IEOR 242: Applications in Data Analysis, Fall 2019

Homework Assignment #5

November 12, 2019

Problem 1: Collaborative Filtering for Recommending Songs to Music Listeners (100 points)

Recommendation engines are vital to many of today's most successful firms. From Amazon to Netflix to Spotify, companies have realized the value of automatically showing customers products they are more interested in purchasing based on the collaborative evaluations of products by customers, and based also on customer-specific and product-specific data as well. In this problem, we provide you with a dataset consisting of three .csv files, obtained from the Million Song Dataset Project, of derived user ratings of songs based on each user's listening behavior. (Note that users do not actually rate songs – rather the "rating" has been determined from their listening behaviors – but we will still use phrases like "user i has rated song j", etc.) Each user only listens to a few songs, and therefore for each user we only have rating data concerning these few songs. Nevertheless, we would like to infer each user's ratings of all songs based on the entirety of the data. The summary goal of this problem will be to recommend specific songs to specific users.

For this homework, we will use the following .csv files:

- `Songs.csv`: each observation corresponds to a song and it includes the song's ID, name, year, artist, and genre. Observations are sorted by song ID, in ascending order.
- `Users.csv`: a list of all user (listener) IDs, in ascending order.
- `MusicRatings.csv`: derived ratings of songs based on each user's listening history. Each observation contains the user ID, the song ID, and the derived rating.

a) (5 points) Load the data into R. How many songs are in this dataset? How many users? What is the range of values that the ratings take on?

The file `MusicRatings.csv` contains the list of observations for this problem. First set the seed using `set.seed(345)`, then split these observations into four sets:

(a) Training set with 84% of the observations

(b) Validation set A to be used for tuning the collaborative filtering model, with 4% of the observations

(c) Validation set B to be used for blending, with 4% of the observations

(d) Testing set with 8% of the observations

Then, use the `Incomplete` function in the `softImpute` package to construct an incomplete training set ratings matrix.

b) (30 points) Let X denote the "complete" ratings matrix, i.e., $X_{i,j}$ denotes either the observed rating if user i actually rated song j or the "hypothetical" such rating if user i has not yet rated song j. We are interested in predicting the values of $X_{i,j}$ that are not observed. Let us first consider the following model:

$$X_{i,j} = \alpha_i + \beta_j + \epsilon_{i,j} \ , \tag{1}$$

where α_i is a coefficient that depends only on the particular row i (i.e., user), β_j is a coefficient that depends only on the particular column j (i.e., song), and $\epsilon_{i,j}$ is a noise term.

i) (5 points) For this dataset, how many total parameters are included in model (1)? How many observations do we have to train the model with?

The function `biScale` in the `softImpute` package fits a model of the form (1) using a least-squares approach. That is, it solves the following optimization problem:

$$\min_{\alpha,\beta} \sum_{(i,j)\in \mathrm{OBS}} (X_{i,j} - \beta_j - \alpha_j)^2 \ ,$$

where, as in the lecture slides, OBS denotes the set of observed entries in the training set (here we set $X_{i,j} \leftarrow \mathrm{OBSR}_{i,j}$ for all $(i,j) \in \mathrm{OBS}$ that are observed, as was done in the lecture). Use the `biScale` function to fit a model of the form (1). Set the parameters of this function to `maxit` = 1000, `row.scale` = FALSE, `col.scale` = FALSE.

ii) (10 points) Using the output of the `biScale` function (check the documentation to see precisely what this function returns, and you may need to look up the `attr` function), what are the three most popular songs after removing for the bias due to users' affinity for rating songs highly (or lowly)? Provide the song ID numbers as well as the song and artist names for these three songs. Explain, in your own words, how your answer relates to model (1). (Hint: you are going to want to append the computed α and β values to the `Users` and `Songs` data frames, respectively. Note that `softImpute` ensures that each row i corresponds to `userID` i, etc.)

iii) (5 points) Likewise, which three users are the most enthused about songs after removing for the bias due to the effect of the popularity of songs? Provide the user ID numbers of these three users. No need to provide an explanation.

iv) (10 points) What is the out-of-sample performance of the fitted model on the previously constructed test set? Report the test set MAE, RMSE, and the OSR^2 values.

c) (35 points) Now let's consider the following model:

$$X_{i,j} = Z_{i,j} + \alpha_i + \beta_j + \epsilon_{i,j} \ , \tag{2}$$

which is the same as part (*b*) except for an additional term $Z_{i,j}$. Here, Z represents the low-rank collaborative filtering model that we discussed in lecture, i.e., we presume that Z is a matrix with *rank at most* k. Equivalently, the number of archetypes as discussed in lecture is at most k.

i) (10 points) For this dataset, how many total parameters are included in model (2) (the answer should depend on k)? How many observations do we have to train the model with?

ii) (15 points) We will fit the model (2) by using the previously computed estimates of α and β from part (*b*). That is, letting $\hat{\alpha}$ and $\hat{\beta}$ denote these estimates, we will use the `softImpute` function to fit a collaborative filtering model on the incomplete training set matrix of *residuals* $X^C_{i,j} := X_{i,j} - \hat{\alpha}_i - \hat{\beta}_j$. Thankfully, this object has already been returned to us by the `biScale` function in part (*b*).

Using the previously constructed validation set A, determine the value of k, i.e., the number of archetypes, that should be selected. Use a plot to justify and explain the value of k that you selected.

iii) (10 points) Build a final collaborative filtering model on the training set using the previously determined value of k. What is the out-of-sample performance of the fitted model on the previously constructed test set? Report the test set MAE, RMSE, and the OSR^2 values. (Note: the `impute` function will work as expected since it will detect the α and β attributes associated with the model returned by the `softImpute` function.)

d) (30 points) In this part of the problem, we will additionally incorporate some of the features associated with songs.

i) (15 points) First, fit two distinct models to predict the rating based on the following independent variables: *(i)* genre of the song, *(ii)* year that the song was released. Be sure to treat *(i)* and *(ii)* as factors/categorical variables and to build the model using only the same training set as before. Report the test set MAE, RMSE, and the OSR^2 values of your two models.

ii) (15 points) Now, use validation set B to perform blending of the collaborative filtering model (2) trained in part (*c*) and the two models trained in part (*i*) above. Report the test set MAE, RMSE, and the OSR^2 values of the blended model. Do the additional features associated with songs add a lot of predictive power on top of the collaborative filtering model?