| For office use only | | Team Control Number | For office use only | |
|---|---|---|---|---|
| T1 | | **36428** | F1 | |
| T2 | | | F2 | |
| T3 | | Problem Chosen | F3 | |
| T4 | | **B** | F4 | |

**2015 Mathematical Contest in Modeling (MCM) Summary Sheet**
(Attach a copy of this page to each copy of your solution paper.)


## Abstract

With the rapid development in Internet technology, the trend of appreciating films online is becoming more and more heated. In order to realize the maximum profits, it is essential for content providers to collect users' information, data, including both their basic personal identities and implicit browsing histories and habits, to analyse their characteristic film styles and favor and eventually recommend potential best choices. Actually, this kind of application pipeline has been widely used in those famous sites like NetFlix[2], Hulu[12] and Amazon[9], etc. In this competition problem, we are required to implement a simple version of such a system. Therefore, we focus our attention on the utilization of raw data provided by MovieLens dataset [10] and model relatively precious and objective features for both users and movies. Later, we are able to analyse any specific user's film favor based on statistic approaches. In addition, with the aid of Support Vector Machine(SVM)[11] and Classification and Regression Tree(C&RT)[8], we innovatively mine the implicit information from rating records and establish a predictive model based on users' and movies' features. Finally, we set experiments to compare our models with the traditional collaborative filtering algorithm[3] and get some interesting conclusions.

# An implement of film recommendation system based on statistical models

Team #36428

January 28, 2015

**Abstract**

With the rapid development in Internet technology, the trend of appreciating films online is becoming more and more heated. In order to realize the maximum profits, it is essential for content providers to collect users' information, data, including both their basic personal identities and implicit browsing histories and habits, to analyse their characteristic film styles and favor and eventually recommend potential best choices. Actually, this kind of application pipeline has been widely used in those famous sites like NetFlix[2], Hulu[12] and Amazon[9], etc. In this competition problem, we are required to implement a simple version of such a system. Therefore, we focus our attention on the utilization of raw data provided by MovieLens dataset [10] and model relatively precious and objective features for both users and movies. Later, we are able to analyse any specific user's film favor based on statistic approaches. In addition, with the aid of Support Vector Machine(SVM)[11] and Classification and Regression Tree(C&RT)[8], we innovatively mine the implicit information from rating records and establish a predictive model based on users' and movies' features. Finally, we set experiments to compare our models with the traditional collaborative filtering algorithm[3] and get some interesting conclusions.

**Key Words:** recommendation system; collaborative filtering; SVM regression; C&RT

# Contents

# 1 Introduction

With the rapid progress in information technology, people have entered into the epoch with overloaded information. From the prospective of the content provider, it is urgent and necessary to provide accurate and specific information to a particular user, mostly corresponding with his or her personal information and browsing records. Such an system is commonly considered as a automatic recommendation based on data mining, machine learning and statistic models[1]. The most well-known and powerful applications of such system in industry fields include Amazon's goods recommendation system, Hulu and NetFlix's film recommendation platform, etc. Actually, no matter we have realized or not, we are experiencing this technology every day.

In this problem, we are generally required to implement such a film recommendation system based on MovieLens[cite] dataset. The MovieLens dataset provides us with 100,000 records of users' rates to over 1,000 films ranging from 1 to 5. In addition, it offers us with users' fundamental personal identities: ages and occupation. Besides, each films are classified into 19 different themes(romantic, drama, comedy...). On the basis of these information, the problem requires us to realize the following three sub-goals.

- **Sub-problem 1:** Establish the users' film favor in mathematical models and analyse some particular users' film characteristic.
- **Sub-problem 2:** Establish the films' theme models and combine with models in Subproblem 1 to recommend 5 films for users mentioned above.
- **Sub-problem 3:** Realize a recommendation system for new users with only register information.

We solve sub-problem 1 by modelling users' feature and films' feature and analysing the relationship between them. In sub-problem 2 and 3, we furthermore try to determine the analytical numerical model between the combination of users, film feature and the final rating score by SVM and C&RT. Finally we test and compare our model with state-of-the-art: collaborative filtering algorithm.

The structure of the paper is organized by three sub-problems, respectively. In each subproblem, we give our own assumptions, models and experiment results. The final conclusion is given in section 5. The codes and are shown in Appendix.

# 2 Solution for sub-problem 1

## 2.1 Analysis and Assumptions

In this sub-problem, our goal is to analyse the film preference model for a specific user. Due to the fact that this sub-problem's target object are some specified individuals, we assume that this sub-problem is meant to exhibit the basic preference for each user. In other words, our ultimate output of this goal is giving out each user's subjective film favor. Thus, there is no need to calculate and deduce the overall preference of all users. The most fundamental idea is to simply add the user's rate together and calculate the average value to represent his or her preference.

Nonetheless, when we recall the actual rating situation: one film may score an abnormal average lower score under thousands of rates due to various reasons(actors, directors, back-

grounds, etc), although, its theme satisfies one particular user's favor well. In this degree, we assume that if a film's average score is low whereas our target user gives a "High Five" for it, the reason lies on he or her *is fascinated by* the films' theme. Therefore, the model should reflect this idea and gives a heavier weight when this situation occurs. In addition, the summation of user's preference on all film themes should be one, naturally, it means a normalization operation.

