Web based Recommender Systems and Rating Prediction

Tho Nguyen
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects
Part of the Computer Sciences Commons

Recommended Citation
DOI: https://doi.org/10.31979/etd.rpf4-6cre
https://scholarworks.sjsu.edu/etd_projects/75

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.
Web based Recommender Systems and Rating Prediction

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Tho Nguyen

September 2009
ABSTRACT

Web based Recommender Systems and Rating Prediction

by Tho Nguyen

This project implements a recommender system on large dataset of Netflix’s movies. This project also tries to improve recommender systems by incorporating confidence interval and genres of movies. This new approach enhances the performance and quality of service of recommender systems and gives better result than Netflix commercial recommender system, Cinematch.
# Table of Contents

1. Introduction............................................................................................................................... 1  
2. Recommender Systems............................................................................................................. 2  
3. User-based Collaborative Filtering.......................................................................................... 4  
4. Item-based Collaborative Filtering .......................................................................................... 5  
5. Fisher Transformation .............................................................................................................. 7  
6. Confidence Intervals.................................................................................................................. 8  
7. Content Based Method ............................................................................................................ 10  
8. Experimental Results ............................................................................................................. 11  
8.1 Pearson Correlation Algorithm ........................................................................................... 12  
8.2 Pearson Correlation with lower limit of confident interval Algorithm .................................. 14  
8.3 Content based Algorithm ....................................................................................................... 17  
8.4 Making Predictions ................................................................................................................ 23  
9. Summary of Results ............................................................................................................... 26  
10. Future Work ............................................................................................................................ 27  
11. Conclusion ............................................................................................................................. 27  
12. References .............................................................................................................................. 28  
Appendix A: How to run recommender systems ........................................................................... 29
LIST OF TABLES

Table 8.1. RMSE of Probe data for each algorithm ........................................ 24

LIST OF FIGURES

Figure 1: The Collaborative Filtering Process .............................................. 4
Figure 2: Similarity computation on selected items ............................... 6
Figure 3: Fisher transformation of Pearson Correlation .......................... 8
Figure 4: Netflix database with movies, users and ratings .................. 10
Figure 5: Recommended movies for selected item rank by Pearson Correlation ................................................................. 13
Figure 6: Recommended movies for selected item rank by lower limit of confidence interval .................................................. 16
Figure 7: XML schema of Netflix movies catalog ............................. 18
Figure 8: Netflix’s movie genres of training data set .......................... 19
Figure 9: Recommended movies for selected item with content based Algorithm .......................................................... 22
Figure 10: RMSE of Netflix’s Probe Data for Pearson correlation Algorithm 24
Figure 11: RMSE of Netflix’s Probe Data for lower limit confidence interval Algorithm .................................................. 25
1. Introduction

Netflix, a rental movie company, announced on October 2, 2006 that it will give $1,000,000 to anyone who can improve the RMSE of its movie recommender system, Cinematch, by more than ten percent. The movies from Netflix were reviewed and rated by its subscribers with number from one to five. One means they dislike the movie and five means they are really like the movie. Netflix has about 12 million subscribers who rating over eighty five thousands movies. The numbers of ratings from the users are about two billion [4]. RMSE is a root mean square error and it is calculated by [4]

\[ RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (r_i - t_i)^2} \]  

\( r_i = \text{prediction} \quad t_i = \text{target} \)

Netflix's contest uses two datasets, a training set and a probe set. The training set used to train the recommender algorithm and test it on the probe set. There are over one hundred million ratings from 480,000 users in the training set. The users rate on eighteen thousands movies. The probe set has 1,500,000 user movie pairs and the true rating is hidden. A simple algorithm which uses average rating of movies from user to predict rating of target movie will give RMSE of 1.054. Netflix's recommender system, Cinematch, gives RMSE of 0.9525. Therefore, to win the contest, the winner algorithm should achieve an RMSE of at least 0.8563, a ten percent improvement. Currently, as of May 11, a team called BellKor in Big Chaos has a highest improved algorithm with RMSE of 0.8616 [4].

This research applies different algorithms to improve recommender system on Netflix's dataset. The methods are including item-based Collaborating Filtering, CF, and content-based technique. For item-based CF, Pearson correlation and its Variance lower limit confidence interval
are used. For content-based method, extra item's profile, genres are computed. The enhanced algorithms gave better improvement over Cinematch.

2. Recommender Systems

Recommender systems analyze user's profile and the relationship between user and target item to help user purchase or rent the item based on user's interest. With the help of computer, recommender systems can analyze huge collection of data based on users' preferences to give good recommended items. Some online company like Netflix and Amazon use recommender systems to help users easy to find items they want on their website [5]. Every time a user logins to their website, a new list of recommended items are showed based on past user’s reviews or purchases. Instead of spend time navigate on the website and search for the items, a recommender system can save time for the user by display the list of items which the user likes based on user’s profile.

Recommender system also can help online companies sell their products better. Example, when I logins to Amazon website, there was a screen protector for ipod classis on my recommended items. I bought an eighty gigabyte ipod classis on Amazon website before and did not think about buying a screen protector for it. When I saw the screen protector for ipod, it made me thought about the protection for my ipod so I bought it. Same thing happens to other websites like newegg.com and buy.com, the users do not think about buying the items until they see them display on their recommended list.

Recommender system can give personalize feeling to the user because it is based on the real input from the user and it is always update. Whenever the user buys or reviews new item, a new recommended list is created for that particular user.
There are two groups in recommender systems, content-based and collaborative filtering (CF) algorithms. Content-based algorithms use user's profile to find matching items with the user. For a twenty three year old user, a content-based algorithm will select all items which are interested by this age. Content-based approach also can use item's profile to recommend item to user. For example, a content-based recommender system can recommend list of movies to user base on movies' genre which user's interest. These user and item's profiles are difficult to collect and need to get from external source [5].

