Evaluation of Collaborative Filtering Recommendation Algorithms

Andrey Vasilyev and Amar Depa

CSci 724- Artificial Intelligence

ABSTRACT

Collaborative filtering is a rapidly advancing research area. Every year several new techniques are proposed and yet it is not clear which of the techniques work best. So in our project we built recommendation system using two popular collaborative filtering algorithms - User based and Item based collaborative filtering algorithms. We measured the accuracy and efficiency of these algorithms using different error and accuracy metrics. For comparison of the results we considered two approaches- Cosine Vector Similarity and Pearson Correlation Coefficient. Our conclusions based on the results of the recommendation algorithms identify which algorithm works best under what conditions.

1. INTRODUCTION

In our day to day life we come across options and choices. What to eat? Which movie to watch? Which camera to buy? With the massive amount of increasing information in the world, decision making is also becoming tedious job to accomplish. Without having prior domain knowledge, it is difficult to make a choice. Thus we see people relies on recommendations from their friends or respective domain people. Technology has reduced barriers to distribute information. It can help us to find the appropriate choice among huge available information.

One of such promising technology is collaborative filtering based recommendation system [1]. Collaborative filtering works by building a database of choices of items by users. A new user, John, is matched against the users in the database to find neighbors which had similar taste to John. Then items liked by neighbors are recommended to John, as he will probably like them too. Collaborative filtering has been very successful in both research and practice. However, there is still important question remained, which of the collaborative filtering techniques works best under what conditions. One difficulty in finding this is that the performance of different methods differs substantially based on different parameters. Typically, factors like number of users, number of items, sparsity level affect different collaborating filtering methods in a different way.

In our project, we address this issue by building and evaluating performance of recommendation systems by applying item based and user based collaborative filtering algorithms on movie data set. In recommendation system for finding neighbors it uses techniques of finding similarity between users or items. We followed two approaches Cosine vector similarity and Pearson correlation coefficient for finding similarity between users or items. Finally, we compared results of two approaches with the help of multiple evaluation measures.

2. ORGANIZATION

The rest of the report is organized as follows. The next section provides the related work done in the area of recommendation systems and typically collaborative filtering recommendation system. Section 4 provides a brief background in collaborative filtering algorithms. We describe collaborative filtering process and then discuss its two types memory based and model based approaches. Section 5 provides experimental evaluation, details of our dataset, evaluation metrics and two similarity functions Cosine vector similarity and Pearson correlation coefficient. After that Section 6 provides out experimental results for prediction and recommendation based on two algorithms user based collaborative filtering (UBCF) and item based collaborative filtering (IBCF) also includes discussion of the results. Final section

7 provides concluding remarks about things we learned, problems we faced and how we solved those problems.

3. RELATED WORK

Earliest implementations [1] of collaborative filtering-based recommender systems relied on the explicit opinions of people from a close community, like office workgroup. However, recommendation systems for large communities cannot rely on each person knowing the others. Later many rating based automated recommendation systems were developed. The Grouplens research lab is the example of that which produces collaborative filtering solutions for movies [1]. Widespread use of recommendations has exposed some of their limitations like problem of sparsity in dataset, problems with high dimensionality. These problems are addressed by several research papers [1].
There are couple of experimental studies available. Breese and team compared memory based and model based methods on three different datasets [2]. A more recent experimental comparison of collaborative filtering algorithms compared user-based, item-based collaborative filtering and SVD focusing on e-commerce applications. They considered precision, recall, F1-measure and rank score as evaluation measures. However, they ignored some of standard evaluation metrics like RMSE or MAE [3].
In our project we tried to explore the extent to which user based and item based recommenders are able to solve these problems.

4. COLLABORATIVE FILTERING ALGORITHMS

Recommendation systems perform data analysis and help users to find items or users that they might be interested in. It is done by either producing predictions or list of Top-N recommended items for a given user. Collaborative filtering (CF) is the most successful recommendation technique [1]. The key idea of CF based algorithms is to provide recommendations based on the opinion of other like-minded users.
In a particular CF based scenarios there is a list of m users U= $\{u_1, u_2, \ldots u_m\}$and a list of n items I = $\{i_1, i_2, \ldots i_n\}$. Each user u_i has expressed his opinions about list of items $I_{ui.}$ Opinions given by user could be in the form of ratings generally in certain numerical scale. Here I_{ui} is subset of I and I_{ui} could hold null value too. An active user u_a is the user who belongs to U for whom the job of collaborative filtering algorithm is to find an item likeliness that can be of two forms.

