Using Social Networks to Improve Movie Rating Predictions

Suhaas Prasad

Introduction

Recommender systems based on collaborative filtering techniques have become a large area of interest ever since the Netflix Prize was announced, in which the system must be able to predict how a user will rate a movie based on the rating histories of that user along with many others. These types of recommendation systems are become ever more popular throughout web-based applications, including e-commerce, news, video streaming, and similar sites. However, it does not appear as though any of these systems have attempted to incorporate a user's social network in determining how they might rate a movie.

The collaborative filtering approach attempts to learn past user-item relationships by treating the problem as a large-scale data mining problem. A common collaborative filtering method is the k-nearest neighbor (kNN) method in which the user-item preference is determined by looking at the ratings of similar users or items. While most neighborhood based method for the Netflix challenge find the item-based approach a better option, in order to incorporate social networks, the approach outlined in this paper uses user similarity to predict user-item preferences. In addition, several methods for using a user's friend network to weigh user similarities and their results are described. The data for the users ratings and social networks have been provided by Flixster, a social movie platform where users can rate and review movies with their friends.

To formally describe the problem, given a set of users, *U*, movies, *M*, a set of training examples of the format (user, movie, rating), and a set of user connections (user1, user2), the task is to make a prediction $P_{u,m}$ for a rating for a user and movie. Then the social network is represented by the matrix $\boldsymbol{A}$, where $A_{u1,u2}$ is 1 if there exists an edge between user u1 and u2 (i.e. u1 and u2 are friends) and 0 otherwise. Moreover the ratings matrix $\boldsymbol{R}$, holds the user-movie ratings, where $R_{u,m}$ indicates the rating given by user u for movie m, which is undefined if no rating exists in the training set and is on a 1 to 5 scale otherwise. The effectiveness of an algorithm given a training set is based on the root mean square error (RMSE) on a test set:

$$rmse = \sqrt{\frac{1}{|S_{\text{test}}|} \sum_{(m,u) \in S_{\text{test}}} (R_{m,u} - P_{m,u})^2}$$

where $|S_{test}|$ is the number of test cases (m,u) in the test set *S*.

K-Nearest Neighbor

The premise of the k-nearest neighbor approach is that similar users will rate items similarly and similar items will be rated similarly; however, we will only be examining the user-user similarities in the following kNN implementation. The method assumes that user rating sets are independent of each other, so one might average the ratings of similar users for a movie to predict how a user would rate that movie. Rather than simply average the ratings, kNN uses a weighted average, where the weights are given by how similar the users are. Therefore, the predicted rating is given by:

$$P_{u,m} = \frac{\sum_{v \in U_m^k} sim(u,v) R_{v,m}}{\sum_{v \in U_m^k} |sim(u,v)|}$$

where U_m^k is the set of *k* users most similar to user u that have rated movie m, and sim(u,v) is the similarity measure of users *u* and *v*. The two most common similarity measures are the cosine similarity and Pearson's correlation coefficient, both of which are implemented and compared here. Cosine similarity is defined as:

$$sim(u,v) = \frac{\sum_{m \in M} R_{u,m} \times R_{v,m}}{\sqrt{\sum_{m \in M} \left(R_{u,m}\right)^2} \times \sqrt{\sum_{m \in M} \left(R_{v,m}\right)^2}}$$

while the closely related Pearson's coefficient ρ_{ij} is defined below:

$$\rho_{ij} = \frac{E[(R_i - \mu_i)(R_j - \mu_j)]}{\sigma_i * \sigma_j}$$

$$E[(R_i - \mu_i)(R_j - \mu_j)] \approx \frac{1}{M} \sum_k (r_{ik} - \mu_i)(r_{jk} - \mu_j)$$

$$\sigma_i \approx \sqrt{\frac{1}{M} \sum_k (r_{ik} - \mu_i)^2} \ , \ \sigma_j \approx \sqrt{\frac{1}{M} \sum_k (r_{jk} - \mu_j)^2}$$

where μ_i is the average rating given by user i. The values of both cosine similarity and Pearson's coefficient lie in the range [-1,1], but can be converted to the [0,1] range if

necessary. The two similarity measures essentially determine how linearly related the two ratings distributions are.