Collaborative Filtering algorithm, another choice for recommender system, uses past user's behaviors to recommend items to user [5]. These behaviors include user's transactions or product rating. Example, the transactions where users buy some products or the number of ratings which users review items. They don't need the explicit profiles of each user or item. For a user X who rate five on all five movies. A CF system will analyze the data and find all users who give the same five movies with rating of five then recommend the list of movies that these same users' interest to user X.

A schematic diagram of CF algorithm is shown in Figure 1. In the picture, we can see the matrix mxn represents user-item data. There is a rating score of each user m on item n at each entry of the matrix. Each individual rating has a numerical scale from 0 to 5. The 0 means the user has not yet rate that item [1].
3. User-based Collaborative Filtering

User-based Collaborative Filtering is one of the most chosen algorithms to use in recommender systems by online companies [8]. It relies on the similar behaviors between each users in the group. These behaviors are including buying or ratings items. The behaviors of various users in one group can help recommending other users in same group to buy or rate different items [12].

There are many algorithms to calculate the similarity between the two users in CF systems. One of them is Pearson correlation algorithm. It is a most chosen algorithm to use in CF systems [2]. Pearson correlation only computes the similarity between the two users who rate a same item. For example, let S is the set of items where both user x and user y rated. Then the Pearson correlation computes the similarity between user x and user y as [2]:

$$
\text{Pearson correlation} = \frac{\sum_{i \in S} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i \in S} (x_i - \bar{x})^2 \sum_{i \in S} (y_i - \bar{y})^2}}
$$
Considering as the most used algorithm in Collaborative Filtering, there are some limitations in user-based approach. The first limitation is the scalability of the algorithm. The computation of user-based CF is more complex when the number of users gets bigger [12]. Therefore, it is difficult to use user-based CF in big online service companies as Amazon and Netflix. User-based CF recommender systems can work very well with a small dataset, but they usually don't work well with a large dataset like Netflix's dataset. Second limitation of user-based CF is performance [12]. Its performance is slow because User-based CF needs to recomputed the similarity of user-user every time it gives new recommendation.

4. Item-based Collaborative Filtering

Instead of computation between two users, the item-based collaborative filtering algorithm computes the similarity between two items. The computation of item-based algorithm is much simpler and more scalability than user-based algorithm. Usually, there is less number of items than users in online service companies. For example, Netflix's dataset has over 480,000 users but there are only 18000 movies.

\[
sim(x, y) = \frac{\sum_{s \in S_{xy}} (r'_{x,s} - \bar{r}_x)(r'_{y,s} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r'_{x,s} - \bar{r}_x)^2 \sum_{s \in S_{xy}} (r'_{y,s} - \bar{r}_y)^2}}
\]  

(3.1)
Figure 2: Similarity computation on selected items.

To compute the similarity between two items, the users who rated both items need to be selected as in Figure 2 [1]. Then the calculation will be used on these users and items. For Pearson correlation algorithm, the similarity of two items is computed by [1]

\[
sim(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_i)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_j)^2}}. \tag{4.1}
\]

Here \( \bar{R}_i \) is average number of item \( i \), \( R_{u,i} \) is number of rating user \( u \) gives on item \( i \).

The prediction of user on target item is computed after we have similarity score of all other items to target item. For the set of all items which rated by the user, the prediction of user \( u \) on item \( i \) is given by [1]
\[ P_{u,i} = \frac{\sum_{\text{all similar items, } N} (s_{i,N} \times R_{u,N})}{\sum_{\text{all similar items, } N} (|s_{i,N}|)} \] (4.2)

Where \( s_{i,N} \) is the similarity between item \( i \) and other item in set \( N \). \( R_{u,N} \) is the rating of user \( u \) on item in set \( N \). Set \( N \) is the set of items which rated by user \( u \).

## 5. Fisher Transformation

Unlike confidence intervals around means, confidence intervals around Pearson correlation \( r \) are not symmetrical. The confidence interval around a Pearson correlation \( r \) is based on Fisher’s transformation. The transformation is given by [13].

\[ z = 0.5 \log_e \left( \frac{1 + r}{1 - r} \right) \] (5.1)

Difference than Pearson correlation, transform value \( z \) is normally distributed with expectation equal to \( 0.5 \ln(1 + p)/(1 - p) \). Where \( p \) is the population correlation and have variance equals to \( 1/(n-3) \) with \( n \) is the sample size. Figure 3 shows the conversion between Pearson correlations to Fisher value \( z \) [13].
Figure 3: Fisher transformation of Pearson Correlation

The x-axis is Pearson correlation and has a range from -1 to 1. Looking at Figure 3, we can see that when Pearson correlation value goes near the outer limit, the fisher value will go to positive and negative infinitive. The transformation value is more stable in the middle of the range.

6. Confidence Intervals

Confidence interval is used to estimate the range of interval for the value of Pearson correlation. With different sample size of items, a ninety five percent of confidence interval will estimate value of Pearson correlation correctly ninety five times out of one hundred trials. To calculate confidence interval, the value need to be normal distributed. Pearson correlation value is not normal distributed, so we need to convert Pearson correlation value to fisher value. Then we take the confidence interval on the fisher value and convert it back to Pearson correlation confidence interval. The steps to do it are [13]
1. Convert Pearson correlation value to fisher value by formula (5.1)

2. Calculate confidence interval on fisher value $z$ with upper and lower limit.

$$
\zeta_l = z_r - z(1-\alpha/2) \sqrt{\frac{1}{n-3}} \\
\zeta_u = z_r + z(1-\alpha/2) \sqrt{\frac{1}{n-3}}
$$

(6.1)

Where $z(1-\alpha/2)$ equals to 1.96 for ninety five percent confident interval.

