Proceedings of the 53rd Hawaii International Conference on System Sciences | 2020

Applying Optimal Weight Combination in Hybrid Recommender Systems

Nicolas Haubner
Karlsruhe Institute of Technology
haubner@fzi.de

Thomas Setzer
Catholic University of Eichstätt-Ingolstadt
thomas.setzer@ku.de

Abstract

We propose a method for learning weighting schemes in weighted hybrid recommender systems (RS) that is based on statistical forecast and portfolio theory. An RS predicts the future preference of a set of items for a user, and recommends the top items. A hybrid RS combines individual RS in making the predictions. To determine the weighting of individual RS, we learn so-called optimal weights from the covariance matrix of available error data of individual RS that minimize the error of a combined RS. We test the method on the well-known MovieLens 1M dataset, and, contrary to the "forecast combination puzzle", stating that a simple average (SA) weighting typically outperforms learned weights, the out-of-sample results show that the learned weights consistently outperform the individually best RS as well as an SA combination.

1. Introduction

A recommender system (RS) is an information system aimed at predicting the utility of an item for a particular user from available data. For instance, RS forecast the "rating" or "preference" a user would give to an item, and then recommend those items with the highest predicted preference [1].

RS are vital to commercial applications, especially in the realm of digital business models, for instance to generate playlists for music or video streaming services such as Spotify or Netflix, for online shops such as Amazon, or content or personal connection recommendation in social media such as LinkedIn or Facebook, and for various other application domains. Specifically, nowadays' affiliate networks offering personalized advertisements are based on RS that compute, e.g., when and how to display which banner advertisement to which customer [2].

To make utility predictions, various paradigms exist that use different analytical methods or consider different input data. Amongst the most prominent paradigms are collaborative filtering (CF), content-based filtering (CB), and knowledge-based systems. For instance, CF estimates utility by finding statistical neighbors of a user with respect to past behavior (items previously purchased, viewed, selected, or rated), and then recommends items the neighbors also liked. CB uses characteristics or features of an item to recommend additional items with similar properties to the ones a user has shown high utility for (e.g. gave high rating). Knowledge-based systems exploit relationships and dependencies between items to make recommendations, from simple logical rules or bills of material to semantic technologies exploiting ontological networks. For each paradigm, there is again a multitude of different algorithms available [3, 4].

It has been shown that the combination of two or more RS approaches into a hybrid recommender system (HRS) can improve predictive accuracy, while again various approaches exist how to combine RS [5].

This is well in line with findings when combining statistical forecasts or human judges (social forecasts), where various studies have shown that combining statistical forecasts can improve forecast accuracy [6, 7, 8, e.g.] or judgmental forecasts [9, 10, 11, e.g.].

Approaches to the hybridization of RS are manifold, ranging from weighted (numerically combining the predictive outcomes of different RS), switching (algorithms for dynamically choosing one particular RS), to cascading (one RS generates a coarse ranking, and another RS refines the ranking) [5].

In this work, we consider the first mentioned hybridization technique, the weighing of individual RS to derive a combined forecast, where we aim at learning the weighting scheme that maximizes the accuracy of the hybrid RS.

In the next section, we review related literature on weighted hybrid RS, the challenges of weighting predictive algorithms, and sketch our contribution in the realm of weighted HRS.

URI: https://hdl.handle.net/10125/63930
978-0-9981331-3-3

2. Related Work

First, we summarize the literature on weighted HRS. Subsequently, we sketch the so-called "forecast combination puzzle", stating that a simple average (SA) weighting is hard to beat in an out-of-sample evaluation. Finally, we summarize the contribution of this work to the literature on weighted HRS.

As aforementioned, HRS combine two or more methods to improve performance by alleviating the drawbacks of the individual methods [5]. E.g., CF suffers from the *cold-start problem*, which means it is impossible to create recommendations for users which have not rated any items yet or items which have not been rated yet. As another example, CB suffers from *over-specialization*, i.e., the items which are recommended to a given user are all similar to items which that user has rated positively, resulting in a lack of serendipity and possibly even the recommendation of substitute products to already consumed ones.

Furthermore, each predictive method uses a certain part of the overall information available to make the prediction, and lacks other information with predictive value exploited by another method based on different data. E.g., CF only uses the user-item rating matrix as an input, while CB also exploits item metadata in order to compute similarities between items.

In a weighted HRS, several individual recommendation techniques are run independently, hence considering different information, and their predictions are then combined using an aggregation function [5] (e.g. a linear combination for a regression task [12] or a majority vote for a classification problem [13]).

The combination of two or more prediction methods in order to improve accuracy has been subject to a lot of research in the fields of statistics and forecasting. In their seminal paper, Bates and Granger [6] introduce Equation (1) for choosing the combination weights in the case of two forecasts[1].

$$w = \frac{\sigma_2^2 - \rho\sigma_1\sigma_2}{\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2} \tag{1}$$

In Equation (1), w represents the weight given to the first forecast. Since the weight vector must sum to 1, in the two-forecast case, the second forecast is given a weight of $1 - w$. σ_i is the standard deviation of past errors of forecast i, and ρ is the correlation between the past errors of the two forecasts. The better forecast, i.e. the one with the lower mean squared error (MSE), receives a higher weight. This weighting strategy is called optimal weights (OW). It minimizes the MSE in-sample, i.e. on the available training data, given that the individual forecasts are unbiased, i.e. do not consistently over- or underestimate the true values, and that the performance of the individual forecasts is consistent over time. If the forecasts are biased, they must be corrected by subtracting the mean bias from each prediction.

