Movie Recommendation System

Course Project–
CS771: Machine Learning Techniques

By ***Group 27****:*

Ayush Jain	(10180)
Garima Gaur	(13111162)
Mridul Verma	(10415)
Sriram Gopalakrishnan	(13111021)

Supervisor:
Prof. Harish Karnick

Abstract

Recommendation systems are a type of information filtering systems that recommend products available in e-shops, entertainment items such as books music, videos, movies, books, news, images, or people, on social networking sites, that are likely to be of interest to the user. Now a days web is widely used as a mean of entertainment, huge number of people watch movies on line and there are a large number of online movie websites, thus a good recommendation sytsem will be of great help for those websites, in order to survive in this competitive world. Here we are building a movie recommendation system using data collected through MovieLens website. We are using collaborative filtering technique to build our recommendation system. In this report we are presenting possible approaches, the approach that we followed , and then the analysis of performance of our recommendation system.

Contents

1 Problem Definition

We need to build a system which predicts whether a *new user* will watch a particular movie. If yes, we are to further predict the rating that the user will give to that movie. To accomplish this task, we will have only the demographic information of the new user with us. This demographic information consists of the user's age, gender, occupation and ZIP code.

2 Data Description

This data set consists of:

1. 100,000 ratings (1-5) from 943 users on 1682 movies.
2. Each user has rated at least 20 movies.
3. Simple demographic info for the users (age, gender, occupation, zip)

This data has been cleaned up – users who had less than 20 ratings or did not have complete demographic information were removed from this data set. Demographic information about the users include their user id, age, gender, occupation and zip code. And movie information includes movie id, title, release date and genre.

2.1 Data Analysis

Prior to approaching any Machine Learning problem, it is essential to analyze the data at hand, to look for patterns that can be helpful in making decisions. Some of the statistics of our data analysis have been shown below:

Rating Distribution

We find the rating distribution in Figure 1 pie chart. Rating 4 has highest percentage with 34% and rating 1 with the lowest with 6%

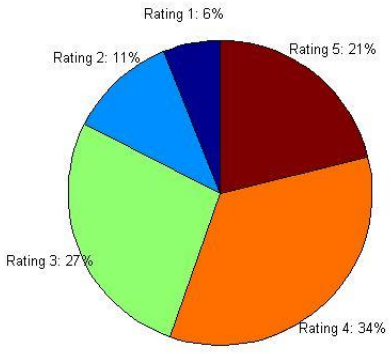

Figure 1: Pie Chart of Rating

Observe that 55% of the ratings in the dataset have been either 4 or 5. This makes sense, because if a user does not like a particular movie, he/she will not care to give it even a low rating.

Number of Ratings per Movie

Atleast 755 movies in the given data had been rated fewer than 20 times . Atleast 553 movies in the given data had been rated fewer than 10 times. Atleast 384 movies in the given data had been rated fewer than 5 times. Histogram shown in Figure 2 frequency of movie and number of ratings for the movie in space of 10.

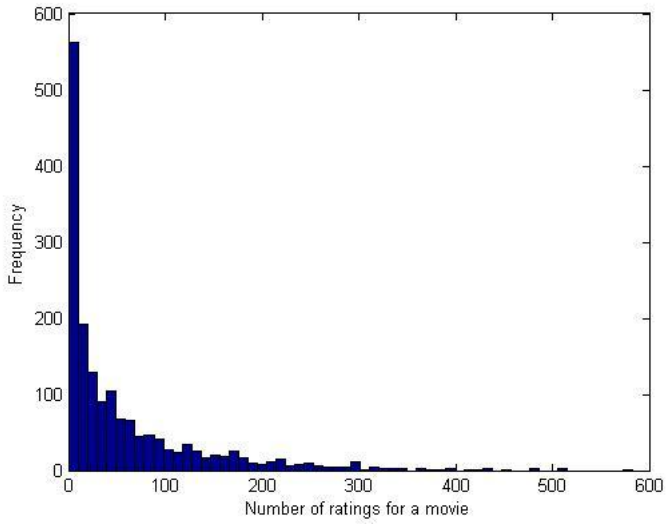

Figure 2: Analysis of number of ratings for a movie

Relation Between Number of Ratings and Average Rating of a Movie

In Figure 3, each point represents a movie. The average rating and the number of ratings have been estimated for each movie. The general trend that is observed is that for movies that have been rated by more users, the average ratings tend to be more, and also show a lot less deviation. Again, this makes sense because movies that receive more ratings tend to be more popular and better.