3. Convert the confidence interval back to Pearson correlation $r$ by

$$
r_l = \tanh(\zeta_l) = \frac{\exp(2\zeta_l) - 1}{\exp(2\zeta_l) + 1} \\
r_u = \tanh(\zeta_u) = \frac{\exp(2\zeta_u) - 1}{\exp(2\zeta_u) + 1}
$$

(6.2)
As shown in Figure 4 and equation (6.1), confidence interval is related with Pearson correlation and the sample size. When the value of Pearson correlation is high, the lower limit of confidence interval also has high value. Same thing for the sample size, with high number of sample size, the value of lower limit of confidence interval is closer to the Pearson correlation. Instead of using Pearson correlation, we can use lower limit of confidence interval to find the similarity between the two items and also take the sample size into the computation.

7. Content Based Method

Content-based recommendation method uses extra information of user's profile or item's profile in the computation. To give recommendation to one user, the profile of target user will analyze and items which matching to user's profile will be selected [2]. For user who likes action movie, all movies with action genre will be selected and recommend to target user. In another example, when user is twelve year old and likes animation movie, then most of the Disney animation movies will be recommended to this user.
Content based algorithm can work best with items that has lots of information like documents or news website. On Google website, a content based algorithm is used to give user news and information based on user’s location. When I logins to Google and go to news page, I can see all the news that is happening in San Jose where I live.

There are some disadvantages with the content based algorithm, because its algorithm is based on user’s or item’s profile. The profile needs to be easy to extract by computer. Therefore, it works well with text or xml file but has difficulty when dealing with media data like movies or pictures.

8. Experimental Results

At the beginning of the project, I try to calculate the Pearson correlation between the two users with equation (3.1) but it did not work. It took over six days and did not finish the computation. So I did more research on papers about user-based Pearson correlation. The computation of all user-user correlation is not possible because the dataset is too big. Netflix’s dataset has over 480 thousand users [4] and to compute Pearson correlation on all user-user pair will have over 1 Terabyte data [5]. So I switch to item-based collaborative filtering approach and use equation (4.1) to get all item-item correlations. To manage the database of Netflix’s training set, I use Netflix recommender framework from Benjamin Meyer [11]. The framework is written in c++ and it converts Netflix data’s text files into 2 binary files, movies.data and users.data. Each file has about 400 Megabyte data. Movies.data contains all movies with users and ratings and users.data contains all users with movies and ratings as figure 4 shows. It is much easier to manage and access data from binary files than mysql database. The runtime for computation is also faster in binary files because they can be loaded into memory while mysql database, over eight Gigabyte, can not be loaded into memory when doing the computation.
8.1 Pearson Correlation Algorithm

I build a web base interface where users can select a movie and view a list of recommended items. The top 30 recommended movies with highest correlations are given when user select movie from the table.

For movie “Lord of the Rings: The Return of the King” (LOTR:ROK), the Pearson correlation is computed by (4.1) and the thirty highest correlations are given in figure 5. “Lord of the Rings: The Two Towers” has the highest Pearson correlation with 0.785264 and have 114244 users who rate on both movies. By looking at figure 5, we can see that for the top thirty
recommended movies for LOTR:ROK, there are eight movies is about Lord of the Ring story.

Moreover, the top five recommended movies in the list are about Lord of the Ring movies.

![Figure 5: Recommended movies for selected item rank by Pearson correlation.](image-url)
8.2 Pearson Correlation with lower limit of confident interval Algorithm

Fisher transformation and Confidence interval has been used to improve the movies recommender system. The lower limit of confidence interval takes size of the number of ratings and Pearson correlation into the computation of recommendation and prediction for the target movie. When the size of the number of ratings is high, the value of lower limit will stay closer to the Pearson correlation. When the size of the number of ratings is low, the value of lower limit will get farther from the Pearson correlation. The movie which has high value in both number of ratings and Pearson correlation with the target movie will have high value in lower limit of confidence interval. This movie will be selected into list of recommended movies and set of neighbor movies to predict the rating of target movie. By ranking the movies with lower limit of confidence interval from high to low value, we can get the better result as example of Figure 6. It also gives better RMSE value over Netflix Probe dataset than the Pearson correlation algorithm, 0.92669 compare to 0.929651.

For movies “Lord of the Rings: The Return of the King”, the fisher transformation of Pearson correlation is computed by equation (5.1) and its highest ranking correlation movie, “Lord of the Ring: The Two Towers”, has fisher transform value of 1.05896. For 95% confidence interval, the lower and upper limit of confidence interval is calculated by equation (6.1) and “Lord of the Ring: The Two Towers” has lower limit of 1.05316 and upper limit of 1.06475. The confidence interval of fisher is converted back to lower and upper limit of Pearson correlation by equation (6.2). Lower limit 1.05316 and upper limit 1.06475 of “Lord of the Ring: The Two Towers” are converted to 0.78303 and 0.787477 for lower and upper confidence
interval of Pearson correlation as figure 6. The source code to calculate the lower and upper limit of Pearson correlation is follow:

```c
float fisher_calculation(float pearson_correlation, int count, float percentage)
{
    float fisher = 0.5 * log( (1 + pearson_correlation) / (1 - pearson_correlation));
    float different = percentage / sqrt(count - 3);
    return fisher + different;
}

float fisher_to_Pearson(float fisher)
{
    return (exp(2*fisher) - 1) / (exp(2*fisher) + 1);
}

float calcLowerLimit(float pearson_correlation, int count)
{
    return fisher_to_Pearson(fisher(pearson_correlation, count, -1.96));
}

float calcUpperLimit(float pearson_correlation, int count)
{
    return fisher_to_Pearson(fisher(pearson_correlation, count, 1.96));
}
```