After Bates and Granger's initial article, a lot of research was produced on the combination of forecasts [7, 14, e.g.]. Interestingly, it was repeatedly reported that using the SA, i.e. giving equal weights to all forecasts, outperformed more sophisticated strategies that learn the weights, such as OW. This observation has been coined "forecast combination puzzle" and has been subject to research for decades.

Smith and Wallis [14] provide a theoretical explanation for the puzzle: For more sophisticated weighting schemes like OW, the weights must be estimated from past errors of the individual forecasts. While Equation (1) is guaranteed to yield minimal MSE in-sample, it does not necessarily deliver an estimate which is optimal out-of-sample, i.e. on unseen test data.

Empirical research in forecast combination typically uses rather small datasets such as time series of economic indicators with only a few dozens or hundreds of observations. The smaller the training sample on which the weights are learned, the higher the estimation uncertainty and the more learned weighting schemes like OW are adjusted to randomness in the training sample rather than to persistent structure in the data (overfitting error). This results in high out-of-sample errors due to this high model variance error component.

Static weights such as SA, on the other hand, are not learned on training data and are therefore not subject to wrong fitting to randomness. Therefore, they do not overfit the training data; they have a sampling-variance-related error of 0.

On the other hand, SA does not exploit any structure from the training data, and therefore SA is subject to an underfitting error: It produces higher in-sample MSE than OW, which produces the lowest in-sample MSE of all linear weight combinations. The difference is, in statistical learning theory, called the model bias. Hence, for which of both combination models, OW versus SA, the aggregate of model bias and model variance is lower, produces the lower MSE out-of-sample. In the literature on statistical forecasting, this is coined the bias-variance trade-off.

The forecast combination puzzle therefore results from the fact that model variance typically exceeds model bias, especially if there is a rather small number of observations, and therefore the SA is hard to beat in

[1] In their original article, there is a sign error ($-$ instead of $+$) in the denominator.

empirical settings.

This is especially the case if the true, unknown out-of-sample optimal weight is close to the SA. However, if the true optimal weight is farther away from the SA, empirical results [14, e.g.] still show that the OW estimate is mostly outperformed by a strategy which is more sophisticated than SA but more robust than OW. This weighting scheme, which was also suggested by Bates and Granger [6], ignores the error correlation between the individual forecasts (assumes $\rho = 0$, even if the true ρ is different) and only uses the inverse ratio of error variances. This strategy, which we will call "modified OW" or "OW ignoring error correlation" in the following, always yields a w in between w_{sa} and w_{ow} (given $p > 0$), which can be seen in Equation (1). It is more sophisticated than SA since it gives more weight to the better forecast, but is more robust than OW because OW can lead to extreme and instable values of w, especially if σ_1 and σ_2 are similar and ρ is close to 1, leading to the denominator of Equation (1) approaching 0.

In this paper, we apply OW to learn weights in HRS. Not a lot of research has been published on the selection of combination weights in weighted HRS. Jahrer et al. [15] compare different weighting strategies like ridge regression, neural networks, or gradient boosted decision trees. Their ridge regression combination is similar to OW, but some differences exist: They solve a ridge regression with a regularization parameter λ in order to avoid overfitting. This approach can lead to weight vectors which do not sum to 1, especially in the case of systematically biased predictors, which are not checked for or preprocessed. Our OW approach instead checks the predictors for mean biases, corrects them if necessary, and then solves a constrained least squares problem with the condition that the resulting weight vector must sum to 1. Furthermore, instead of learning the weight vector on the whole training set using cross-validation, they train the individual methods on the training set and learn the combination weights on a holdout set which is rather small in relation to the training set and might not properly represent the whole dataset. This way, the resulting weight vector might overfit to the holdout set.

We show how OW can be derived from a large user-item rating matrix and demonstrate that, in contrast to the usual findings in the realm of the combination puzzle, that OW are beneficial to combine individual RS given the large sets of training data, which are mostly available in RS. We propose a simple and computationally efficient combination strategy for HRS that can be expected to outperform other weighting schemes given sufficient training sample size. In addition, we analyze the bias-variance trade-off along the dimension n_{train}, i.e. how the performance of OW develops in relation to SA and the more robust modified OW when the number of available observations decreases.

3. Methodology

This section explains how we calculate optimal combination weights of different RS methods based on the errors that those methods produce, and then evaluate OW by comparing it to SA and the individual methods.

In a first step, we check whether the assumptions under which OW produce in-sample optimal weights, namely that the individual RS are unbiased, are met. In case these assumptions are violated, we preprocess and transform the data aimed at meeting the presumptions.

Second, we split the available data into a training set of the size n_{train} and a test set of the size n_{test}. The training set is used to learn optimal combination weights, while the test set is used to evaluate the combined predictions.