- **Prediction**: It is a numerical value which shows the predicted likeliness of item for active user. This predicted value falls in the same range as opinion values provided by active users (for example from 1 to 5).
- **Recommendation (Top-N):** It is the list of N items that the active user will like the most. Here recommended list must be on items not already purchased by active user. This form of CF algorithms is also called as Top-N recommendation.

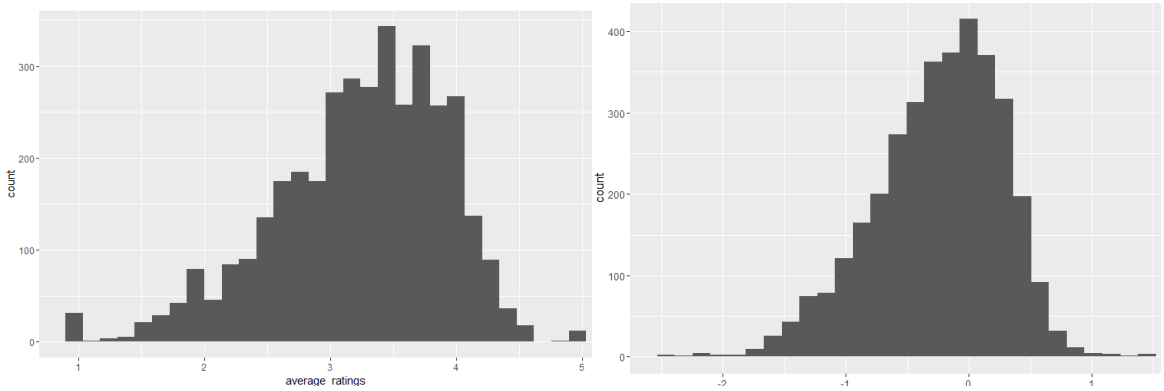
Number of collaborative filtering algorithms are broadly classified into two categories. Memory based (user based) filtering algorithms and model based (item based) filtering algorithms [1].

5. EXPERIMENTAL EVALUATION

5.1 Dataset: In order to know how accurate our recommendation algorithms and their predictions we conducted experiments. We used movie ratings data set [8]. We preferred movies which are rated by more than 50 users and users who rated more than 100 movies to get more promising results. Our input file had CSV format and consisted of three columns: user ID, movie ID and rating. We trained our model with 90% of the data and predicted for remaining 10%. When we obtained predicted results we could compare them with actual values and determine the accuracy of our recommendation algorithms.
5.2 Challenges of CF algorithms: The main challenges we faced were data sparsity, scalability and bias ratings.

- **Data sparsity:** It is related with the fact that users cannot provide recommendation for every single movie and when the list of rating is converted into matrix it can have some empty ratings. The matrix with sparse data takes more memory and, as result of this, more processing time for algorithm to generate results. To resolve this issue we used "Real Rating Matrix" class from "Recommenderlab" package in R studio to store the matrix in compact format.
- **Scalability:** As for data scalability, it is usually caused by increasing number of users and movies that can happen in any online recommendation system. When more users and movies are added to the system, it takes more time for recommendation algorithms to process data and provide predictions. Since we specified the date ranges of 5400 users and 3500 movies on our data set we were able to limit scalability of our data within predefined ranges.
- **Biased Rating:** In addition to all this, our initial data set MovieLens contained bias ratings, caused by users who consistently provided only low or high ratings to all of their movies in comparison with all other users. In order to improve performance and accuracy of our recommendation algorithms we had to remove this rating bias by normalization process.

The normalization is calculated by means of the formula: $h(r_{ui}) = r_{ui} - \bar{r}_u$, here $\bar{r}_u$ is the mean of all available ratings in row u of the user-item rating matrix R. According to this formula, mean average of all ratings is subtracted from the rating of every individual user. To normalize our data set we used "Z-score" method which, in addition to the standard formula, to reduce variance further. Z-score method is calculated as (actual rating – mean of all available ratings)/standard deviation. Before normalization we built graph (left hand side graph below) based on average ratings so that we could see outliers and distribution of ratings on the scale from 1 to 5. Outlier is a value that "lies outside" (is much smaller or larger than) most of the other values in a set of data. We can see the spikes around the rating 1 and between the ratings from 3 to 4. In order to get rid of such spikes (i.e. bias ratings) we normalize our data by using Z-score method from "Recommenderlab" package. Right hand side graph shows data after normalization without outliers.



5.3 Similarity Measures: In recommendation system for finding neighbors it uses techniques of finding similarity between users or items. We followed two approaches Cosine vector similarity and Pearson correlation coefficient for finding similarity between users or items.