By using fisher transformation and ranking the movies by lower limit of confidence interval, we can have better recommended movies as in figure 6. We can see that for the top thirty recommended movies for LOTR:ROK, there are nine movies is about Lord of the Ring story. Moreover, the top eight recommended movies are about Lord of the Rings story.
### 30 Recommend movies for: Lord of the Rings: The Return of the King (2003)

<table>
<thead>
<tr>
<th>Movies</th>
<th>pearson correlation</th>
<th>Number of Rating</th>
<th>Lower Limit</th>
<th>Upper Limit</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lord of the Rings: The Two Towers (2002)</td>
<td>0.785264</td>
<td>114244</td>
<td>0.78903</td>
<td>0.787477</td>
<td>0.00444615</td>
</tr>
<tr>
<td>Lord of the Rings: The Fellowship of the Ring (2001)</td>
<td>0.790399</td>
<td>111120</td>
<td>0.757908</td>
<td>0.762698</td>
<td>0.00485024</td>
</tr>
<tr>
<td>Lord of the Rings: The Return of the King: Extended Edition (2003)</td>
<td>0.749107</td>
<td>66208</td>
<td>0.745145</td>
<td>0.754331</td>
<td>0.00086573</td>
</tr>
<tr>
<td>Lord of the Rings: The Two Towers: Extended Edition (2002)</td>
<td>0.721617</td>
<td>68246</td>
<td>0.718204</td>
<td>0.725391</td>
<td>0.00718743</td>
</tr>
<tr>
<td>The Lord of the Rings: The Fellowship of the Ring: Extended Edition (2001)</td>
<td>0.707142</td>
<td>66818</td>
<td>0.70394</td>
<td>0.711503</td>
<td>0.00756335</td>
</tr>
<tr>
<td>Lord of the Rings: The Return of the King: Bonus Material (2003)</td>
<td>0.467648</td>
<td>908</td>
<td>0.415221</td>
<td>0.519879</td>
<td>0.101759</td>
</tr>
<tr>
<td>Lord of the Rings: The Two Towers: Bonus Material (2002)</td>
<td>0.437632</td>
<td>1272</td>
<td>0.390893</td>
<td>0.474987</td>
<td>0.0703043</td>
</tr>
<tr>
<td>Lord of the Rings: The Fellowship of the Ring: Bonus Material (2001)</td>
<td>0.436736</td>
<td>1512</td>
<td>0.39502</td>
<td>0.470654</td>
<td>0.0616344</td>
</tr>
<tr>
<td>Without Conscience (2004)</td>
<td>0.695058</td>
<td>20</td>
<td>0.664707</td>
<td>0.869991</td>
<td>0.505284</td>
</tr>
<tr>
<td>Best Motoring: Drift Bible (2003)</td>
<td>0.559706</td>
<td>43</td>
<td>0.321767</td>
<td>0.736281</td>
<td>0.424514</td>
</tr>
<tr>
<td>Connections 3 (1979)</td>
<td>0.550697</td>
<td>42</td>
<td>0.296759</td>
<td>0.732287</td>
<td>0.435539</td>
</tr>
<tr>
<td>The Westerner (1940)</td>
<td>0.500247</td>
<td>49</td>
<td>0.254902</td>
<td>0.685079</td>
<td>0.430176</td>
</tr>
<tr>
<td>The Hiding Place (1975)</td>
<td>0.581179</td>
<td>26</td>
<td>0.250132</td>
<td>0.790563</td>
<td>0.540431</td>
</tr>
<tr>
<td>Star Wars: Episode VI: Return of the Jedi (1983)</td>
<td>0.251832</td>
<td>65317</td>
<td>0.244736</td>
<td>0.2581</td>
<td>0.0143647</td>
</tr>
<tr>
<td>Survival Island, IMAX (1986)</td>
<td>0.36108</td>
<td>232</td>
<td>0.244485</td>
<td>0.468638</td>
<td>0.224033</td>
</tr>
<tr>
<td>Burn Up Scramble (2004)</td>
<td>0.43322</td>
<td>83</td>
<td>0.241227</td>
<td>0.643437</td>
<td>0.30311</td>
</tr>
<tr>
<td>Dora the Explorer: Super Babies (2005)</td>
<td>0.343157</td>
<td>307</td>
<td>0.240451</td>
<td>0.436284</td>
<td>0.197813</td>
</tr>
<tr>
<td>Thomas &amp; Friends: 10 Years of Thomas (1995)</td>
<td>0.523518</td>
<td>57</td>
<td>0.240248</td>
<td>0.724023</td>
<td>0.484375</td>
</tr>
<tr>
<td>X2: X-Men United (2003)</td>
<td>0.245783</td>
<td>70713</td>
<td>0.238845</td>
<td>0.252696</td>
<td>0.0138509</td>
</tr>
<tr>
<td>Star Wars: Episode V: The Empire Strikes Back (1980)</td>
<td>0.245211</td>
<td>68849</td>
<td>0.238177</td>
<td>0.252219</td>
<td>0.0140413</td>
</tr>
<tr>
<td>Beauty and the Beast (1987)</td>
<td>0.494782</td>
<td>45</td>
<td>0.235437</td>
<td>0.688347</td>
<td>0.40281</td>
</tr>
<tr>
<td>Gundam: The 08th MS Team: The Movie: Miller's Report (1990)</td>
<td>0.410177</td>
<td>56</td>
<td>0.234073</td>
<td>0.653911</td>
<td>0.419838</td>
</tr>
<tr>
<td>Divergence Eve (2003)</td>
<td>0.42936</td>
<td>57</td>
<td>0.230371</td>
<td>0.562537</td>
<td>0.348505</td>
</tr>
<tr>
<td>Harry Potter and the Chamber of Secrets (2002)</td>
<td>0.242023</td>
<td>70357</td>
<td>0.233238</td>
<td>0.247214</td>
<td>0.0139256</td>
</tr>
<tr>
<td>Spider-Man (2004)</td>
<td>0.238367</td>
<td>85252</td>
<td>0.232983</td>
<td>0.24575</td>
<td>0.0127868</td>
</tr>
<tr>
<td>National Geographic: Beyond the Movie: The Lord of the Rings (2001)</td>
<td>0.271251</td>
<td>2083</td>
<td>0.239683</td>
<td>0.310582</td>
<td>0.0795894</td>
</tr>
<tr>
<td>Snowy River: The McGregor Saga &quot;The Race&quot; (1994)</td>
<td>0.408275</td>
<td>100</td>
<td>0.230324</td>
<td>0.658084</td>
<td>0.32948</td>
</tr>
<tr>
<td>Harry Potter and the Prisoner of Azkaban (2004)</td>
<td>0.239653</td>
<td>61780</td>
<td>0.231915</td>
<td>0.244803</td>
<td>0.0148878</td>
</tr>
<tr>
<td>Red Green: Splited and Mounted 2 (1981)</td>
<td>0.351012</td>
<td>338</td>
<td>0.22827</td>
<td>0.417308</td>
<td>0.151037</td>
</tr>
<tr>
<td>Star Wars: Episode IV: A New Hope (1977)</td>
<td>0.234848</td>
<td>63028</td>
<td>0.224084</td>
<td>0.238881</td>
<td>0.0147775</td>
</tr>
</tbody>
</table>