Third, we run a k-fold cross-validation on the training data. In each iteration, we train the individual methods on $k-1$ folds of the training set and evaluate their predictions on the remaining fold. Using the error vector of each RS method, we can estimate the error covariance. The error covariance is averaged over the k iterations and then used to compute OW. This approach is taken since, as described above, OW is only guaranteed to be the optimal combination strategy in-sample, i.e. on the training set. Using cross-validation to compute a pseudo-out-of-sample estimate of the error covariance allows us to derive a more robust weighting strategy which does not suffer as much from overfitting as in-sample OW.

Finally, we re-train all components on the whole training set and let them make predictions on the test set, which has not been used at all so far. We compare the performance – measured by root mean squared error (RMSE) – of the individual RS with the performance of combinations with OW and SA weighting.

4. Experimental Design

We now describe the design of the experiments we conduct to analyze the out-of-sample accuracy using OW compared to SA and the individual models, respectively. First, we characterize the dataset used in the experiments. Second, we describe the individual RS methods which we apply and combine.

4.1. Dataset

For our experiments, we use the MovieLens 1M dataset [16]. The data consists of 1,000,209 ratings of 3,900 movies made by 6,040 users. The ratings are based on a five-star scale, i.e., they have integer values from 1 (worst) to 5 (best). We represent the ratings in the user-item matrix R, where each row vector contains the ratings by one user and each column contains the ratings for one item. An entry r_{ui} contains the rating for item i by user u, or a missing value in case no no rating has been given yet. In our experiments, we remove a subset of the available ratings in R as a test set, which are then the "missing" values to be predicted out-of-sample. We denote the prediction of user u's rating for item i by $\hat{r}_{ui}$.

In addition to the ratings, the dataset contains metadata for users and items: Each user is described by his or her gender, age (seven groups), occupation, and ZIP code; each movie is represented by its title, genre(s), and year of release. We denote the user (item) metadata as a matrix U (I), where each row vector represents a user (item).

4.2. Individual RS methods

We compare and combine four different methods, which are briefly described in the following. They are chosen such that each method either uses different input data or makes different assumptions about the underlying preference model.

4.2.1. Neighborhood-based collaborative filtering ("knn") This prediction technique exploits (only) the user-item matrix R. The assumption is that users who behaved in a similar way in the past will also behave in a similar way in the future. There exists user-based and item-based CF.

User-based CF computes similarities between users based on their rating vectors, using a similarity metric such as the cosine similarity. For a given user and a given item, the rating prediction is a weighted average of the ratings for this item given by the most similar users (the "neighbors"). The ratings of neighbors are weighted by their similarity to the given user. This means that items are recommended which are popular in a neighborhood of like-minded users [17, e.g.].

Item-based CF is analogous to user-based CF, only with the terms "user" and "item" exchanged.

4.2.2. Model-based collaborative filtering ("svd") While neighborhood-based CF is based on a nearest-neighbors regressor, model-based CF methods are trained to learn a model that describes the user preferences. Any predictive model can be used, but the most popular method is matrix factorization.

In matrix factorization CF, the user-item matrix R is decomposed into a matrix of user factors and a matrix of item factors. Each user and each item are represented by f latent factors (f is a hyper-parameter which can e.g. be learned using cross-validation). The factor matrices are learned, e.g. by using an alternating least squares procedure. The prediction matrix $\hat{R}$ is then derived by multiplying the two factor matrices [18, e.g.].

Matrix factorization CF has empirically been shown to be one of the most accurate and also computationally efficient RS algorithms.

4.2.3. Content-based filtering ("cb") CB works in a similar way to item-based CF. The difference is that the similarities between items are not computed based on ratings, but instead based on the item metadata I. For a given user, items are recommended which are most similar, based on their properties (for movies, e.g. genre and year of release), to items which the user has liked in the past [19, e.g.].

By computing similarities based on I, CB integrates independent information which CF does not exploit.

4.2.4. Demographic filtering ("df") Demographic filtering (DF), like CB, exploits independent metadata in addition to R. It is similar to user-based CF, but the similarities between users are computed based on the user metadata U (e.g. age, gender, occupation). Users are recommended items which are popular in their demographic neighborhood [20, e.g.].

4.3. Preparation

For each of the individual RS methods, we first test the OW model assumption of no mean bias, i.e., whether the mean prediction errors of the RS methods are approximately 0.

Table 1 displays the mean errors for all methods, as well as the standard errors. Mean errors are very close to 0 and standard errors very low, so we include all of the individual RS in our experiments without a prior mean-debiasing (error centering), as we expect the impact to be negligible.

To parameterize the RS methods and assess their out-of-sample performance as well as the performance of their combinations, we apply the approach described in Section 3. We use 75% of the ratings as training data and 25% as test data, which for the full MovieLens dataset results in $n_{train} \approx 750,000$ and $n_{test} \approx$

Predictor	Mean prediction error	Standard error
svd	0.0002	0.0007
knn	-0.0155	0.0007
df	-0.0025	0.0010
cb	0.0081	0.0010

Table 1. Mean prediction errors of all individual RS with standard errors.

$250,000$. For the pseudo-out-of-sample estimation of the error covariance, we run a cross-validation with $k = 5$ folds.

In addition, to analyze the behavior of OW also on smaller sets of available training data, we conduct the experiments with two subsets of the data, where we use only 50% and 10% of the available ratings, respectively.