According to analysis above, we summarize our assumptions as below:

- **Assumption 1-1:** The user's preference has nothing to do with his or her age, occupation, merely depend on his or her history rating results.
- **Assumption 1-2:** The higher the ratio between the user's score to a film and its average score, the deeper the user prefers it.
- **Assumption 1-3:** The summation of one individual's preference on all the themes should be 1.

## 2.2 Models

Suppose we have the film set $F = (f_1, f_2, \ldots, f_n)$, which collects all the films mentioned in the dataset and $U = (u_1, u_2, \ldots, u_m)$ to denote user set. For each rating record, it is merely a dual function between $U$ and $F$, we define $R(i, j)$ is user $u_i's$ rate to film $f_j$.

$$R(i,j) = \begin{cases} 0 & \text{no rating record in dataset from } u_i \text{ to } m_j, \\ score & \text{exist rating record in dataset from } u_i \text{ to } m_j. \end{cases} \tag{1}$$

First, we calculate the general average score $Avg_{f_j}$ for each film in film set.

$$AVG(f_j) = \sum_{i=1}^{m} \frac{R(i,j)}{count(R(i,j) \neq 0)} \tag{2}$$

Then, we calculate the summation of "scaled-version" rate score multiplied by a film-style "mask-vector" $S(f_j)$, a $s-$ length binary vector defined in file:$u.item$(in this case s equals to 18):

$$UFEAT(u_i)' = \sum_{j=1}^{n} \frac{S(f_j) * R(i,j)}{AVG(f_j)} \tag{3}$$

After normalized the summation of each dimension in $UFEAT(u_i)'$ to 1, just as mentioned in**Assumption 1-3**, we seize the answer in this sub-problem: $UFEAT(u_i)$.

## 2.3 Experiment Result

The matrix for specialized 10 people(108, 133, 228, 232, 336, 338, 545, 613, 696, 777) can been viewed in table 1: for a particular user, we quantify his or her film favor in a column. For example, user 108 prefers Drama(0.18) at most, and we can deduce that user 108 has never rate and appreciate Fantasy, Film-Noir and Horror films(their indexes are zero). In addition, we visualize the answer for user 108 and user 133 in figure 1.

| | 108 | 133 | 228 | 232 | 336 | 338 | 545 | 613 | 696 | 777 |
|---|---|---|---|---|---|---|---|---|---|---|
| Action | 0.13 | 0.12 | 0.06 | 0.08 | 0.08 | 0.03 | 0.20 | 0.10 | 0.06 | 0.06 |
| Adventure | 0.08 | 0.05 | 0.08 | 0.04 | 0.04 | 0.02 | 0.13 | 0.05 | 0.03 | 0.02 |
| Animation | 0.01 | 0.00 | 0.00 | 0.01 | 0.00 | 0.03 | 0.03 | 0.02 | 0.00 | 0.01 |
| Children | 0.02 | 0.07 | 0.05 | 0.03 | 0.02 | 0.01 | 0.05 | 0.02 | 0.00 | 0.01 |
| Comedy | 0.12 | 0.11 | 0.06 | 0.12 | 0.44 | 0.21 | 0.11 | 0.09 | 0.04 | 0.22 |
| Crime | 0.02 | 0.03 | 0.02 | 0.03 | 0.03 | 0.02 | 0.02 | 0.07 | 0.08 | 0.04 |
| Documentary | 0.00 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Drama | 0.18 | 0.19 | 0.38 | 0.29 | 0.12 | 0.22 | 0.08 | 0.25 | 0.38 | 0.32 |
| Fantasy | 0.00 | 0.02 | 0.00 | 0.01 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 |
| Film-Noir | 0.00 | 0.00 | 0.00 | 0.01 | 0.00 | 0.03 | 0.00 | 0.02 | 0.02 | 0.00 |
| Horror | 0.00 | 0.01 | 0.03 | 0.00 | 0.01 | 0.01 | 0.04 | 0.01 | 0.04 | 0.02 |
| Musician | 0.03 | 0.00 | 0.00 | 0.04 | 0.01 | 0.01 | 0.03 | 0.00 | 0.00 | 0.01 |
| Mystery | 0.02 | 0.05 | 0.01 | 0.02 | 0.01 | 0.06 | 0.01 | 0.02 | 0.10 | 0.01 |
| Romance | 0.14 | 0.09 | 0.11 | 0.13 | 0.13 | 0.16 | 0.06 | 0.09 | 0.07 | 0.06 |
| Sci-Fi | 0.08 | 0.08 | 0.02 | 0.06 | 0.03 | 0.03 | 0.09 | 0.08 | 0.00 | 0.03 |
| Thriller | 0.08 | 0.12 | 0.06 | 0.05 | 0.05 | 0.09 | 0.08 | 0.09 | 0.11 | 0.10 |
| War | 0.07 | 0.05 | 0.11 | 0.07 | 0.01 | 0.06 | 0.05 | 0.07 | 0.08 | 0.10 |
| Western | 0.00 | 0.00 | 0.00 | 0.01 | 0.02 | 0.01 | 0.02 | 0.02 | 0.00 | 0.00 |

