Movie Recommendation System

Group 24
Ashudeep Singh
Shashwat Chandra
Gidijala Chandra Sekhar
Kamlesh Verma

1 Introduction

Recommendation systems are special types of expert systems in the sense that they combine the knowledge of the expert in a given domain (for the product type being recommended) with the user's preferences to filter the available information and provide the user with the most relevant information. Personalization of the recommendations works by filtering a candidate set of items (such as products, movies, web pages) through some representation of a personal profile. A content based recommendation system uses the user's past history to recommend new items where as a collaborative approach uses the preferences of other people with similar tastes for recommending items to the user. [**?**]

The project selected by our group involves building a system that uses the large *movielens* dataset to predict for an unknown user whether he is likely to see a given movie, and if yes, the rating the user will give the movie.

To solve these problems, we decided to divide the problem into two parts. The first involves finding the similarities in whether a user will see a movie or not. Since it is not accurate to assume that the user doesn't want to see any of the movies he hasn't rated, we use a binary classification based on the rating given by users in the Learning Set to a movie they saw. This classification basically divides the movies watched by users into two categories:

- Movies liked by the user (A similar user is more likely to see this movie)
- Movies disliked by the user (A similar user is less likely to see the movie)

The second part of the problem takes a *regression* approach. We shall consider the actual rating given to a movie, and use that to categorize both whether a new user will watch a given movie, and what rating he may give it. This will be a better model to consider cases where a new user would want to see a movie that he is sure he dislikes (for example, maybe because he sees all movies starring a particular actor).

2 The Dataset

The MovieLens dataset was collected by GroupLens Research Project at University of Minnesota [1][2]. The dataset consists of:

- 100k ratings between 1 and 5 from 943 users for 1682 movies.
- Each user has rated atleast 20 movies
- Simple demographic features for users (age, gender, occupation, zip)
- Meta-information about the movies (genre, release date, title)
- A rating given by the user to the movie. This was on a scale of 1 to 5; 1 being the worst rating.

The hundred thousand rating set consisted of just around 6% of the User x Movies space.
The mean of the ratings is 3.53 and the standard deviation about the mean is 1.12.
The dataset provided also has 5 seperate sets each to be used while training and testing seperately.

3 Preprocessing

Our task was to predict whether a new user shall watch a movie or not, and then rate the movies he was likely to watch. The only information given to us regarding the new user were his/her demographics: the age, the gender, the zip code, and the occupation. To use these demographics, the following processes were applied on the training and test sets:

1. The age of each user was normalized to a value from 0 to 1.
2. The gender was represented as a numeral: 0 for Male and 1 for Female
3. The occupations were numbered from 0 to 20 according to their similarity to other occupations. This was done using systemic knowledge of the occupations. These occupation values were then normalized to a value from 0 to 1.
4. The ZIP codes were converted to Latitude, Longitude pairs using the United States Postal Service database. The distance calculated between two zip codes was normalized using the distance between Alaska and Florida. This was done since the data entries were from USA and Canada only.

The movie database was categorized by genre. Each genre was given a 0 or 1 value depending on whether elements of that genre were in the movie or not.

4 Recommending a movie for a new user

When a new user arrives to the system, we only have his/her demographic features. So all the recommendations have to be based on these. We use two techniques to recommend movies to this new user.

4.1 k-NN (k- Nearest neighbors)

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method for classification and regression that predicts objects' values or class memberships based on the k closest training examples in the feature space. [4]

Here we use the neighbors of this user in the user demographic feature space and recommend the movies that they like. Tests performed on the number of neighbors to select showed us that the optimal number was 20. We hence looked at movies watched by the 20 nearest neighbors to the given user, and returned the movies seen by at least 50% of them. These movies were returned in decreasing order of popularity.

This method predicts cases where a popular movie with a bad rating may still be expected to be viewed by the user. An approach of recommending movies with high ratings only would ignore such movies, leading to erroneous results.

4.2 Decision Trees

A $J48$ decision tree was constructed with the following feature set:

- Age
- Gender
- Occupation
- Genre
- Release Date

This tree was supposed to use whether the movie has a rating of 3, 4, or 5; or not as the classifier, and predict the same for a new user/movie pair

For a new user, his/her details along with every movie details were fed into the tree, and the movies that returned a 1 were recommended to him. As mentioned above, this methodology is expected to miss popular movies with bad ratings.

