (λ=1.0)	0.1066	0.1066	0.1072	0.1072
R2 (λ=0.0)	0.1007	0.0226	0.1007	0.0226
Baseline (λ=0.5)	0.1509	0.1142	0.1599	0.1219
Best static	0.1524	0.1200	0.1632	0.1219
(best λ)	(0.4)	(0.7)	(0.3)	(0.5)
Perfect correlation	0.1608	0.1188	0.1640	0.1237
PC-OM	0.1202	0.1164	0.1254	0.1199
PC-FW	0.1189	0.1143	0.1263	0.1219
AvgNeighDeg-OM	0.1489▽▽	**0.1195**▽△	0.1599▽▽	0.1131▼▼
BetCentrality-OM	0.1443▼▼	0.1132▼▽	0.1487▼▼	0.1114▼▼
ClustCoeff-OM	0.1465▼▽	0.1123▼▽	0.1483▼▼	0.1108▼▼
Degree-OM	0.1472▽▽	**0.1154**▽△	**0.1614**▽△	0.1107▼▼
EgoCompSize-OM	0.1461▽▽	**0.1158**▽△	0.1596▽▽	0.1140▼▼
HITS-OM	0.1485▽▽	**0.1200**▲	0.1467▼▼	0.1134▼▼
PageRank-OM	0.1471▼▽	**0.1167**▽△	0.1579▽▽	0.1123▼▼
TwoHopNeigh-OM	0.1478▽▽	**0.1171**▽△	0.1585▼▽	0.1118▼▼
AvgNeighDeg-FW	**0.1518**▽△	**0.1191**▽▲	**0.1623**▽△	0.1204▼▼
BetCentrality-FW	0.1491▼▽	**0.1180**▽▲	0.1577▼▽	0.1213▼▼
ClustCoeff-FW	0.1500▽▽	**0.1182**▽▲	0.1566▼▼	0.1189▼▼
Degree-FW	0.1489▽▽	**0.1191**▽▲	**0.1627**▽△	0.1208▼▼
EgoCompSize-FW	0.1489▽▽	**0.1193**▽▲	**0.1618**▽△	0.1210▼▼
HITS-FW	0.1482▼▼	**0.1195**▽▲	0.1564▼▼	0.1202▼▼
PageRank-FW	0.1491▼▽	**0.1186**▽▲	**0.1610**▽△	0.1211▼▼
TwoHopNeigh-FW	0.1500▽▽	**0.1195**▽▲	**0.1619**▽△	0.1211▼▼

Table 7.15. Dynamic ensemble performance values (P@10) using the AR methodology with social-based user predictors (CAMRa, collaborative dataset).

7.3.4 Discussion

The analysis of the results presented in this chapter shows that ensembles can indeed benefit from a dynamic weighting of their recommenders. In particular, we have seen that when these weights come from performance predictors, which previously had shown significant correlation with the performance of individual recommenders, the resulting dynamic ensemble tends to outperform static combinations of the recommenders. In this context, in order to obtain successful hybridisations, we have to take several variables into account, which correspond to three stages proposed in our framework: the correlation between the predictor and the combined recommenders, the relative performance of such recommenders, the strategy to normalise the predictor's values, and the weight distribution among recommenders.

The relative performance of the recommenders has proven to be decisive, since in some cases, hybridisation does not make sense to begin with, when the difference in performance between the recommenders is significant and systematic, and thus, dynamic ensembles cannot obtain the best performance result, although they may outperform static ensembles. Performance prediction normalisation and weight distribution, on the other hand, do make a difference in the results. Although no explicit results are presented in this work regarding different normalisation approaches, previously conducted experiments showed us that score normalisation produce worse results than rank normalisation. Finally, the weight distribution strategy is not as critical as other stages of our framework, but helps to obtain much better results, specifically, when the one minus strategy (OM) is used.

The obtained results have also shown that more complex formalisations and probability models do not necessarily lead to better results, with respect to the adaptation and definition of the user and item clarity performance predictors. In this adaptation, various configurations were available, and we experimented with further extensions of different language models for the same clarity model, using rating and log-based information. Additionally, several graph-based metrics were tested, where the concept of the user's strength in a social network is modelled in different ways.

