Collaborative Filtering in Recommender Systems:

a Short Introduction

Norm Matloff
Dept. of Computer Science
University of California, Davis
matloff@cs.ucdavis.edu

December 3, 2016

Abstract

There is a strong interest in the machine learning community in recommender systems, especially using collaborative filtering. A rich variety of methods has been proposed. Here we briefly introduce several classes of methods.

1 Overview

Say we have user ratings data, with Y_{ij} denoting the rating user i gives to item j, for $i = 1, ..., n,\ j = 1, ..., m$. The term *rating* here can be interpreted quite broadly — it could for instance be a binary variable indicating whether there is (or will be in the future) a link between two vertices in a graph — but for now, we will take as a concrete example one of the most famous recommender system data sets, MovieLens,[1] in which users rate films. This is a very active branch of machine learning research, and has attracted considerable attention in the popular press [6].

Let M denote the matrix of the Y_{ij}. Note that most of M will be undefined, as each user typically has rated only a small fraction of the available items. This sparsity of the ratings matrix is a central issue. The goals include **matrix completion**, i.e. predicting the missing Y_{ij} values, and understanding the underlying processes, i.e. identifying factors affecting ratings. This process is called *collaborative filtering*, alluding to a "collaboration" between users in predicting ratings.

In this paper, we will first give an overview of collaborative filtering methods in Section 2, then show some of the details. Finally, we introduce our **rectools** software package in Section 4.

2 Collaborative Filtering

We are concerned here with *collaborative filtering*, meaning that the predictions for a given Y_{ij} will be informed by prediction patterns of similar users. We will focus here on three general categories of such methods.[2]

[1] http://grouplens.org/datasets/movielens/

[2] There is some inconsistency in the literature regarding the definition of collaborative filtering. Some authors use the term only for the nearest-neighbor method, the first of our three categories. Others use the term for any method involving user data, which will be our meaning of the term here.

2.1 Quick Summary

Below is a very quick summary of the three categories of methods, to find a predicted value $\widehat{Y}_{ij}$ for Y_{ij}.

- *Nearest-neighbor methods:*
  To predict the rating of item j by user i, find the other users in the data set who have rated item j and who have similar rating patterns on other items to user i. Take $\widehat{Y}_{ij}$ to be the average of the ratings of item j among these neighbors. We will refer to this as the k-Nearest Neighbor (kNN) method.
- *Matrix factorization methods:*
  Assume the matrix M can be approximated as

$$M \approx UV \tag{1}$$

  where the matrices U and V of sizes $n \times k$ and $k \times m$ are calculated from the known Y_{ij} values. Typically k is much smaller than n and m. Then set

$$\widehat{Y}_{ij} = U_{i.}V_{.j} \tag{2}$$

  the dot product of row i of U and column j of V.
- *Statistical random effects models:*
  The basic model is

$$Y_{ij} = \text{overall mean} + \text{user i effect} + \text{item j effect} + \text{noise} \tag{3}$$

  The latent user and item effects can be estimated by Maximum Likelihood methods, but the much faster Method of Moments (MM) approach has less stringent assumptions, and yields closed-form expressions,

$$\widehat{Y}_{ij} = Y_{i.} + Y_{.j} - Y_{..} \tag{4}$$

  where the three quantities on the far right are the mean in row i, the mean in column j and the overall mean, among known values in M.

The first category is very much in line with statistical thinking, and the third is explicitly so. The second category is not statistical as described — and as typically perceived in the field — but it too can be viewed in this way, as will be seen below.

Note that RS usage involves two stages:

- **Exploratory stage:** Here the RS administrator explores various approaches, e.g. kNN, MF and RE, and any associated parameters, such as the rank k of U and V in the MF method. Eventually the administrator settles on some method and parameter values.
- **Production stage:** Here new cases come in continually, and the model chosen in the exploratory stage is applied to these cases for predictions one-by-one. The model from the first stage may be updated occasionally with this new data.

3 Details of Collaborative Filtering Methods

3.1 Nearest-Neighbor Methods

To predict the rating of a certain item by a certain user, we compare her ratings vector to those for all our users in our data, finding the ones closest to the given user, among those in which a rating for this item are available. Early work in the field tended to use this approach.

Of course, one of the issues that arises is the definition of "closest," a problem especially in miew of the fact that most of our data is missing. We will return to this issue below.

3.2 Matrix Factorization Methods

From a statistical point of view, matrix factorization methods for recommender systems amount to a principal components analysis (PCA) of the data, or something similar.

3.2.1 Intuition

Row i of UV is a linear combination of the rows of V. Since row i of UV is also the approximation of row i of M, the rows of V then form a basis for a space approximating the rows of M. In other words:

(a) The rows of V serve as synthetic typical users, with each row consisting of the ratings such a sythetic user gives to the various items.

(b) For any real user i in our data set, row i of U gives us weights with which we can express this user's item ratings as a linear combination of the sythetic users.

3.2.2 What Actually Is Done

Our digression to the matrix $\widetilde{M}\widetilde{M}'$ above was done for the purpose of developing intuition. Now let's look at M itself.

