Statistical Signal Processing

Final Project

Movie Recommender System

Amir Hossein Afsharinejad

Sheng Zhao

Instructor:

Prof. Davenport

Spring 2014

1 Introduction

The Netflix challenge has brought tremendous amount of interest into the field of recommendation system. In this project, the goal is implement a movie recommender system to predict how users will rate on an unwatched movie. We are given the information on the users, their preference on some other movies, and description of the movies.

There are three common approaches to this problem: collaborative filtering and content-based and hybrid filtering. Collaborative filtering is a method of predicting user's interests by collecting preferences from other users. The underlying assumption is that if person *A* has the same opinion as person *B* on an issue, then A is very likely to have B's opinion on a different issue. Content-based filtering method is based on a description of the item and a profile of the user's preference. It tries to recommend items that are similar to those that a user liked in the past.

In this report, we explored two methods. The first one is a collaborative filtering approach called the matrix factorization. The second one is a weighted hybrid approach, a combination of collaborative and content-based filtering. It incorporates content features into the collaborative similarity matrix.

2 Matrix Factorization

At first, we implemented a simple matrix-factorization-based recommender system using gradient descent. In this approach, we assume that the rating matrix M is a low-rank matrix. Intuitively, this assumption is based on another assumption that only a small number of factors (e.g, genre, released year, user preferences etc.) affects the user to like a movie or not. We define r as the number of factors. Then, an item profile $V \in R^{1682\,x\,r}$ and a user profile $U \in R^{943\,x\,r}$ are learned from the rating matrix M. The estimated rating is equal to the inner product of two length r vectors, one representing user profile and the other representing item profile. The mathematical expression of a predicted rating for user u and movie i is as following:

$$M_{u,i} \approx \sum_{k=1}^{r} U_{u,k}\, V_{i,k} \quad (1)$$

The next step is to fit each element of U and V, which is done by minimizing the squared reconstruction error (SRE) over all training points:

$$SRE(U,V) = \sum_{(u,i)\in M} \left(M_{u,i} - U_u^T V_i\right)^2 = \sum_{(u,i)\in M} \left(M_{u,i} - \sum_{k=1}^{r} U_{u,k}^T V_{i,k}\right)^2 \quad (2)$$

where U_u is the uth row of U and V_i is the ith row of V. Also, we usually add regularization terms to avoid overfitting. Equation (2) looks very similar to linear regression, the only difference is that in linear regression we only have one variable to be learned, while here both U and V need to be learned.

There will be no closed form solution for U and V as they are interrelated. Thus, this is the part where we use gradient descent. The following equations show the implementation:

$$U_{u,k} \leftarrow U_{u,k} - \mu \frac{\delta SRE(U,V)}{\delta U_{u,k}}$$
$$V_{i,k} \leftarrow V_{i,k} - \mu \frac{\delta SRE(U,V)}{\delta V_{i,k}} \quad (3)$$

where the update rate is determined by μ. So, first we take the partial derivative of SRE in (2) and then update U and V using (3). After doing the calculations, our final update formula will be as following:

$$U_{u,k} \leftarrow U_{u,k} + 2\mu \left(\sum_{i|(u,i)\in M} e_{u,i} * V_{i,k} \right) - 2\lambda U_{u,k}$$

$$V_{i,k} \leftarrow V_{i,k} + 2\mu \left(\sum_{i|(u,i)\in M} e_{u,i} * U_{u,k} \right) - 2\lambda V_{i,k} \quad (4)$$

At the end of each iterative gradient descent loop, the RMSE is measured. If the change in RMSE is smaller than a threshold ε, then convergence has been reached and the iteration can stop.

3 Hybrid Approach

3.1 Collaborative Filtering

The general idea of our project is to implement one of the most popular non-probabilistic collaborative filtering methods which is the Nearest Neighborhood (NNH) algorithm. There are two basic NNH techniques: 1- user-based nearest neighborhood 2- item-based nearest neighborhood.