5. Experimental Results

First, we provide and discuss results with the complete dataset. Afterwards, we discuss the results with the rating subsets in order to show the effects of a decreasing n_{train}.

Table 2 displays the aggregated experimental outcomes with the complete MovieLens dataset. The first column denotes the weighting scheme used, i.e., whether the individual RS methods (rows 1-4), SA over mutual combinations of the methods (rows 5-11), the modified OW scheme ignoring error correlation (rows 12-18), or an OW combination scheme (rows 19-25) has been applied. The second column displays the individual methods used or combined. For better comparison, there is one additional row ("Total") for SA, modified OW, and OW, respectively, which reports the aggregate over all pairs. The third column shows the out-of-sample RMSE of each method or the methods' pairwise combination over ten random train-test splits together with the standard deviation of the RMSE.

In order to make the results more accessible, in the fourth column, for each of the combinations with learned weights we additionally display the percentage difference in RMSE to the respective SA combination of the same RS methods. For instance, a value of -5% means that the learned weights resulted in an RMSE 5% lower than the one obtained when using SA.

Finally, the two outer-right columns display the learned weight w together with the weight's standard deviation over the ten runs, and the error correlation ρ of the two combined methods, also with its standard deviation.

Starting the results discussion with the individual methods, the table shows that the sole application of *cb* results in the highest RMSE of 1.0224, while *svd* achieves, with 0.878, the lowest RMSE of all individual methods.

In line with expectation from the literature on forecast combination, SA combination further reduces the RMSE obtained with all individual RS methods. While this is expected for *cb* with the individually highest RMSE, where an SA with any of the individually better-performing methods decreases RMSE compared to *cb*, also the RMSE of *svd* is slightly reduced by an SA combination with *knn* from 0.8780 to 0.8738.

The results in column "Δ SA" show that the RMSE with SA combination is further reduced by using an OW combination ignoring error correlation for all pairs. In total, over all six mutual RS method combinations, the average RMSE is reduced by 0.47%.

However, the RMSE with the OW ignoring error correlations is strongly reduced when learning and applying OW considering error correlation, as can be seen in the seven rows on the bottom of the table. The overall lowest RMSE of 0.8725 is obtained with an OW combination of *svd* and *knn*. The lowest improvements are observed when combining *svd* and *knn* (-0.15%), while the highest improvements are made when combining *svd* and *cb* (-3.66%) or *knn* and *cb* (-3.40%). On average over all six pairs, applying OW results in a RMSE that is, on average, 1.97% lower than the RMSE with SA.

At a first glance, the results contradict empirical findings in forecast combination research, indicating that SA is hard to beat out-of-sample, and if so, then rather by the modified OW which ignores ρ than by OW.

Looking at the table in more detail, we observe that OW is the dominant combination strategy for each pair of forecasts. However, for some combinations the difference to SA is small and the difference to the individual methods is comparably large, while for other combinations the difference to SA is large and the difference to the individual methods is smaller.

In the cases where the weight value w_{ow} learned by OW approaches $w_{sa} = 0.5$ (combination of *svd* and *knn*, $w_{ow} = 0.6$, and combination of *df* and *cb*, $w_{ow} = 0.61$), the RMSE difference between OW and SA is rather small (0.15% and 0.29%, respectively). Obviously, in case OW are similar to SA weights, the RMSE difference between both combination schemes is also small, but the learned weights still improve the results of an SA combination.

On the other hand, for those combinations where OW is farther away from SA (the other four combinations), the difference in RMSE between OW and SA is larger (between 2.06% and 3.66%).

The forecast combination puzzle states that typically the estimated OW are too instable to improve the RMSE

Weighting	Method(s)	RMSE (std.)	Δ SA	w (std.)	ρ (std.)
Individual methods	cb	1.0224 (0.0009)			
	df	0.9768 (0.0011)			
	knn	0.8964 (0.0012)			
	svd	0.8780 (0.0016)			
Simple average ($w = 0.5$)	df + cb	0.9488 (0.0007)			
	knn + cb	0.9279 (0.0010)			
	knn + df	0.9155 (0.0012)			
	svd + cb	0.9103 (0.0012)			
	svd + df	0.8962 (0.0011)			
	svd + knn	0.8738 (0.0014)			
	Total	0.9121 (0.0011)			
OW ignoring error correlation	df + cb	0.9478 (0.0007)	-0.10%	0.52 (0.0002)	
	knn + cb	0.9202 (0.0010)	-0.83%	0.56 (0.0002)	
	knn + df	0.9123 (0.0012)	-0.35%	0.54 (0.0001)	
	svd + cb	0.9007 (0.0012)	-1.05%	0.57 (0.0002)	
	svd + df	0.8917 (0.0011)	-0.49%	0.55 (0.0003)	
	svd + knn	0.8737 (0.0014)	-0.01%	0.51 (0.0002)	
	Total	0.9078 (0.0011)	-0.47%		
OW	df + cb	0.9460 (0.0007)	-0.29%	0.61 (0.0011)	0.80 (0.0003)
	knn + cb	0.8964 (0.0010)	-3.40%	0.96 (0.0009)	0.87 (0.0002)
	knn + df	0.8966 (0.0011)	-2.06%	0.95 (0.0013)	0.91 (0.0002)
	svd + cb	0.8770 (0.0014)	-3.66%	0.92 (0.0008)	0.84 (0.0006)
	svd + df	0.8761 (0.0012)	-2.24%	0.86 (0.0021)	0.88 (0.0003)
	svd + knn	0.8725 (0.0015)	-0.15%	0.60 (0.0044)	0.95 (0.0002)
	Total	0.8941 (0.0012)	-1.97%		