3 Traditional Approaches

Traditional, recommendation systems rely on creating user and movie 'profiles'. These profiles can then be used in any of the following three ways :

3.1 Content Based Filtering

In this approach, recommendation system makes use of profiles of users, which depend upon the user's tastes. Taste is based on how the user rated items.

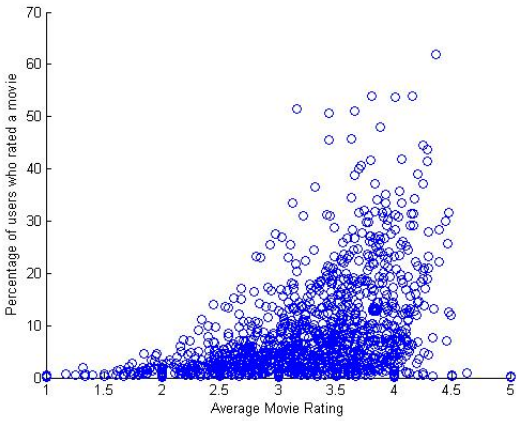

Figure 3: Scatter Plot of Number of Ratings vs. Average Movie Rating

This type of system would find movies that the user has previously watched and liked. Then, it would look for movies that have the same actors/directors, or belong to the same genre and suggest those movies to the user.

3.2 Collaborative Filtering

The idea of collaborative filtering is to find the group of users having similar interest or taste. Here, the taste of the user is determined by the movies that ratings that he has given to various movies. Among the methods of collaborative filtering, we can distinguish most popular approaches: user-based, item-based and hybrid approaches.

1. **User-based approach**
   This technique relies on the idea that if two users have given similar ratings to same movies, then they have similar tastes. Such users build a group or a so called neighborhood. A new user is likely to watch the same movie as their neighbors. In the user-based approach, the users perform the main role. If certain majority of the customers has the same taste then they join into one group. A new user is like to watch movies that are already watched the user of that group to which new belongs to.

2. **Item-based approach**
   In item-based collaborative filtering techniques, movies that have received similar ratings from users are grouped together in a neighbourhood. If a user likes a particular movie, we can recommend him similar movies in the same neighbourhood.

3. **Hybrid Approaches:**
   Hybrid approach tend to combine Item-based and User-Based collaborative filering techniques in novel ways. These approaches can be combined at the end, i.e. getting results from each and then combining them. Or, they can be much more interleaved.

For the problem at hand, none of these traditional techniques is directly applicable. While these techniques have been shown to provide good results when some rating history is available for the user, they are not really meant for predicting ratings for a new user.

4 System Details

4.1 Overview of the Approach

An overview of our approach has been provided in Figure 4.

We begin by normalising the ratings given by all users. Thereafter, *rating profiles* are created for each user, using the ratings given by the user to different movies and the relevant information regarding that movie. Based on these rating profiles, users are clustered into neighbourhoods, where users belonging to a particular cluster have similar rating profiles. Now, for each cluster, we build a *rating profile* and a *genre profile*. Next, we build a Decision Tree to assign users to different clusters, using only their demographic information.

Now for a new incoming user, we use the Decision Tree to assign that user a neighbourhood/cluster. Using this cluster's *genre profile*, we decide whether the user is likely to see a given movie or not. If yes, we use the cluster's rating profile to predict her rating for a particular movie.

In the ensuing sections, each of these steps has been explained in sufficient detail.