5 Predicting a rating given by a user to a movie

5.1 Collaborative Filtering using SVD

With the motivation to find out some intrinsic features that a movie (or a user) will hold, which is not explicitly shown in the data as a feature, we try out finding an approximate decomposition of the USER x MOVIE matrix into U, S and V^T, called its SVD-Decomposition. It is understood amongst the Recommendation System experts and researchers [3] that the matrix U holds some instrinsic features/qualities about the USERS and the matrix V holds some intrinsic features/qualities about the MOVIES like the actors/actresses, director, awards that directly has an impact on their liking/disliking by any user.

5.1.1 SVD Decomposition

The Singular Value Decomposition (SVD) is a well known matrix factorization technique that factors an m by n matrix X into three matrices as follows:

$$\underset{m \times n}{\overset{X}{\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{m \times r}{\overset{U}{\begin{pmatrix} u_{11} & \dots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix}}} \underset{r \times r}{\overset{S}{\begin{pmatrix} s_{11} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix}}} \underset{r \times n}{\overset{V^{\mathsf{T}}}{\begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}}}$$

The matrix S is a diagonal matrix containing the singular values of the matrix X. There are exactly r singular values, where r is the rank of X. The rank of a matrix is the number of linearly independent rows or columns in the matrix. Recall that two vectors are linearly independent if they can not be written as the sum or scalar multiple of any other vectors in the space. Observe that linear independence somehow captures the notion of a feature or agglomerative item that we are trying to get at. To return to our previous example, if every user who liked Star Wars also liked The Matrix, the two movie vectors would be linearly dependent and would only contribute one to the rank.

We can do more, however. We would really like to compare movies if most users who like one also like the other. To accomplish we can simply keep the first k singular values in S, where k¡r. This will give us the *best rank-k approximation* to X, and thus has effectively reduced the dimensionality of our original space. Thus we have:

$$\underset{m \times n}{\hat{X}} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix} \approx \underset{m \times r}{U} \begin{pmatrix} u_{11} & \cdots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix} \underset{r \times r}{S} \begin{pmatrix} s_{11} & 0 & \cdots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix} \underset{r \times n}{V^{\top}} \begin{pmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}$$

Now from this new matrix $\hat{X}$ which is an approximation of the original matrix U, we can recommend ratings to the users i.e. the element $U_{i,j}$ represents the rating recommended for User i for the Movie j.

5.2 SVD Decomposition with Linear Regression Seeding

The matrix User x Movie is only filled upto around 6%, but before decomposing it using the SVD Decomposition we now fill in the missing values using a Linear Regression based model trained for every user on the basis of the movies he/she has already rated. For training our feature-set is the user-demographics along with movie genres and the labels are the actual ratings, training a linear regression model, we fill in the values for the movies that the user hasn't rated and hence filling up the matrix. Then following the approach same as in previous subsection, where the values seeded were the row or column averages.

6 Results

The following results were achieved using the various techniques mentioned above.

6.1 Movie Recommendation

To validate the accuracy of our movie recommendation system, we took new users from the additional datasets provided to us, and verified the predicted recommended movies with the movies actually rated by those users. The percentages reported here are the number of correctly predicted movies, over the number of movies rated by the new user.

Technique	Accuracy	Notes
k-NN	73%	k=20 Cross-validation accuracy The accuracy means that 73% of the movies recommended to a user were actually in his watched list.
Decision Trees	82%	Cross-validation accuracy. The accuracy shows the classifiers performance on seperating the 3,4,5 rated movies from 1,2 rated movies.

6.2 Movie Rating

To validate the accuracy of these ratings, we took new users, and rated the movies watched by them using our algorithms. These predicted ratings were compared with the actual ratings and the *Root Mean Squared Error (RMSE)* and *Mean Absolute Error (MAE)* were calculated.