Table 1: The analytical result for specific users' preference

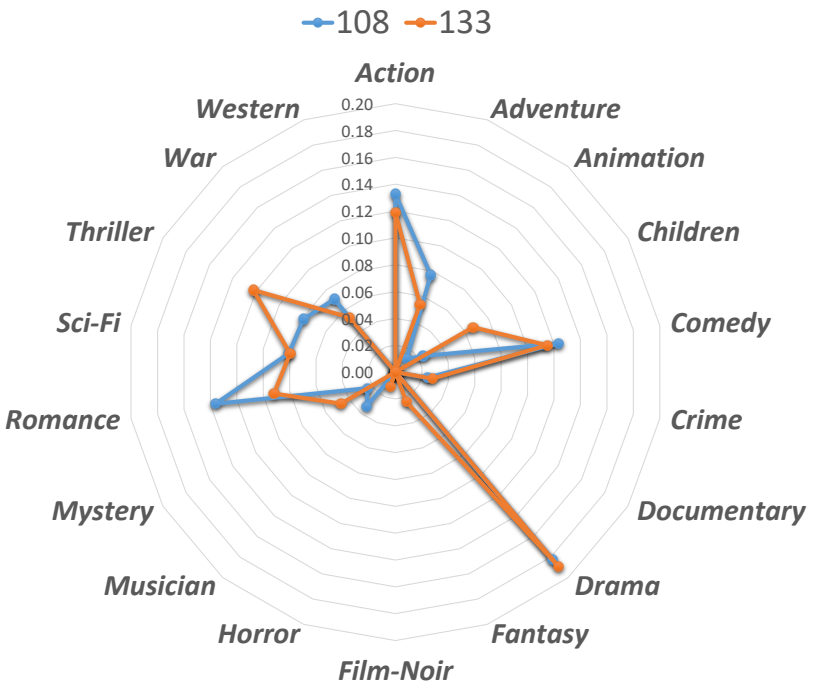


Figure 1: The rader figure of two user preference

# 3 Solution for sub-problem 2

## 3.1 Analysis and Assumptions

In sub-problem 2, the demand is even more critical: we need to exploit the internal information for each film and eventually implement the function of recommend five books for specific users. In this aspect, our model is implemented in two steps: the first step is to modelling film datasets; the second is to exploring the relationship between film model and user model with the aid of rating records.

First, let us consider the film-modelling task, which is similar as the one in sub-problem 1. In order to simplify the model, we give such an assumption just like the one mentioned before[7]:

- **Assumption 2-1:** Each film can been appreciated from the following angles for a single individual separately: personal preference, age and occupation.

We construct three relationships between film and personal characteristics: film-theme favor, film-age and film-occupation. The first relationship is calculated as a "user-preference weight score". For a particular film $f_j$, we deduce its film-theme favor as follow:

$$FFEAT(f_j)_{favor} = \frac{\sum_{i=1}^{m} S(f_j) \times R(i,j) \times UFEAT(u_i)}{\sum_{i=1}^{m} UFEAT(u_i)} \tag{4}$$

In order to seize film-age feature, we manually divide ages into six parts:0 16, 17 24, 25 32, 33 40, 41 48 and over 49, indexed from 1 to 6, respectively. The film-age feature is then calculated by simply counting the average rating scores in each age groups and finally we are able to get the 6-dimension feature: $FFEAT(f_j)_{age}$. The same idea is applied on acquiring $FFEAT(f_j)_{occupation}$, a 21-dimension vector(the number of occupation provided in $u.data$ is 21. Finally, we combine $FFEAT(f_j)_{favor}, FFEAT(f_j)_{age}, FFEAT(f_j)_{occupation}$ together as $FFEAT(f_j)$, which is the model of all films.

The second task is explore the internal relationship between user feature and film feature, which is the core of the whole system. We give a assumption as following:

- **Assumption 2-2:** A pair of rating score $R(u_i, f_j)$ is a predictable result corresponding with both user feature: $UUFEAT(u_i)$ and film feature: $FFEAT(f_j)$.

Meanwhile, another perspective of this sub-problem is the state-of-the-art method: collaborative filtering algorithm, a user-angle approach[9, 3]. The basic idea is that we can always find the similar class users groups by comparing *"similarity degree"* among user features: $UUFEAT(ui)$. And we design our recommendation system in another way on the basis on the following assumption:

- **Assumption 2-3:** Given two users features $UUFEAT(u_i), UUFEAT(u_j)$, if their similarity in quantity is less than one particular threshold given in advance, we can conclude they have similar film preference and should recommend them with counterparts' high-rating films.

## 3.2 Models

We employee two heated Machine Learning model to construct our model. And compare the performance with the traditional Collaborative Filtering Algorithm[cite]. We introduce them

one-by-one in each subsection. Note that the first two models are based on **Assumption 2-2** and **Assumption 2-3**. The last model is a comparing model transplanted from the state-of-the-art.

### 3.2.1 Support Vector Machine regression

The Support Vector Machine(SVM)[cite] is a classical classifier/regression widely used in machine learning, pattern recognition fields[cite]. The basic idea of SVM is quite similar with the one of Linear Regression. Nevertheless, the extended core idea in SVM is the "maximized-margin" theory: suppose we are trying to find a best separation hyperplane in $\mathbb{R}^n$, in SVM theory, the best solution should not only meet up the least error, but also maximize the margin among the hyperplane and all the given points. In other words, the hyperplane is carefully restricted in avoid of over-fitting issues.[cite] In figure 2, we provide a simple case in $\mathbb{R}^2$: in this case the third column is the better case compared with the first two columns.