Table 2. Aggregated predictive results of the individual RS algorithms and their pairwise combination using OW and SA on the MovieLens 1M dataset.

with SA out-of-sample. Concretely, the puzzle states that when OW is close to SA, SA will outperform OW, and when OW is farther away from SA, the modified OW is to be preferred. Our results contradict this advice: OW is the best weighting strategy for all cases, and the farther w_{ow} is away from SA, the larger the performance improvement by using OW over SA.

It is also worthwile comparing the OW combinations to the respective individual methods. The OW combination of *df* and *cb* yields the highest improvement (3.08%). This is in part caused by the relatively low error correlation of $\rho = 0.8$. For the combinations of *knn* with *cb* and *df*, respectively, the RMSE is virtually the same as when using *knn* alone, while combining *svd* with *cb* or *df* results in slight improvements. One reason for this, again, is that the errors of *knn* are more highly correlated with those of *cb* and *df* than it is the case for *svd*. Finally, combining the two best individual methods, *svd* and *knn*, yields an improvement of 0.63% over using only *svd*, which is one of the most accurate RS algorithms.

Figure 1 provides more insights on the findings displayed in Table 2. Each subplot represents a combination of one pair of prediction methods. For each combination we determine the out-of-sample RMSE as a function of the weighting parameter w; hence, the curves show which RMSE would have been observed with a particular w, again averaged over ten runs. The vertical lines mark the values of w which result from using OW (dashed line), modified OW (dotted line), and the true out-of-sample optimal value of w (dash-dotted line), which is unknown upfront. SA ($w = 0.5$) is marked on the horizontal axis.

In each subplot, a characteristic U-shape can be observed, which shows that the combination of two methods yields a smaller RMSE than each of the individual methods. The graphs show that the OW estimate is close to the out-of-sample optimal weight.

This is especially the case for those combinations where the error correlation ρ is relatively low ($\rho_{df,cb} = 0.8$, $\rho_{svd,cb} = 0.84$). For the combinations with higher correlation, spec. $\rho_{svd,knn} = 0.95$, the OW estimate is farther off the real out-of-sample optimal weights (the estimate is more instable), but still dominates SA, the modified OW, and the individual methods.

As aforementioned, we ran the same experiments on 50% as well as 10% of the available data in order to analyze the performance relationships with increasing overfitting and instability of the OW estimates with decreasing training set size.

The results for 50% of the data are reported in Table 3. The structure of the table is the same as in Table 2. We also observe similar results as with the full dataset; in particular, SA improves the best individual method, OW estimates ignoring error correlation improve over SA, while the lowest RMSE are achieved with the OW combinations, as before. However, the error difference is smaller than on the full dataset. The OW estimate is similarly accurate as with the full dataset, but the true, out-of-sample optimal weight is closer to SA for all combinations except *df* and *cb*.

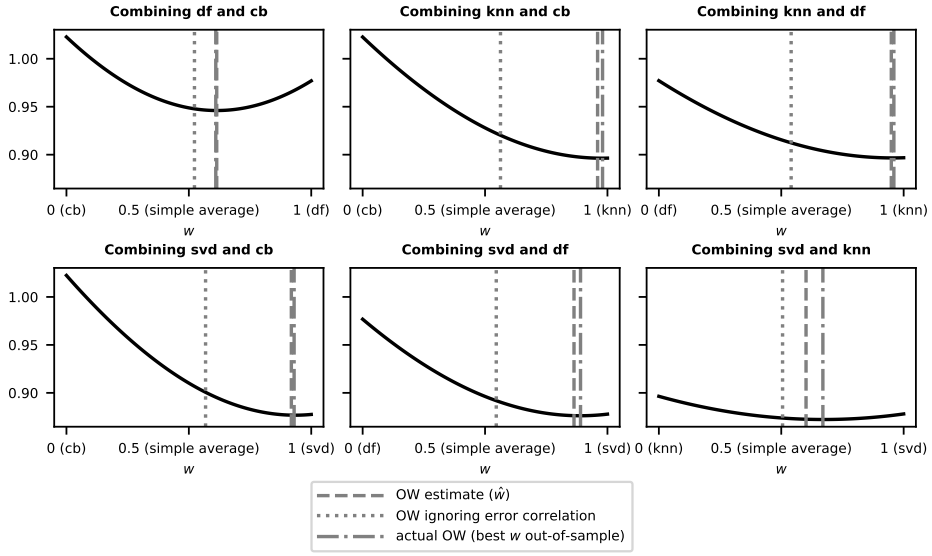

Figure 1. Comparison of OW, SA, individual methods, and OW ignoring error correlation for pairwise combinations of methods on the MovieLens 1M dataset.

