Movie Rating Predictor Using Nonnegative Matrix Factorization on Tag Data

Claire Chang, Thaxter Shaw, and TJ Tsai[1]

***Abstract*— The goal of recommendation systems is to suggest suitable content such as movies, groceries, and more. This article aims to predict what someone will rate a movie, which could in turn be applied to a movie recommendation system. We propose a method of addressing this problem**

I. INTRODUCTION

The goal of this paper is to predict what someone will rate a movie on a five-star scale. This prediction can be used in recommendation systems for not only movies, but also grocery shopping, music, and more [1].

There are several obstacles that make this problem challenging. One issue, as investigated by Bell et al. is the effect of time. Over time, users may change their baseline rating (for example, have higher expectations of graphics) and begin to prefer different genres and actors [2]. Movies themselves also become more or less popular over time [2].

One approach that has been taken is to use a "neighborhood approach" that compares different movies, and recommends a highly rated movie that is similar to other movies a user has liked. Another approach is to use collaborative filtering (CF) methods such as nonnegative matrix factorization (NMF) and singular value decomposition (SVD) [3]. We could also use a hybrid approach of the neighborhood and CF approahces [4].

In this paper, we use data from MovieLens. The dataset we use contains ratings of over 1000 movies made by various users, filtered such that each user has rated at least 20 movies. The dataset also relates each movie to a series of short tags, such as "sci-fi," "cliché," or "overrated." This data was collected through users applying labels to movies, and machine learning [5].

Our system uses NMF on this tag data to create clusters of tags, which we will equate to "genres". We then use these genres predict ratings. Finally, we use mean movie ratings to adjust this predicted rating.

II. SYSTEM DESCRIPTION

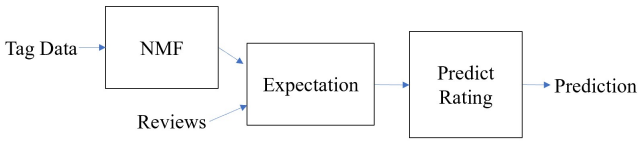


Fig. 1: Block diagram of proposed system.

[1]T. Tsai is with the Department of Engineering at Harvey Mudd College, 301 Platt Blvd., Claremont, CA 91711. E-mail: ttsai@hmc.edu

The overall architecture for our system is shown in Figure 1.

A. Nonnegative Matrix Factorization

We begin with the tag data, stored as a matrix of t rows and m columns. Here, t is the number of tags and m is the number of movies in the dataset. We perform NMF on the tag data matrix, which gives us a template matrix W and an activation matrix H. In W, we have t rows and R columns, where R represents the number of templates. In H, we have R rows and m columns.

Each template is comprised of clusters of tags and represents a genre. For example, the romantic comedy genre might contain the tags "cliché," "romance," "Ryan Gosling," and "comedy." In H, each movie is broken down into its component genres.

B. Expectation

Next, we assemble a new matrix of reviews. This matrix has m rows and p columns, where p is the number of reviewers, and stores all ratings made by each user.

Then, we create another matrix M, which represents each reviewer's affinity to each genre. M has R rows and p columns. Each entry in M is an rating out of five stars indicating what a reviewer is expected to rate a typical movie from a given genre. We get this information by taking the "expectation" of all the reviewer's ratings. In this expectation, ratings of movies more relevant to the genre of interest (as defined in H) are given more weight.

C. Predict Rating

Finally, we generate a predicted rating for a movie and reviewer from a query. For example, suppose we want to predict what reviewer A will rate the movie *La La Land*. To do this, we will use a "genre fingerprint" for our movie.

Recall that H is an $R \times m$ matrix, representing the relevance of each genre to each movie. We can $L1$ normalize the columns of H to create a matrix storing the genre fingerprints of each movie. These fingerprints indicate the breakdown of the movie into its component genres, such that each genre represents a proportion of the entire movie. See Figure 2 for examples of genre fingerprints.

For our example, let's assume that *La La Land*'s fingerprint is 75% romantic comedy and 25% drama. Then, based on reviewer A's preferences for these genres (from matrix M), we can generate a preliminary prediction for their rating.

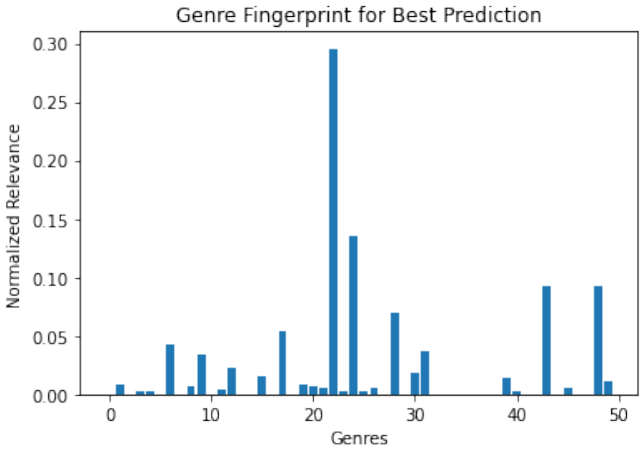
Finally, we look at the mean movie rating for *La La Land*. If this mean rating is higher than our preliminary prediction,

we round up to the nearest half-star. Otherwise, we round down.