4.2 Ratings Normalization

The intuition behind normalising the ratings is that each user has her own bias while rating. Some users tend to give more favourable ratings, while others are more critical. Besides, same rating can mean have different meanings for different users. While a rating of 4 might be considered very good by a user, it may be considered only above-average by someone else. This intuition is backed by some analysis of the data, as shown in Figure 5. The average ratings of users vary across a wide spectrum. Hence, our normalisation scheme is outlined below.

Notations: R is the set of all ratings. In particular $R_{u=i}$ represents the set of ratings given by user i, and $R_{m=j}$ represents the set of ratings given to movie j.

1. For each user, we calculate the mean and standard deviation of his ratings as:

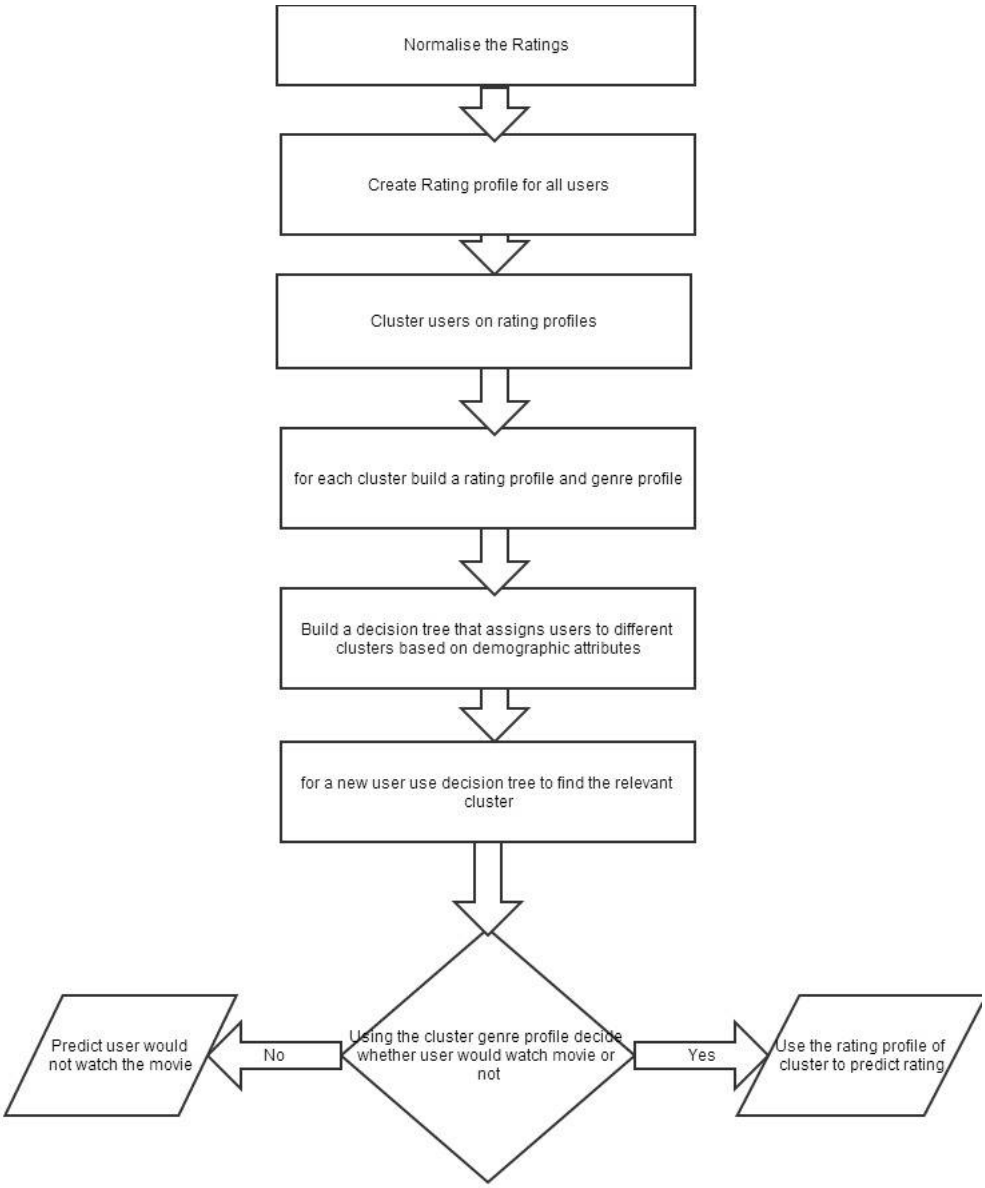

Figure 4: Overview of our Approach

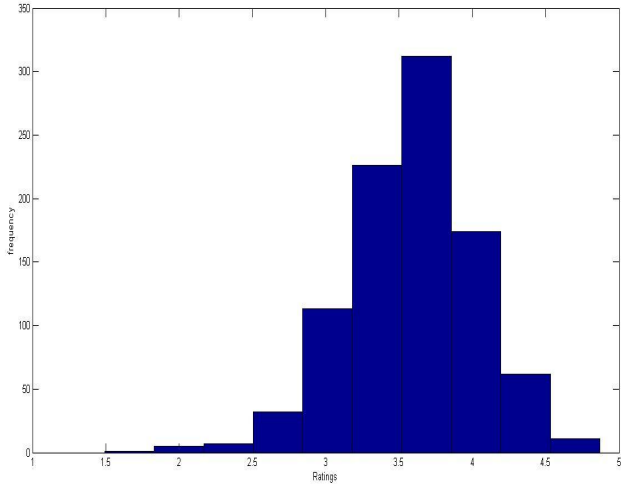

Figure 5: Rating Trend before normalization

$$\mu_i = \frac{1}{|R_{u=i}|} \sum_{r \in R_{u=i}} r$$
$$\sigma_i = \sqrt{\frac{\sum_{r \in R_{u=i}} (r - \mu_{u=i})^2}{|R_{u=i}|}}$$

The mean ratings and standard deviations for all set of users is then calculated as:

$$\hat{\mu_u} = \frac{1}{|U|} \sum_{u \in U} \mu_u$$
$$\hat{\sigma_u} = \frac{1}{|U|} \sum_{u \in U} \sigma_u$$

