Optimal Recommender Systems Blending

Fabio Roda
LIX, École Polytechnique
F-91128
Palaiseau,France
roda@lix.polytechnique.fr

Alberto Costa
LIX, École Polytechnique
F-91128
Palaiseau,France
costa@lix.polytechnique.fr

Leo Liberti
LIX, École Polytechnique
F-91128
Palaiseau,France
liberti@lix.polytechnique.fr

ABSTRACT

In the Recommender Systems field *ensemble techniques* gain growing interest. This approach is based on the idea of mixing many recommenders and to get an average prediction from all of them. Even if it is useful this process may be very expensive from a computational point of view. We propose the use of Operations Research techniques in order to optimize the balance of different predictors and to accelerate it. We show that this problem can be generalized, thus we provide a mathematical framework which helps to find further improvements.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering; H.3.4 [**Systems and Software**]: Performance evaluation.; G.1.6 [**Optimization**]: Nonlinear programming

General Terms

Algorithms, Measurement, Experimentation.

Keywords

Recommender Systems, Collaborative Filtering, Optimization.

1. INTRODUCTION

Recommender Systems represent already a successful technology and have a strong foundation [6, 12, 11]. By the way, many research groups are proposing possible extensions in order to overcome some limitations [1]. Recent trends suggest the use of ensemble techniques in order to improve the quality of recommendations. This approach is based on the idea of mixing many recommenders and to get an average prediction from all of them. Instead of using only one predictor, many are used together and they cooperate to produce the ultimate recommendation which is some kind of combination (*blend*) of the ones proposed by each of them. Bigchaos team (one of winners of the Netflix Prize) remarks this interesting ideas [2]: *"Blending techniques were used to combine the independently trained predictors. ...An ideal solution would be to train all models in parallel and treat the ensemble as one big model. The big problem is that training 100+ models in parallel and tuning all parameters simultaneously is computationally not feasible."* The blending requires to look for an optimal balance between the recommenders (since the ultimate outcome is produce by the cooperation of all of them) but it is computationally difficult.

Most of the related works propose step by step methods to improve the blending but rarely settle this problem as a real optimization one. We think that Operations Research (OR) techniques could be conveniently exploited to formalize and solve it. In very general term, we propose to consider the predictors as a vector of functions which have to be combined by means of a proper operator in order to minimize a convenient error function which measures the performance of the system. This combination depends on some design variables which play the role of decision variables of an optimization problem. Our work aim to furnish a theoretical framework to better understand the optimal way to mix different recommenders and to suggest possible stategies to manage the computational complexity of this process.

WIMS '11, May 25-27, 2011 Sogndal, Norway

2. THE BLENDING PROBLEM

In order to formally define our problem in OR style we have to identify these key elements:

- objective function;
- decision variables;
- constraints.

Typically, the objective function has to be minimized and represents *the task*, decision variables have to be chosen in a feasible region and represent *choices* and constraints determine the feasible area and represent *limits*.
We begin by introducing a very general formalization (basically a schema) and refine it step by step.

2.1 Problem Schema

Let M be a set of predictors, Pr_m an element of this set, $\beta(M)$ a blending operator which mixes different predictors, λ a vector of design variables, $P_n(\lambda)$ constraints that λ must respect and $\mathcal{E}$ an error function which measures the quality of recommendations.
Thus, the schema of the Blending Problem is:

$$\left.\begin{array}{ll}\min\limits_{\lambda} & \mathcal{E}(\beta(M,\lambda)) \\ \text{s.t.} & P_n(\lambda)\end{array}\right\} \quad (1)$$

2.2 Error Function

First of all we have to define the error function $\mathcal{E}$ which expresses the error done by the system. Many different measures are used in order to evaluate the performance of filtering algorithms employed by Recommender Systems and some metrics fit better for top-N recommendation, and others for prediction. The clamour of the Netflix competition has made one metric notorius: accuracy computed as the square root of the averaged squared difference between each prediction and the actual rating (the root mean squared error or "RMSE"). We believe also that it represents a good choice (see [9] for a discussion on this topic).

Let the r_{ui} denote the actual rating provided by a certain user u for an item i, with $i = 1, 2, ..., n_u$ ($n_u \leq n$, where n is the number of all available items) and let p_{ui} denote the prediction generated by a certain algorithm for the same user and the same item. RMSE, relating to user u, is defined by:

$$RMSE_u = \sqrt{\frac{\sum_{i=1}^{n_u} (r_{ui} - p_{ui})^2}{n_u}} \quad (2)$$

The total RMSE can be obtained as an average of the RMSE of all users:

$$RMSE = \sqrt{\frac{\sum_u \sum_{i=1}^{n_u} (r_{ui} - p_{ui})^2}{\sum_u n_u}} \quad (3)$$

2.3 Blending Function

In order to mix different predictors we may simply calculate their linear combination. Thus, we adopt the *weighted average* as a specialization of the blending operator β. Let p_{mui} be prediction generated by predictor Pr_m for the user u and the item i. The RMSE for a user becomes:

$$RMSE_u = \sqrt{\frac{\sum_{i=1}^{n_u} \left(r_{ui} - \frac{\sum_{m=1}^{M} w_{mu} p_{mui}}{\sum_{m=1}^{M} w_{mu}}\right)^2}{n_u}} \quad (4)$$

We look for the vector of weights $[w_{1u}, \ldots, w_{Mu}]$ which would have minimized the error. An important remark is that the weights are the decision variables and (4) is the objective function; thus ratings and predictions are known parameters. Moreover, we underline that we need a different vector of weights for each user. Basically, we have to solve as many specific optimization problems as the total number of users; in the reminder we call it *personalized blending*.