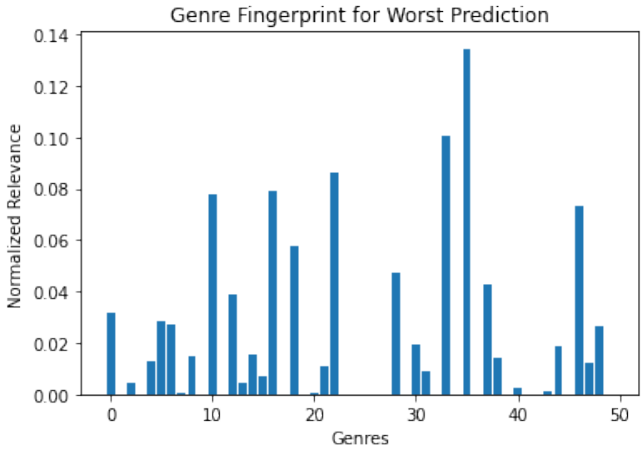
misc

III. RESULTS

IV. CONCLUSION

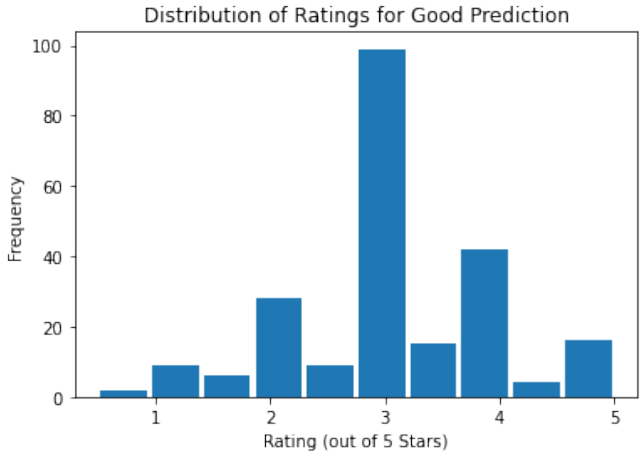


(a) Fingerprint of movie from good prediction.

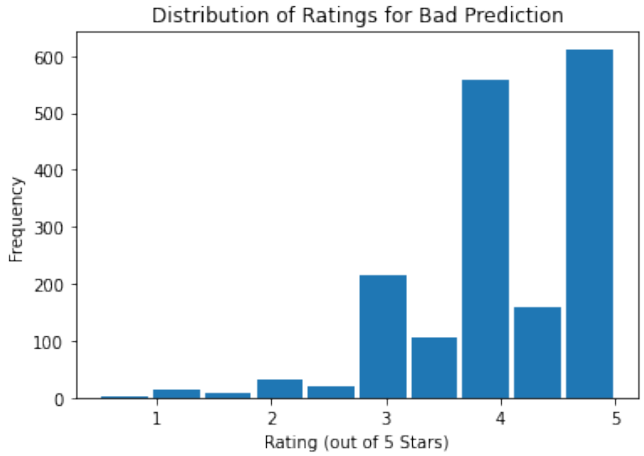


(b) Fingerprint of movie from bad prediction.

Fig. 2: Fingerprints of two movies. A fingerprint is the breakdown of the movie into genres. We can see that the fingerprint of the movie from the bad prediction has a wider distribution.



(a) Distribution of ratings for a well-predicted movie. The predicted and true ratings were both 3.0 stars.



(b) Distribution of ratings for a badly-predicted movie. The predicted and true ratings were 4.5 and 0.5 stars respectively, while the most popular ratings seem to have been 4.0 and 5.0 stars.

Fig. 3: Rating distributions for two movies.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[2] R. M. Bell, Y. Koren, and C. Volinsky, "The bellkor 2008 solution to the netflix prize," 2008.

[3] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, no. 12, p. 61–70, Dec. 1992. [Online]. Available: https://doi.org/10.1145/138859.138867

[4] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 426–434, 2008.

[5] M. D. Ekstrand, "Lenskit for python: Next-generation software for recommender systems experiments," *The 29th ACM International Conference on Information and Knowledge Management*, p. 2999–3006, 2020.