Weighting	Method(s)	RMSE (std.)	Δ SA	w (std.)	ρ (std.)
Individual methods	cb	1.0279 (0.0020)			
	df	0.9801 (0.0012)			
	knn	0.9125 (0.0013)			
	svd	0.9181 (0.0018)			
Simple average ($w = 0.5$)	df + cb	0.9518 (0.0015)			
	knn + cb	0.9384 (0.0013)			
	knn + df	0.9273 (0.0011)			
	svd + cb	0.9388 (0.0021)			
	svd + df	0.9239 (0.0012)			
	svd + knn	0.9062 (0.0015)			
	Total	0.9311 (0.0015)			
OW ignoring error correlation	df + cb	0.9507 (0.0015)	-0.12%	0.52 (0.0002)	
	knn + cb	0.9318 (0.0013)	-0.71%	0.56 (0.0002)	
	knn + df	0.9250 (0.0011)	-0.24%	0.53 (0.0002)	
	svd + cb	0.9331 (0.0021)	-0.62%	0.55 (0.0003)	
	svd + df	0.9222 (0.0012)	-0.19%	0.53 (0.0004)	
	svd + knn	0.9062 (0.0015)	-0.00%	0.50 (0.0003)	
	Total	0.9281 (0.0015)	-0.32%		
OW	df + cb	0.9488 (0.0016)	-0.32%	0.62 (0.0012)	0.79 (0.0004)
	knn + cb	0.9121 (0.0013)	-2.81%	0.91 (0.0011)	0.86 (0.0004)
	knn + df	0.9119 (0.0012)	-1.66%	0.94 (0.0014)	0.93 (0.0002)
	svd + cb	0.9158 (0.0022)	-2.45%	0.90 (0.0017)	0.87 (0.0004)
	svd + df	0.9140 (0.0012)	-1.07%	0.79 (0.0030)	0.90 (0.0005)
	svd + knn	0.9062 (0.0015)	-0.01%	0.36 (0.0074)	0.96 (0.0002)
	Total	0.9181 (0.0015)	-1.39%		

Table 3. Results analogous to Table 2 for 50% of the training data. Principle relationships are similar to the full dataset, but the differences are smaller.

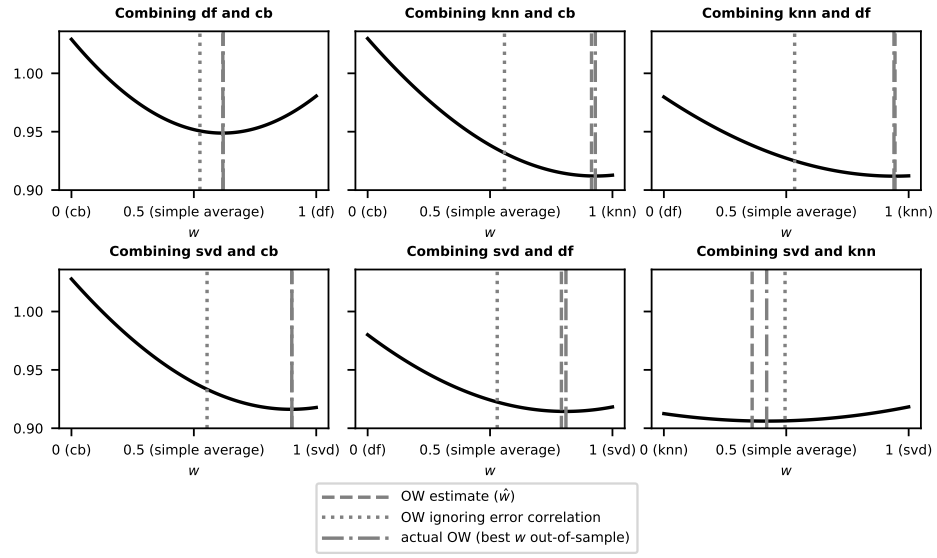

Figure 2. Display analogous to Figure 1 for 50% of the training data. Results are similar to the full dataset, but the actual out-of-sample best weights are closer to SA.

This can be seen in Figure 2, again depicting the RMSE as a function of w but for the 50 % sample. The graphs show relationships similar to the ones observed on the full dataset presented in Figure 1, but the actual, out-of-sample best weight is closer to the middle and therefore closer to $w_{sa} = 0.5$ for almost all subplots. Therefore, the difference in performance is smaller than on the full dataset.

Finally, the results for 10% of the data are reported in Table 4. The table shows that even when using only $\frac{1}{10}$ of the available data, OW is still the combination strategy that results in the lowest RMSE and outperforms SA as well as modified OW. However, as expected, the error difference is smaller than for 50% and 100% of the data, since for the decreased number of observations the model variance (overfitting of weight estimates) is stronger.

OW only dominates for five mutual combinations of methods, while for *knn* and *df* SA would have outperformed estimated OW combinations, specifically with OW estimates considering error correlations, which are, for this combination, dominated also by more robust OW estimates ignoring correlations.

The increasing instability of the OW estimates can also be seen in Figure 3. We observe that the distance between the true out-of-sample optimal weight and the OW estimate increased. For instance, for the combination of *knn* and *cb*, the OW estimate is off by 0.07 $(0.68 - 0.61)$ on the 10% dataset, whereas for the whole dataset, it is only off by 0.02 $(0.98 - 0.96)$. Similarly, for the combination of *knn* and *df*, the OW estimate is off by 0.12 $(0.59 - 0.47)$ on the 10% sample, in comparison to only 0.01 $(0.96 - 0.95)$ on the full dataset. This is also the one combination where the OW estimate is outperformed on the 10% sample by both SA and OW ignoring error correlation, as described above.