2. It is important to note that for users who have seen very few movies, it might be the case that all these movies have been very good. Likewise, the standard deviation of his movies can also be too high or too low. If a user has seen a sufficiently large number of movies, the distributions of these movies would have settled down to its general pattern. Based on these observations, the means and standard deviations for all users are recalculated as:

$$\mu_i' = \frac{|R_{u=i}|\mu_i + \lambda\mu_i}{|R_{u=i}| + \lambda}$$
$$\sigma_i{}' = \frac{|R_{u=i}|\sigma_i + \lambda\sigma_i}{|R_{u=i}| + \lambda}$$

Here, $|R_{u=i}|$ is the number of ratings given by user i, λ is a constant that has been set to 25. Note that if the number of user's ratings are much more than λ, these quantities would converge to their previous values. On the other hand, if they are comparable to λ, these quantities will be a weighted average of the previous values and the global values.

3. The normalised rating for rating by user i for movie j is given by:

$$\hat{R}_{u=i,m=j} = \frac{R_{u=i,m=j} - \mu'_i}{\sigma'_i}$$

4. The final predictions need to be converted to the original rating scale of 1-5. To achieve this, we assume that the bias of the unknown user is same as the global characteristics. So the final rating of the unknown user is given by:

$$R = \left(\hat{R} \times \hat{\sigma_u}\right) + \hat{\mu_u}$$

After normalizing, the distribution of the ratings looks as shown in Figure 6.