Figure 6: Recommended movies for selected item rank by lower limit of confidence interval.
8.3 Content based Algorithm

To use content based algorithm, extra information of items need to be collected from the web. Netflix provides an API to collect its entire catalog titles including genre of movies into an xml file. The source code to collect the catalog is

```perl
#!/usr/bin/perl
use WWW::Netflix::API;
my %variable = do('vars.inc');
my $netflix_data = WWW::Netflix::API->new({
    consumer_key => $variable {consumer_key},
    consumer_secret => $variable {consumer_secret},
    content_filter => 'catalog.xml',
});
$netflix_data->REST->Catalog->Titles->Index;
$netflix_data->Get();
```

The xml file, catalog.xml, is about three hundred Megabyte and contains movie’s title, genre, and released year as in Figure 7. I try to use java DOM to parse the xml file, but it is too big to load all of it into memory. DOM needs over three Gigabyte to build a tree in memory and my computer has only three Gigabyte, so it is out of memory. A java parser will parse the xml file and collects all the genre of movies in Netflix’s training set and imports them into mysql table as in Figure 8. There are total of 501 different genres including action, drama, horror, fantasy, etc.
Figure 7: XML schema of Netflix movies catalog.
Figure 8: Netflix’s movie genres of training data set.
The content based algorithm finds and matches genres of all the movies to target movie. “Lord of the Rings: The Return of the King” movie has five genres and they are “Action and Adventure”, “Fantasy”, “Action Sci-Fi and Fantasy”, “Dramas Based on the Book”, “Dramas Based on Classic Literature”. The algorithm finds all the movies in the database and matches them with these five genres. Then it ranks the movies based on the number of matching genres. In Figure 9, the top thirty movies with common genres to LOTR:ROK are given. We can see that the algorithm work really well for LOTR:ROK because the top nine movies is relate with LOTR:ROK and about Lord of the Rings story.

The php code to display Figure 9 is follow:

```php
<html>
<head><title>Netflix Movies</title></head>
<body>

<?
// database server
$dbServer='mh213a.cs.sjsu.edu';

// username and password setup
$username='username';
$password='password';
$database='netflix';

// get movies id from user input
$movies_id = (isset($_GET['movies_id']))?$_GET['movies_id']:'';

// connect to database
$link = mysql_connect($dbServer,$username,$password) or die("Could not connect");
@mysql_select_db($database) or die( "Unable to select database");

// run the CalGenre script to generate movie list based on movies_id and store them to /movies_genres.txt
exec("java CalGenre $movies_id");

// open the text file to parse the movies for display
$file_handle = fopen("movies_genres.txt", "r");
$i=0;
$line_of_text = fgets($file_handle);
```
$values = explode("n", $line_of_text);
// get the title and released of movie from movie id.
$query="SELECT title,released FROM movies_title WHERE movieid=$values[0];";
$result=mysql_query($query);
$movie_title=mysql_result($result,0,"title");
$released=mysql_result($result,0,"released");

<h1>30 Recommend movies for: <? echo $movie_title." (".$released." )"; ?></h1>
<table border="2" cellspacing="2" cellpadding="2">
<tr>
<th><font face="Arial, Helvetica, sans-serif">Movies</font></th>
<th><font face="Arial, Helvetica, sans-serif">Number of Same Genres</font></th>
</tr>

<?
// get only the first 30 movies on the list
while (!feof($file_handle) and ($i <30)) {
    $line_of_text = fgets($file_handle);
    $values = explode(",", $line_of_text);
    $query2="SELECT title,released FROM movies_title WHERE movieid=$values[0];";
    $result2=mysql_query($query2);
    $movie_title=mysql_result($result2,0,"title");
    $released=mysql_result($result2,0,"released");

    <tr>
    <td><font face="Arial, Helvetica, sans-serif"><? echo $values[1]; ?></font></td>
    </tr>
    $i++;
}
?>
</table>
</body>
</html>