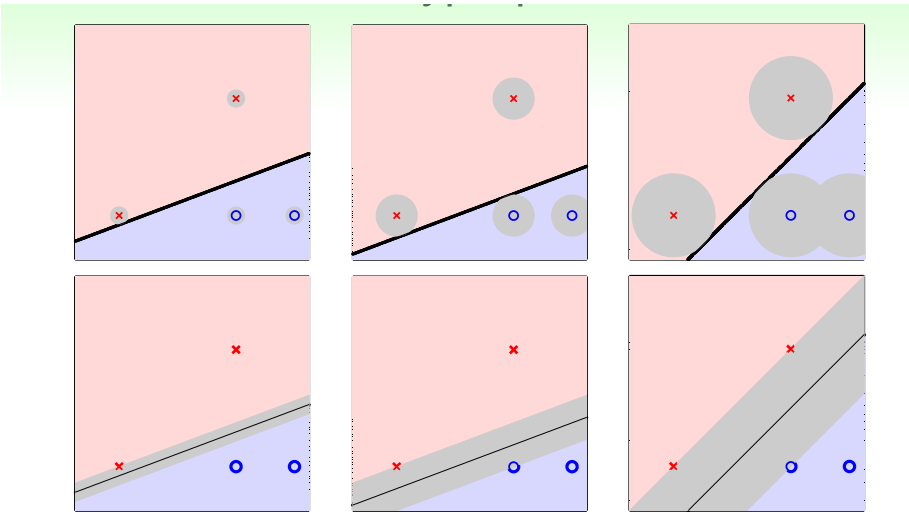

Figure 2: A simple case of hyperplane choices in $\mathbb{R}^2$

By utilize a batch of records as training data(600000) in $u.data$, we have actually acquired a joint feature of $UUFEAT$ and $FFEAT$, denoted as $X = x_i|i = 1, 2, 3...p$ and their regression value $Y = y_i|i = 1, 2, 3...p$. We train the SVM parameter $w, b$ by minimize following object function. Actually, the first term is just the common object: least error measure; the second term can be seen as a L2-regularization operation: we want to lower the risk of over-fitting by reduce the norm of $w$. Overall, the objective function is convex and can be solved with convergent solutions.

$$\min_{w,b} \sum_{i=1}^{p} \left(1 - y_i \left(w^T x_i + b\right)\right) + \frac{1}{2} w^T w \tag{5}$$

In fact, the upper SVM framework has been perfectly implemented by libsvm[4], a integral and powerful SVM version in Matlab. We train our SVM model under it with default RBF kernel[5] and some other parameter tuning settings. As long as we have acquired $w, b$, for a giving person $u_i$, we can calculate all the feature matches $X_{u_i}$ by exhausting all the films in dataset and calculating the potential $Y_{u_i}$. Eventually, as the problem required, we give out the top-five films with highest predicted rating scores.

### 3.2.2 Classification and Regression Tree

### 3.2.3 Collaborative Filtering Algorithm

As mentioned in **Assumption 2-3**, the core idea of collaborative filtering algorithm is to calculate the similarity between any given two users with their features. In this case, we synthesize users' preference feature $UUFEAT(u_i)$, age and occupation together into a ultimate feature vector $UF(u_i)$. By calculating the Pearson coefficient[6] between $UF(u_i), UF(u_j)$:

$$sim\,(i,j) = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{UF(u_i)-\overline{UF(u_i)}}{S_{UF(u_i)}}\right)\left(\frac{UF(u_j)-\overline{UF(u_j)}}{S_{UF(u_j)}}\right) \tag{6}$$

$$where, \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

$$and, S_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i-\overline{X}\right)^2}$$

we are able to quantify the similarity degree between $u_i$ and $u_j$.

One particular special step in this model is to calculate the similarity between different jobs, which is normally a non-numerical answer. However, according to the fuzzy theory, we calculate it in semi-quantity form by *Microsoft Bing* search engine: we enter two occupations into search box and take the resultant search items number as their similarity(those items with two mutual occupations). After normalization operation, we achieve Table 2, a symmetric matrix to define occupation similarity.

| | technician | other | writer | executive | administrator | |
|---|---|---|---|---|---|---|
| technician | 1.00 | 0.13 | 0.13 | 0.14 | 0.13 | |
| other | 0.13 | 0.13 | 0.13 | 0.13 | 0.13 | |
| writer | 0.13 | 0.13 | 1.00 | 0.16 | 0.16 | |
| executive | 0.14 | 0.13 | 0.16 | 1.00 | 0.10 | |
| administrator | 0.13 | 0.13 | 0.16 | 0.10 | 1.00 | |
| student | 0.12 | 0.13 | 0.10 | 0.16 | 0.13 | |

A part of the occupation similarity matrix

Finally, we rate all the films in the dataset by the Perason coefficient-weighted sum of all users rates in $u$

## 3.3 Experiment Result

The summary of recommendation result for specific user is shown in Table 3 as below.