User-based methods find the most like-minded users for a given user and then use their ratings to predict the rating for active user on an unseen item. To get more precise results, weights are assigned to ratings of nearest neighbors; i.e. most similar users . These weights are set according to the similarity of neighbors to the given user. Item-based nearest neighbor algorithms are

similar to user-based ones. The difference is that we estimate the ratings based on the similarities between items and not users.

3.2 Feature Combination

Before we talk about how to calculate similarities and to predict missing ratings, our approach needs to take into account the movie and user features to be Hybrid; i.e. to use collaborative filtering as a base and then appending content-based information. For this purpose, we want to combine the original rating matrix with all extracted feature information in a single model (two matrices). Figure 1 illustrates how to extend the matrix into two matrices with combined movie ratings and features [2].

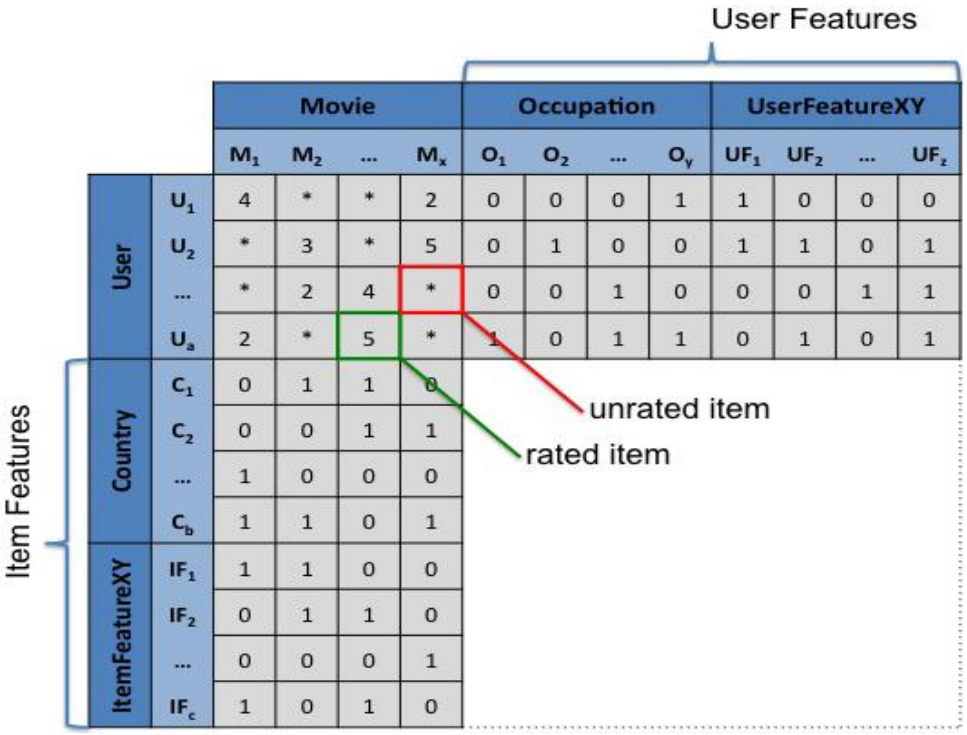

Fig.1- 2D-Matrix with User and Movie Features

As one can see, two matrices are generated, one for user and one for movie. The user matrix is made from horizontally concatenating the 943x1682 ratings matrix and the 943x5 user data. The movie matrix is made from vertically concatenating the ratings to the 23x1682 movie data.

3.3 Reformat Data

Since the user and movie data have non-numerical values such as user's occupation, gender, or movie's release day, we need to reformat the data into binary numbers before we can extend the matrix as in section 3.2. We do this by generating vectors of zeroes and ones similar to what we have in movieData matrix for genre data.

In reformatted user data, gender is made into the male and female column vectors. Age is separated into four age groups. The underlying assumption is that people in a similar age group are likely to have similar preferences on movies. The zip code is not considered for user data. Despite neighboring zip code often means the two regions are physically close to each other, it is difficult to assume that the neighboring zip code number mean similar socioeconomic status or preference on movies.