<table>
<thead>
<tr>
<th>Movies</th>
<th>Common Genres</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lord of the Rings: The Two Towers (2002)</td>
<td>5</td>
</tr>
<tr>
<td>Journey to the Center of the Earth (1959)</td>
<td>3</td>
</tr>
<tr>
<td>The Librarian: Quest for the Spear (2004)</td>
<td>3</td>
</tr>
<tr>
<td>Journey to the Center of the Earth (1999)</td>
<td>3</td>
</tr>
<tr>
<td>Masters of the Universe (1987)</td>
<td>3</td>
</tr>
<tr>
<td>The Shape of Things to Come (1979)</td>
<td>3</td>
</tr>
<tr>
<td>Mysterious Island (1961)</td>
<td>3</td>
</tr>
<tr>
<td>The Time Machine (2002)</td>
<td>3</td>
</tr>
<tr>
<td>The Sword and the Sorcerer (1982)</td>
<td>3</td>
</tr>
<tr>
<td>Call of the Wild (1972)</td>
<td>3</td>
</tr>
<tr>
<td>Beowulf (1999)</td>
<td>3</td>
</tr>
<tr>
<td>The Beastmaster (1982)</td>
<td>3</td>
</tr>
<tr>
<td>Around the World in 80 Days (1989)</td>
<td>3</td>
</tr>
<tr>
<td>Spider-Man: Bonus Material (2002)</td>
<td>3</td>
</tr>
<tr>
<td>The Long Ships (1964)</td>
<td>3</td>
</tr>
<tr>
<td>No Escape (1984)</td>
<td>3</td>
</tr>
<tr>
<td>The Heroic Tito (1992)</td>
<td>3</td>
</tr>
<tr>
<td>Lost Horizon (1937)</td>
<td>3</td>
</tr>
<tr>
<td>The Rocketeer (1991)</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 9: Recommended movies for selected item with content based Algorithm.
8.4 Making Predictions

The prediction algorithm is shown in equation (4.2). For each movie i that is unrated, Collaborative finds the subset of the similar movies that predict for i. This subset of movies is sorted with respect to the degree of the Pearson correlation and thirty movies with highest Pearson correlation will be used as Neighbor movies in equation (4.2).

For lower limit confidence interval algorithm, the subset of movies is sorted with respect to the degree of the lower limit confidence interval and thirty movies with highest lower limit confidence interval will be used as Neighbor movies in equation (4.2).

For Pearson correlation algorithm, the RMSE of Netflix’s Probe Data is 0.929651 as in Figure 10. The algorithm took 2759 seconds to make 1408395 predictions. It is about 510 predictions for every second, and it is a very slow process. Pearson correlation algorithm has a better improvement over Netflix’s Cinematch.
Figure 10: RMSE of Netflix’s Probe Data for Pearson correlation Algorithm.

Total predictions: 1408395
rmse: 0.929651
Errors: 0

kNN took: 2759 seconds
For lower limit confidence interval algorithm, the RMSE of Netflix’s Probe Data is 0.930027 as in Figure 11. The algorithm took 2189 seconds to make 1408395 predictions.

Figure 11: RMSE of Netflix’s Probe Data for lower limit confidence interval Algorithm.
Table 8.1 shows the RMSE results of all algorithms on the Probe Data of Netflix. Looking at the result, we can see that both item based Collaborative Filtering methods give better improvement over Netflix’s Cinematch algorithm.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE of Probe Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average rating of movie</td>
<td>1.0540</td>
</tr>
<tr>
<td>Netflix’s Cinematch</td>
<td>0.9474</td>
</tr>
<tr>
<td>Pearson correlation</td>
<td>0.929651</td>
</tr>
<tr>
<td>Pearson correlation with lower limit confidence interval</td>
<td>0.92669</td>
</tr>
</tbody>
</table>

Table 8.1: RMSE of Probe data for each algorithm.

9. Summary of Results

For Collaborative filtering approach, the lower limit of confidence interval algorithm gives better result than the traditional Pearson correlation. Confidence interval algorithm takes into account both of the size of the users who rate movie and the value of Pearson correlation. The movie with more numbers of ratings usually is a popular movie and it gets a high ranking in recommended list. The RMSE over Probe data of the confidence interval algorithm also has more improvement over Pearson correlation algorithm, 0.92669 over 0.929651. Both the item-based CF algorithms give better RMSE result than Netflix’s algorithm as in Table 8.1.

The content-based approach has better real-time performance than item-based CF methods when giving the list of recommended movies. It takes CF methods over one minute to do the calculation and display thirty recommended movies while the content-based method can do it in less than ten seconds. Because the content-based algorithm has extra genre information of the movie, it gives same or better recommended list than item-based CF algorithm.
10. Future Work

Due to the huge size of dataset, many algorithms cannot be used such as user-based Collaborative Filtering. It takes lots of time to do the calculation on Netflix’s dataset. It needs seven hours to import all dataset into mysql database and eight hours to calculate all Pearson correlations between each items. To recommend movies to the user, it takes over one minute to do calculation and display the result. We need to find a better way to improve the respond time for each query. Otherwise, user cannot wait that long for any web service.

11. Conclusion

This project has attempted a new approach in doing recommender systems on a large dataset. Confidence interval and extra information like genres have been incorporated into recommender system and give better improvement over Netflix’s algorithm. Both item-based algorithms improve RMSE over Netflix’s algorithm by 1.9 percent.