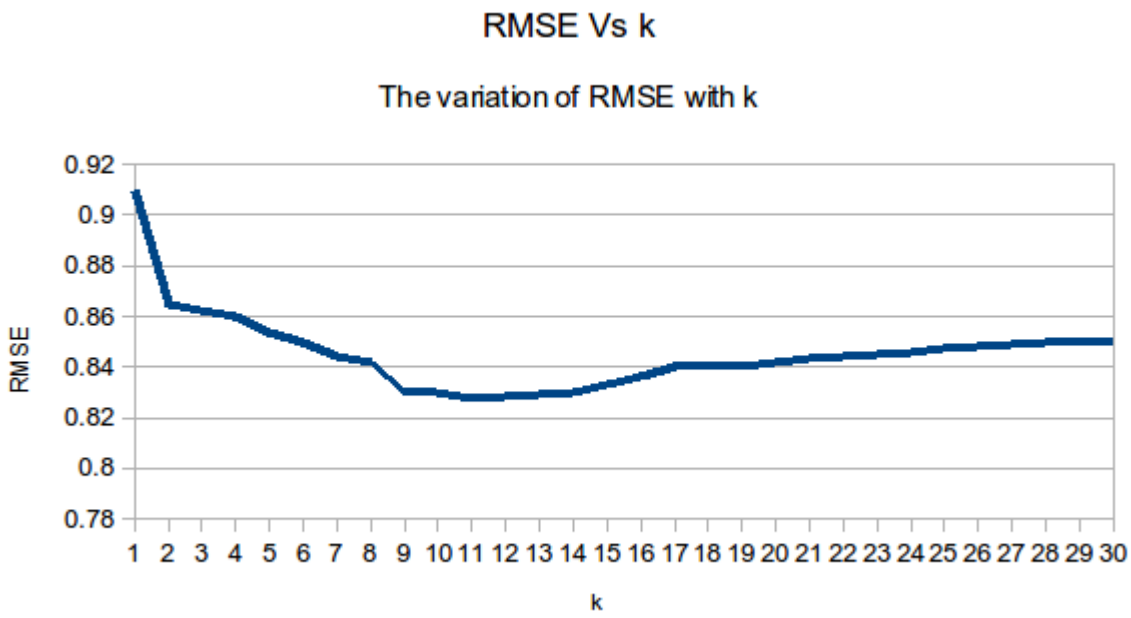
$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i' - y_i)^2}{n}} \text{ and } MAE = \sum_{i=1}^{n}\frac{|y_i' - y_i|}{n},$$

where y' is the estimated rating and y the actual rating.

Note that the mean of the ratings is 3.53 *and the standard deviation about the mean is* 1.12*. An algorithm with a Root Mean Squared Error of more than* 1.12 *would mean that predicting a rating of* 3.53 *for every movie would perform better than that algorithm.*

We computed the RMSE′s variation with k for the SVD-based approach. Below the graph obtained. Observe that the minimum value obtained was for $k = 12$.

Technique	RMSE	MAE	Notes
k-NN	1.10	0.841	Using k=25
SVD	0.835	0.643	This is the cross-validated value calculated on the additional datasets provided.
SVD seeded using Linear Regression	0.828	0.642	This is the cross-validated value calculated on the additional datasets provided.



7 Conclusion

The process of recommending a movie can make the difference between a good media website and a great media website. The Netflix challenge, offered a USD 1 Million prize for improving their recommendation system by 10% clearly shows the impact such a recommendation can make. Our results clearly show that, with the advent of better and faster computers, the mining of user demographic data and user preferences can give good predictions to whether a user will watch a particular movie or not, and if he/she shall, the rating given to the movie by that user.

Our k-NN algorithm used user demographics in a unique way, and these modifications on occupation and zip-code led to an accuracy increase from 72% to 76%, showing that better demographics will lead to a more comprehensive understanding of user preferences.

Our best result was obtained using the User-Movie matrix factorization, which was also the method used by *Simon Funk*, the winner of the Netflix challenge. The best result of $RMSE = 0.828$ shows that our predictions of the rating are off, on average, by less than 1 rating point on the scale from 1 to 5. Predicting whether the user will like the movie or not using a binary classifier gave an accuracy of 82% which shows that a difference of 1 point in the ratings generally doesn't matter much to users.

8 Acknowledgements

The authors of this paper would like to thank Dr. Harish Karnick for giving us an opportunity to carry out this project under his guidance; and also for his invaluable teaching and contribution to the understanding of the algorithms used in this paper. We would also like to thank the GroupLens Research Project at University of Minnesota for providing access to their database.

References

[1] O'Connor, M., & Herlocker, J. (1999, August). Clustering items for collaborative filtering. In Proceedings of the ACM SIGIR workshop on recommender systems (Vol. 128). UC Berkeley.

[2] Konstan, J. A., Riedl, J., Borchers, A., & Herlocker, J. L. (1998). Recommender systems: A grouplens perspective. In Recommender Systems: Papers from the 1998 Workshop (AAAI Technical Report WS-98-08) (pp. 60-64).

[3] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Application of dimensionality reduction in recommender system-a case study (No. TR-00-043). MINNESOTA UNIV MINNEAPOLIS DEPT OF COMPUTER SCIENCE.

[4] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (pp. 285-295). ACM.