ESANN'2007 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2007, d-side publi., ISBN 2-930307-07-2.

Collaborative Filtering with interlaced Generalized Linear Models

Nicolas Delannay[*], Michel Verleysen
Université catholique de Louvain (UCL)
DICE - Machine Learning Group
Place du levant 3, 1348 Louvain-la-Neuve, Belgium
Nicolas.Delannay@uclouvain.be, verleysen@dice.ucl.ac.be

Abstract. Collaborative Filtering (CF) aims at finding patterns in a sparse matrix of contingency. It can be used for example to mine the ratings given by users on a set of items. In this paper, we introduce a new model for CF based on the Generalized Linear Models formalism. Interestingly, it shares specificities of the model-based and the factorization approaches. The model is simple, and yet it performs very well on the popular MovieLens and Jester datasets.

1 Introduction

The Internet is bringing together communities of users, allowing them to share their opinions and experiences on particular sets of items (e.g. songs, books,...). Collaborative Filtering (CF) tries to discover patterns within the experiences expressed by the community. The patterns can then be used to automatically link users to the items interesting for them. Typically, it requires the users to encode their experiences over items with ratings (higher values for appreciated items). In this framework, ratings are said to be *explicit*. CF can also be used in other contexts like text mining with a bag-of-words representation. We will not tackle this application in this paper.

Many approaches where proposed to perform CF. Only a very brief overview of the domain is given here. Early methods relied on a nearest neighbor principle: the user should get advice from other most similar users [1], [2]. The main challenge is to define a good similarity between users in particular when they share only very few rated items.

Model-based approaches try to merge the user ratings in typical rating patterns. The aspect model [3] is a good example of the approach. For this model, identified patterns can be interpreted as particular aspects of the rating process, and users are represented by their sensibilities to the different aspects. Other interesting variants of this model were proposed in [4].

CF can also be handled by factorization of the sparse matrix of ratings. The goal is then to find two compact factor matrices minimizing a reconstruction error. The choice of the reconstruction error and the definition of compactness are critical to obtain an efficient algorithm. The Principal Component factorization and its generalization to the exponential family of distribution [5] is a good starting point. Another objective function based on the maximum-margin factorization of a matrix [6], [7], has shown to perform very well.

The aim of this paper is to develop a model for CF based on the well-known Generalized Linear Models [8]. The model lies midway between the model-based approach

[*]N.D. is Research Fellow of the F.N.R.S.

and the matrix factorization one. What makes the model appealing is its simple formulation and the possibility to handle large datasets, without trading off the level of prediction performances.

This paper focuses only on the prediction of explicit ratings. Obviously, rating prediction is rarely the ultimate goal of CF. In general, we are rather interested in making good recommendations. For this task, other valuable criteria were proposed, like the prediction of the best short ranked list, or the serendipity. However, evaluating objectively recommendations is still an open issue. Also, the manner data are collected often strongly biases the results. For a good review of the problem see [9].

The paper is organized as follows. In the next section the general problem of rating prediction is formulated. In section 3, the model is developed, and its optimization is discussed in section 4. Section 5 presents experiments on two datasets.

2 Collaborative Filtering for rating prediction

As exposed in the introduction, our aim is to design a model capable of predicting accurately the ratings. The pedicted ratings can then be used for recommendation purposes.

In the following, the set of items will be noted $Y = \{y_m\}_{m=1}^{M}$ and the set of registered users $U = \{u_n\}_{n=1}^{N}$. The ratings already expressed by these users are stored in a list of triplets $R = \{(u, y, r)_l\}_{l=1}^{L}$. The ratings r can take value in a (discrete or continuous) ordered set. To avoid conflicts between subscripts, the rating index l will be noted on the left of the symbol. For example, the user who made the lth rating is ${}_l u$.When no particular element of the set is designated, the subscript will be omitted.

Let $\mathcal{S}$ be the model. For any couple (u, y), the model returns a prediction $s(u, y)$. The quality of the prediction is evaluated by a loss function $\lambda(r(u, y), s(u, y))$, where $r(u, y)$ is the true observed rating. For concision, (u, y) is implicit in the succeeding formulas. Ideally, the model should try to minimize the expected loss function $\mathbb{E}\{\lambda(r, s)\}$ over independent ratings (i.e. not used for learning). In practice, this objective is approached by minimizing a regularized empirical loss

$$\mathcal{L}(\mathcal{S}) = \sum_{l} \lambda({}_l r, {}_l s) + \rho(\mathcal{S}) \ , \tag{1}$$

where the regularizer $\rho(\mathcal{S})$ penalizes the complexity of the model.

3 Interlaced Generalized Linear Models

The fundamental point to setup a model for CF is to choose how to encode the information about users and items. The concept of aspect representation is intuitively appealing. Indeed, items could be represented by their intrinsic quality evaluated from the point of view of different aspects, and users would be encoded with a sensibility to each aspect. Consequently, when there is a match between a user sensibilities and an item qualities, the model should predict a high rating.