It is a challenge to implement a recommender system to work on this scale of data. I need to use different language, such as java, c++, perl, php, to manage the data and have efficient computation.
12. References


Appendix A: How to run recommender systems

1) Item based collaborative filtering with Pearson correlation:

Run `pearsonmovies movie_id`
Example: `pearsonmovies 14240`
To output text file `output.txt` with the top 30 recommended movies for `movie_id 14240`

```
output.txt file:
14240
11521,0.785264,114244,0.785264,0.787477,0.00221294
2452,0.760399,111120,0.760399,0.762868,0.00246906
14961,0.749107,66208,0.749107,0.752431,0.00332379
7057,0.721817,68248,0.721817,0.725391,0.00357425
7230,0.707742,66918,0.707742,0.711503,0.00376141
15521,0.695058,20,0.695058,0.869991,0.174934
9979,0.581179,26,0.581179,0.790563,0.209383
10336,0.559706,43,0.559706,0.736281,0.176575
6725,0.550997,42,0.550997,0.732297,0.1813
15571,0.541402,13,0.541402,0.841398,0.299997
4457,0.523518,37,0.523518,0.724623,0.201105
17616,0.500247,49,0.500247,0.685078,0.184831
12341,0.496717,34,0.496717,0.714816,0.2181
14124,0.494782,45,0.494782,0.688347,0.193565
8430,0.470177,55,0.470177,0.653911,0.183734
10352,0.467648,908,0.467648,0.516979,0.0493313
11883,0.461182,53,0.461182,0.650403,0.189221
14071,0.454057,30,0.454057,0.699847,0.24579
11598,0.452459,45,0.452459,0.685355,0.206076
17337,0.443348,65,0.443348,0.620188,0.17684
4908,0.437958,65,0.437958,0.616055,0.178097
8091,0.437632,1727,0.437632,0.479997,0.037364
8737,0.436805,55,0.436805,0.629193,0.192388
10313,0.436736,1512,0.436736,0.476654,0.0399185
9943,0.435509,51,0.435509,0.634896,0.199387
11856,0.435207,38,0.435207,0.662965,0.227488
7408,0.434322,83,0.434322,0.594337,0.160015
8145,0.433127,51,0.433127,0.633141,0.200014
8144,0.426825,68,0.426825,0.603806,0.176981
12964,0.426196,25,0.426196,0.702952,0.276756
```

movie_id, pearson correlation, Number of Rating, lower limit, upper limit, width

Run php script `netflixmovies.php?movies_id=movies_id` to display the top 30 recommended movies for the selected movie_id on the website as in figure 5.
<?
$dbServer='mh213a.cs.sjsu.edu';
$username='username';
$password='password';
$database='netflix';
$movies_id = (isset($_GET['movies_id']))?$_GET['movies_id']:'';
$link = mysql_connect($dbServer,$username,$password) or die("Could not connect");
@mysql_select_db($database) or die("Unable to select database");
exec("pearsonmovies.sh $movies_id");
$file_handle = fopen("output.txt", "r");

$i=0;
$line_of_text = fgets($file_handle);
$values = explode("\n", $line_of_text);
$query="SELECT title,released FROM movies_title WHERE movieid=$values[0];";
$result=mysql_query($query);
$movel_title=mysql_result($result,0,"title");
$released=mysql_result($result,0,"released");
?>
<h1>30 Recommend movies for: <? echo $moviel_title." (".$released. "); ?></h1>
<table border="2" cellspacing="2" cellpadding="2">
<tr>
<th><font face="Arial, Helvetica, sans-serif">Movies</font></th>
<th><font face="Arial, Helvetica, sans-serif">pearson correlation</font></th>
<th><font face="Arial, Helvetica, sans-serif">Number of Rating</font></th>
<th><font face="Arial, Helvetica, sans-serif">Lower Limit</font></th>
<th><font face="Arial, Helvetica, sans-serif">Upper Limit</font></th>
<th><font face="Arial, Helvetica, sans-serif">Width</font></th>
</tr>
<?
while (!feof($file_handle)) and ($i <30)) {
    $line_of_text = fgets($file_handle);
    $values = explode("", $line_of_text);
    $query2="SELECT title,released FROM movies_title WHERE movieid=$values[0];";
    $result2=mysql_query($query2);
    $moviel_title=mysql_result($result2,0,"title");
    $released=mysql_result($result2,0,"released");
</tr>
<?
}<table>
Run pearsonprediction to generate text file outputpearsonpred.txt as in figure 10.

2) Item based collaborative filtering with lower limit confidence interval:
Run confidencemovies movie_id
Example: confidencemovies 14240
To output text file confidence.txt with the top 30 recommended movies for movie_id 14240

confidence.txt file:
14240
11521,0.785264,114244,0.78303,0.787477,0.00444615
2452,0.760399,111120,0.757908,0.762868,0.00496024
14961,0.749107,66208,0.745745,0.752431,0.0068573
7057,0.721817,68248,0.718204,0.725391,0.00718743
7230,0.707742,66918,0.70394,0.711503,0.00756335
10352,0.467648,908,0.415221,0.516979,0.101759
8091,0.437632,1727,0.398693,0.474997,0.0763043
10313,0.436736,1512,0.39502,0.476654,0.0816344
15521,0.695058,20,0.364707,0.869991,0.505284
10336,0.559706,43,0.311767,0.736281,0.424514
6725,0.550997,42,0.296759,0.732297,0.435539
17616,0.500247,49,0.254902,0.685078,0.430176
9979,0.581179,26,0.250132,0.790563,0.540431
9628,0.251932,65317,0.244736,0.2591,0.0143647
15790,0.36189,232,0.244485,0.468838,0.224535
1710,0.343157,307,0.240451,0.438264,0.197813
14124,0.494782,45,0.235437,0.688347,0.45291
8430,0.470177,55,0.234073,0.653911,0.419838
7279,0.42353,87,0.233731,0.582237,0.348505
movie_id, pearson correlation, Number of Rating, lower limit, upper limit, width

Run php script confidencemovies.php?movies_id=movies_id to display the top 30 recommended movies for the selected movie_id on the website as in figure 6.