6. Conclusion and Discussion

Our experimental findings show that OW estimates can be learned in the realm of RS that significantly outperform the individually best RS as well as the SA of RS algorithms. While this finding contradicts the forecast combination puzzle, stating that most likely SA will outperform learned weights, this outcome is explained by the fact that (i) the presumptions of OW optimality on training data are approximately met, and (ii) that a comparably large set of training observations is available, which is typically not available in business forecast scenarios usually considered in the forecasting literature.

As large sets of observations are often available for RS, e.g. in the realm of media platforms or e-commerce shops, OW are likely to be a beneficial choice for weighted HRS in many practical settings. This result is of importance as of the key role of RS in today's digital world, and as OW estimates can be computed efficiently. Applying OW is a viable strategy in weighted HRS, where, surprisingly, today there is very little research on strategies for choosing the weight.

It is worthwhile to mention that, considering decreasing benefits of using OW with decreasing

Weighting	Method(s)	RMSE (std.)	Δ SA	w (std.)	ρ (std.)
Individual methods	cb	1.0649 (0.0040)			
	df	1.0062 (0.0041)			
	knn	0.9869 (0.0033)			
	svd	0.9534 (0.0031)			
Simple average ($w = 0.5$)	df + cb	0.9696 (0.0036)			
	knn + cb	0.9736 (0.0044)			
	knn + df	0.9727 (0.0042)			
	svd + cb	0.9810 (0.0038)			
	svd + df	0.9583 (0.0039)			
	svd + knn	0.9517 (0.0028)			
	Total	0.9678 (0.0038)			
OW ignoring error correlation	df + cb	0.9680 (0.0036)	-0.16%	0.53 (0.0008)	
	knn + cb	0.9717 (0.0044)	-0.19%	0.53 (0.0008)	
	knn + df	0.9728 (0.0042)	0.01%	0.50 (0.0007)	
	svd + cb	0.9751 (0.0038)	-0.60%	0.56 (0.0007)	
	svd + df	0.9570 (0.0038)	-0.14%	0.53 (0.0006)	
	svd + knn	0.9508 (0.0028)	-0.10%	0.53 (0.0006)	
	Total	0.9659 (0.0038)	-0.20%		
OW	df + cb	0.9661 (0.0038)	-0.36%	0.62 (0.0031)	0.75 (0.0009)
	knn + cb	0.9675 (0.0045)	-0.62%	0.61 (0.0039)	0.78 (0.0011)
	knn + df	0.9733 (0.0042)	0.06%	0.47 (0.0064)	0.88 (0.0013)
	svd + cb	0.9544 (0.0033)	-2.71%	0.97 (0.0059)	0.89 (0.0007)
	svd + df	0.9511 (0.0038)	-0.75%	0.78 (0.0063)	0.91 (0.0005)
	svd + knn	0.9482 (0.0028)	-0.36%	0.80 (0.0081)	0.90 (0.0010)
	Total	0.9601 (0.0038)	-0.80%		

Table 4. Results analogous to Table 2 for 10% of the training data. OW still dominates SA and the modified OW overall, but the gap is smaller due to the higher model variance.

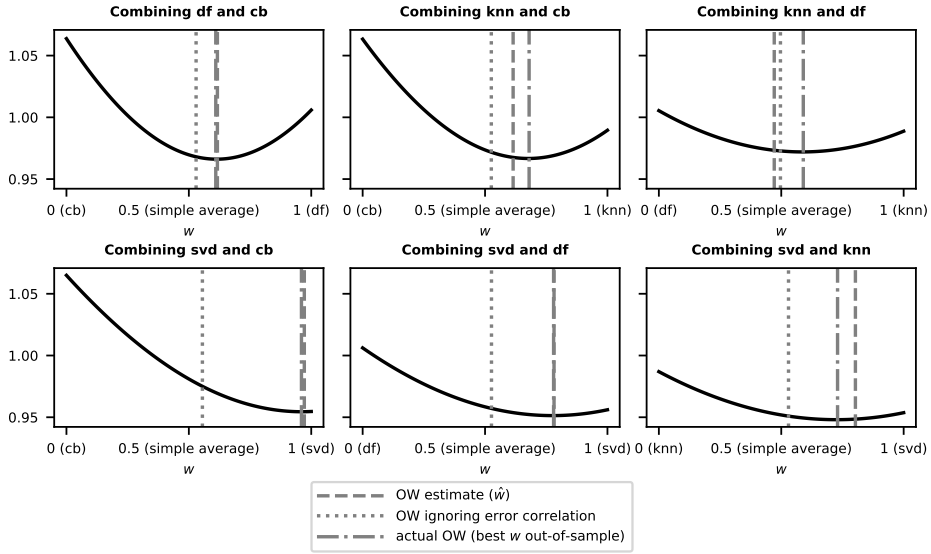

Figure 3. Display analogous to Figure 1 for 10% of the training data. The smaller n_{train} increases estimation uncertainty for OW.

training sample size, OW will not be the optimal choice for sample sizes below a certain threshold, e.g. for small enterprises or platforms with few error observations, where one would then face the combination puzzle. Our results show that we started to experience the puzzle when reducing the training set size to 10% of available rating data. In such cases, model variance with OW increases and regularization is required, for instance by shrinking OW towards SA by a degree that can be learned using cross-validation. Shrinkage has been shown in the realm of business forecasting to be a beneficial strategy in case the number of observations decreases or the number of forecasts increases [21, e.g.].