| User ID | **SVM regression(no scaled score)** | **CRT** | **Collaborative filtering(no scaled score)** |
|---|---|---|---|
| 108 | Film Id: 347 predicted score:4.9245 | | Film Id: 814 predicted score:5.3837 |
| | Film Id: 7 predicted score:4.9103 | | Film Id: 1201 predicted score:5.3081 |
| | Film Id: 813 predicted score:4.6293 | | Film Id: 1122 predicted score:4.9761 |
| | Film Id: 317 predicted score:4.6005 | | Film Id: 1536 predicted score:4.8043 |
| | Film Id: 1405 predicted score:4.6005 | | Film Id: 1293 predicted score:4.5755 |
| | | | |
| 133 | Film Id: 192 predicted score:4.6002 | | Film Id: 1122 predicted score:5.0261 |
| | Film Id: 258 predicted score:4.6002 | | Film Id: 814 predicted score:5.025 |
| | Film Id: 193 predicted score:4.6001 | | Film Id: 1201 predicted score:4.9492 |

| | | | |
|---|---|---|---|
| | Film Id: 64 predicted score:4.6001 | | Film Id: 1536 predicted score:4.7916 |
| | Film Id: 357 predicted score:4.6001 | | Film Id: 1293 predicted score:4.6294 |
| | | | |
| 228 | Film Id: 1135 predicted score:8.2108 | | Film Id: 1653 predicted score:4.8226 |
| | Film Id: 589 predicted score:6.7022 | | Film Id: 1599 predicted score:4.1469 |
| | Film Id: 404 predicted score:5.288 | | Film Id: 1467 predicted score:3.9926 |
| | Film Id: 14 predicted score:5.1448 | | Film Id: 1594 predicted score:3.9886 |
| | Film Id: 742 predicted score:5.0866 | | Film Id: 1536 predicted score:3.8258 |
| | | | |
| 232 | Film Id: 56 predicted score:4.6005 | | Film Id: 1201 predicted score:4.9919 |
| | Film Id: 1149 predicted score:4.6002 | | Film Id: 1536 predicted score:4.9518 |
| | Film Id: 923 predicted score:4.6002 | | Film Id: 1122 predicted score:4.912 |
| | Film Id: 48 predicted score:4.6002 | | Film Id: 814 predicted score:4.8787 |
| | Film Id: 170 predicted score:4.6002 | | Film Id: 1599 predicted score:4.7434 |
| | | | |
| 336 | Film Id: 204 predicted score:4.6004 | | Film Id: 1467 predicted score:4.0908 |
| | Film Id: 153 predicted score:4.6003 | | Film Id: 1599 predicted score:3.8075 |
| | Film Id: 216 predicted score:4.6002 | | Film Id: 1189 predicted score:3.7232 |
| | Film Id: 42 predicted score:4.6002 | | Film Id: 1536 predicted score:3.4807 |
| | Film Id: 762 predicted score:4.6002 | | Film Id: 1594 predicted score:3.3989 |
| | | | |
| 338 | Film Id: 603 predicted score:4.6005 | | Film Id: 1536 predicted score:4.8951 |
| | Film Id: 663 predicted score:4.6005 | | Film Id: 1201 predicted score:4.8449 |
| | Film Id: 170 predicted score:4.6003 | | Film Id: 1599 predicted score:4.8054 |
| | Film Id: 408 predicted score:4.6003 | | Film Id: 814 predicted score:4.7241 |
| | Film Id: 197 predicted score:4.6002 | | Film Id: 1122 predicted score:4.6906 |
| | | | |
| 545 | Film Id: 238 predicted score:5.5413 | | Film Id: 814 predicted score:4.3884 |
| | Film Id: 121 predicted score:4.6002 | | Film Id: 1122 predicted score:4.2495 |
| | Film Id: 257 predicted score:4.6001 | | Film Id: 1467 predicted score:4.2378 |
| | Film Id: 472 predicted score:4.6001 | | Film Id: 1189 predicted score:4.2147 |
| | Film Id: 230 predicted score:4.6001 | | Film Id: 1293 predicted score:3.9502 |
| 613 | Film Id: 194 predicted score:4.6004 | | Film Id: 1536 predicted score:4.918 |
| | Film Id: 178 predicted score:4.6002 | | Film Id: 1201 predicted score:4.888 |
| | Film Id: 887 predicted score:4.6002 | | Film Id: 1122 predicted score:4.8509 |
| | Film Id: 750 predicted score:4.6001 | | Film Id: 814 predicted score:4.8425 |
| | Film Id: 357 predicted score:4.6001 | | Film Id: 1293 predicted score:4.6415 |
| | | | |
| 696 | Film Id: 258 predicted score:4.9815 | | Film Id: 1201 predicted score:4.9422 |
| | Film Id: 276 predicted score:4.6344 | | Film Id: 1536 predicted score:4.8192 |
| | Film Id: 9 predicted score:4.6002 | | Film Id: 1122 predicted score:4.8121 |
| | Film Id: 166 predicted score:4.6001 | | Film Id: 814 predicted score:4.7529 |
| | Film Id: 134 predicted score:4.6001 | | Film Id: 1293 predicted score:4.6233 |
| | | | |
| 777 | Film Id: 880 predicted score:4.6005 | | Film Id: 1201 predicted score:4.9848 |
| | Film Id: 168 predicted score:4.6004 | | Film Id: 814 predicted score:4.9263 |
| | Film Id: 56 predicted score:4.5999 | | Film Id: 1122 predicted score:4.9224 |
| | Film Id: 87 predicted score:4.5999 | | Film Id: 1536 predicted score:4.9125 |

| | Film Id: 117 predicted score:4.5997 | | Film Id: 1599 predicted score:4.629 |
|---|---|---|---|

Table 3: Recommendation result with RAW score under three models.