5.3.1 Cosine Vector Similarity: In case of Item Based Algorithm, Cosine similarity represents two items as two vectors in the m dimensional user-space. The similarity between them is measured by computing the cosine of the angel between vectors ***i*** and ***j*** based on the formula [4]

$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{||\vec{i}||_2 * ||\vec{j}||_2}$$

The cosine similarity between users is calculated in a similar way. The similarity between two users $\boldsymbol{u_x}$ and $\boldsymbol{u_y}$ are defined as:

$$\text{sim}_{\text{Cosine}}(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\|\|\vec{y}\|}$$

where $\vec{x} = r_x$ and $\vec{y} = r_y$ represent the row vectors in R with the two users' profile vectors. These measure provide values in the range from 0 to 1. If the value gets closer to 1 then similarity between entities gets higher.

5.3.2 Pearson Correlation Coefficient: Pearson similarity is calculated between two variables as the covariance of the two variables divided by the product of their standard deviations.

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}$$

It is a measure of the strength of a linear relation between 2 variables. When the value is zero then there is no relationship between the variables. When it approached -1 or +1 then there is a positive or negative relationship between the variables respectively. That means positive value indicates similar interest (taste) and negative value indicated dissimilar interest (taste).

6. EXPERIMENTAL RESULTS

For this experiment we divided our data set into training and test data set. In this section we present our experiment results by applying UBCF and IBCF algorithms on the movie data set mentioned in the previous section. We used two similarity methods for comparison. Our results are divided into two sections. In first section Prediction, we considered MAE as a measure to compare the predicted ratings accuracy. In second section we compare the recommended items using precision and recall as well as using area under curve.

6.1 Predictions:

We predicted ratings for the movies to active user using both user based collaborative filtering algorithm (UBCF) and item based collaborative filtering algorithm (IBCF) based on two similarity methods (Cosine vector similarity and Pearson correlation coefficient). We used 10 fold cross validation and computed average error across all 10 folds. The advantage of this method is that every data point gets to be in a test set exactly once, and gets to be in a training set k-1 times. We obtained result of the predictions in the form of real rating matrix.

After prediction we measured how accurate our predictions were for User based and Item based algorithms using mean average error (MAE) using two similarity measures Cosine and Pearson. We chose MAE over mean square error (MSE) and root mean square error (RMSE) because MSE and RMSE tend to exaggerate the effects of possible outlier whereas, MAE is less sensitive to outliers

```
> rownames(error) <- c("UBCF-Cos","UBCF-Cor","IBCF-Cos","IBCF-Cor")
> error
              RMSE      MSE       MAE
UBCF-Cos 1.022681 1.045877 0.8159263
UBCF-Cor 1.025515 1.051680 0.8177486
IBCF-Cos 1.143625 1.307879 0.8249689
IBCF-Cor 1.368528 1.872868 1.0050422
>
```

From the error metrics looking at MAE, UBCF performed very well compared to IBCF for both similarity methods Cosine vector (cos) and Pearson correlation (cor).

6.2 Recommendations

6.2.1 Computation time: We recommended top-N from 10 to 100 movies to active user that he will like the most. We compared performance of UBCF and IBCF algorithms in the form of time taken by these two algorithms to build the model and predict the recommendations. In order to build the model we used 90% of the data and predicted for 10% data.

From above image we can observe that IBCF algorithm took more time for modeling whereas UBCF algorithm took more time for prediction. Hence we can conclude that, when we have more number of items to recommend, IBCF is better choice even though it takes more time for modeling it will take less time for recommendation. On the other hand, if we have less number of items to recommend UBCF is the better choice

6.2.2 Confusion Matrix: We first determined the confusion matrix to compare the performance of item based and user based collaborative filtering algorithms based on two similarity methods to recommend movies. Confusion matrix is a table, which contains true and false positives (TP, FP), true and false negatives (TN, FN), precision, recall, false positive rate (FPR), true positive rate (TPR).

Method	**Precision**	**recall**	**FPR**	**TPR**
UBCF_Cos_1	0.464	0.005	0.005	0.0003
UBCF_Cos_100	0.23	0.22	0.22	0.046
UBCF_Cor_1	0.38	0.004	0.004	0.0003
UBCF_Cor_100	0.205	0.189	0.189	0.0479
IBCF_Cos_1	0.208	0.002	0.002	0.0004
IBCF_Cos_100	0.17	0.15	0.15	0.05
IBCF_Cor_1	0.27	0.003	0.003	0.0004
IBCF_Cor_100	0.121	0.099	0.099	0.047

From the above table we can observe that user based collaborative filtering algorithm (UBCF_Cos_1) using cosine method has more precision and recall when we recommend less number of movies(Top 1) to the user. As well as when we recommend more number of movies (Top 100) also the user based collaborative filtering algorithm using cosine similarity method (UBCF_Cos_100) has more precision and recall.