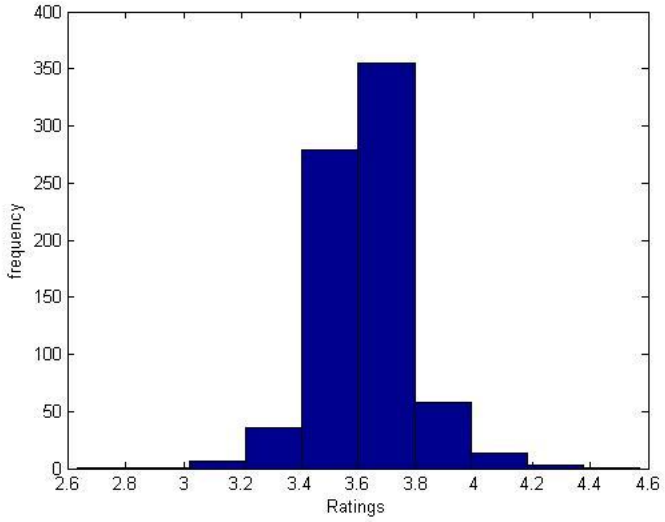

Figure 6: Rating Trend after normalization

4.3 Building Rating Profile for All Users

To build rating profiles for users, we first represent movies in a 19-dimensional vector using their genre information. There are 19 genres in total. So a movie j is represented by a 19-dimensional bit-vector mp_j, where a 1 at position n indicates that the movie belongs to genre n. Note that in the given dataset, a movie can belong to more than one dataset. Let $M_{u=i}$ denote the set of movies rated bu user i. Then the rating profile of a user i is computed as:

$$rp_i = \frac{\sum_{j \in M_{u=i}} R_{u=i,m=j} \times mp_j}{|M_{u=i}|}$$

Here $R_{u=i,m=j}$ is the rating given by user i to movie j. It can be clearly seen that rp_i represents the average rating given by user i, to each genre of movies.

4.4 Clustering of Users

All the users are then divided into clusters. Each user is represented by her rating profile, and the **K-Means Clustering** algorithm is used to cluster the users. The number of clusters has been kept at 20 in our experiments. This is motivated by the fact that there are 19 genres to which a movie can belong.

At the end of this step, for each user, we have an associated cluster number. Users in a particular cluster tend to rate a particular genre of movies similarly. And the clusters are differentiated on the bases of higher ratings given to genres. This fact is specially visible in Figure 7, where users of Cluster 1 show strong inclination towards Romance movies, while those belonging to Cluster 4 are biased dowards Drama and those belonging to Cluster 7 have a particular preference for Comedy.