From the table above, we can observe some interesting phenomenon and conclusions. First and foremost, the SVM regression method and Collaborative filtering method give us absolutely different results. In our SVM regression algorithm, we can note that for each person, the recommendation result is quite personalized. When we go on a step further to trace each film by id with $u.item$, we find that they are quite coherence with person's characteristic. To exemplifier this idea, we take User 133 as an instance: user 133 is a 53-year-old engineer; our SVM recommendation system presents film 347(comedy and drama), film 7(drama and sci-fi), film 813(documentary), etc as results. According to our experience, the results are quite coherent with the identity of a middle-age engineer.

However, when we scan over the results generated by collaborative filtering model, we can find that most users are recommended with similar films(1201,1122,814, etc), although their rank are different. When we further track those films are mostly with very high average rates in their own themes(You can imagine those most heated films like *Titanic, Avater* as examples). Recalling the tip that these data are collected in a relatively short-term time interval(**September 19th, 1997 through April 22nd, 1998**), this result is also somewhat persuasive. Generally speaking, as far as we are concerned, our own SVM regression model is better than the collaborative model.

# 4 Solution for sub-problem 3

## 4.1 Analysis and Assumptions

## 4.2 Models

## 4.3 Experiment Result

# 5 Conclusion

In this paper, we successfully implement a automatic film recommendation based on MovieLens dataset. We analysis both users' models and films' models based on the dataset and people's own experience in film-rating experiences. Furthermore, we exploit the deeper relationship between users' models and the films' one; by employing SVM regression model, we beat the collaborative filtering algorithm, the state-of-the-art in default parameter settings. In the further research, we will polish up our system by some other popular and powerful statistical models, such as Convolution Neural Network(CNN)[cite], a powerful weapon in deep learning fields and evaluate their performance with our present framework.

# Appendices

## Appendix A Codes

### A.1 Codes for sub-problem 1

```
clear
clc
[udata,uitem,uuser]=fun_readFiles('../dataset/u.data','../dataset/u.item','../dataset/joblist.txt'


FilmNum=size(uitem,1);
FilmAverScore=zeros( FilmNum,1 );
FilmTempCounter=zeros( FilmNum,1 );

for i=1:size(udata,1)
   FilmAverScore( udata( i ,2) ) = FilmAverScore( udata( i ,2) )+udata( i ,3);
   FilmTempCounter( udata( i ,2) )=FilmTempCounter( udata( i ,2) )+1;
end

FilmAverScore=FilmAverScore./FilmTempCounter;

UserNum=max(udata(:,1));
UserAverScore=zeros(UserNum,18);

for i=1:size(udata,1)
     UserAverScore( udata(i,1) ,: )=UserAverScore(udata(i,1) ,:) +(udata(i,3)./ FilmAverScore( uda
end

 %¡øÐÐźéÒżžŕ
 for i=1:size(UserAverScore,1)
     UserAverScore(i,:)=UserAverScore(i,:)./sum(UserAverScore(i,:));

 end

 %ijÇÂij£ÍźÛůÖÊý
 FilmScore_Preference=zeros(size(uitem));
 %ijÇÂij£ÍźÛůÖÊýÖÐÃ£ÿöÎżÖÃţÄ£ÍźÛůÖÊý
 FilmWeight=zeros(size(uitem));

 %ÇótçÓřÔÚÃ£ÿöÌâšÄÉÏţÄţÃůÖ
 for m=1:size(udata,1)
     %user: udata(m,1)
     %film:udata(m,2)
     %score: udata(m,3)
     %ÕâÿötçórţÄÀàšðitem: uitem(udata(m,2),:)
     %ÕâÿöótżğÿøţÄÈÍÖţ: UserAverScore(udata(m,1),:)
     FilmScore_Preference(udata(m,2),:)=FilmScore_Preference(udata(m,2),:)+ UserAverScore(udata(m,
     FilmWeight(udata(m,2),:)= FilmWeight(udata(m,2),:)+UserAverScore(udata(m,1),:).*uitem(udata(m

 end

FilmScore_Preference=FilmScore_Preference./FilmWeight;
FilmScore_Preference(isnan(FilmScore_Preference))=0;

AgeDimension=6;
OccupationDimension=21;
FilmScore_Age=zeros(FilmNum,AgeDimension);
FilmScore_Occupation=zeros(FilmNum,OccupationDimension);
FilmScore_Age_Counter=zeros(FilmNum,AgeDimension);
```

```
FilmScore_Occupation_Counter=zeros(FilmNum,OccupationDimension);

%ÏÂÃært'ÄêÁäűÎžÍÖřÒţżőůÖ
for m=1:size(udata,1)
    %user: udata(m,1)
     %film:udata(m,2)
     %score: udata(m,3)
    %age: fun_ageSegmentation(uuser(udata(m,1),2))
    %Occupation:uuser(udata(m,1),3)

    %ÊŒÏÈijÆËãřt'ÄêÁä
    FilmScore_Age_Counter( udata(m,2) ,fun_ageSegmentation(uuser(udata(m,1),2)))=FilmScore_Age_Cou
    FilmScore_Age( udata(m,2) ,fun_ageSegmentation(uuser(udata(m,1),2)))=FilmScore_Age( udata(m,2)
    %ÈžžóijÆËãřt'ÖřÒţ
    FilmScore_Occupation_Counter( udata(m,2) ,uuser(udata(m,1),3))=FilmScore_Occupation_Counter( u
    FilmScore_Occupation( udata(m,2) ,uuser(udata(m,1),3))= FilmScore_Occupation( udata(m,2) ,uuse
end

FilmScore_Age=FilmScore_Age./FilmScore_Age_Counter;
FilmScore_Occupation=FilmScore_Occupation./FilmScore_Occupation_Counter;
FilmScore_Age(isnan(FilmScore_Age))=0;
FilmScore_Occupation(isnan(FilmScore_Occupation))=0;
```