6.2.2 Recommendation Results: Figure 4 shows the graph for precision against recall using two similarity methods (Cosine vector similarity (cos) and Pearson correlation coefficient (cor)). Labels on the curve represent the number of movies recommending to the user. From the graph it can be observed that user based collaborative filtering algorithm has more precision and recall than item based collaborative filtering algorithm whether we recommend less number of movies such as 1 or more number of movies such as 100. Thus user based collaborative filtering algorithm with cosine distance outperforms the all other algorithms.

Figure 4: Precision vs Recall

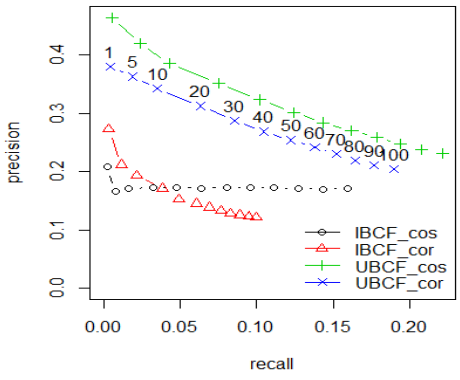


Figure 5: ROC Curve

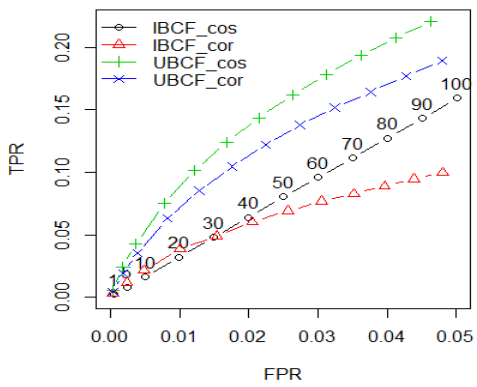


Figure 5 represents the graph for false positive rate (FPR) vs true positive rate (TPR) using two similarity methods (Cosine vector similarity (cos) and Pearson correlation coefficient (cor)). We can also call this as ROC curve. Labels on the curve represent the number of movies recommending to the user. In this we use area under curve (AUC) to compare models. From the graph we can observe that the user based collaborative filtering algorithm with both similarity methods has more Area under the curve than Item based collaborative filtering algorithm.

6.3 Discussion: According to our experimental results UBCF is better than IBCF in terms of error metrics, and accuracy metrics. Then why preference is given to IBCF instead of UBCF in real world recommendation systems? The answer is when and how the recommendations are generated. UBCF saves the whole matrix and then generates the recommendation and predict by finding the closest user. IBCF saves only k closest items in the matrix and doesn't have to save everything. It is pre-calculated and predict simply reads off the closest items [9].

7. CONCLUSION

While doing research for collaborative filtering algorithms we faced different challenges. The first challenge was data preparation and analysis. We had to figure out how to remove data sparsity, get rid of bias rating and split out data for training and prediction. The next challenge was to build model with appropriate parameters and amount of data we considered. Working on big data was the challenge as it was taking ample amount of time to run every possible combination of input parameters for our algorithms. Beside this we had to figure out how to interpret results using different measures. Built-in packages in R studio helped us to overcome these challenges effectively. We were able to visualize our data and perform normalization to solve problems related to dataset. We learned the concepts of recommendation systems algorithms, different measures to use for evaluation of models built from a very good textbook on building recommendation system [11]. After many trials of inputs to the parameters we were able to figure out optimal parameters for two of our algorithms that we built and analyzed. During this whole process we not only learned very basic concepts on how recommendation system works but also we were able to implement functional recommendation system and provide convincing analysis of our experiments.

8. REFERENCES:

[1] http://files.grouplens.org/papers/www10_sarwar.pdf

[2] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering, Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, 1998

[3] Z. Huang, D. Zeng, and H. Chen. A comparison of collaborative filtering recommendation algorithms for e-commerce. IEEE Intelligent Systems, 22:68-78, 2007

[4] Badrul Sarwar, George Karypis (2001). ItemBased Collaborative Filtering Recommendation Algorithms. Scientific Article, GroupLens Research Group, pp. 4

[5] Michael Hahsler. Recommenderlab: A Framework for Developing andTesting Recommendation Algorithms. Tutorial, Southern Methodist University, pp. 5

[6] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1):5–53, 2004

[7] Paolo Cremonesi, Roberto Turrin. (2010) An evaluation Methodology for Collaborative Recommender Systems. WHITEPAPER March 2010. pp – 5

[8] https://inclass.kaggle.com/c/predict-movie-ratings/data

[9] http://www.r-bloggers.com/testing-recommender-systems-in-r/

[10] RecommenderLab 39 pages

[11] rc book