Incorporating Social Networks

In addition to the traditional kNN, in which the similarity of two users is determined by their ratings histories, one new approach examined involves increasing the similarity if the two users are friends. The premise here is that users who are friends with each other will tend to rate similar movies similarly. Therefore the new similarity *s* given sim(u,v) from the previous section can be defined as:

$$s = sim(u,v) + \left(1 - sim(u,v)\right) \times w$$

where *w* and sim(u,v) lie on the range [0,1] and w = 0 if u,v are not friends and > 0 otherwise. This essentially adjusts the similarity closer to 1 if users u and v are friends.

Another approach investigated involves interpolating the ratings of a user's extended friends network. Rather than use kNN, this method looks only at the ratings of users within three degrees of separation from the user and averages the ratings of those users:

$$P_{u,m} = \frac{1}{|F(u)|} \sum_{v \in F(u)} R_{v,m}$$

where F(u) is the set of extended friends of user u that have rated movie m.

kNN Post Processing

Some of the modifications for kNN that are necessary to improve the RMSE include addressing special cases such as users with no recorded ratings, rounding predictions that are close to an integer, and more.

For the case of newcomers, in which a user in the test set has no recorded ratings in the training set, the traditional approach is to use the average rating for that movie, since the similarity measure is useless:

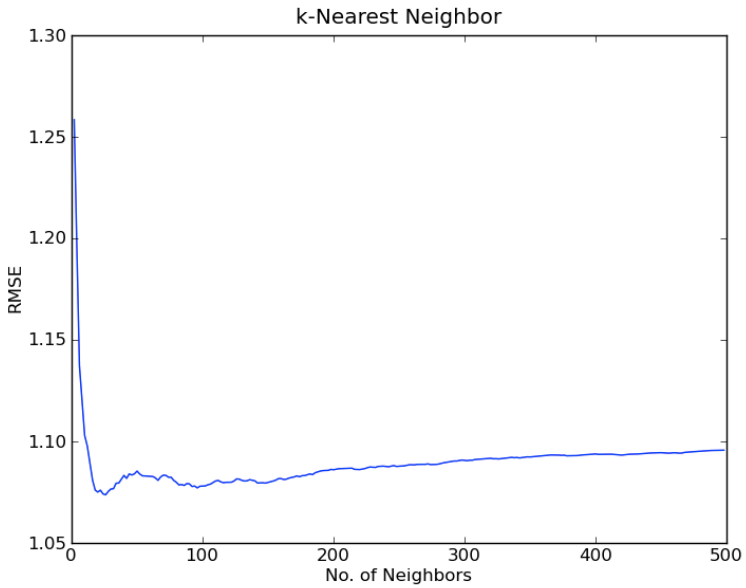
$$P_{u,m} = \frac{1}{|U(m)|} \sum_{v \in U(m)} R_{v,m}$$

where U(m) is the set of users that have rated movie m. However, in this case, we can use the user's friends to gather a set of similar users from which to take an average rating.