Table.1- Reformatted User features

Gender		Age				Occupation		
Male	Female	0-17	18-30	31-50	>50	artist	lawyer	etc.
1	0	0	1	0	0	0	0	1
0	1	1	0	0	0	0	1	0
1	0	0	0	1	0	1	0	0
0	1	0	0	0	1	0	1	0

In movie data, the release date is converted to release year for simplicity. Then, the year is categorized into four groups. The boundary is determined by looking at the distribution of the years. The cutoff is made to have similar number of movies released in each group. The movie title and URL are not considered due to complexity of parsing words and putting them into categories. The genre is not modified as it meets the format requirement of our method.

Table.2- Reformatted Movie features

Year				Genre		
<40s	40s-70s	70s-90s	>90s	Action	Drama	etc.
0	1	0	0	0	1	0
1	0	0	0	1	0	0
0	0	1	0	1	1	0
0	0	0	1	0	1	1

3.4 Similarity Matrix

Now, using the knowledge from the "3.1 Collaborative Filtering" section and rows of user matrix, we can generate user-user similarity matrix using Equation 5, where element (i,j) is the similarity between user i and user j and $vec(i)$ is the ith row of the user matrix.

$$Sim(i,j) = \frac{vec(i) \cdot vec(j)}{\|vec(i)\| \, \|vec(j)\|} \quad (5)$$

To reduce the computation time for generating the similarity matrix, we explored the symmetry in the matrix. This matrix is symmetric $Sim(i,j) = Sim(j,i)$, and we set the diagonal $Sim(i,i)$ to 0 because the similarity of a user/movie with itself is meaningless. The movie-movie similarity matrix is generated in the exact same way, except for the fact that we use movie matrix instead and $vec(i)$is the ith column of movie matrix.

3.5 Prediction

For each missing rating in the original ratings matrix, we will make a prediction based on the formula below:

$$pred(u,i) = \frac{\sum_{n \in neighbors(u)} Sim(u,n) \cdot r_{ni}}{\sum_{n \in neighbors(u)} |Sim(u,n)|} \quad (6)$$

If we are looking for how user *u* is going to rate movie *i*, we find K other users who have highest similarity with user *u* and look for their opinions on the movie *i*. The weight of other user's rating depends on how similar each user is to user *u* ($Sim(u,n)$ in (6)). More similar a user is to user *u*, higher the weight of that user's rating. Also, r_{ni} is the rating one of the most similar users n has provided for movie *i.* We perform similar prediction algorithm to movie-movie-similarity matrix. So now, we have generated two prediction matrices, one from user-user similarity matrix, and the other from movie-movie similarity matrix. The final prediction will be a weighted version of both predictions that minimizes RMSE.

3.6 Bound the prediction

Since the rating prediction should range between 1 and 5, any predicted rating outside the range are adjusted accordingly. From experimentation, our pre-bounded result's minimum is 0.999 and maximum is 5.0, so the bound does not significantly improve/change performance.

4 Results

4.1 Accuracy Measurement

The metric to measure the accuracy of the predicted ratings is root-mean-square error (RMSE). Lower the RMSE, better the prediction.

$$RMSE = \sqrt{\frac{\sum_n (\hat{\theta}_n - \theta_n)^2}{n}} \quad (7)$$

θ_n : the nth observed rating.

$\hat{\theta}_n$: the nth predicted rating.

n : the number of non-zero ratings.

4.2 Dataset

In the given dataset, there are 80,000 nonzero ratings (one to five stars) from 943 users on 1682 movies in a 943 by 1682 matrix M called the ratings matrix. This matrix is sparse in nature, as only around 1% of the cells in it are observed. Now, the objective is to predict ratings for users on unseen movies; i.e. the other 99% of the values in M. We are also given a 943 by 5 user data matrix containing the user profiles and a 1682 by 23 movie data matrix containing movie features.

4.3 Matrix Factorization

As Matrix Factorization method is a collaborative filtering method, it only uses the ratings matrix. In this algorithm, we randomly select 64,000 nonzero ratings (out of 80,000) and use them as training data and also the 16,000 nonzero ratings left would contain our testing data. The learning rate μ is set to 0.0005 and regularizer λ is set to 0.01.