These intuitive thoughts are now formalized. Each user u_n is associated with a vector of features $\phi_n \in \Re^K$ (the sensibilities), and each item y_m with a vector of features $\omega_m \in \Re^K$ (the qualities). For the moment, K is assumed to be fixed. The core of the model is to express the interaction between a user and an item by a linear dot product of their respective features, $a = \phi^{\mathrm{T}}\omega$, called the activation. A smooth link function is added between the activation and the prediction, $s = g(a)$. Finally, the decrepancy between prediction and true rating is modeled with a so-called conditional noise distribution $p(r|s;\psi)$, where ψ parametrizes the variance of this noise.

Obviously, this is the same formulation as a Generalized Linear Model (GLM) [8]. One important distinction however is that both factors ϕ and ω in every activation are unknown. In fact, we have a standard GLM for each feature vector taken separately and for this reason the model is named *interlaced* GLM.

In order to link the model with the goal of rating predictions, we still need to precise which loss function and regularizer are used. A natural choice for the loss function is the negative log-likelihood of the noise distribution:

$$\lambda(r, s) = -\log p(r|s;\psi) \ . \tag{2}$$

Continuing with a probabilistic formulation, the regularizer can be expressed with a prior distribution over the feature representation,

$$\rho(\mathcal{S}) = -\log p(\{\phi_n\}, \{\omega_m\}|\alpha) \ , \tag{3}$$

where α is a set of hyperparameters associated to the prior distribution.

There is no a priori best choice for the link function $g(\cdot)$, the noise distribution and the prior features distribution. The assumption of Gaussian noise distribution is frequent for regression problems. In this case, the prediction s is simply the mean of the Gaussian. Besides, if the identity is used for the link function, we obtain a standard linear regression. To avoid predictions outside the interval of allowed ratings, it could be a good idea to have a saturated link function, for example the logistic $s = (1 + \exp(-a))^{-1}$. When the ratings can only take integer value $\{0, 1, \ldots, V\}$, a binomial noise distribution certainly makes sense.

The first role of the prior distribution over features is to circumvent overfitting. A classic choice is the isotropic Gaussian distribution, with precision specified by the hyperparameter α. It corresponds to a ridge regularization. One may use different values of α for each feature vector:

$$p(\{\phi_n\}, \{\omega_m\}|\alpha) = \prod_n \mathcal{N}(\phi_n|0, \alpha_n^u \mathbf{I}_K) \prod_m \mathcal{N}(\omega_m|0, \alpha_m^y \mathbf{I}_K) \ , \tag{4}$$

where $\mathbf{I}_K$ is the KxK identity matrix. Users with less ratings are more subject to overfitting and should thus have a higher precision parameter α_n^u.

The second role of the prior distribution is to encode prior knowledge or constraints. For example, there are reasons to think that users sensibilities should only take positive values. Then, a natural prior distribution would be the Gamma distribution.

The model was developed with intuitive arguments, trying to get the feeling about users and items representation similarly to a model-based approach for CF. But at the end, we have clearly got to a matrix factorization approach. Indeed, the goal is to decompose the sparse matrix of ratings $\mathbf{R}$ in two factor matrices of users and item features $\mathbf{R} \approx \Phi\Omega^{\mathrm{T}}$, where the reconstruction error is specified by the choice of the link function and the noise distribution. The compactness is favoured by the prior distribution.

4 Model Fitting

There are two levels in the optimization of the model parameters. The inner level corresponds to fitting the features ϕ and ω. The outer level corresponds to the hyperparameters, namely the number of features K, the noise distribution parameter ψ, and the prior distribution parameters α.

Let us first consider that the hyperparameters are fixed. A way to optimize the features is to update one feature vector at a time. For example, the optimization of the regularized loss (1) with respect to ω_m is

$$\omega_m^{update} = \arg\min_{\omega_m} \sum_{l \in L_m^y} \lambda({}_l r, {}_l s) + \rho(\omega_m) \ , \tag{5}$$

where L_m^u is the set of indexes associated to the ratings of item y_m. As stated in the previous section, this problem is a standard GLM. There exist very efficient iterative algorithms to solve it. Each feature vector is to be updated in turn and the complete procedure must be repeated until convergence. We observed that the model converge in general in less than 15 complete runs, and it can be applied on large datasets as memory requirements are small.

For the adjustment of the hyperparameters, we propose a pragmatic approach. The first thing to note is that the number K of features acts on the flexibility of the model in a manner similar to the prior distribution. It is thus sensible to fix K sufficiently large and constrain the flexibility through the prior distribution adjustment. For simplicity, one can work with a precision hyperparameter α^u common to all users, and also with a common α^y for the items. Besides, as user and item features interact multiplicatively, it is sufficient to set the precision over users (or items) to an arbitrary value and adjust only the precision of the other set.

Another thing to note is that the variance hyperparameter ψ has a dual influence to the precision hyperparameters. Small noise variance must be compensated by a weak prior distribution (i.e. a small precision), although the duality is exact only for Gaussian noise and Gaussian prior distributions. It is convenient to fix ψ to an estimate of the noise variance and adjust a single precision hyperparameter of features by cross-validation. We apply this methodology on two experiments in the next section.