In our future research, encouraged by the experimental outcomes presented before, we will follow two directions. First, we will study the combination of three or more RS methods. Recent literature on forecast combination motivates the (average) combination of more than two forecasts, so called select-crowds [11]. Fortunately, OW can also be computed for an arbitrary number of individual RS using Equation (2) [22, e.g.].

$$w = \frac{\Sigma_e^{-1}\iota}{\iota'\Sigma_e^{-1}\iota} \tag{2}$$

In Equation (2), Σ_e is the covariance matrix of errors of the individual forecasts and ι is a column vector of p ones, where p is the number of forecasts. Combining three or more RS, we expect further accuracy improvements, and we will analyze under which conditions that is the case.

Second, we will study whether the shrinkage of OW toward SA might further improve accuracy in the realm of RS, specifically with decreasing training set size. We will explore linear shrinkage methods, i.e., linear combinations of OW and SA weights, as well as non-linear approaches to shrink OW towards SA.

References

[1] B. Xiao and I. Benbasat, "E-commerce Product Recommendation Agents: Use, Characteristics, and Impact," *MIS Q.*, vol. 31, pp. 137–209, Mar. 2007.

[2] H. Jafarkarimi, A. T. H. Sim, and R. Saadatdoost, "A Nave Recommendation Model for Large Databases," *International Journal of Information and Education Technology*, vol. 2, no. 3, p. 4, 2012.

[3] C. C. Aggarwal, *Recommender Systems*. Cham: Springer International Publishing, 2016.

[4] F. Ricci, L. Rokach, and B. Shapira, "Recommender Systems: Introduction and Challenges," in *Recommender Systems Handbook* (F. Ricci, L. Rokach, and B. Shapira, eds.), pp. 1–34, Boston, MA: Springer US, 2015.

[5] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction*, vol. 12, pp. 331–370, Nov. 2002.

[6] J. M. Bates and C. W. J. Granger, "The Combination of Forecasts," *OR*, vol. 20, p. 451, Dec. 1969.

[7] R. T. Clemen, "Combining forecasts: A review and annotated bibliography," *International Journal of Forecasting*, vol. 5, pp. 559–583, Jan. 1989.

[8] S. Makridakis and M. Hibon, "The M3-Competition: results, conclusions and implications," *International Journal of Forecasting*, vol. 16, pp. 451–476, Oct. 2000.

[9] G. W. Hill, "Group versus individual performance: Are N+1 heads better than one?," *Psychological Bulletin*, vol. 91, no. 3, pp. 517–539, 1982.

[10] D. Gigone and R. Hastie, "Proper analysis of the accuracy of group judgments," *Psychological Bulletin*, vol. 121, no. 1, pp. 149–167, 1997.

[11] A. E. Mannes, J. B. Soll, and R. P. Larrick, "The wisdom of select crowds.," *Journal of Personality and Social Psychology*, vol. 107, no. 2, pp. 276–299, 2014.

[12] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin, "Combing Content-Based and Collaborative Filters in an Online Newspaper," in *Proceedings of SIGIR 99 Workshop on Recommender Systems: Algorithms and Evaluation*, (Berkeley, CA), p. 14, 1999.

[13] M. J. Pazzani, "A Framework for Collaborative, Content-Based and Demographic Filtering," *Artificial Intelligence Review*, vol. 13, pp. 393–408, Dec. 1999.

[14] J. Smith and K. F. Wallis, "A Simple Explanation of the Forecast Combination Puzzle," *Oxford Bulletin of Economics and Statistics*, vol. 71, pp. 331–355, June 2009.

[15] M. Jahrer, A. Töscher, and R. Legenstein, "Combining predictions for accurate recommender systems," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 693–702, ACM, 2010.

[16] F. M. Harper and J. A. Konstan, "The MovieLens Datasets: History and Context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, pp. 1–19, Dec. 2015.

[17] X. Ning, C. Desrosiers, and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," in *Recommender systems handbook*, pp. 37–76, Springer, 2015.

[18] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender systems handbook*, pp. 77–118, Springer, 2015.

[19] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The Adaptive Web: Methods and Strategies of Web Personalization* (P. Brusilovsky, A. Kobsa, and W. Nejdl, eds.), pp. 325–341, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[20] B. Krulwich, "Lifestyle finder," *AI Magazine*, vol. 18, no. 2, pp. 37–46, 1997.

[21] S. M. Blanc and T. Setzer, "When to choose the simple average in forecast combination," *Journal of Business Research*, vol. 69, pp. 3951–3962, Oct. 2016.

[22] A. Timmermann, "Chapter 4 Forecast Combinations," in *Handbook of Economic Forecasting* (G. Elliott, C. W. J. Granger, and A. Timmermann, eds.), vol. 1, pp. 135–196, Elsevier, Jan. 2006.