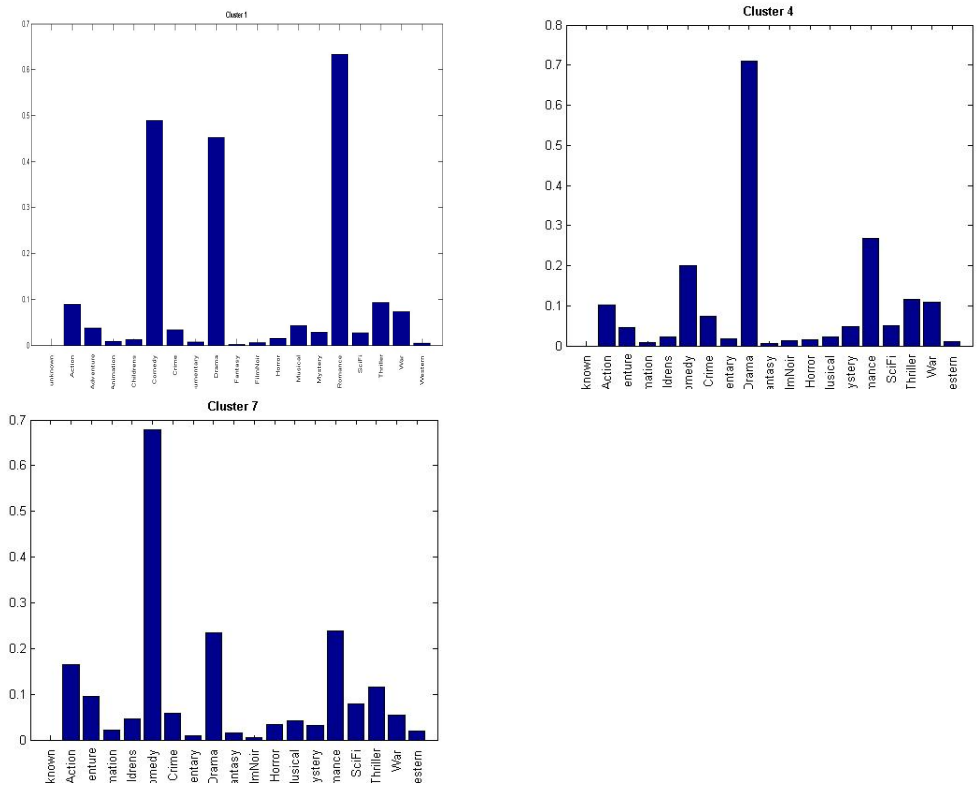

Figure 7: User behaviour In Different Clusters

4.5 Cluster-Based Rating and Genre Profiles

Given the rating profile of all users rp, the rating profiles of clusters are a straighforward extension. The rating profile of a cluster C is given by:

$$RP_C = \frac{\sum_{i \in C} rp_i}{|C|}$$

Here, $|C|$ represents the number of users belonging to cluster C.

To create the cluster-based genre profiles, we need the user's genre profiles which are defined as:

$$gp_i = \frac{\sum_{j \in M_{u=i}} mp_j}{|M_{u=i}|}$$

The notations used are the same as in precious sections. The genre-profile of a user essentially gives information about what fraction of movies that she has watched, belonged to a particular genre. Given the user's genre profiles, the cluster-based genre profiles can be analogously defined as:

$$GP_C = \frac{\sum_{i \in C} gp_i}{|C|}$$

4.6 Building a Decision Tree

At the end of the previous stage, we have a cluster number associated with each user. Now, to predict the ratings for a new user, we need to decide the cluster she belongs to. Since clusters were formed on the basis of rating profiles, and a new user has no rating profile, this poses a problem.

To solve this, we build a **Decision Tree Classifier**. The cluster number that users belong to is the *class* of that user. And each user is defined by his demographic attributes. Once the tree has been built, it would use the demographic attributes of an unknown user to assign her a cluster.

4.7 Finding the Cluster for a New User

As explained in the previous subsection, we use the Decision Tree to find the classifier that an unknown user belongs to.

4.8 Predicting Whether a User Will See a Movie

Once a new user has been assigned to a particular cluster, the next task is to use her *neighbours* to decide whether the user will watch a particular movie or not. To accomplish this task, we use the cluster-based genre profiles created earlier. Recall that a genre profile indicated that out of all the movies seen by a user, what fraction belonged to a particular genre. Cluster-based genre profiles also carry the same intuition.

Consider the case when the new user has been assigned a cluster C, having genre profile GP_C. We are to predict whether this user will watch a movie j.

Movie j will have its own 19-dimensional vector mp_j as explained earlier, that reprents the genres to which movie j belongs. Now, the probability P that the user will watch movie j is calculated as:

$$P = \frac{mp_j.GP_C}{|mp_j|}$$

Here, . represents the vector dot-product and $|mp_j|$ is the number of genres to which movie j belongs. If P is greater than a threshold, the system predicts that the user will watch the movie. Otherwise, the system predicts that the user will not predict the movie.

4.9 Predicting the Rating Given by the User

Once it has been ascertained that a new user will watch a particular movie, the next task is to predict whether she will like it or not. This is done by predicting the rating that the user would give to the movie. To do this, we use the Cluster-Based Rating Profiles that were created preciously. Recall that a rating profile indicated the average rating given to movies of a particular genre.

Once again, consider the case when the new user has been assigned a cluster C, having rating profile RP_C. We are to predict whether this user will watch a movie j. Now, the probability R that the user will watch movie j is calculated as:

$$R = \frac{mp_j.RP_C}{|mp_j|}$$

5 Performance Evaluation

5.1 Experimental Conditions

To test our system, we split the dataset on the basis of users. 80% of the users were randomly assigned to the training set and remaining 20% were assigned to the test set. To counter the effects of 'lucky split', this procedure was carried out 5 times and the average figures in these trials were reported.