Singular value decomposition (SVD) is similar to PCA, writing M as

$$M = QSR \tag{5}$$

where Q and R are orthogonal, and S is a diagonal matrix consisting of the *singular values*, which are the square roots of the eigenvalues/variances mentioned above. Again discarding the small singular values, retaining the k largest, we have matrices U and V such that

$$M \approx UV \tag{6}$$

for matrices U, $n \times k$, and V, $k \times r$, of rank k. U and V are calculated from available data, after which the product UV gives us predicted values for the unknown Y_{ij}:

$$\widehat{Y}_{ij} = U_{i.}V_{.j} \tag{7}$$

where the "hat" means predicted value and the notations $i.$ and $.j$ mean row i and column j of the given matrix.

Here k is the analog of 75 in our intuitive discussion above, as follows. In (6), row i of M, the ratings of user i, is now give approximately as a linear combination of the rows of V (with the coefficients being row i of U). Thus the rows of the matrix V serve as k "representative" users to serve in place of our original n. (They are actually composites of users.) Similarly, the columns of U serve as k "representative" films, in place of our original m. The idea of avoiding overfitting in general is to use fewer predictors, and here in this context of predicting ratings, a smaller k fits that description. Thus k hopefully much less than the rank of M.

On the other hand, if k is too small, we are underfitting. Finding a good "middle ground" for k is one of the major challenges in statistics, and certainly is so here.[3]

3.2.3 One More Refinement

Since the matrix M has nonnegative entries, we might wish to find the matrices U and V with that same property. This is called nonnegative matrix factorization (NMF). The appeal of NMF here is that it makes the "k representative users" and "k representative films" interpretations above more plausible.

[3]Another approach to dealing with overfitting is *regularization*, in which the matrices U and V are constrained from becoming too large. We do not cover this approach here.

NMF is probably the most popular method today.[4]

3.3 Statistical Random Effects Models

A popular extension of matrix factorization methods is to incorporate user and item *biases*, reflecting the tendency of some users to be more generous in rating than others, and the tendency of some items to be rated higher than others.

These quantities will be defined below, but for now, let's just give them names, α_i for user i and β_j for item j. But the point is that the popular approach is to first subtract them from the known Y_{ij}, and then use matrix factorization as above.

Random effects models form a branch of traditional statistical methodology, ironically useful in a modern application such as recommender systems. The model is

$$Y_{ij} = \mu + \alpha_i + \beta_j + \epsilon_{ij} \tag{8}$$

with μ fixed but unknown, and with the other quantities random and unobservable, with mean 0. The term *random effects* here refers to the fact that the users in our data are treated as a random sample from some population of users, with a similar treatment for items.

The quantity μ is then the overall mean rating, among all possible users and items; α_i is the difference between the mean rating that user i would give all possible items; and β_j is the mean rating that all possible users would give to item j. The ϵ term accounts for variability not incorporated in the other quantities.

The three random quantities are assumed independent over all i and j. In the Maximum Likelihood formulation, the random quantities are also assumed to be normally distributed.

According to the history given in [5], random effects models of various kinds were used in the early history of recommender systems, but rejected on computational grounds when data sets became much larger. More recently, [2] and [5] proposed circumventing these problems by using Method of Moments (MM) estimation.

MM matches expressions for population moments (means, variances etc.) with their sample analogs. In our setting here, it is quite straightforward to derive closed-form expressions for the estimators:

$$\widehat{\mu} = Y_{\cdot\cdot} \tag{9}$$

$$\widehat{\alpha}_i = Y_{i\cdot} - Y_{\cdot\cdot} \tag{10}$$

$$\widehat{\beta}_j = Y_{\cdot j} - Y_{\cdot\cdot} \tag{11}$$

where $Y_{\cdot\cdot}$ is the overall sample mean in our data, $Y_{i\cdot}$ is the sample mean for user i in our sample, and $Y_{\cdot j}$ is the sample mean for item j.

The missing Y_{ij} are then predicted by

$$\widehat{Y}_{ij} = \widehat{\mu} + \widehat{\alpha}_i + \widehat{\beta}_j = Y_{i\cdot} + Y_{\cdot j} - Y_{\cdot\cdot} \tag{12}$$

4 The rectools Package

This is an R package for recommender systems, developed by the present author and Pooja Rajkumar [4], downloadable from **https://github.com/Pooja-Rajkumar/rectools**. It offers all three categories of collaborative

[4] See [3] for a tutorial showing the use of NMF in other applications. Also, [1] develops conditions under which NMF gives the "correct" factorization, though in practice this is just assumed, and the result is for exact, not approximate, factorization.

filtering methodology discussed here, with some novel enhancements and some entirely new methods. The research proposed in Section 3 will be incorporated into this software (some of it is already in the package in experimental form).

One of the key features of the package is graphics. These are aimed at model checking and exploring possible subpopulations. There are also miscellaneous features, such as a focus group finder, which identifies "typical" users.

References

[1] D. Donoho and V. Stodden, *When Does Non-Negative Matrix Factorization Give a Correct Decomposition into Parts?*, NIPS 2003.

[2] K. Gao and A. Owen, *Efficient Moment Calculations for Variance Components in Large Unbalanced Crossed Random Effects Models*, technical report, 2016.

[3] N. Matloff, *Quick Introduction to Nonnegative Matrix Factorization*, http://heather.cs.ucdavis.edu/NMFTutorial.pdf, 2016.

[4] P. Rajkumar and N. Matloff, Rectools: an Advanced Recommender System, *useR! 2016*.

[5] P. Perry, Fast Moment-Based Estimation for Hierarchical Models, JRSS-B, 2016.

[6] C. Thompson, If You Liked This, Youre Sure to Love That, *New York Times Magazine*, November 21, 2008.