## A.2 Codes for sub-problem 2, 3

```
%For Problem 2: concate person's feture and movie feature together,
%using svm regression model(libsvm) to automaticly calculate the weight.
function [] = svm_regression_generator()
  try
      addpath(genpath('libsvm-3.18/'));
  catch
  end

  dataset_path='../dataset/';
  %cd(dataset_path);
  load([dataset_path,'FilmFeature.mat']);
%load film info.
  load([dataset_path,'UserFeature.mat']);
%load film preference info.
  [udata,uitem,uuser]=fun_readFiles('../dataset/u.data','../dataset/u.item','../dataset/joblist.tx
%load user info.

  feat=zeros(size(udata,1),72);score=[];  %the svm regression feature and corresponding output sco
  for i=1:size(udata,1)
    rate_rec_tmp=udata(i,:);
    %disp(num2str(rate_rec_tmp(3)));
%     feat_tmp=[UserAverScore(rate_rec_tmp(1),:),  ...
%user-preference info.
%               get_user_age_info(rate_rec_tmp(1),uuser),...
%user-age info.
%               get_user_job_info(rate_rec_tmp(1),uuser),...
%user-job info.
%               FilmScore_Preference(rate_rec_tmp(2),:),...
%film-preference score.
%               FilmScore_Age(rate_rec_tmp(2),:),...
%film-age score.
%               FilmScore_Occupation(rate_rec_tmp(2),:)];
%film-occupation socre.

%improved version: fusion user-preference and
  %film-preference:
```

```
 feat_tmp=[get_fusion_feat(UserAverScore(rate_rec_tmp(1),:),FilmScore_Preference(rate_rec_tmp(2),:
...    %user-preference info.
              get_user_age_info(rate_rec_tmp(1),uuser),...
%user-age info.
              get_user_job_info(rate_rec_tmp(1),uuser),...
%user-rate_rec_tmp(2)ob info.
              FilmScore_Age(rate_rec_tmp(2),:),...
%film-age score.
              FilmScore_Occupation(rate_rec_tmp(2),:)];
%film-occupation socre.

     %film's final score.

     score(i,:)=rate_rec_tmp(3);
      feat(i,:)=feat_tmp;
  end

% GA Algorithm to search for bestc and best g for svm regression model.
ga_option.maxgen = 100;
ga_option.sizepop = 20;
ga_option.pCrossover = 0.4;
ga_option.pMutation = 0.01;
ga_option.cbound = [0.1,100];
ga_option.gbound = [0.01,100];
ga_option.v = 3;
[bestCVmse,bestc,bestg,ga_option] = gaSVMcgForRegress(score(1:2000,:),feat(1:2000,:),ga_option);

  cmd = [' -s 3 -p 0.4 -h 0 -c ',num2str(bestc),' -g ',num2str(bestg)];
%using best param to train svm regression.
 % cmd = ['-s 3 -p 0.4 -h 0'];
 % cmd=[' '];
  model = svmtrain(score(1:60000,:), feat(1:60000,:), cmd);

  save([dataset_path,'svm_regression_model.mat'],'model');

  [predict,accuracy,decision_value] = svmpredict(score(60000:100000,:),feat(60000:100000,:),model)
%using 60000-100000 record as cross-validation.
  predict=round(predict);
  compare=[predict,score(60000:100000)];

  right=0;
  for i=1:size(compare,1)
     if (compare(i,1)==compare(i,2))
         right=right+1;
     end
  end

  disp(num2str(right/size(compare,1)));
end

function [age_feat]=get_user_age_info(userid,uuser)
 age_feat=zeros(1,6);
 age_feat(1,fun_ageSegmentation(uuser(userid,2)))=1;
 %age_feat=uuser(userid,2);
end

function [job_feat]=get_user_job_info(userid,uuser)
  job_feat=zeros(1,21);
  job_feat(1,uuser(userid,3))=1;
end

function [fusion_feat]=get_fusion_feat(user_pre,film_pre)
  fusion_feat=user_pre.*film_pre./(max(user_pre)*max(film_pre));
```

```
  for i=1:size(fusion_feat,2)
    if (isnan(fusion_feat(1,i)))
        fusion_feat(1,i)=0;
    end
   end


end
```