To evaluate our performance on rating prediction, the performance metric we use is the Normalised Root Mean Squared Error (NRMSE). For a rating R and predicted ratings $\hat{R}$, the root mean squred error is defined as:

$$RMSE = \frac{\|R - \hat{R}\|^2}{|R|}$$

The $NRMSE$ is given by:

$$NRMSE = \frac{RMSE}{r_{max} - r_{min}}$$

where, r_{max} and r_{min} are the maximum (5) and minimum (1) ratings respectively. THE NMRSE is calculated on the movies that the user is predicted to watch by the system.

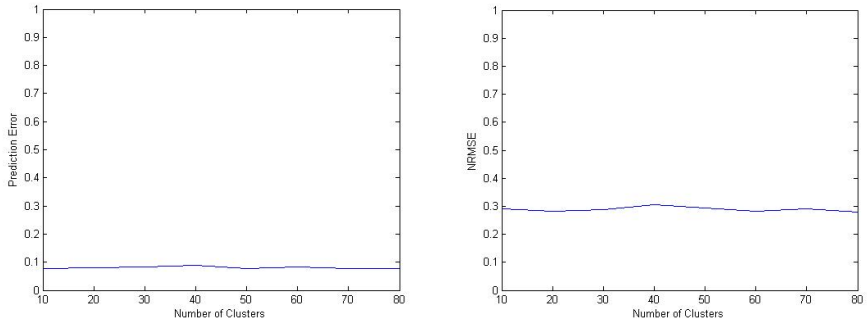

Figure 8: Effect of Number of Clusters

5.2 Results

The probability threshold was set at 0.1 for predicting whether a user sees a movie or not. The results are noted below:

1. **Predicting Whether a User Sees a Movie or Not:**
   Error Rate = 8.2252%

2. **NRMSE in predicting ratings:** 0.2867

5.3 Effect of Normalisation

To judge the effectiveness of our normalisation scheme, we repeat the above experiment, keeping all the other parameters same, but without normalising the data. We observe that the error rate in prediction rises to 9.73%. Also, the NRMSE shoots up all the way to 0.7647.

Thus, the normalisation scheme that we propose is very effective.

5.4 Effect of Number of Clusters

We also varied the number of clusters to see its effect on performance. The variation of NMRSE with the number of clusters was noted. The results have been shown in Figure 8. Increasing the number of clusters beyond 10 does not have any major changes in the error rates.

6 Conclusions

The problem at hand was to predict whether a new user would watch a movie, and if yes, the problem further asked to predict the user's rating. Because all this prediction was to be done for a new user, traditional techniques that relied on rating history could not be used. Furthermore, the performance of traditional similarity metrics (like Cosine Similarity, etc) on categorical variables is

not straight-forward, and each of the ways to do so (like Bit Vectors) suffer some or the other advantage.

This motivated us to design a new system. The system that we designed used Machine Learning Algorithms like Decision Trees and Clustering. At each stage, the design is motivated by relevant data analysis. The system that we designed has shown good performance for both the tasks. Also, the design of the system has one distinct significant advantage. *This system works, not only for a new user, but also for a new movie and for movies that have not received many ratings.* This is possible because all we are using in this system is the genre information related to the movie.

7 Future Work

In this work, we classified users into different clusters using decision tree. We also performed some preliminary tests with SVMs. The results did not fare out well as in this case the resulting classes of the new users only belonged to a same particular subset of the clusters (at each and every iteration) even when randomly changing the test users. Nonetheless, more extensive testing is required to be more certain.

As has been noted ealier, movies with large number of ratings are usually rated higher and movies with lower rating are usually rated by few number of people. We want to use this trend in appropriately scaling the probabilities of users watching a particular movie.

We also want to analyse individual user bias with respect to movie and explore whether we can use it while predicting whether that user would watch a movie or not and predict the rating.