We find that different formulations for the user-based performance clarity predictor consistently obtain the best results in different situations for rating-based preference information. We also experimented with item-based predictors, and found that the UserItem, URItem, and RatUser predictors were noticeably better than the rest of the formulations. When log-based information is exploited, the ItemTime and TimeSimple predictors obtained better results than other predictors not based on the clarity concept, such as the Autocorrelation function. Moreover, regarding the social-based ensembles, the HITS, two-hop neighbourhood, and average neighbour degree approaches clearly outperform the ensemble weighted by the rest of the predictors and, in most of the cases, also outperform the baseline static ensemble.

These results are, in general, consistent with the correlation values between the predictors' output values and the recommenders' performance values. Figure 7.2 shows a summary of the results presented in this and previous chapters, where the difference in correlation is plotted against the gain (or loss) in performance with respect to the baseline. For this figure, the best and worst dynamic ensembles were selected from Table 7.2, Table 7.11 and Table 7.15. In the figure we may observe the trend that the larger the difference in correlation, the better the improvement over the baseline, which is in concordance with the requirement that both correlations should not be very similar. These results provide some insights in order to understand which features may help configure well performing dynamic recommender ensembles, where performance predictors have emerged as a clear useful characteristic.

7.4 Conclusions

In this chapter we have explored how the performance of a recommender ensemble can be improved by dynamically assigning the weights of its recommenders, by analysing the performance correlation between the values of a performance predictor and the performance of an individual recommender. In this way, we have proposed a dynamic hybrid framework that let decide when and how dynamic hybridisation should be done.

Drawing from the performance predictors proposed in the previous chapter, we have conducted several experiments in order to assess whether recommender en-

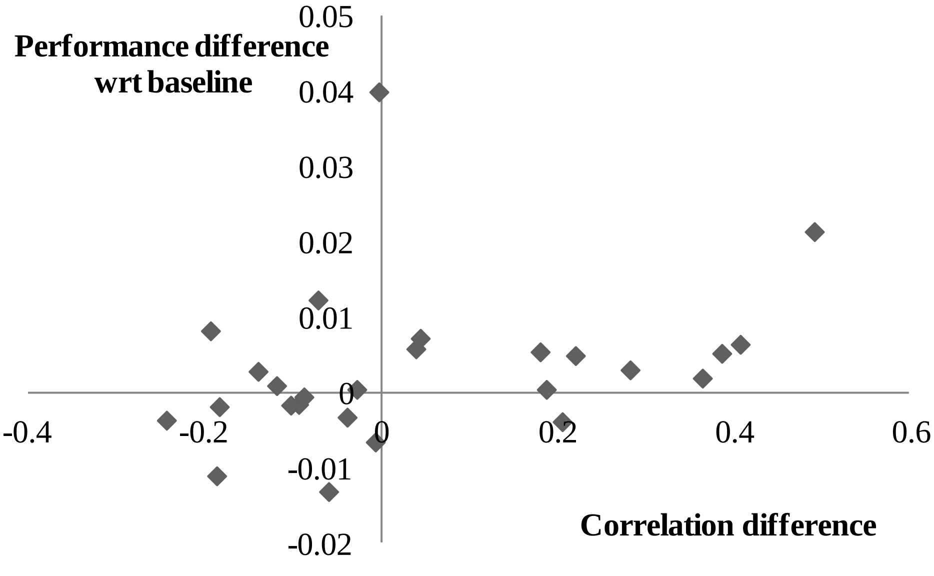

Figure 7.2. For each best and worst dynamic ensemble in Table 7.2, Table 7.11 and Table 7.15, this graph plots the difference in correlation between each predictor and a recommender against the difference in performance between the ensemble and the baseline.

sembles can benefit from dynamic weights according to such predictors. The results obtained in our experiments indicate that a strong correlation with performance tends to correspond with enhancements in ensembles by using the predictor for weight adjustment. The dynamic ensembles usually outperformed the baseline static ensemble for different recommender combinations, supporting their effectiveness in different situations.

In future work we aim to evaluate our framework with more than two recommenders in an ensemble, and more than one performance predictor, eventually, one for each recommender. We also plan to test different normalisation strategies of the predictor's values, where several assumptions about the ideal weight distribution can be verified, such as whether the user's rating distribution or the recommender's output are beneficial for the final performance of the ensemble. Moreover, Machine Learning approaches could also be used to learn the best weights in a user (or item) basis. Despite being more time consuming, these techniques may also achieve good results in terms of performance of the dynamic ensemble, although they are usually more prone to overfit the learned weights.