Confidencemovies.php file:
<html>
<head><title>Netflix Movies</title></head>
<body>
<?
$dbServer='mh213a.cs.sjsu.edu';
$username='username';
$password='password';
$database='netflix';

$movies_id = (isset($_GET['movies_id']))?$_GET['movies_id']:";
$link = mysql_connect($dbServer,$username,$password) or die("Could not connect");
@mysql_select_db($database) or die("Unable to select database");
echo("./confidencemovies.sh $movies_id");
$file_handle = fopen("confidence.txt", "r");

$i=0;
$line_of_text = fgets($file_handle);
$values = explode("\n", $line_of_text);
$query="SELECT title,released FROM movies_title WHERE movieid=$values[0];";
$result=mysql_query($query);
$movie_title=mysql_result($result,0,"title");
$released=mysql_result($result,0,"released");
?>
<h1>30 Recommend movies for: <? echo $movie_title." (".$released." )"; ?></h1>
<table border="2" cellspacing="2" cellpadding="2">
<tr>
<th><font face="Arial, Helvetica, sans-serif">Movies</font></th>
<th><font face="Arial, Helvetica, sans-serif">pearson correlation</font></th>
<th><font face="Arial, Helvetica, sans-serif">Number of Rating</font></th>
<th><font face="Arial, Helvetica, sans-serif">Lower Limit</font></th>
<th><font face="Arial, Helvetica, sans-serif">Upper Limit</font></th>
<th><font face="Arial, Helvetica, sans-serif">Width</font></th>
</tr>
</table>

32
Run confidenceprediction to generate text file confidenceprediction.txt as in figure 11.

3) Content based Algorithm:
Run java CalGenre movie_id
Example: java 14240 to generate text file movies_genres.txt with the top 30 recommended movies for movie_id 14240

movies_genres.txt file:
14240
13,5
2452,5
7057,5
7230,5
8091,5
10313,5
10352,5
Run php script genremovies.php?movies_id=movie_id to display the top 30 recommended movies for selected movie_id on the website as in figure 9.

4) Netflix dataset:
• training_set folder contains all 17770 movie rating files with Quadruples of <movie_id, user_id, rating, date>
• example mv_0000001.txt contains
  1:  
  1488844,3,2005-09-06  
  822109,5,2005-05-13  
  885013,4,2005-10-19  
  30878,4,2005-12-26  
  823519,3,2004-05-03  
  893988,3,2005-11-17  
  124105,4,2004-08-05  
  1248029,3,2004-04-22  
  1842128,4,2004-05-09  
  2238063,3,2005-05-11  
  1503895,4,2005-05-19  
  2207774,5,2005-06-06  
  …
mv_0000002.txt contains

<table>
<thead>
<tr>
<th>Movie ID</th>
<th>Rating</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>2059652</td>
<td>4</td>
<td>2005-09-05</td>
</tr>
<tr>
<td>1666394</td>
<td>3</td>
<td>2005-04-19</td>
</tr>
<tr>
<td>1759415</td>
<td>4</td>
<td>2005-04-22</td>
</tr>
<tr>
<td>1959936</td>
<td>5</td>
<td>2005-11-21</td>
</tr>
<tr>
<td>998862</td>
<td>4</td>
<td>2004-11-13</td>
</tr>
<tr>
<td>2625420</td>
<td>2</td>
<td>2004-12-06</td>
</tr>
<tr>
<td>573975</td>
<td>3</td>
<td>2005-07-21</td>
</tr>
</tbody>
</table>

- Movies.data is a binary file that contains all movies with users and ratings
- users.data is a binary file that contains all users with movies and ratings as figure 4 shows

movie_titles.txt contains movie_id, released year, movie title data

<table>
<thead>
<tr>
<th>Movie ID</th>
<th>Released Year</th>
<th>Movie Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2003</td>
<td>Dinosaur Planet</td>
</tr>
<tr>
<td>2</td>
<td>2004</td>
<td>Isle of Man TT 2004 Review</td>
</tr>
<tr>
<td>3</td>
<td>1997</td>
<td>Character</td>
</tr>
<tr>
<td>4</td>
<td>1994</td>
<td>Paula Abdul's Get Up &amp; Dance</td>
</tr>
<tr>
<td>5</td>
<td>2004</td>
<td>The Rise and Fall of ECW</td>
</tr>
<tr>
<td>6</td>
<td>1997</td>
<td>Sick</td>
</tr>
<tr>
<td>7</td>
<td>1992</td>
<td>8 Man</td>
</tr>
<tr>
<td>8</td>
<td>2004</td>
<td>What the #$*! Do We Know!?</td>
</tr>
<tr>
<td>9</td>
<td>1991</td>
<td>Class of Nuke 'Em High 2</td>
</tr>
<tr>
<td>10</td>
<td>2001</td>
<td>Fighter</td>
</tr>
<tr>
<td>11</td>
<td>1999</td>
<td>Full Frame: Documentary Shorts</td>
</tr>
<tr>
<td>12</td>
<td>1947</td>
<td>My Favorite Brunette</td>
</tr>
</tbody>
</table>

- Probe.txt: probe data where the algorithms run on. It contains only movie_id and user_id without rating number.

<table>
<thead>
<tr>
<th>Movie ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>30878</td>
</tr>
<tr>
<td>2647871</td>
</tr>
<tr>
<td>1283744</td>
</tr>
<tr>
<td>2488120</td>
</tr>
<tr>
<td>317050</td>
</tr>
</tbody>
</table>

...
• Catalog.xml: contains movie’s title, genre, and released year as in Figure 7.
• Movies_title_all.sql: mysql database file contains all 17770 movies with movie_id, released year, movie_title, genre_1, genre_2, etc...
• pearson.data contains pearson correlation value between each movies.