5 Experiments and Discussion

The model is tested on two publically available datasets: MovieLens [1] and Jester [2]. The goal is to be able to compare the prediction performances with other performances found in the literature. Three different configurations of the model are compared. Also, we follow the procedure described in [4] to evaluate the performances.

5.1 Datasets

- MovieLens: The dataset is distributed by GroupLens Research at the University of Minnesota. It contains 6040 users, 3900 movies (the items), and approximately

[1]http://www.grouplens.org/
[2]http://www.ieor.berkeley.edu/ goldberg/jester-data/

	MovieLens		**Jester**	
	MAE	RMSE	MAE	RMSE
Common Pref.	0.718	0.937	4.50	5.30
Gauss-linear	0.657	0.891	3.27	4.17
Binom-Gamm	0.648	0.875	3.26	4.24

Table 1: Rating prediction performances over MovieLens and Jester datasets for three different model configurations.

1 million discrete ratings on a scale from 1 to 5.

- Jester: For online recommendation of jokes [10], the dataset contains 73,421 users, 100 jokes (the items), and 4.1 millions continuous ratings on the interval $[-10, 10]$. In the experiment, only 18,000 users were randomly selected.

5.2 Model configuration

- Common preference: Baseline configuration giving a starting point for the comparison of performances. In this configuration, $K = 2$ and we constrain $\phi_{n1} = 1$ and $\omega_{m2} = 1$. This way, there is no direct interaction between the users and items features. Consequently, all users have the same order of preference over the items. We use an identity link function and Gaussian noise/prior distributions.
- Gauss-Linear : Standard linear regression configuration with an identity link function and Gaussian noise/prior distributions like in the previous configuration but this time, $K = 15$ and there is no constraint.
- Binom-Gamm: This configuration uses a Binomial noise distribution, a logistic link function, the user features have a Gamma prior distribution, and the item features have a Gaussian prior distribution.

5.3 Procedure

The users in the datasets are divided randomly into three groups of equal size. For each user, one of his ratings is left aside for test. The models are evaluated with a 3-fold cross-validation procedure. Each of the three user sets is in turn used for evaluating the predictions. The model is learned with the other two sets, using their test ratings to select the precision hyperparameter. Then, the features of users in the third set are learned (keeping the item features unchanged), and a prediction is returned for the test ratings of this same set. Finally, the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) averaged over the three cross-validations are computed.

5.4 Results and discussion

The prediction performances are shown in Table 1. In comparison, the best performances found in the literature (to our knowledge) are for MovieLens a MAE of 0.652

(in [7]) and for Jester a MAE of 3.26 (in [11]). Our performances are comparable. We see that the Binom-Gamm configuration performs slightly better than the Gauss-linear one. It seems thus advantageous to respect the bounds of the rating scale within the noise distribution, and to force the user features to be positive. On the other hand, the Gauss-linear configuration is clearly faster to learn.

The common preference configuration is weaker than the other configurations. Nevertheless, the difference is relatively small (in particular for MovieLens), meaning that the dominant patterns in the dataset are global averages. This observation is far from the belief that people's tastes are very diverse and contradictory. We think that expression of diversity is hindered by the constraint to rate items on a single scale.

6 Conclusion

In this paper, we proposed to perform Collaborative Filtering with a model relying on the Generalized Linear Model formalism. It is mathematically well founded, can be applied on large datasets and performs comparably to other state-of-the-art methods. Moreover, prior knowledge can be incorporated naturally with the prior distributions.

Experiments show that the ratings appear to encode mainly a common appreciation over items, and only slightly the users own preferences. This fact limits the use of CF for discovering user interests on the basis of their ratings. We would like to focus more on this task in future work. A step in this direction would be to include in the model the sparsity structure of the rating matrix.

References

[1] P. Resnik and al. Grouplens: An open architecture for collaborative filtering of netnews. In *ACM, Conference on Computer Supported Cooperative Work*, pages 175–186. ACM, 1994.

[2] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[3] Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[4] Benjamin Marlin. Collaborative filtering: A machine learning perspective. Master thesis, University of Toronto, 2004.

[5] M. Collins, S. Dasgupta, and R. E. Schapire. A generalization of principal components analysis to the exponential family. In T. G. Dietterich and al., editors, *NIPS 14*. MIT Press, 2002.

[6] N. Srebro, J. D. M. Rennie, and T. S. Jaakola. Maximum-margin matrix factorization. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *NIPS 17*. MIT Press, 2005.

[7] D. DeCoste. Collaborative prediction using ensembles of maximum margin matrix factorizations. In *ICML '06*, pages 249–256. ACM Press, 2006.

[8] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman and Hall, New York, 1989.

[9] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

[10] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[11] John Canny. Collaborative filtering with privacy via factor analysis. In *SIGIR '02*, pages 238–245. ACM Press, 2002.