---

```
%For Problem 2: concate person's feture and movie feature together,
%using svm regression model(libsvm) to automaticly calculate the weight.
function [] = svm_regression_generator()
  try
      addpath(genpath('libsvm-3.18/'));
  catch
  end

  dataset_path='../dataset/';
  %cd(dataset_path);
  load([dataset_path,'FilmFeature.mat']);
%load film info.
  load([dataset_path,'UserFeature.mat']);
%load film preference info.
  [udata,uitem,uuser]=fun_readFiles('../dataset/u.data','../dataset/u.item','../dataset/joblist.tx
%load user info.

  feat=zeros(size(udata,1),72);score=[];  %the svm regression feature and corresponding output sco
  for i=1:size(udata,1)
    rate_rec_tmp=udata(i,:);
    %disp(num2str(rate_rec_tmp(3)));
%     feat_tmp=[UserAverScore(rate_rec_tmp(1),:),  ...
%user-preference info.
%               get_user_age_info(rate_rec_tmp(1),uuser),...
%user-age info.
%               get_user_job_info(rate_rec_tmp(1),uuser),...
%user-job info.
%               FilmScore_Preference(rate_rec_tmp(2),:),...
%film-preference score.
%               FilmScore_Age(rate_rec_tmp(2),:),...
%film-age score.
%               FilmScore_Occupation(rate_rec_tmp(2),:)];
%film-occupation socre.

%improved version: fusion user-preference and
  %film-preference:
 feat_tmp=[get_fusion_feat(UserAverScore(rate_rec_tmp(1),:),FilmScore_Preference(rate_rec_tmp(2),:
...    %user-preference info.
              get_user_age_info(rate_rec_tmp(1),uuser),...
%user-age info.
              get_user_job_info(rate_rec_tmp(1),uuser),...
%user-rate_rec_tmp(2)ob info.
              FilmScore_Age(rate_rec_tmp(2),:),...
%film-age score.
              FilmScore_Occupation(rate_rec_tmp(2),:)];
%film-occupation socre.

     %film's final score.

     score(i,:)=rate_rec_tmp(3);
      feat(i,:)=feat_tmp;
  end
```

```
% GA Algorithm to search for bestc and best g for svm regression model.
ga_option.maxgen = 100;
ga_option.sizepop = 20;
ga_option.pCrossover = 0.4;
ga_option.pMutation = 0.01;
ga_option.cbound = [0.1,100];
ga_option.gbound = [0.01,100];
ga_option.v = 3;
[bestCVmse,bestc,bestg,ga_option] = gaSVMcgForRegress(score(1:2000,:),feat(1:2000,:),ga_option);

  cmd = [' -s 3 -p 0.4 -h 0 -c ',num2str(bestc),' -g ',num2str(bestg)];
%using best param to train svm regression.
 % cmd = ['-s 3 -p 0.4 -h 0'];
 % cmd=[' '];
  model = svmtrain(score(1:60000,:), feat(1:60000,:), cmd);

  save([dataset_path,'svm_regression_model.mat'],'model');

  [predict,accuracy,decision_value] = svmpredict(score(60000:100000,:),feat(60000:100000,:),model)
%using 60000-100000 record as cross-validation.
  predict=round(predict);
  compare=[predict,score(60000:100000)];

  right=0;
  for i=1:size(compare,1)
     if (compare(i,1)==compare(i,2))
         right=right+1;
     end
  end

  disp(num2str(right/size(compare,1)));
end

function [age_feat]=get_user_age_info(userid,uuser)
 age_feat=zeros(1,6);
 age_feat(1,fun_ageSegmentation(uuser(userid,2)))=1;
 %age_feat=uuser(userid,2);
end

function [job_feat]=get_user_job_info(userid,uuser)
  job_feat=zeros(1,21);
  job_feat(1,uuser(userid,3))=1;
end

function [fusion_feat]=get_fusion_feat(user_pre,film_pre)
  fusion_feat=user_pre.*film_pre./(max(user_pre)*max(film_pre));

  for i=1:size(fusion_feat,2)
    if (isnan(fusion_feat(1,i)))
        fusion_feat(1,i)=0;
    end
   end


end
```

# References

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and*

*Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[2] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[3] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[4] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[5] Kai-Min Chung, Wei-Chun Kao, Chia-Liang Sun, Li-Lun Wang, and Chih-Jen Lin. Radius margin bounds for support vector machines with the rbf kernel. *Neural Computation*, 15(11):2643–2681, 2003.

[6] Bernhard G Häne, Kathrin Jäger, and Hans G Drexler. The pearson product-moment correlation coefficient is better suited for identification of dna fingerprint profiles than band matching algorithms. *Electrophoresis*, 14(1):967–972, 1993.

[7] Jason J Jung. Attribute selection-based recommendation framework for short-head user group: An empirical study by movielens and imdb. *Expert Systems with Applications*, 39(4):4049–4054, 2012.

[8] Roger J Lewis. An introduction to classification and regression tree (cart) analysis. In *Annual Meeting of the Society for Academic Emergency Medicine in San Francisco, California*, pages 1–14. Citeseer, 2000.

[9] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[10] Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266. ACM, 2003.

[11] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[12] Hua Zheng, Dong Wang, Qi Zhang, Hang Li, and Tinghao Yang. Do clicks measure recommendation relevancy?: an empirical user study. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 249–252. ACM, 2010.