On the other hand, a different approach is the *global blending*. We look for a single vector of weights, that is independent from the users. In this case, the RMSE is:

$$RMSE = \sqrt{\frac{\sum_u \sum_{i=1}^{n_u} \left(r_{ui} - \frac{\sum_{m=1}^{M} w_m p_{mui}}{\sum_{m=1}^{M} w_m}\right)^2}{\sum_u n_u}} \quad (5)$$

The evaluation of RMSE is typically performed using the "leave-n-out" approach [4], where a part of the dataset is hidden and the the rest is used as a training set for the recommender, which tries to predict properly the withheld ratings.

2.4 Reformulations

The objective (4) is a non linear form which is not easy to optimize, so we look for another way to formulate the problem. A first observation is that we can simplify the equations of RMSE ignoring the square root (because we consider only positive values). Secondly, we can introduce a constraint on the sum of weights, so for the *personalized blending* we get only one optimization problem:

$$\left.\begin{array}{ll}\min\limits_{w} & \frac{\sum_{i=1}^{n_u} (r_{ui} - \sum_{m=1}^{M} w_{mu} p_{mui})^2}{n_u} \\ \text{s.t.} & \sum_{m=1}^{M} w_{mu} = 1\end{array}\right\} \quad (6)$$

We recall that we should solve one optimization problem like (6) for each user. On the other hand, for the *global blending* we get:

$$\left.\begin{array}{ll}\min\limits_{w} & \frac{\sum_{i=1}^{n_u} (r_{ui} - \sum_{m=1}^{M} w_m p_{mui})^2}{n_u} \\ \text{s.t.} & \sum_{m=1}^{M} w_m = 1\end{array}\right\} \quad (7)$$

We plan to use these formulations to perform initial computational tests. Anyway, if we want be general we may relax the constraint, and recognise that we got a *sum of squares linear fractional problem* whose general mathematical programming problem (SSLFP) can be formulated as follows:

$$min \sum_{i=1}^{n} \left(\frac{\sum_{j=1}^{n} a_{ij} x_j + b}{d^T + e} - p_i \right)^2 \quad (8)$$

$$x_i \geq 0 \quad \forall i \leq n$$

Where $x \in \mathbb{R}^n, a, d \in \mathbb{R}^n$ are real coefficients, $b, e \in \mathbb{R}, p \in \mathbb{R}^n$. This formulation is a general, hence very useful in order to look for good reformulations. *"It is well known that several different formulations may share the same numerical properties (feasible region, optima) though some of them are easier to solve than others with respect to the most efficient available algorithms. ... When a problem with a given formulation P is cast into a different formulation Q, we say that Q is a reformulation of P."* [8].

2.5 Conclusion and future work

We plan to move towards two different objectives. Firstly we will perform computational tests based on (6) and (7). This activity is planned as follows:

- identification of the set of predictors, including well-known ones as KNN [5] and Slope-One [7];
- choice of the dataset, for example the ones provided by GroupLens (`http://www.grouplens.org`);
- selection of the solvers for the problems (6) and (7); since they are non-linear problems, we can use well-known non-linear solvers like COUENNE [3] or BARON [10];
- comparison of the results obtained with (6) and (7); in our opinion, (6) allows to obtain a better level of accuracy with respect to (7), while the time complexity of (6) is greater than the complexity of (7) by a factor that is the total number of user.

Secondly, we want to investigate the best way to reformulate the problem starting from (8).

3. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] A.Töscher, M.Jahrer, and R.M.Bell. The bigchaos solution to the netflix grand prize. pages 1–52, 2009.

[3] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4):597–634, 2009.

[4] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering, 1998.

[5] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In G. Cooper and S. Moral, editors, *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, San Francisco, 1998. Morgan Kaufmann.

[6] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[7] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of SIAM Data Mining (SDM'05)*, 2005.

[8] L. Liberti. Reformulations in mathematical programming: Definitions. In *CTW08 Proceedings*, pages 66–70, New York, NY, USA, 2008. G. Righini (ed.).

[9] S. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 1097–1101, Montréal, Québec, Canada, 22/04/2006 2006. ACM, ACM.

[10] N. Sahinidis and M. Tawarmalani. *BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2005.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[12] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating 'word of mouth'. In *Conference proceedings on Human factors in computing systems (CHI'95)*, pages 210–217, Denver, CO, May 1995. ACM Press/Addison-Wesley Publishing Co.