Table 3- Matrix Factorization results (RMSE)

Set/Low Rank	1	3	5	7	10
Train	0.9143	0.8803	0.8648	0.8588	0.8684
Test	0.9606	0.9577	0.9583	0.9569	0.9582

4.4 Hybrid

In the hybrid method, all three matrices are needed. We randomly select 56,000 nonzero ratings as the training set, 12,000 nonzero ratings as the holdout set, and 12,000 nonzero ratings as the testing set. We have determined the optimal parameters and weights from the holdout set. The weight distribution for age, gender and occupation is 3,4 and 5 respectively. The higher the number, heavier the weight. We anticipate the user's occupation has higher impact than the gender. The weight distribution for year is 7 and for genre is 5. Finally, using the holdout set we found out that the nearest neighbor K=16 produces the best results.

The final prediction is composed of predictions from user-user and movie-movie similarity matrices. From experimenting with the weights, having 70% from the user prediction and 30% from the movie prediction yields the smallest RMSE of **0.9306**.

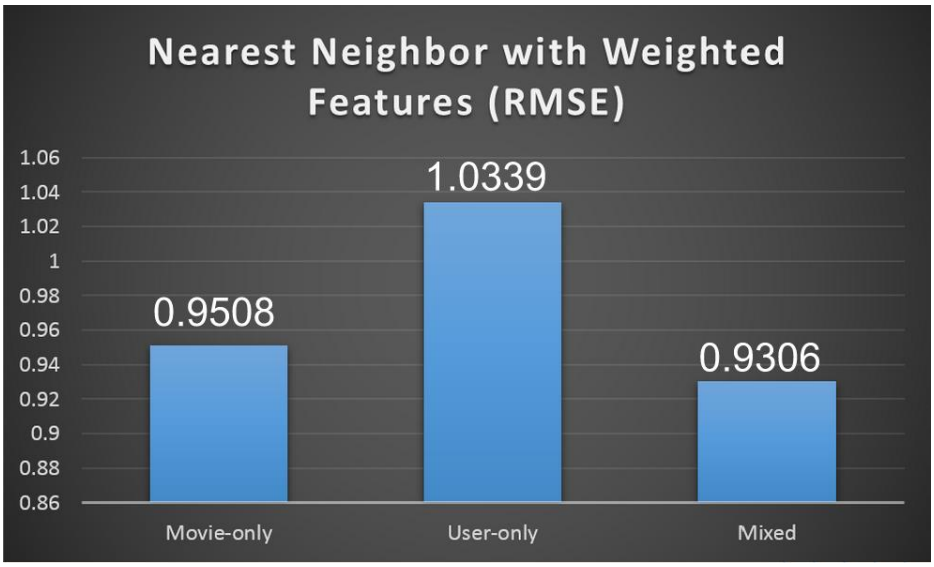

Fig.3- NNH with weighted Features RMSE

5 Conclusion

In the report, we explored the existing matrix factorization method and our customized hybrid method, which combines collaborative and content-based filtering. While both methods yield reasonable results, there are advantages and disadvantages in both methods.

In matrix factorization, it only needs the ratings matrix. When movie features or user profiles are not available, matrix factorization really shines. However, when there is additional movie or user information, it is not capable of incorporating those information. Computationally, the iterative gradient descent requires tremendous amount of time to execute.

The hybrid method is where we put majority of our effort in. We have created the user and the movie similarity matrix. Applying nearest neighbors on the similarity matrices, we were able to make two predictions, one from the user similarity matrix and the other from the movie similarity matrix. Then, the final prediction is a weighted combination of prediction from the movie and prediction from the user. This weighted prediction yield results much better than using the movie or user data alone.

6 References

[1]- Badaro, G.; Hajj, H.; El-Hajj, W.; Nachman, L., "**A hybrid approach with collaborative filtering for recommender systems**," Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International , vol., no., pp.349,354, 1-5 July 2013

[2]-Spiegel.S: "**A Hybrid Approach to Recommender Systems based on Matrix Factorization**",Technical University of Berlin