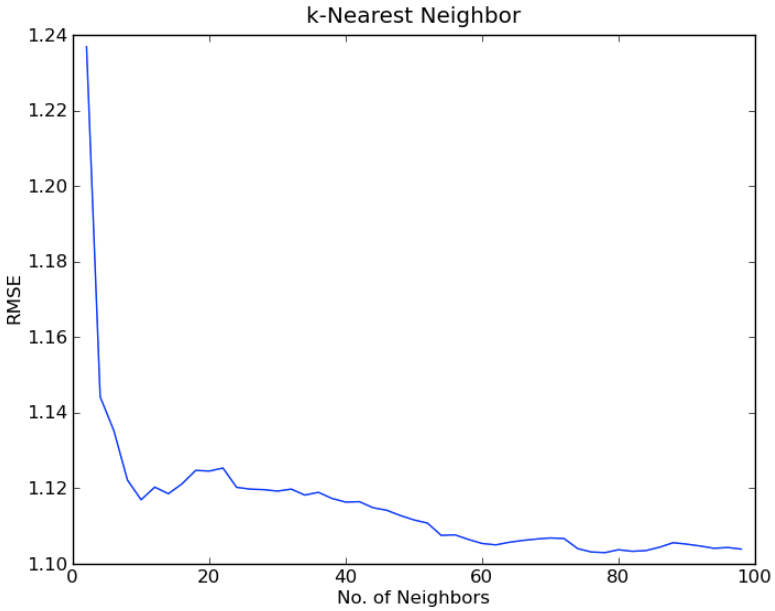
Results

Using the Flixster data with 8000 users and 5600 movies, the results of running the kNN algorithms with the following parameters are as follows. For the baseline item-based kNN, the RMSE seen was approximately 0.989. Using this as a comparison for the user-based method, we see that item-based kNN far outperforms any user-based approach.

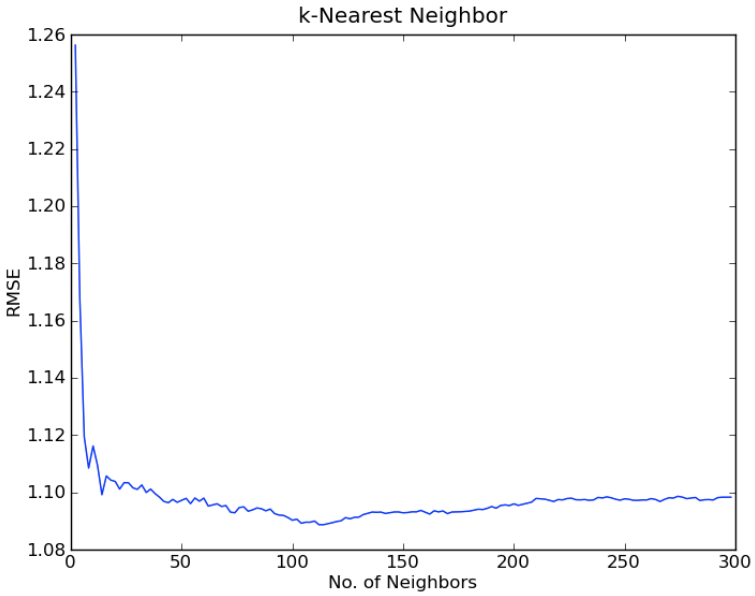
Pearson Coefficient:



Cosine Similarity:



Pearson /w Friend Weighting of 0.25



As far as can be seen, increasing the weight of friends' ratings in kNN does not seem to have any significant effect on the RSME. The use of Pearson's coefficient clearly surpasses that of the cosine similarity, and post-processing tricks appear to only slightly increase the RSME. Moreover the approach that interpolates one's friends ratings rather than using kNN performs significantly worse with an RSME of 1.48.

Conclusion and Future Work

While the results show no improvement by taking one's social network into account, the results are not conclusive. The similar results with and without increasing the similarity of users who are friends may be due to the sparseness of the social graph in the data. Given that the average degree of the network is 4.38, the principles of kNN would likely overshadow the use of a user's friends' ratings. Moreover, Flixster users may not necessarily represent one's true social graph. A better approach would be to use Facebook users who use the Flixster network; however, this data was unavailable due to Facebook restrictions. One of the primary goals for future work would be to construct a data set of only socially active users to see if there is a significant increase in accuracy when testing on these users. Another goal is to try blended models using item-based collaborative filtering methods in addition to these user-based methods that take a user's social graph into account.

References

1. J. Golbeck, J Hendler, "FilmTrust: Movie Recommendations using Trust in Web-based Social Networks", Proceedings of the IEEE Consumer Communications and Networking Conference , January 2006.
2. Z. Wen, "Recommendation System Based on Collaborative Filtering", Stanford CS229 Projects, 2008.
3. T. Hong, D. Tsamis, "Use of KNN for the Netflix Prize", CS229 Projects. 2006.
4. Y. Zhou, D. Wilkinson, R. Schreiber, R. Pan. "Large-Scale Parallel Collaborative Filtering for the Netflix Prize", AAIM 2008: 337-348.